A generic approach to topic models and its application to virtual communities

Von der Fakultät für Mathematik und Informatik
der Universität Leipzig
angenommene

D I S S E R T A T I O N

zur Erlangung des akademischen Grades

DOKTOR-INGENIEUR
(Dr.-Ing.)

im Fachgebiet
Informatik
vorgelegt von

Dipl.-Ing. Gregor Heinrich

Die Annahme der Dissertation wurde empfohlen von:

1. Prof. Dr. Gerhard Heyer, Universität Leipzig

2. Prof. Dr. Alexander Mehler, Goethe-Universität Frankfurt am Main

Die Verleihung des akademischen Grades erfolgt mit Bestehen
der Verteidigung am 28. November 2012

Abstract

This thesis investigates a generic model of topic models in order to facilitate their design and implementation. Topic models are probabilistic representations of grouped discrete data. Applied to text, the basic topic model represents documents as mixtures of topics – probability distributions over the vocabulary. In many cases, there exists a semantic relationship between terms that have high probability within the same topic. This phenomenon, which is rooted in the word co-occurrence patterns in the text, can be used for information retrieval and knowledge discovery in databases, and a large body of work extends the basic topic model, mostly modelling structures in the data beyond term co-occurrence or analysing different data modalities jointly to discover their inter-relations. While these approaches have been very successful individually, an analysis of topic models as a generic model class does not yet exist.

Such an analysis is undertaken in this thesis, based on the conjecture that important properties may be generic across models and that this, in turn, may lead to practical simplifications in the derivation of model properties, inference algorithms and finally design methods. As an exemplary application domain, virtual communities are considered, like those arising in large organisations, the scientific community and the "Social Web".

Work pursues a three-step strategy: In the initial Modelling part, theories of (1) virtual communities and (2) topic models are developed. For virtual communities, this thesis posits that a large part of the their available knowledge can be expressed by three types of entity and their inter-relations: actors (i.e., people or agents), media (i.e., documents and other information sources) and qualities (units of knowledge representation). For topic models, this thesis builds a generic representation that consists of networks of discrete mixtures, "networks of mixed membership" (NoMMs), and shows that this covers a large set of real-world models.

In the subsequent Inference part, generic algorithms for Gibbs sampling and variational inference are developed, revealing a general structure of model properties analogous to the structure of NoMMs. A central result of this work is a Gibbs meta-sampler that allows implementation of inference algorithms from NoMM structures directly. To improve scalability, variations of the generic sampling algorithm are studied that are based on parallelisations and accelerations of the serial algorithm, leading to significant speed-up of model implementations.

The final Application part combines the previous results to a design method that allows composition of topic models from modular NoMM structures, aligning them with structures in the data and monitoring model properties at each construction step. A case study applies this method to a scientific virtual community, and novel models for expert finding are developed that use semantic tags in addition to document text and authorship information to improve retrieval results and topic coherence.

Bibliographische Daten

Heinrich, Gregor
A generic approach to topic models and its application to virtual communities
(Ein generischer Ansatz für Topic-Modelle und seine Anwendung auf virtuelle Gemeinschaften)
Universität Leipzig, Dissertation
270 + xviii S., 90 Abb., 318 Lit., 5 Anh.

Selbständigkeitserklärung

Hiermit erkläre ich, die vorliegende Dissertation selbständig und ohne unzulässige fremde Hilfe angefertigt zu haben. Ich habe keine anderen als die angeführten Quellen und Hilfsmittel benutzt und sämtliche Textstellen, die wörtlich oder sinngemäß aus veröffentlichten oder unveröffentlichten Schriften entnommen wurden, und alle Angaben, die auf mündlichen Auskünften beruhen, als solche kenntlich gemacht. Ebenfalls sind alle von anderen Personen bereitgestellten Materialen oder erbrachten Dienstleistungen als solche gekennzeichnet.

Leipzig, 3. Januar 2012

Gregor Heinrich

Wissenschaftlicher Werdegang

Aug. 2012 – heute	Leiter Forschung und Entwicklung (CTO), semafora systems GmbH, Darmstadt: Semantic Web Technologien inkl. Reasoner; industrielle Anwendungen inkl. semantische Suche mit Ontologiesupport, Terminologie- und Ontologiemanagement; Management
Mai 2007 – Aug. 2012	Leiter Forschung und Entwicklung (CTO), vsonix GmbH, Darmstadt: Suche in Video-Metadaten; Nutzerinteraktion; skalierbare Signalverarbeitung; Projektmanagement
Okt. 2005 – Feb. 2008	Wissenschaftlicher Mitarbeiter, Universität Leipzig: Semantische Suche in Communitydaten, insbesondere verteilte Suche in Peer-to-Peer-Netzwerken und Topic-Modelle; Projektmanagement
Feb. 2000 – Mrz. 2002	Guest Professor, International Certificate Program for New Media, Fraunhofer Center for Research in Computer Graphics und Rhode Island School of Design, Providence, Rhode Island, USA: Serverseitige Weblogik
Apr. 1999 – Dez. 2004	Wissenschaftlicher Mitarbeiter, Fraunhofer-Institut für Graphische Datenverarbeitung (IGD), Darmstadt: Projektmanagement; Suche und Informationsvisualisierung für Wissensinfrastruktur; Audiovisuelle virtuelle Umgebungen
Mai 1997 – Jan. 1998	Research Scientist, Sensimetrics Corporation, Cambridge, Massachusetts, USA: Sprachdetektion und -klassifikation, Simulation binauraler Modelle
Okt. 1991 – Jan. 1998	Studium Nachrichtentechnik, Technische Universität Darmstadt: Diplomarbeit “Speech Detection Algorithms”

Acknowledgments

First and foremost, I would like to thank my supervisor Prof. Gerhard Heyer for his thematic and personal support, suggestions, encouragement and last but not least giving me the opportunity and freedom to pursue my thesis research in the setting and pace compatible with my external project work.

I am also very grateful to Prof. Alexander Mehler for the highly valuable suggestions in his review of my thesis.

There is no such thing as successful research without creative context, and many fruitful discussions contributed to my research. At the Department of Natural Language Processing of the University of Leipzig, I wish to thank all colleagues for their support and the exchange of many ideas, especially Florian Holz and Sven Teresniak. Of the former Leipzig colleagues, furthermore Chris Biemann and Frieder Witschel have been (and still are) vivid discussion partners, helping ask the right questions and much more.

What cannot be underestimated for the success of this thesis is the environment at Fraunhofer IGD and vsonix GmbH in Darmstadt, especially with Volker Hahn and Christoph Jung, as well as the support that came from Prof. Georgios Sakas and Prof. Michael Goesele. Also, let's not forget the encouragement by Ido Iurgel in the more difficult phases of the thesis. In addition, I am grateful to Prof. J. L. Encarnaçao for effectively putting me on the right research path.

However, all the thesis work would not have been possible without such a supportive family, and I would like to thank my parents Christa and Jens Heinrich as well as my brother Mathias Heinrich for always being there in all situations. This goes even more for my girlfriend Sandra Deutsch who I owe so much for her love, encouragement, support and also endurance, especially during the many "final" steps of my research and thesis work.

Contents

List of Figures

Chapter 1

Introduction

> *Innovation can be systematically managed if one knows where and how to look.*
> — Peter F. Drucker [1985].

Since antiquity, innovation has been the driving force to virtually any economy, and the "raw material" of innovation is the knowledge of its originators. Among the sources of knowledge, information exchange and collaboration are the most vital ones: In the hands of the right people, relevant information turns into knowledge, and during collaboration between the right people, new knowledge is created as the basis for novel engineering approaches, scientific ideas, business concepts, etc.

Virtual communities have become a central factor to support this exchange and collaboration [Scarbrough & Swan 2001, Huysman et al. 2003]. Virtual communities are groups of people that share a common interest and collaborate via electronic means. Because electronic media and communication permeate much of our current work and social environments, virtual communities have become ubiquitous, either implicitly (e.g., as the people connected by email communication or in the citation network of a digital library) or explicitly (e.g., by membership in a social network portal, a "Web 2.0" community or an organisation).

As a consequence of this omni-presence, content creation and consumption, electronic communication and collaboration can be interpreted as changes of the state of a virtual community: People leave traces that may be potentially relevant as knowledge sources. This includes "cues" that help find the right people, as well as resources that help make available the right information to these experts. More generally, tracking the structure and communication of the community may give important insights into the origination of innovation, as both the social system and its innovative and scientific output are inseparably connected [Ziman 2000].

Discovery of existing knowledge sources therefore depicts one of the main tasks when virtual communities are to be used to promote innovation. The choice of appropriate tools may, however, be a difficult one: Not only is the amount of data associated with virtual communities often huge, but also information may change quickly as the community evolves and it may be highly specialised. This often renders approaches uneconomical that lead to high-quality results in other fields, such as the knowledge-based methods of the "Semantic Web" [Biemann 2005, Staab & Studer 2009] or topic maps [Maicher & Park 2006]: Too much human intervention is required to create and maintain a structure of the knowledge domain (ontologies, topic map structures,

controlled vocabularies) as well as class–instance associations and other reference data that any automatic analysis is based on.

What may be better suited to such community scenarios are approaches that limit their inputs to the data and a model that "explains" them. Because for such "model-based" methods, assumptions on the data are known already at development time – in the form of the model – rather than at training time as in the case of knowledge-based methods, learning from data is a highly automatic process. However, the application of such methods to language data – in most cases text – needs to take into account some intricacies that may be summarised as follows:

- *"Sparsity problem"*: Text data are high-dimensional, and Zipf's law [Zipf 1949, Manning & Schütze 1999, Biemann & Quasthoff 2009] indicates that the majority of terms empirically observed in text are infrequent. The resulting term–document matrices consequently have a small portion of non-zero entries, and this sparsity typically grows with the size of the corpus.

- *"Vocabulary problem"*: Although there exists overlap between semantic and literal similarity, they are not identical in language data. Linguistic phenomena like polysemy or synonymy are examples where literal identity is associated with semantic difference and vice versa [Fellbaum 1998], leading to semantic ambiguity of text data. This is one of the main challenges in information retrieval [Manning & Schütze 1999].

It is clear that virtual communities do not only exhibit text as a single data modality. Rather, the typical case is that data are multi-modal, still having language at its core due to its central role in human communication, but complemented by varying types of meta-data. Depending on the actual scenario, these other "modalities" reach from authorship and co-citation relations to social interaction data and preference profiles and include attributes as diverse as geographical location, semantic annotations, activity profiles or even program code. In short, community data may be described as a multi-modal compound of unary and relational data types.

What simplifies the wide range of possibilities is that many of these modalities to a certain extent have similar properties to text, in most cases sharing the power-law distributions (link degrees, authorship, tag frequencies) and sparsity properties [Börner et al. 2004, Heyer 2011]. Furthermore, in many cases they behave like semantic data in the sense that the occurrence of a given feature (corresponding to a word) may have different meanings. For example, a link between persons in a social network may have various reasons, and to disambiguate them is one direction of current research.

Topic models. Out of the many methods that have been used in information retrieval [Baeza-Yates & Ribeiro-Neto 1999, Manning et al. 2008], natural language processing [Manning & Schütze 1999] and knowledge discovery in databases [Gaber 2010], one particularly suitable approach to reduce sparsity and vocabulary problems is the usage of *topic models*. Topic models are unsupervised methods to analyse text and other data that represent content as weighted sums of "latent topics" (a.k.a. "latent concepts" or "components"). In this thesis, we subsume under topic models different approaches that have followed the seminal approach of latent semantic analysis (LSA) [Deerwester et al. 1990], notably probabilistic methods like probabilistic LSA [Hofmann 2001] and latent Dirichlet allocation (LDA) [Blei et al. 2003b]. In particular, the most recent Bayesian approach LDA will be in focus.

Topic models decrease the sparsity problem described above by reducing the dimensionality of the data. In particular, topic models have very close relationship to principal components analysis (PCA): The singular value decomposition (SVD) that underlies the LSA method is identical to PCA under certain assumptions, and [Buntine & Jakulin 2005] show that LDA is a form of multinomial PCA.

Furthermore, topic models have been shown to reduce the vocabulary problem. By expressing documents as combinations of topics rather than lexical entities, it is possible to avoid vocabulary mismatch between semantically similar text fragments if topics are composed of semantically similar words, including synonymous ones. If words with several meanings appear in multiple topics, each one associated with the respective (semantically) similar words, the combination of topics to represent documents may be used to disambiguate polysemic words, as well, assuming that the query is semantically coherent. Remarkably, the methods used to establish topic models are based purely on the co-occurrences found in the data and no background knowledge is required, which is owed to the fact that word meaning is contextual [Firth 1957]. In the literature, there have been various empirical results on the semantic meaning of topics, including [Landauer & Dumais 1997] and [Griffiths et al. 2007]. They support the general idea of imagining text to be generated from a latent structure that comprises the actual meaning and is observed with additions of what may be called "word choice noise".

1.1 Topic models for virtual communities

In summary, topic models appear as an appropriate means of model-based analysis of document collections. Extending plain text applications to the various modalities of data available in communities leads to topic models that additionally target various types of meta-data as well as relational structures and have been targeted by various research efforts. Such models – especially those that extend the Bayesian model of LDA – have been published for analysis of authorship information in the author–topic model [Rosen-Zvi et al. 2004], to extract hierarchies of topics with the pachinko allocation model [Li & McCallum 2006] and different media types, such as the correspondence-LDA model [Barnard et al. 2003] for images.

Specifically for virtual community analysis, a wide range of results have been achieved, using various models that start from LDA and relatively simple extensions of the author–topic model to topic and role discovery [McCallum et al. 2004; 2005], and lead to much more complex models that handle unaligned corpora from different sub-communities and multilingual approaches [Boyd-Graber & Blei 2009, Mimno et al. 2009, Ni et al. 2009, Zhao & Xing 2007], document networks via citations [Dietz et al. 2007, Chang & Blei 2009], bibliometrics [Mann et al. 2006], trends and history of ideas [Wang & McCallum 2006, Hall et al. 2008], group and relation analysis [Wang et al. 2006], micro-blogging [Ramage et al. 2010], meeting and discourse analysis [Huang & Renals 2008, Purver et al. 2006], sentiment analysis [Mei et al. 2007a, Blei & McAuliffe 2007], social network analysis [Sinkkonen et al. 2008, Wahabzada et al. 2010, Xu et al. 2006] and many others.

In summary, since its early beginnings in the 1990's [Deerwester et al. 1990], topic modelling has grown to a large research area that applies a set of different modelling ideas to a wide number of problems, among which those in community knowledge discovery, like the ones mentioned above, form an important research strand. From this and from specific results in information

retrieval tasks [Wei 2007], one may conclude that topic models have indeed fulfilled the promise to reduce some of the problems at hand. As a result, presently topic models are making their way from a research subject into wider usage in industry and real-world applications in data mining, information retrieval and knowledge discovery.

Discussion. Before this background of a transition to "main-stream" usage, one may dare a look at the methods that topic models are currently approached with. Virtually all of the models mentioned above are inherited from the basic principle of LDA that proposed the use of mixtures of discrete clusters to describe observations in a Bayesian approach [Blei et al. 2003a]. They further share the use of conjugate distributions and therefore numerically well-behaved properties.

In the research literature, based on these core assumptions a certain informal "standard" has established itself to develop models, which may be recognised from almost any of the above papers. Presentation typically starts from the "basis" model, LDA, and loosely transcends from there by stating a generative process of how the data are (or may theoretically have been) produced, using methods of statistics along with a Bayesian network graphical representation. The models and their algorithmic implementations are then derived from scratch, instantiating more or less complex machine learning techniques like variational inference [Wainwright & Jordan 2003, Blei et al. 2003b] or Gibbs sampling [Geman & Geman 1984, Griffiths & Steyvers 2004].

While this approach rests on strong methodological feet, its complexity may on the other hand be an obstacle for a wider application of topic models: Designing and implementing models or deciding which models may be helpful in a particular situation currently requires expert knowledge in machine learning and statistics. This may be the reason why researchers from other fields tend to re-use simple topic model structures like LDA, avoiding more complex structures that might better suit their needs.

For the development of novel topic models, a benefit might therefore be derived from a separation of knowledge domains between topic model users who develop models with an application viewpoint and the respective expertise on one hand, and researchers who derive inference algorithms and apply a machine learning viewpoint on the other, which may be extended by aspects from fields such as distributed computing architectures to answer questions of scalability [Nallapati et al. 2007, Newman et al. 2009].

One solution for such a separation would be to focus less on the differences between models as in current literature but explicitly take into account their common properties and construct a "grammar" or meta-model for the models at hand. This in turn may be used to create some "front-end" to topic modelling that hides the underlying intricacies while exposing the grammar of topic models to application developers. Developers then express the problem within this restricted grammar and leave the rest to the envisioned development system.

This itself raises the question what such a grammar or meta-model of topic models would look like and to what extent it may simplify the implementation of inference algorithms, potentially with scalability improved by exploiting some generic properties. Furthermore, when dealing with data of virtual communities, characterising their domain-specific properties may help structure topic models.

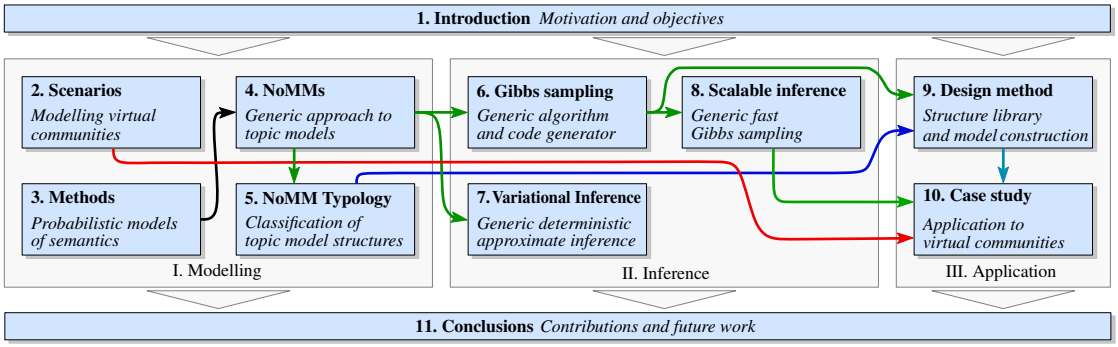


Figure 1.1: Chapter organisation and main dependencies.

1.2 Research objectives

In summary, what is missing in the state of the art is a solution that allows simple access to topic modelling and hides its complexities, including those of inference and scalability. The conjecture is that the common model properties indeed allow such simplifications. In addition to confirming this conjecture, it is of interest to investigate the connections between the data present in virtual community scenarios and the structures of topic models that permit their analysis. From this, research questions may be formulated:

- *Community modelling:* How may virtual communities be described in a generic way, also including the complex nature of knowledge whose hidden aspects may be captured in community data like those of the "Social Web" and digital libraries?

- *Topic modelling:* What is the "grammar" of topic models? Are there properties of topic models that may be derived on the basis of such a generic representation, which is not possible with today's methods? How may this be used to create a simplified approach to topic modelling?

- *Inference:* Based on the grammar of topic models, how may inference for topic models be generalised? How may this help simplify the implementation of algorithms? And are there possibilities to generalise scalability approaches like those for distributed and parallel computing architectures?

Success in answering these questions will lead to simplifications in the development of topic models, possibly without the need of page-long mathematical derivations as in today's research literature – and with largely reduced implementation effort. This may contribute to an end-to-end development tool for topic modelling.

1.3 Thesis structure

This thesis is structured into three main parts, Modelling, Inference and Application, a concluding chapter as well as an appendix. The following outline and Fig. 1.1 summarise the contents.

Part I. Modelling
In the Modelling part, Chapters 2 to 5, virtual community scenarios as the application domain and topic models as the methodology at hand are reviewed and models for them developed. The part also gives an overview of the state of the art.

Chapter 2: A model of community knowledge. This chapter formulates an abstract characterisation of knowledge in virtual communities appropriate for retrieval and knowledge discovery purposes. It is argued that a large part of the knowledge in a virtual community can be represented by three entity types – actors, media and qualities (AMQ) – as well as the relations that connect these entities. The AMQ representation provides the basis for both the work in later chapters and classification of existing approaches.

Chapter 3: Methods: Probabilistic models of semantics. This chapter introduces the general idea of representing text semantics by probabilistic means, focussing on latent semantics as a core methodology of this thesis. After introducing basic semantic and probabilistic concepts, the relation between different language and topic models is discussed, and based on this latent Dirichlet allocation (LDA) is introduced. Subsequently, besides discussing non-parametric extensions of LDA, this chapter treats querying and analysis methods for topics. All subsequent work builds on the methods laid out in this chapter.

Chapter 4: A generic approach to topic models: Networks of mixed membership. This chapter develops a generic model of topic models. To define the problem space, general characteristics for this class of models are derived, which give rise to a representation of topic models as "networks of mixed membership" (NoMMs), a domain-specific compact alternative to Bayesian networks.

Chapter 5: A typology of NoMM structures. The generic approach to topic models may help simplify derivation of new models. However, its potential advantages are not limited to that. Based on the state of the art in topic modelling, this chapter presents a typology of NoMM structures that serves as a classification of state-of-the-art models and as the basis for a "library" of building blocks for new models.

Part II. Inference
In the Inference part, Chapters 6 to 8, inference methods are investigated for the generic approach to topic models derived in the Modelling part.

Chapter 6: Gibbs sampling for NoMMs. This chapter contributes a generic derivation of Gibbs sampling for topic models, which are here represented as NoMMs. In addition to deriving Gibbs full conditional distributions, formulations for predictive inference and convergence monitoring are given, all valid across NoMM structures. These findings are the basis for a Gibbs sampling tool that is able to produce model-specific algorithm source code based on an easy-to-use model specification language.

Chapter 7: Variational inference for NoMMs. In this chapter, an alternative approach to inference in NoMMs is derived using variational inference. Opposed to Gibbs sampling, variational inference has not been widely used for more complex topic models. Central to the approach is the usage of a "topic field", a multi-way array of likelihoods for all latent configurations of a set of hidden variables that explicitly models their dependencies.

Chapter 8: Fast sampling methods for NoMMs. Based on the generic model of topic models and its Gibbs sampling formulation developed in Chapters 4 and 6, fast sampling schemes are introduced using serial and parallel methodologies. Furthermore, independent sampling of dependency groups is analysed for its convergence behaviour. The *speedups* achieved allow usage of models, especially with several dependent topics, for a much wider range of applications because scalability issues with existing implementations can be reduced.

Part III. Application
The Application part, Chapters 9 and 10, uses the results of Modelling and Inference parts for topic modelling in community data. This is also used to illustrate how model design simplifies when the NoMM representation is employed.

Chapter 9: Towards model design using NoMMs. This chapter investigates how topic models may be constructed using the advantages of the NoMM representation. In particular, a design method is proposed that is based on assembling models from NoMM sub-structures, associating them with structures in the data. Using the numerical properties of Gibbs full conditionals and data likelihood as predictors of model behaviour, models can be constructed in a controlled way than using today's approach.

Chapter 10: Case study: Topic modelling for virtual communities. To validate the "tools" developed in the thesis, they are applied to a concrete application scenario: expert finding using document content and semantic annotation information. This chapter demonstrates a "round-trip" process to topic modelling, starting from AMQ models as input to the design method, designing the model structure, implementing the algorithms using the Gibbs meta-sampler and finally evaluating the models. Besides serving as exemplary validation for the NoMM design process developed in this thesis, the proposed expert–tag–topic models are contributions in their own right.

Chapter 11: Conclusions. Discusses the results and contributions of the thesis and proposes future directions.

Appendix. The Appendix presents background information in order to keep this thesis self-contained. It includes an overview of notation in Appendix A, a review of exponential-family distributions and conjugacy in Appendix B, and details about implementations done in this thesis as well as the machinery and data used in Appendices C and D, respectively. Finally, some details are given on the application models in Appendix E.

Part I

Modelling

The Modelling part is concerned with introducing background, analysing exising work and formulating generic models from it. Chapter 2 investigates the data structures in the application scenario of communities and formalises a generic structure that may be used as a model for particular knowledge discovery applications. Chapter 3 then reviews methods to analyse the actual data: topic models. Based on this, more complex models are analysed in Chapter 4 and formalised into a generic model of topic models and a representation as networks of mixed membership (NoMMs). Using this representation, Chapter 5 reviews existing work in terms of the NoMM structures contained, attempting to give a point of departure for re-using research results from the literature on a modular basis in new model designs.

Background:

- Virtual communities → Chapter 2
- Latent semantics and topic models→ Chapter 3
- Topic model state of the art → Chapter 5

Main contributions:

- Generic community model → Chapter 2
- Generic topic model → Chapter 4
- Network of mixed membership (NoMM) representation → Chapter 4
- Structural analysis of NoMMs → Chapter 5

Chapter 2

Scenarios: Modelling virtual communities

This chapter formulates an abstract characterisation of knowledge in virtual communities appropriate for retrieval and knowledge discovery purposes. It is argued that a large part of the knowledge in a virtual community can be represented by three entity types – actors, media and qualities (AMQ) – as well as the relations that connect these entities. The AMQ representation provides the basis for both the work in later chapters and classification of existing approaches.[1]

2.1 Introduction

Virtual communities have become a major factor for the design of information systems and Web-based applications, giving rise to community-based information infrastructure. This development is apparent for instance in the emergence of social computing and the "Social Web" or "Web 2.0" [Wang et al. 2007], as well as in the paradigm shift in enterprise knowledge management from techno-centric approaches towards tacit knowledge and social capital [Huysman et al. 2003, Gottschalk 2005]. It even extends to the increasing importance of peer-to-peer systems [Witschel & Böhme 2005] and collaboration grids [Stockinger 2007] for information management and processing, where peers or grid nodes act in lieu of humans and can be considered members of generalised virtual communities.

A major reason for this development is that the information available from a community-based system includes an added value that is impossible to generate or capture using classical, purely content-based approaches because it directly gains from the intelligence, creativity and social behaviour of people. On the "supply" or authoring side, this added value consists of processes to generate content interesting for the community, which is done by providing infrastructure to easily author or make available contributions, to ask or answer questions (Web 2.0, Social Web, peer-to-peer), to supply and exchange explicit and tacit knowledge (knowledge sharing infrastructure) or to offer computational services (grid or cloud services). On the other, the "demand" or retrieval side, the added value consists of processes to provide social decision support, which is done by collaborative filtering, feedback mechanisms and importance measures that often use the

[1] The model has been published in [Heinrich 2010] (originally presented in 2007).

relational structure of the community (social recommenders, reputation systems). Furthermore, by offering suitable infrastructure to facilitate and amplify social processes, community-based systems tend to reinforce the identification of their users as members of a community and thus create the motivation to contribute to the community.

Objective. In this chapter, we investigate the question of how to capture some of the added values that community-based approaches offer and how to combine them with content-based approaches in a reasonably compact model. Such a model does allow description of the structure of community-based information either for comparison and classification of existing systems, or as a structural basis for new developments.

The final goal that this work contributes to is to build infrastructure for *access* to the knowledge that exists within the community, i.e., the demand side of the infrastructure. However, one of the features that makes users of community-based systems unique compared to those of others, is that they act on both sides of the system, supply and demand, i.e., are contributors and consumers alike. Looking at retrieval approaches cannot therefore completely exclude the content creation processes in communities.

Chapter outline. This chapter continues with a more detailed discussion of the requirements needed for the envisioned model in Section 2.2. As the core contribution of this chapter, the model itself is proposed in Section 2.3 and applied to a practical example in Section 2.4. Finally, after a review of related work in Section 2.5, the present approach and future directions are discussed in Section 2.6.

2.2 Requirements

This section derives the properties of the envisioned model of virtual communities by specifying requirements. There are three major types of requirements: First we need to identify the scenarios that should be covered (Section 2.2.1) and need to define how specific they should be modelled (Section 2.2.2). Finally, the types of community knowledge of the scenarios that should be addressed in the model need to be considered (Section 2.2.3).

2.2.1 Requirement 1: Usage scenarios covered

The main purpose of the envisioned model is to *represent a virtual community to support information access scenarios and associated knowledge discovery tasks*. Such tasks are for instance to find community members and documents according to certain criteria. The scenarios for information access include:

- *Expert finders:* Systems that allow to find people who have expert knowledge in a given topic, based on profile information, document content and authoring information (e.g., ArnetMiner [Tang et al. 2007], Webwijs [Balog et al. 2007], ExpertFinder [Hogan & Harth 2007], XperT [Heinrich 2004]);

- *Digital libraries:* Systems that allow to access documents where the community consists of the authors who mutually cite their articles or monographs (ArXiV, the ACL Anthology [Radev et al. 2009], CiteSeer [Giles et al. 1998], the ACM Digital Library [White 2001], DBLP [Ley & Reuther 2006]);

- *Collaborative authoring:* Systems that allow community members to contribute to a collaborative information repository, either *ad hoc* asynchronous communication (mailing lists, forums) or as "Web 2.0" or "Social Web" tools like blogs, wikis [Wang et al. 2007, Nemoto et al. 2011]) and other approaches that make accessible and allow users to contribute content (Twitter, flickr and YouTube) or meta-content (structured data as in IMDB, and/or bookmarks, citations and tags as in CiteULike and del.icio.us or Bibsonomy [Jäschke et al. 2007]);

- *Social network platforms:* Systems that allow self-authored personal profiles and connections that developed from dynamic contact lists (Facebook, Xing, LinkedIn, myspace, cf. [Chang & Blei 2009, Papagelis et al. 2011] for research work); these systems may be considered types of collaborative authoring platforms with a focus on the social links rather than the content;

- *Peer-to-peer systems:* Systems that distribute content over a community of peer modules (that can themselves represent a community of people) and can be considered "generalised communities" (SemPIR [Witschel & Böhme 2005, Witschel, Holz, Heinrich & Teresniak 2008, Holz, Witschel, Heinrich, Heyer & Teresniak 2007]). For storage and computation services more generally, this extends to computation grids and cloud computing.

2.2.2 Requirement 2: Scope and specificity

The model is intended for usage primarily in early stages of system design where considerations on the target behaviour and design are being undertaken (cf. [Barna et al. 2003]). Therefore, the model should be generic enough to be *independent of scenario specifics* like particular types of persons or documents. This also makes it suitable for representation and comparison of a wide variety of existing and new systems and scenarios. In fact, the result may be a form of data model or ontology whose structure can be specialised for particular scenarios in question, similar to meta-modelling methodologies (cf. [Bezivin 2006] or the Unified Modeling Language (UML [Weilkiens 2007]).

2.2.3 Requirement 3: Information and knowledge types addressed

Many systems for community-based information infrastructure are not only used to retrieve information but actually locate knowledge. For these cases, it is inevitable to not only *represent documented information as explicit knowledge*, but to also *integrate tacit knowledge* [Nonaka & Takeuchi 1995, Boisot 1999] and possibly *social capital* [Lesser 2000] into the model. This way, a great part of the added value ascribed to community-based approaches can be captured, as suggested by the literature (see, e.g., [Wenger 1998, Huysman et al. 2003] covering communities of practice).

The access to knowledge needs a few more remarks. By definition, tacit knowledge – or "knowing" as a process rather than a state [Polanyi 1974] – is restricted to individuals or groups of individuals and it is in most cases difficult to write down (to "externalise" [Nonaka & Takeuchi 1995]) because it depends on intangible factors like experience, procedural knowledge, special talents, cultural background and norms that can only be made available to others by direct interaction. Social capital as a form of collective tacit knowledge [Davenport et al. 2003] supports

this interaction by holding together communities. Furthermore, it may be regarded as the basis for offers of help and collaboration needed to achieve shared goals or to solve problems [Putnam 2000]. Social capital also entails "relational capital" within a social network.

To give an example, in an online forum the experience of an expert answering a complex question cannot be written down in its entirety; it is tacit. Furthermore, the fact that such exchange works at all is often a result of the social capital established within the community, which is tacit as well. The predominant way to approach the problem of making available such tacit knowledge is to identify the expert, e.g., from a profile, from previous answers or articles, or by explicit recommendation, possibly confirmed by the location within a social network that reflects the communication within the community.

Therefore, "cues" to tacit knowledge (and social capital) in the community are important auxiliaries, and a model covering discovery of tacit knowledge must incorporate them. The types of such cues are manifold, and we take an "inductive" approach and summarise common cases:

- *Authoring and reference information:* Tacit knowledge leaves traces in documents that are created in the community, either by experts themselves ("authoring") or via reference in documents by other authors, such as in reports and in scientific citations ("reference"). This is not restricted to text content, like scientific authoring, but also in non-textual media, for instance in the way a movie is edited by an expert editor. Authoring and reference information is captured "en passant" from the existing processes in the community.

- *Profile information:* The existence of expertise can be catalogued using questionnaires, interviews, structured CVs and other means that are "actively" or explicitly applied to capture the existence of tacit knowledge, as in many knowledge and skills management approaches. However, many tacit skills are unknown even to the expert, and in these cases, only by interaction and problem solving can tacit knowledge be located and may be "implicitly" captured by tracking collaboration. Both active and implicit methods to capture specific traits and properties of users contribute to profiles.

- *Social network information:*[2] Important pieces of tacit community knowledge are identified from the structure of the community, i.e. the position of individuals and groups in the social network. Depending on the type of relations available as a representation of the real social network, this may allow identification of experts by their embedding into clusters of other experts, as well as possibly capture cues of social capital, such as trust, recommendation and reputation, which are important prerequisites to sharing of tacit knowledge through collaboration.

[2] In "generalised communities" like peer-to-peer networks, social network information does not represent social capital proper but similarly, relational properties of the network are used as cues to identify items of interest.

2.3 The actors–media–qualities model

The requirements collected in Section 2.2 yield a set of qualitative input factors to develop the envisioned model. With a focus on information access, Requirement 1 implies the need for a semantic representation of the items in the model that can be used to evaluate the relevance to a query, i.e., some sort of profile or set of "qualities" that can be associated with the items. The generic scope (avoidance of scenario specifics) from Requirement 2 implies what in ontology design is called "minimal ontological commitment", i.e., the restriction of the model to a minimum of elements [Gruber 1994]. Requirement 3 implies on one hand the representation of explicit knowledge items, which can be modelled as documents or, more generally, "media", on the other cues of tacit knowledge, i.e., the types of sources in the list in Section 2.2 need to be included. Authoring and reference demand for a connection of documents or media with community members, and profiles can be considered special cases of documents. Finally, social network information indicates the appropriateness of a graph-based representation of the community.

In fact, such a graph representation is flexible enough to be the structural basis for the entire model. Authoring information or other associations can be expressed by including into the network media items and connections with the authors. Moreover, semantic or other qualities associated with the items can be directly included as nodes in the graph representation.

In the next subsections, we define the model in terms of a graph structure. We first introduce node types in Section 2.3.1, its edge types in Section 2.3.2 and finally the complete model structure in Section 2.3.3.

2.3.1 Defining entities

We define AMQ entity types similar to classes in ontology or software analysis and denote them by calligraphic letters like $\mathcal{A}$. Our model supports inheritance relationships (*is_a*), i.e., entity types may have a hierarchy of subtypes that are denoted in italic type, $A \subset \mathcal{A}$, etc. Furthermore, it supports aggregation relationships (*has_a*). Instances, i.e., objects that represent the actual data, will then be denoted in lower case, $a \in A \subset \mathcal{A}$. Considering type hierarchy, for simplicity we will use the shorthand $a \in \mathcal{A}$ to denote that a is an instance of some type $A \subset \mathcal{A}$. Three root entity types are proposed to model a community according to the above requirements: actors, media and qualities.

Actors, $a \in \mathcal{A}$**,** are entities that represent everyone/everything acting in an autonomous way, which implies actions like to *write*, *collaborate*, *query*, *study* or to *assess*, in addition to actual knowledge (to *know*). These actions will later be defined as relations. Actors bear tacit knowledge, and naturally represent people and groups of people that engage in knowledge sharing and interaction. As a special case, intelligent agents can be considered actors although they are often represented via explicit rule sets. Subsumption of actors is defined (an author *is a*n actor), as is aggregation (a group *has a* number of authors).

Media, $m \in \mathcal{M}$**,** are entities that contain information and "react" to actors. Media bear explicit, i.e., verbal or numerical knowledge, and can be thought of as a generalisation of documents to all formats that can contain explicit knowledge or serve as cues to knowledge, including for example project and course documentation, audio tracks and images, video clips and other artefacts. In addition, user queries fit into this scheme as "reciprocal" media (requesting rather than providing

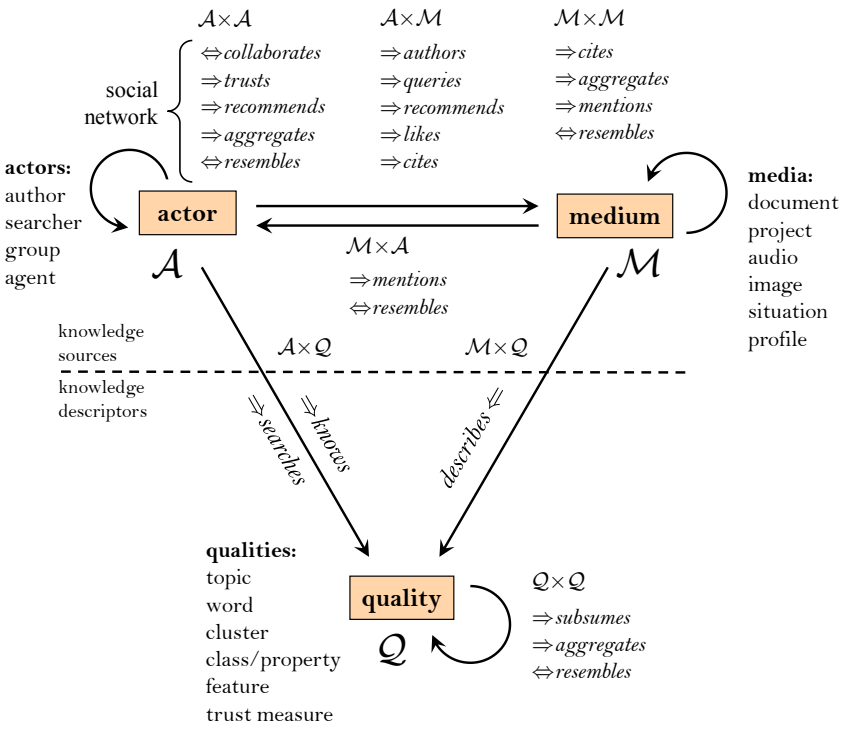


Figure 2.1: Structure of the AMQ model, with example entity and relation (italics) types. Directed relations between classes denoted with ⇒, undirected ones with ⇔.

knowledge). Furthermore, user profiles behave like media. Like actors, media allow subsumption (a document *is a* medium) and aggregation (a book *has a* number of chapters).

Qualities, $q \in \mathcal{Q}$**,** are entities that provide a set of attributes to describe actors, media and other qualities in a way that inference can be performed on them. Such inference includes comparison using a distance or similarity metric and consequently retrieval. Qualities as the units of knowledge representation are one key to mapping existing knowledge discovery and information retrieval methods into the model and can be used develop new models. Considering existing language and semantic representation frameworks, full-text approaches use inverted indices as representations of qualities. In Semantic Web-based information retrieval, reasoning requires qualities to be defined with formal semantics, i.e., $\mathcal{Q}$ is the representation of an ontology that can be queried or reasoned upon. And when using latent-semantic models like latent Dirichlet allocation [Blei et al. 2002], automatically inferred latent variables (commonly referred to as latent topics) are the qualities to describe the entities in the model. Finally, other types of representations may be added as qualities, such as vectors for content-based features when dealing with multimedia data. Generally, different types of qualities may be combined.

For a schematic view of these entities, see Fig. 2.1, where they are displayed as rectangles, along with examples of typical classes they subsume. With regard to its three root entity types, we call the model the *actors–media–qualities model* or *AMQ model*.

2.3.2 Defining relations

With the entities allowing the representation of real-world items in the model as instances of one of the actors, media and qualities types (or subtypes), relations between them provide the actual information used for knowledge discovery and retrieval. Relations in the AMQ model are typed and weighted, $R(x, y) : \mathcal{X} \times \mathcal{Y} \rightarrow I_R$ where $I_R \subset \mathbb{R}$ is the set of allowed values. Depending on the relation R, this interval can be discrete and binary, $I_R = \{0, 1\}$, probabilistic, $I_R = [0, 1]$, some distance or similarity measure, $I_R = [0, \infty)$, or any other set depending on the semantics of the relation. Typically, relations restrict the node types $\mathcal{X}$ and $\mathcal{Y}$ that they map to each other (domain and range), and may be directed or undirected. Typical relation types as given in Fig. 2.1 will be discussed in the next paragraphs.

Media–quality relations, $R(m, q) : \mathcal{M} \times \mathcal{Q} \rightarrow I_R$, describe the semantics of media. Explicit knowledge cues in media items use the media–quality relation *describes*(m, q), denoting explicit knowledge in a particular subject. Here, $I_R = [0, 1]$ is a function that maps to a relevance value for the subject, and to describe this subject, weights are established from the medium to all possible qualities q_i that this subject is composed of. Combining the weightings of an media–quality relation over several qualities $\vec{q} = \{q_i\}_i, i \in [1, K]$ can be expressed by introducing a shorthand for a vector weighting function $R(m, \vec{q}) : \mathcal{M} \times \mathcal{Q} \rightarrow (I_R)^K$ over all qualities q_i. This can for instance represent a vector of topic probabilities or a set of binary association functions with the range of ontology classes. When trying to retrieve relevant documents, the subject is expressed as a set of qualities, which themselves are extracted from a query text or other object (as a "reciprocal" medium).

Actor–quality relations, $R(a, q) : \mathcal{A} \times \mathcal{Q} \rightarrow I_R$, describe the semantics of knowledge associated with an actor. The central relation with respect to knowledge cues is the actor–quality relation *knows*(a, q), which, however, is not directly observable. In most approaches in the literature, the *knows*(a, q) relation is inferred, for instance from *author*ship: For example, in the Webwijs [Balog et al. 2007] and XperT [Heinrich 2004] systems, knowledge cues from documents are used via the *describes*(m, q) relation, and experts are inferred via the actor–media relation *authors*(a, a). The author-topic model [Rosen-Zvi et al. 2004], however, directly extracts latent topics for actors, thus implementing the *knows*(a, q) relation directly. Opposite to this "supply" dimension of knowledge, the "demand" of specific knowledge can be evaluated for actors, which is reflected by the actor–quality relation *searches*(a, q) that can be inferred via the *describes*(m, q) and *queries*(m, q) relations of associated media (for all explanations, see Fig. 2.1). Comparing the qualities of both supply and demand dimensions enables the functionality of matchmaking systems like that in [Reichling et al. 2005].

Actor–media relations, $R(a, m) : \mathcal{A} \times \mathcal{M} \rightarrow I_R$ or $R(m, a) : \mathcal{M} \times \mathcal{A} \rightarrow I_R$, describe the association of an actor with a medium or vice versa. Actor–media relations usually derive from authoring and reference information (*authors*(a, m), *cites*(m, m'), *recommends*(a, m), etc.). Furthermore, query actions by actors are a special types of relation, *queries*(m, q). Typically, such information is often explicit and can be extracted a priori as a basis for inference. Inferred actor–media relations are used in collaborative approaches to express recommendation (*recommends*(a, m)) and preference (*likes*(a, m)).

Media–media relations, $R(m, m') : \mathcal{M} \times \mathcal{M} \rightarrow I_R$, describe mutual relationships between media. Media–media relations play an important role in citation networks and digital libraries, both as references and aggregation. Like actor–media relations, they are often explicit and can be used as basis for inference. An important inferred relation for retrieval is similarity (*resembles*(m, m)).

Actor–actor relations, $R(a, a') : \mathcal{A} \times \mathcal{A} \rightarrow I_R$, describe social structure of the community and other relations between actors. Actor–actor relations can represent information on social capital in the community. For example, Opal [Heinrich et al. 2005a] uses relational cues such as friends, colleagues, and co-workers as well as ratings between collaboration candidates [Davenport & McLaughlin 2004]. Depending on the application case, different types of inference are possible. An example is to find an actor who is an expert in a topic and trusted by reputable actors. The inference based on explicit cues is the same as described for actor–quality relations, but the set of relevant actors now is filtered via appropriate network criteria, such as shortest path or reputation measures that aggregate weighted ratings (see, e.g., [Pujol et al. 2003] and references therein). An alternative way to perform inference in an integrated manner is to use statistical relational learning techniques that integrate semantic and relational steps of inference (see, e.g., [Neville 2006]).

Quality–quality relations, $R(q, q') : \mathcal{Q} \times \mathcal{Q} \rightarrow I_R$, map knowledge description frameworks into the AMQ model. For instance, when using ontologies quality–quality relations may include RDFS or OWL relations (e.g., *rdfs:subClass*, properties or aggregations). The AMQ model deliberately avoids to make any commitment on the formalism for such relations, allowing the incorporation of axiomatic descriptions of qualities, for instance to use the results of Semantic-Web inference or subsumption relations between latent topics in a topic hierarchy. Furthermore, quality–quality relations are the place in the model where distance measures fit in to compare actors and media as the knowledge sources in the model, with ontology approaches on one hand (leading to binary results) and real-valued retrieval functions modelling relevance on the other. For instance, two actors in an expert finder system may be similar in terms of their knowledge if the qualities they are described with are similar.

2.3.3 AMQ graphs and inference

In order to complete the AMQ model, all data are joined in a graph structure [Diestel 2005], which is the basis to formulate inference problems. More specifically, taking ideas from ontology modelling, e.g., [McGuinness & van Harmelen 2004], schema and instance structures are distinguished.

Schema graph. The structure that combines the entities and relations discussed in the last sections is defined as an AMQ schema graph, $\mathcal{G}(\mathcal{V}, \mathcal{E})$, with the vertex set consisting of the three entity types (possibly their subtypes), $\mathcal{V} = \mathcal{A} \cup \mathcal{M} \cup \mathcal{Q}$, and the edge set mapping to the various relation types between them, $\mathcal{E} : \mathcal{V} \times \mathcal{V} \rightarrow \mathcal{R}$, with $\mathcal{R} \supset R$ denoting the set of relation types. Fig. 2.1 can be understood as a simplified example of a schema graph where the different node and edge types are collapsed into the clique of root entity types. This will be extended by a more expressive graphical notation in Section 2.4.

Instance graph. While the schema graph reflects the structure of the data about the community, the corresponding instance graph $G(V, E)$ fills this AMQ schema with data, leading to a kind of

generalised co-citation graph or social network [Wasserman & Faust 1994]. The instance graph contains typed objects as vertices, $v \in V$, where each vertex v has a type that is a member of $\mathcal{V}$ in the schema, as well as edges between the objects, $e \in E$, where each e maps to a relation type $R \in \mathcal{R}$ and a weight, i.e., $E : V \times V \rightarrow \mathcal{R} \times \mathbb{R}$ with $\mathbb{R} \supset I_R$ subsuming the range of all weighting functions $R(x, y)$. Note that this definition can be easily extended to hypergraphs by allowing edge sets with higher vertex counts per edge and to vector relation types as described above.

In order to represent a virtual community for concrete knowledge discovery or retrieval scenarios, typically only a small set of entity and relation types need to be included in a particular instance of the schema, depending on the available information on the community and the retrieval mechanisms required to fulfil the envisioned retrieval tasks.

Inference in AMQ models is the process of identifying or creating entities or relations in the AMQ instance graph by analysis of its semantic or structural properties. Semantics here refers to the qualities associated with entities (e.g., topics associated with a document), and structure to the general topology of the AMQ graph (e.g., co-citation, social network) spanned by the different data available.

More formally, inference algorithms can be defined as transformations from a given instance graph structure $G(V, E)$ to another structure that adds the inferred items to G: $G' = G(V, E) \cup G(\hat{V}, \hat{E})$.

In this way, standard methods of information retrieval and inference may be expressed in the AMQ model, providing a method to classify or unify existing algorithms, possibly creating a library of standard algorithms for re-use. Furthermore, the method allows definition of novel inference schemes that may use combinations of existing algorithms or lead to completely new approaches. As the AMQ model itself makes no commitment on the type of inference used, the range of possibilities is wide. In the next section, this will be explained with an example.

2.4 Example: The ACL Anthology as an AMQ model

An illustrative application for an AMQ model is the ACL Anthology (ACLA) digital library[3], which offers research publications in the scientific field of computational linguistics online and is available as a full-text corpus [Radev et al. 2009]. The authors of the publications in the ACLA can be considered to form a scientific virtual community whose members venture to create new knowledge by using and extending existing sources.

Being only one example among a set of scientific digital libraries (e.g., CiteSeer [Giles et al. 1998], the CORA[4], NIPS[5] datasets or the digital libraries in Section 2.2), beside the semantic content of titles, abstracts and full-texts, ACLA gives access to authorship, co-citation, publication time and venue or conference information and thus can be used to track the knowledge creation process in the community, to identify relevant and influential papers or perform other knowledge discovery or retrieval tasks.

Before we analyse discovery and retrieval tasks possible on these data in Section 2.4.2, we characterise the structure of the ACLA data as an AMQ model in Section 2.4.1.

[3] `http://aclweb.org/anthology-new/`. More information on this data set is found in Appendix D.2.

[4] `http://www.cs.umass.edu/~mccallum/code-data.html`.

[5] `http://nips.cc/books`, see Appendix D.2.

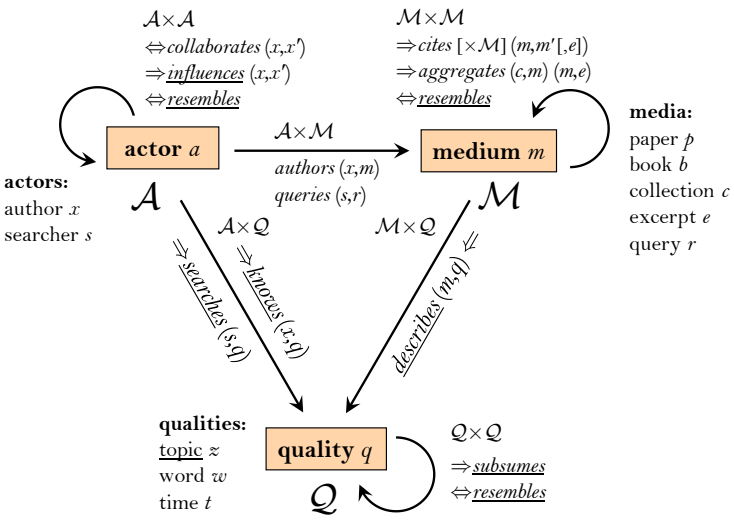


Figure 2.2: AMQ schema considered for the ACL Anthology and other digital library corpora.

2.4.1 Schema

The AMQ schema structure is shown in Fig. 2.2. Compared to Fig. 2.1, only the entities and relations relevant to the ACLA scenario are displayed, but on the other hand, Fig. 2.2 adds some additional information on the schema:

- Entities are added symbols, like x for an author as a special type of actor, a.
- Relations: Symbols are used to restrict the domain and range of the different relations to subclasses of the top types. For example, $\mathit{authors}(x, m)$ applies to authors x (as opposed to searchers, s) but to all possible media types, m. This notation allows to retain the collapsed straight-forward graphic representation of the AMQ schema. If no constraints are specified, a relation applies to the root types.
- Underlined types are considered inferred, whereas the others can directly be extracted from the available data.

Regarding actors, there are authors and searchers. The searcher entity $s \in \mathcal{A}$ is added to represent a querying process of someone retrieving data from the corpus. Looking at media entities seems self-explanatory, an excerpt $e \in \mathcal{M}$ being part of a medium ($\mathit{aggregates}(m, e)$) that can serve as context for a reference to another medium. Then optionally the binary $\mathit{cites}(m, m')$ relation between two documents is extended to a ternary $\mathit{cites}(m, m', e)$ relation with additional excerpt e as part of m (leading to a hyper-graph structure). Finally, there are three different types of quality: topic, word and time. While the latter two can be directly read from the corpus by indexing or from metadata, topics are themselves inferred entities, e.g., extracted by latent Dirichlet allocation or the author–topic model (see qualities definition in Section 2.3.1). The quality–quality relation *subsumes* can be applied for instance to a topic for hierarchical topic models, to a word when using some semantic hierarchy and to nest periods of time.

2.4.2 Retrieval and discovery tasks

On the ACLA data, various existing and novel retrieval and knowledge discovery methods can be applied, and here we view them from the perspective of how they are expressed in terms of the AMQ model.

Media retrieval is to find documents etc. for a given query. This requires initial indexing, which creates *describes* relations between media and qualities (words for inverted index or vector-space models [Baeza-Yates & Ribeiro-Neto 1999, van Rijsbergen 2004], inferred topics for latent semantics, etc.). During search, the same is done for a query, completing the graph with the respective weighting. The actual ranking of relevant documents is then based on an appropriate distance measure between the weightings of the *describes* relation, using vector weighting function $describes(m, \vec{q}) : \mathcal{M} \times \mathcal{Q} \rightarrow \mathbb{R}^K$ with K qualities for a given medium m. To obtain a medium m from a query r, this may be written symbolically as:

$$\text{document } m \Leftarrow \text{query } r :: \mathrm{D}\{describes(m, \vec{q}) \parallel describes(r, \vec{q})\} \; .$$

This notation reads: The value for document m "is obtained from" ($\Leftarrow$) query r "as a function of" (::) the distance ($\mathrm{D}\{\cdot\|\cdot\}$) between the weights of the $describes(\cdot, \vec{q})$ relations for m and r. Below, the distance $\mathrm{D}\{a\|b\}$ is used as a shorthand for any $resembles(a, b)$ relation.

In the example, both words and topics as qualities allow combination of literal and latent-semantic search in an appropriate retrieval function or distance measure. For latent semantics, the weighting function $describes(m, \vec{q})$ represents for instance the probability distribution $p(z|m)$ over K topics and implies the existence of topic distributions $p(w|z)$ that map words to topics (cf. Chapter 3). Graphically, this task and the following ones are depicted in Fig. 2.3.

Expert finding. Extending document retrieval to scenarios like expert finding (see Section 2.2) is straight-forward: Ranking for retrieval of authors x for a query r is done via distances between $knows(x, r)$ relations, or alternatively $describes(m, q)$ relations of documents authored by a particular person, which are identified via the $authors(a, m)$ relation:

$$\text{author } x \Leftarrow \text{query } r :: \mathrm{D}\{knows(x, q) \parallel describes(r, q)\};$$
$$knows(x, q) :: describes(m, q)\ \forall\{m : \exists\, authors(x, m)\}$$

where the second line indicates that the $knows(x, q)$ relation is a function of the $describes(m, q)$ relations "for all" ($\forall$) documents m "such that" (:) author x "exists" ($\exists$) among the authors of m.

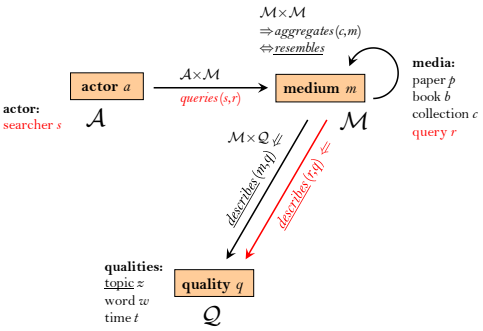
For the actual implementations of the associated algorithms, numerous possibilities exist in the literature, e.g., the mentioned [Rosen-Zvi et al. 2004] or [Heinrich 2004] that make use of latent topic distributions $p(z|x)$ for authors.[6]

Advanced tasks. Beyond this, various other discovery tasks can be performed using the ACLA data and corresponding schema in Fig. 2.3, for instance:

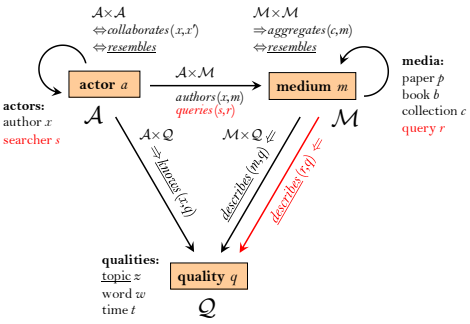
- Semantic matching: For a given document, the distance to other documents is inferred:

  $$\text{document } m' \Leftarrow \text{document } m :: resembles(m, m');$$
  $$resembles(m, m') :: \mathrm{D}\{describes(m, q) \parallel describes(m', q)\}$$

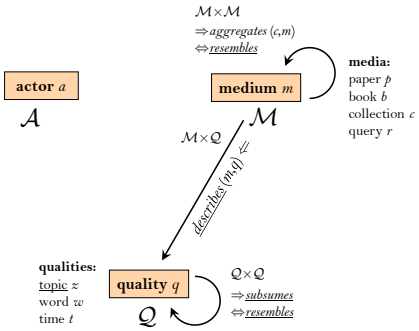
[6] In the approach described in [Heinrich 2004], the probability $p(z|x)$ is determined via the mean of $p(z|x) \propto \sum_d p(z|d)p(d)$ in an LDA model, whereas [Rosen-Zvi et al. 2004] directly estimate $p(z|x)$, cf. Chapter 4.



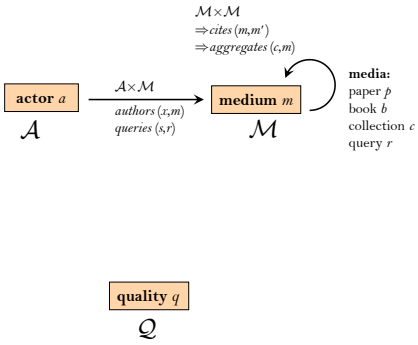
(a) Media retrieval



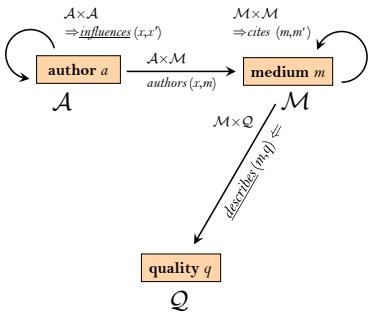
(b) Expert finding



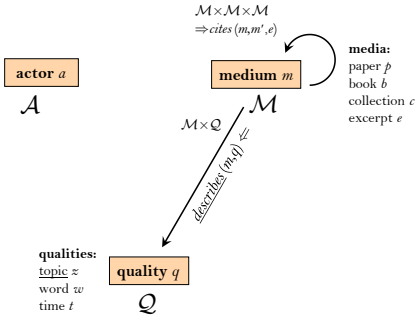
(c) Semantic matching



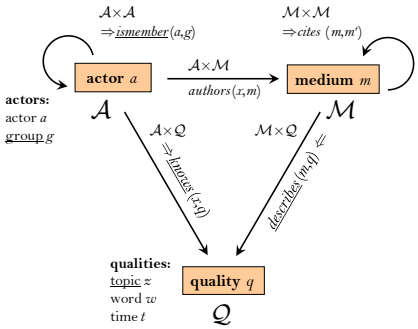
(d) Author co-citation matching



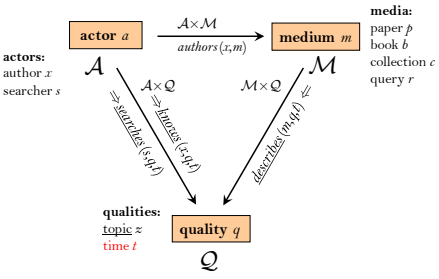
(e) Author influence



(f) Semantic citation influence



(g) Sub-community detection



(h) Dynamic model

Figure 2.3: AMQ schemas for digital library retrieval and knowledge discovery tasks.

In the AMQ schema, the possibility of hierarchical topics and aggregated documents is shown. Furthermore, similar to the analogy between media retrieval and expert finding, semantic matching may be changed to actor matching: Pairs of actors are ranked via actor–quality relations, inferring *resembles* relations via their $knows(x, q)$ relations.

- Co-citation matching: The similarity between the subgraph structures spanned by *cites* relations around two documents is inferred, e.g., defining a distance based on intersection:

  document $m' \Leftarrow$ document $m :: \mathrm{D}\{cites(m', m'') \parallel cites(m, m'')\}$

- Document citation influence: The influence of a document along $cites(m, m')$ relations is inferred, e.g., using graph importance measures like degree centrality [Wasserman & Faust 1994] or PageRank [Brin & Page 1998]:

  $CI(m) \Leftarrow$ document $m :: cites(m', m) \times CI(m')$,

  which reads: citation influence of m is obtained from the document m as a function of the $cites(m', m)$ relation (the incoming links) and the citation influence of these incoming links. The $\times$ symbol is used to denote combination, not necessarily a product.

- Author influence: Document citation influence may be mapped to their authors:

  $CI(x) \Leftarrow$ author $x :: CI(m)\ \forall\{m : \exists\ authors(x, m)\}$

  Author influence may be specific to a particular influenced author x' ($influences(x, x')$).

- Semantic citation influence: Combining the influence of citations with some semantic context e ($describes(e, q)$), possibly the text of the citing paragraph via $cites(m, m', e)$:

  $SCI(m, q) \Leftarrow$ document $m :: cites(m', m, q) \times CI(m')$;
  $cites(m', m, q) :: cites(m', m, e) \times describes(e, q)$

- Sub-community detection: Similar interests ($searches(a, q)$) and/or knowledge ($knows(a, q)$) can be clustered into communities with an entity group $g \in \mathcal{A}$ and $aggregates(g, a)$.

  group $g \Leftarrow$ actor $a :: knows(a, q) \times cites(m', m) \times authors(a, m)$

- Dynamic models: Combining the above models with temporal information, e.g., to analyse the evolution of *describes* or *searches* relations as $describes(m, q, t)$ or $searches(m, q, t)$, putting each relation in time context.

With relevance feedback or preference information included in the data, this list could be extended by collaborative approaches like recommender systems where actors rate media via $recommends(s, m)$ and inference yields a $likes(s, m)$ relation, creating profile clusters similar to the groups g above.

2.5 Related work

Regarding previous approaches to define a generic structural basis of information access that uses the typical data available in virtual communities, existing work turned out to be surprisingly scarce.

For the purpose of information retrieval (IR), different meta-modelling approaches have been proposed that appear relevant. A classical one is the application of the spreading activation principle in cognitive psychology [Collins & Loftus 1975] to information retrieval: From a point of interest in a directed graph of interconnected documents and terms, items are considered relevant that are reached by spreading "activation energy" along the vertices of the graph, using a distance-based decay function [Preece 1981]. Based on this, [Witschel 2007] proposed a graph-based meta model of IR tasks and models under a common representation: feedback (on perceived retrieval result relevance), associative retrieval (finding items similar to relevant ones) and browsing (interactively exploring the document collection). The framework models different types of entity, such as terms and documents, and the interrelation between entities of the same type is modelled by a weighted directed graph on a distinct "level", while entities of different types are connected by weighted bipartite graphs between levels. The resulting "multi-level association graph" (MLAG) can be used to define particular retrieval tasks based on the spreading activation approach. In principle, it is possible to represent the information of AMQ graphs in MLAG form and define particular retrieval algorithms this way. However, the AMQ model is more structured in terms of its application to social communities, and algorithms are not defined on the basis of spreading activation energy but more generically as arbitrary operations on the graph structure. In the words of the MLAG model, this results in changing weights and adding vertices on and between the levels.

From a different perspective, the AMQ approach can be thought of as an extension of work on social networks [Wasserman & Faust 1994] by documents and items of knowledge representation. In particular, social networks analysis may provide important results on the properties of AMQ graphs. In connection to this, the properties of citation graphs are of interest, and regarding actor–media graphs, [Börner et al. 2004] defines a model for a network generation process for graphs of authors and documents along what in the AMQ model is defined as $cites(m, m')$ and *authors(a,m)* relations, showing that the small-world properties of the generated graph resembles that of observations.

Other research is relevant mostly considering the way data is represented. There are close relationships with ontology modelling [Gruber 1994], particularly the Web ontology language (OWL [McGuinness & van Harmelen 2004, Hitzler et al. 2009]): Regarding entities and their types, OWL defines individuals that belong to classes that themselves support subsumption and aggregation as well as other relations called properties to link individuals to each other (object properties) or to data values (datatype properties). The AMQ model takes up the individual and class concepts but is limited to the object properties as the basis for its relations. Because OWL and other ontology languages are focussed on logical reasoning, the concept of weighted relations cannot be modelled in a simple and expressive way but rather requires workarounds like reification (the transformation of relations to actual ontology classes). Because the possibility of weighted relations is at the core of the AMQ model and it does not restrict inference methods to logic reasoning, the AMQ model has been defined as a more generic graph structure.

Moreover, the AMQ model can be considered an application of entity-relationship modelling [Chen 1976] to community-based information retrieval tasks, providing a "template" to designing domain-specific database schemes. In a similar direction, the approach has some relations to meta-modelling [Bezivin 2006] as it can be used to derive models from a template model structure.

Finally, some relationship may be seen to Peirce's theory of signs [Atkin 2010], which posits that a sign is composed of three basic elements: the *sign* itself as an observation, an *object* that is shown on the sign and an *interpretant* that decodes the meaning of the sign in context. In a coarse-grained correspondence, qualities take the role of *objects* and media that of *signs*. For actors, there exists a dichotomy of roles: They are *interpretants* in their active role to use knowledge and have *sign* character when their own knowledge is being interpreted.

2.6 Conclusions

In this chapter, the "AMQ model" was developed, a representation of virtual communities that can be used as the basis for information systems to support retrieval and knowledge discovery in those communities. The model can be considered an attempt to characterise the domain of virtual communities from a data structure viewpoint, considering the most important factors of explicit and tacit knowledge. Yet the model stays conceptually simple and does not claim completeness for all imaginable scenarios in virtual communities. Rather, it attempts to pragmatically characterise the domain of applications that this thesis investigates, namely such that infer similarity, relevance, classifications and other information from relations between people, documents and semantics.

Although the AMQ model is generic in terms of the quality dimensions, this thesis will only use a special case in the future: latent topics. For this approach, the subsequent Chapters 3–9 will develop the necessary methods. We will join them with the AMQ model in Chapter 10.

Chapter 3

Methods: Probabilistic models of semantics

This chapter introduces the general idea of representing text semantics by probabilistic means, focussing on latent semantics as a core methodology of this thesis. After introducing basic semantic and probabilistic concepts, the relation between different language and topic models is discussed, and based on this latent Dirichlet allocation (LDA) is introduced. Subsequently, besides discussing non-parametric extensions of LDA, this chapter treats querying and analysis methods for topics. All subsequent work builds on the methods laid out in this chapter.[1]

3.1 Introduction

When communicating with other people, humans transform mental models into language and at the receiving end, people transform this stream back to their own mental models [Anderson 1976, Johnson-Laird 1983]. These transformations between typically complex cognitive relationships have a serialised stream of words in the middle and are by no means lossless: Between the original idea in the mind of the speaker or writer and that of the listener or reader, lexical noise is added to the transmitted information that leads to ambiguity when no additional contextual information is available. Fig. 3.1 illustrates this special case of Shannon's classical noisy-channel model [MacKay 2003, Shannon 1949].

On the word level, important sources of this linguistic ambiguity are the phenomena of polysemy, i.e., the existence of different meanings for one word or phrase, and synonymy, i.e., the existence of similar or identical meanings for different words, but generally this extends to other relations like antonymy (opposite meaning), hypo-/hyperonymy (specialisation), mero-/holonymy (part-of relation), etc. (cf. [Cruse 1986]). An example is given in Fig. 3.2, where the node with the polysemic word "bar" is connected with synonyms that correspond to some of its different meanings. By having background knowledge and contextual information, this ambiguity can be reduced. For instance, in a text about a hotel, the meaning of the word "bar" most likely refers to a location where drinks are served. Synonymy, on the other hand, causes problems in human-computer interaction because there is a strong tendency that in a community of people

[1] This chapter is based on the papers [Heinrich 2009b] (LDA, estimation), [Heinrich et al. 2005b, Heinrich 2011b] (topic examples) and [Heinrich 2011a] (HDP).

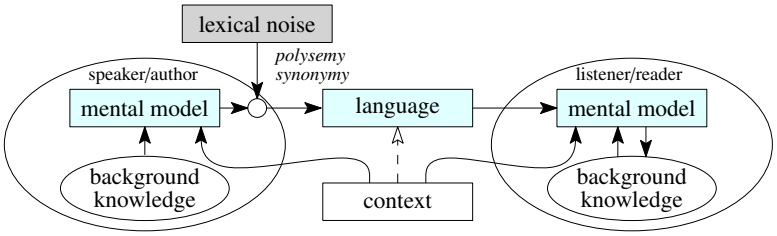


Figure 3.1: Lexical noise in language communication.

with different personal, cultural and educational backgrounds, a variety of synonymous terms or similar formulations is used to describe identical things. This is referred to as the "vocabulary problem" [Furnas et al. 1987], a central problem in information retrieval.

Both of these problems are examples that motivate a non-literal, concept-based semantic approach to language processing as part of retrieval and knowledge discovery solutions, and in this chapter, the idea of "latent semantics" is reviewed.

Chapter outline. In particular, different approaches of semantic representation are reviewed in Section 3.2. Subsequently, the "classical" method of latent semantic analysis is outlined in Section 3.3 as the conceptual basis of probabilistic latent semantic approaches pursued in this thesis. Foundations of these approaches are introduced in Section 3.4 where probability distributions and stochastic independence rules are reviewed, and in Section 3.5 where the relation between different language and topic models is discussed. As a central part of this chapter, generative probabilistic topic models are introduced in Section 3.6, and in Section 3.7 analysis methods are discussed.

3.2 Semantic representation

In Section 3.1, the vocabulary problem has been described as one of the most prominent issues in dealing with language data especially for information retrieval and knowledge discovery. Concept-oriented approaches aim at solving the vocabulary problem by using a non-lexical representation that tries to capture the semantics of the language observed in semantic units referred to as "concepts". By finding a representation of the semantics of a given literal term (or phrase) from its context, the current meaning among multiple possibilities is identified and compared to that of other items. This requires some initial context extraction: The semantic context of a given potentially polysemic term or phrase must be identified and represented. This context is for instance the document a given word appears in. Based on this, semantic disambiguation can be performed: The actual meaning of the term or phrase must be identified and represented given the contextual information. If a document has a certain thematic focus, the semantics of a word contained in it will most likely match this theme. Conversely, it is necessary to take into account synonymous terms for a given meaning of a term to allow non-literal matching.

Concept-based retrieval. In information retrieval, indexing and querying are the most important processing steps, and the associated indexing and querying process for concept-based retrieval is shown in Fig. 3.3. All of the N information items on the right-hand side have some semantic representation that needs to be extracted from the data. In connection to this, we can define a

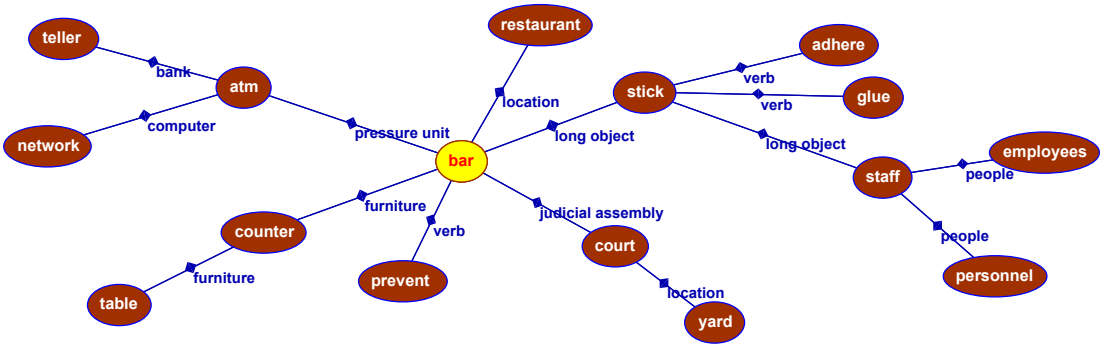


Figure 3.2: Polyseme and synonym relations of the word “bar”.

unit of semantics of a certain object or idea as a set of concepts, and when comparing both a query (information need) and an item in the database, similarity between the associated concepts is measured. In principle, this approach should be superior to standard retrieval techniques that are based on literal matching of words in models like inverted indices or the vector-space model (VSM) [Baeza-Yates & Ribeiro-Neto 1999, Salton & McGill 1983], because they cannot disambiguate different meanings and consequently lead to less precise search results for polysemes and a reduced recall for synonyms.

To solve these tasks and consequently create information systems that are more robust to the natural ambiguity inherent in language communication, different methods exist. The two most prominent directions of research can be termed “formal semantics” and “latent semantics”. Both differ largely in their representation of concepts, expressiveness and computational scalability, and in the next paragraphs, an outline is given.

3.2.1 Formal semantics and ontologies

Formal semantics focuses on an abstract conceptual structure, explicitly defining relations among concepts, and its most prominent approach is the logic-based framework of ontology processing used for the Semantic Web [Berners-Lee et al. 2001], which shall be described here as representative for similar abstract systems of semantic representation (see, e.g., [Sowa 2000]).

Traditionally, ontology is a discipline of philosophy (e.g., [Wittgenstein 1922, Russell 1956]), and from these roots, computer science has derived a framework to represent knowledge domains for automatic reasoning by defining “the basic terms and relations comprising the vocabulary of a topic area, as well as the rules for combining terms and relations to define extensions to the vocabulary” [Neches et al. 1991]. In the seminal work of [Gruber 1994], an ontology is defined as “an explicit specification of conceptualization” where “explicit” marks the formal character of the specification and “conceptualization” refers to the system of concepts, objects and their relations needed to describe the domain. More specifically, an ontology comprises (1) a set of concepts, (2) actual instances of these concepts (individuals, percepts), (3) relationships between the concepts, (4) some functions on mapping between concepts and (5) a set of axioms that constrain concepts, relations and functions and are stated in first-order logic [Tamma 2001]. Depending on the level of expressiveness required, real applications may limit the “implementation” to a subset of these five constituent parts. Consequently, the degree of expressiveness can be described as a continuum from simple glossaries and controlled vocabularies via thesauri to more formal approaches that

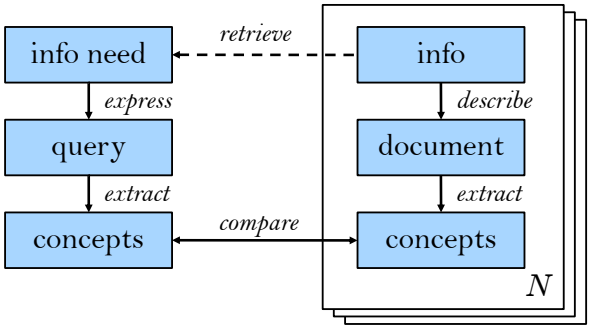


Figure 3.3: Concept-based retrieval.

include formal structures (e.g., WordNet [Fellbaum 1998] and topic maps [Park & Hunting 2002, Smolnik 2006]) to frame-based languages with the five parts listed above [Lassila & McGuinness 2001]. A formalism widely used to represent such high levels of expressiveness is the Web ontology language (OWL [McGuinness & van Harmelen 2004]; current version: OWL 2 [Hitzler et al. 2009]) with associated description logics (DLs), a family of knowledge representation formalisms [Baader et al. 2007] that defines which types of axioms are legal, including cardinality, disjointness, inverse and transitivity. Based on the knowledge represented in this formal way, reasoning can be performed [Dentler et al. 2011, Sirin et al. 2007, Parsia & Sirin 2003, Motik & Sattler 2006], and in practical applications, this is based on subsets of the axiom types allowed in OWL 2, trading off a part of the expressiveness against reasoning efficiency (see, e.g., [Dentler et al. 2011]).

Returning to the example in Fig. 3.2, in formal semantics the concept (class) "bar" may be represented as a subclass (*is-a* relation) of "restaurant" with some property (*has-a* relation) "hasMenu" with cardinality 1 and target class "menu". This class *has-a* number of items ("hasBeverage") with target class "beverage" and cardinality constraint ≥ 1 and "hasDish" with target class "dish" and cardinality ≥ 0, as a bar may choose to not serve food. An "eatery" subclass of "restaurant" may constrain this to ≥ 1, and reasoning may conclude that an individual restaurant (a particular instance of class "restaurant") that has no "dish" on its "menu" is of subclass "bar".

While formal semantics allows complex reasoning over the known instances and their classes, ontologies are difficult to extract from raw data. Due to the high complexity during creation and maintenance of ontologies, they are constructed manually from the consensus between people [Davies et al. 2002, Holsapple & Joshi 2002]. Automatic methods are used as a supplement rather than a substitute, and although considerable effort can be automated in ontology learning or refinement by applying machine learning techniques to unstructured data [Navigli et al. 2011, Wei et al. 2010, Maedche & Staab 2009, Biemann 2005], there is a strong tradeoff between expressiveness of the ontology and complexity of the learning process. Thus, ontology learning mostly covers lower levels of expressiveness [Buitelaar et al. 2005].

Another limitation of typical ontology representations like OWL 2 and their associated reasoning systems is rooted in their focus on first-order logic. This makes representation of uncertain or weighted facts difficult, which sparked research on probabilistic ontologies like the PR-OWL language [Costa & Laskey 2006]. Inference on such extended ontology representations, e.g., building on work like [Laskey 2008] or [Giugno & Lukasiewicz 2002], may provide the flexibility needed for many real-world situations and any uncertain facts they include.

3.2.2 Latent semantics

Latent semantics may be seen as a counterpart to formal semantics in the sense that it does not use formal logic inference to process assertions on data, but uses concepts with "soft associations" to terms and documents. One may consider this a form of implicit conceptualisation, as opposed to the explicit approach that defines ontologies; cf. Section 3.2.1. To represent a domain, the method has a greater emphasis on associative relations between the concepts and terms, possibly including contextual information. Reusing the example from above, the word "bar" may co-occur with "menu" if the word "restaurant" was already observed in the same context more likely than if it was observed with the word "atm" (for the unit of pressure) or "stick" in the same context (see Fig. 3.2). Synonymy can be modelled as weighted relations between different terms and one concept and polysemy between one term and different concepts.

Concepts, in latent semantics commonly referred to as *topics*, thus can capture the information that on the ontology level thesauri do, i.e., on a relatively low level of expressiveness, but may be used to infer or mutually map more expressive ontology structures [Reisinger & Pasca 2009, Spiliopoulos et al. 2007].

Representation of documents (or context in general) is achieved by weighted relationships to latent concepts in the same way that latent concepts are represented via their weighted relationships to terms, as visualised in Fig. 3.4. An advantage is that the unsupervised nature of latent semantics makes the method an appropriate choice in situations where larger amounts of data need to be semantically analysed without human intervention.

More concretely, latent semantics can be defined over the role of its concepts as semantic representation of words and documents by weighted combinations of latent concepts [Deerwester et al. 1990, Berry et al. 1994]. In latent-semantic modelling, a given text is assumed to have a semantic structure and added lexical noise, i.e., variability in word choice. Only the sum of semantic structure and noise can be observed; the semantic structure is hidden and therefore called *latent* semantics of the text (cf. Fig. 3.1).

In this thesis, latent semantics is the major means of semantic representation because of its straight-forward way of concept-based representation of documents and terms as well as other entities (cf. Chapter 2), as well as its ability to extract concepts from these data in an entirely automatic manner. Approaches to realise this extraction will be discussed in the next sections.

3.3 Latent semantic analysis

There exist a number of concept extraction methods that build on the idea of latent structure. They are based on some representation of the vector-space model of text [Salton & McGill 1983], which defines a term-document matrix, $\underline{A}$, with each term and each document represented by a row and a column vector, respectively. Each matrix element contains the weighted occurrence frequency of a term in a document, and the order of words is discarded, which follows the "bag of words" assumption of information retrieval [Baeza-Yates & Ribeiro-Neto 1999]. For this weighting, often models like tf-idf are used, controlling the importance of terms according to their occurrence in context of the complete document and corpus.

We will explain the extraction of latent concepts using the original approach, which coined the terms latent semantic analysis (LSA) and, synonymously in the information retrieval literature, latent semantic indexing (LSI) [Deerwester et al. 1990, Berry et al. 1994, Berry & Browne 2005].

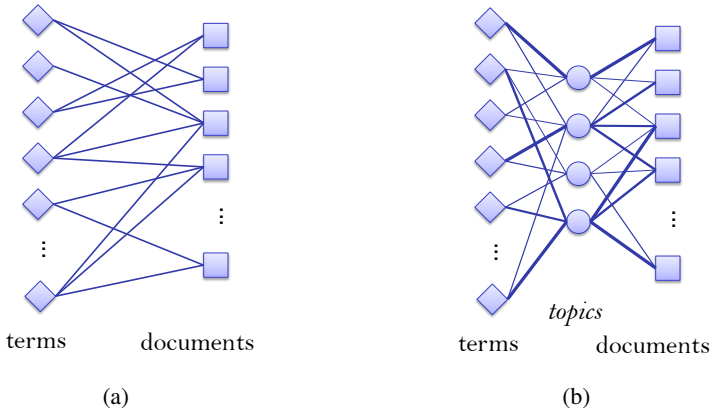


Figure 3.4: Association graphs for text: (a) bipartite term–document association graph, (b) tripartite term–topic–document association graph.

It is based on the singular value decomposition (SVD), which factors an arbitrary matrix $\underline{A}$ into three components: $\underline{A} = \underline{U}\ \underline{S}\ \underline{V}^\top$ where $\underline{U}$ and $\underline{V}$ are matrices whose columns are orthonormal and $\underline{S} = \text{diag}\ \vec{\sigma}$ is a diagonal matrix that contains the singular values.

The idea of LSA is to reduce the rank of the matrix by truncating the diagonal matrix $\underline{S}$ (and therefore the columns of $\underline{U}$ and $\underline{V}$) to k dimensions, $\underline{S}^* = \text{diag}\ (\sigma_1, \sigma_2, \ldots, \sigma_K, 0, 0, \ldots)$, as can be seen in Fig. 3.5:

$$\underline{A} \approx \underline{A}^* = \underline{U}^*\ \underline{S}^*\ \underline{V}^{*\top}\,. \tag{3.1}$$

In the seminal work of [Deerwester et al. 1990], this had been motivated by the guarantee of the SVD to find the best low-rank approximation of any matrix with respect to the L_2 norm, according to the Eckart-Young theorem [Eckart & Young 1936, Horn & Johnson 1985].[2] That is, by using the SVD, a more compact representation of $\underline{A}$ is constructed that retains the overall structure of the data but limits local effects, similar to the usage of SVD as tool for principal components analysis of linear models.

The effect of this truncation is that each word $\vec{t}_i$, originally represented by the i'th row of the original matrix, $\vec{e}_i^{\top}\underline{A}$, is now approximated by a projection on the space spanned by the k orthonormal columns of $\underline{V}^*$,

$$\vec{t}_i \approx (\vec{e}_i^{\top}\underline{U}^*\ \underline{S}^*) \cdot \underline{V}^{*\top} = \vec{t}_i^{\,*} \cdot \underline{V}^{*\top}\,. \tag{3.2}$$

Correspondingly, each document $\vec{d}_j$, originally represented by the j'th column of the original matrix, $\underline{A}\vec{e}_j$, is now approximated by a projection on the space spanned by the orthonormal columns of $\underline{U}^*$,

$$\vec{d}_j \approx \underline{U}^* \cdot (\underline{S}^*\ \underline{V}^{*\top}\ \vec{e}_j) = \underline{U}^* \cdot \vec{d}_j^{\,*}. \tag{3.3}$$

Similarity and querying. The rank-reduced spaces spanned by the orthonormal bases $\underline{U}^*$ and $\underline{V}^*$ correspond to K latent concepts, and similarity (or scoring) calculations can be done within these projections, using the concept vectors for terms, $\vec{t}_i^{\,*}$, and documents, $\vec{d}_j^{\,*}$, just as in the vector-space model. That is, the "natural" similarity metric defined on the rank-reduced spaces is the cosine

[2] For any matrix $\underline{B}$ with rank $\underline{B} = K$, $\|\underline{A} - \underline{B}\|_2 = \min$ if $\underline{B} = \underline{A}^*$.

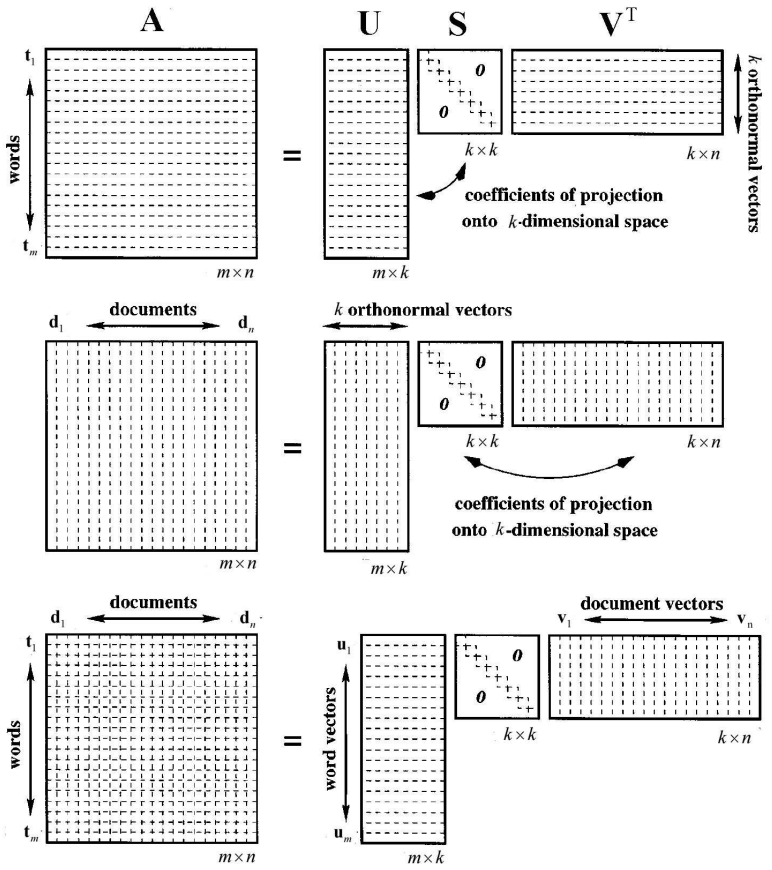


Figure 3.5: Matrices in latent semantic analysis based on the singular value decomposition. From [Bellegarda 2000].

distance [Berry & Browne 2005], $\cos(\vec{x},\vec{y}) = \vec{x}\vec{y}/(|\vec{x}||\vec{y}|)$ for any non-zero vectors $\vec{x}$ and $\vec{y}$.

For querying, a query term vector, $\vec{q}$, is projected into the latent-semantic space, leading to the similarity

$$s_{q,j} = \cos(\vec{q}^*, \vec{d}_j^*) = \cos(\underline{U}^{*\top}\vec{q},\ \vec{d}_j^*)\,, \tag{3.4}$$

which uses the orthogonality property $\underline{U}^{*\top} = \underline{U}^{*-1}$. By considering any term in the query in the context of the complete query text, both the semantic context extraction and disambiguation described above are achieved.

Explanation. The remarkable finding with the truncated SVD approach is that much of the semantic ambiguity in the text can be resolved without any linguistic background information, solely by analysing the weighted occurrence frequencies of terms in the given text corpus [Landauer & Dumais 1997, Griffiths et al. 2007]. An explanation of this effect is that high variability of word choice compared to the lower variability of document semantics seems to coincide with the filtering and transformation effect of SVD also exploited for principal components analysis. As explained in Section 3.1 and especially in Fig. 3.1, when trying to put a mental model into language, the author or speaker needs to choose from a set of possible words,

which are either dictated by context or by style to avoid repetitions.[3] In effect, LSA extracts an averaged pattern of the language used in a given corpus – the latent semantic structure – that words tend to be similar if they co-occur in the same context (document), and documents tend to be similar if they share members of sets of typically co-occurring words [Schütze 1992]. In addition, higher-order co-occurrence seems to play an important role [Kontostathis & Pottenger 2006].

Interestingly, this behaviour could be shown to mimic human cognitive processes. For example, in assessing synonymy, the model performed on par with the average of moderately advanced human learners of English that took the TOEFL test [Landauer & Dumais 1994], although other empirical work discusses this critically, arguing that such results strongly depend on conditions like part-of-speech type and are quantitatively imprecise [Rapp 2003, Wandmacher 2005, Wandmacher et al. 2008]. Generally, however, the connection between human word association and LSA results is consensus in the research community, see, e.g., [Foltz & Dumais 1992, Story 1996, Landauer & Dumais 1997, Landauer et al. 1997, Kintsch 2001, Landauer 2002], and [Griffiths et al. 2007] give an overview of semantic representation from the viewpoint of cognitive psychology.

Alternative approaches to extract and represent the latent semantics of a text according to above definitions include the generalisation of SVD to non-negative matrix factorisation (see, e.g., the semi-discrete decomposition [Kolda & O'Leary 1998]). Another closely related approach, hyperspace analogue to language (HAL) [Burgess et al. 1998] creates word weights by sliding an N-point window over texts and counting the co-occurrence of terms in this window by the number of words they are separated. This relaxes the bag-of-words assumption and replaces the document as a unit of semantic context by the window. A notable extension to LSA is semi-supervised latent semantic analysis [Yu et al. 2005], which permits inclusion of additional information on the texts as input to an extended LSA procedure.

Opposed to these methods rooted in linear algebra, latent semantic extraction has been viewed also as a statistical problem, making use of an extensive toolbox of methods. Among the most important methods are probabilistic latent semantic analysis [Hofmann 2001], admixture modelling [Pritchard et al. 2000], and latent Dirichlet allocation [Blei et al. 2002, Griffiths & Steyvers 2002, Buntine 2002, Griffiths & Steyvers 2004], which today is often simply considered the "topic model". These methods will be described in the next sections, along with an introduction to probabilistic representation of language.

3.4 Probabilistic representation of discrete data

Text as discrete data can be represented probabilistically, which leads to statistical language models. For this, discrete probability distributions over the categories of the data, i.e., the words in the text, are the tool of choice. In this section, first this multinomial distribution and its maximum likelihood estimator are introduced, followed by the Dirichlet distribution and an account of multinomial parameter estimation with a Dirichlet prior. Finally, important concepts to connect different random variables are introduced: Bayesian networks and how they encode independence between variables.

[3] By conjecture, the stylistic rule to avoid lexical repetition in prose text might have its roots in the need to resolve polysemic ambiguity.

3.4.1 Multinomial distribution and the maximum likelihood estimator

One statistical model to represent text data under the bag-of-words assumption is the multinomial distribution, which corresponds to rolling a die with as many faces as there are categories (terms, topics) in the data. For a random observation x, the probability of outcome k in this multinomial experiment is simply:[4]

$$p(x{=}k|\vec{\vartheta}) = \text{Mult}(x{=}k|\vec{\vartheta}) \triangleq \vartheta_k \ , \quad \sum_k \vartheta_k = 1 \ , \ \vartheta_k \in [0,1] \tag{3.5}$$

with some multinomial parameters $\vec{\vartheta} = \{\vartheta_k\}_{k=1}^K$ that describe the fairness of the die.

Considering multiple observations (e.g., a document), $\vec{x} = \{x_i\}_{i=1}^N$, the representation of the vector-space model (VSM) by a sum of basis vectors $\sum_i \vec{e}_{x_i}$ is replaced by the likelihood of the repeated multinomial experiment leading to observation $\vec{x} = \{x_i\}_{i=1}^N$, for which we define the multinomial likelihood $\text{Mult}(\vec{x}|\vec{\vartheta})$:

$$p(\vec{x}|\vec{\vartheta}) = \text{Mult}(\vec{x}|\vec{\vartheta}) \triangleq \prod_{i=1}^N \vartheta_{x_i} \tag{3.6}$$

$$= \prod_{k=1}^K \vartheta_k^{n_k} \ , \quad n_k = \sum_i^N \delta(x_i - k) \ , \tag{3.7}$$

where n_k is the number of times that an $x_i \in \vec{x}$ has taken value k throughout the experiments and $\delta(x)$ is the Kronecker delta defined as $\delta(x) = 1$ iff $x = 0$, otherwise 0.[5]

Parameter estimation. A central task when using the multinomial model is to estimate the parameter $\vec{\vartheta}$ from the data $\vec{x}$. For this, several methods exist that are named after the quantities in Bayes' rule, generically applied to a set of observed data, $\mathcal{X}$, that are the instantiations of some random variable, X, distributed with parameters θ:[6]

$$p(\theta \,|\, \mathcal{X}) = \frac{p(\mathcal{X} \,|\, \theta) \cdot p(\theta)}{p(\mathcal{X})}, \tag{3.8}$$

and we define the corresponding terminology:

$$\text{posterior} = \frac{\text{likelihood} \cdot \text{prior}}{\text{evidence}}. \tag{3.9}$$

[4] This is equivalent to the discrete distribution; referring to "multinomial" follows the established terminology in the topic modelling literature, cf. [Hofmann 1999b, Blei et al. 2002].

[5] Note the distinction from the multinomial distribution of counts $p(\vec{n}|\vec{\vartheta}, N) = \binom{N}{\vec{n}} \prod_{k=1}^K \vartheta_k^{n_k} \triangleq \text{Mult}(\vec{n}|\vec{\vartheta}, N)$, $\binom{N}{\vec{n}} = N!/\prod_k n_k!$, which describes the probability of *any* $\vec{x}$ that leads to the configuration of counts $\vec{n} = \{n_k\}_k$. While this generalises the binomial distribution, the distribution $\text{Mult}(\vec{x}|\vec{\vartheta})$ is defined to generalise repeated Bernoulli trials, i.e., throwing a coin. The additional factor $\binom{N}{\vec{n}}$, the multinomial coefficient, determines the number of possible configurations $\vec{x}$ that can lead to $\vec{n}$, which generalises binomial coefficients from $K = 2$ to arbitrary K.

[6] Note the difference between multinomial parameters $\vec{\vartheta}$ and the generic set of parameters θ.

Maximum likelihood (ML) estimation finds the parameters θ that maximise the likelihood,

$$L(\theta \,|\, \mathcal{X}) \triangleq p(\mathcal{X} \,|\, \theta) = \bigcap_{x \in \mathcal{X}} \{X = x \,|\, \theta\} = \prod_{x \in \mathcal{X}} p(x \,|\, \theta)\,, \tag{3.10}$$

i.e., the probability of the joint event that X generates the data $\mathcal{X}$ given that draws are independent and identically distributed (i.i.d.). Because of the product in (3.10), it is often simpler to use the log likelihood, $\mathcal{L} \triangleq \log L$. The ML estimation problem then can be written as:

$$\hat{\theta}^{\mathrm{ML}} = \operatorname*{argmax}_{\theta} \mathcal{L}(\theta \,|\, \mathcal{X}) = \operatorname*{argmax}_{\theta} \sum_{x \in \mathcal{X}} \log p(x \,|\, \theta)\,. \tag{3.11}$$

The common way to obtain the parameter estimates is to solve the system:

$$\frac{\partial \mathcal{L}(\theta \,|\, \mathcal{X})}{\partial \theta_k} \stackrel{!}{=} 0 \quad \forall \theta_k \in \theta\,, \tag{3.12}$$

and for multinomial/discrete observations with $\mathcal{X} \equiv \vec{x}$ and $\theta \equiv \vec{\vartheta}$ from (3.7), the ML estimator is obtained by using (3.12) and adding Lagrange multipliers, $\lambda(1 - \sum_k \vartheta_k)$, to accommodate the probability constraint of the multinomial, $\sum_k \vartheta_k = 1$:

$$\frac{\partial \mathcal{L}(\vec{\vartheta} \,|\, \vec{x})}{\partial \vartheta_k} = \frac{\partial}{\partial \vartheta_k} \sum_{k=1}^{K} n_k \log \vartheta_k + \lambda\Big(1 - \sum_{k=1}^{K} \vartheta_k\Big) = \frac{n_k}{\vartheta_k} - \lambda \stackrel{!}{=} 0 \tag{3.13}$$

$$\vartheta_k = \frac{n_k}{\lambda}, \quad \sum_{k}^{K} \frac{n_k}{\lambda} = 1 \quad \Leftrightarrow \quad \vartheta_k^{\mathrm{ML}} = \frac{n_k}{N}\,, \tag{3.14}$$

which is simply the ratio of occurrences of a particular value k to the total number of samples. To put some numbers into this, imagine a die rolled 10 times, resulting in the observations $\vec{x} = \{1, 4, 3, 2, 2, 2, 5, 3, 1, 5\}$, which would yield $\vec{\vartheta}^{\mathrm{ML}} = \{.2, .3, .2, .1, .2, 0\}$. With the small number of observations, the die estimated with maximum likelihood clearly feels deformed. As we know that a die is usually not as bad as that, it is useful to encode this prior knowledge into an estimator.

3.4.2 Dirichlet distribution, MAP estimator and posterior inference

Extensions of the maximum likelihood estimator that allow encoding of prior knowledge in the estimation are the maximum a posterior (MAP) estimator and posterior inference.

Conjugate priors. For both estimation methods, an appropriate prior distribution is necessary that allows to add knowledge to the estimator in the form of parameters. For multinomial parameters $\vec{\vartheta}$, in principle any distribution over values on the simplex $\sum_k \vartheta_k = 1$ can be used, and the choice should be guided by both modelling and computation considerations.

For the multinomial as an exponential-family distribution [Wainwright & Jordan 2003, Beal 2003], a numerically appealing choice is the conjugate-exponential ("conjugate") prior.

Conjugate priors on parameters of a likelihood have the interesting property that the product of prior and likelihood (as shown on the numerator of (3.8)) maintains the algebraic form of the prior and simply re-parametrises it with the data encoded in the likelihood. This makes conjugate priors algebraically particularly convenient. In this thesis, most of the inference is

based on conjugate priors, and more details on the theory of this and exponential families is given in Appendix B, including a derivation of the relationship between the multinomial and Dirichlet distributions.

Dirichlet distribution. For the parameters $\vec{\vartheta}$ of the multinomial distribution, the conjugate prior is the Dirichlet distribution:

$$p(\vec{\vartheta}|\vec{\alpha}) = \mathrm{Dir}(\vec{\vartheta}|\vec{\alpha}) \triangleq \frac{\Gamma(\sum_{k=1}^{K} \alpha_k)}{\prod_{k=1}^{K} \Gamma(\alpha_k)} \prod_{k=1}^{K} \vartheta_k^{\alpha_k - 1} \tag{3.15}$$

$$= \frac{1}{\Delta(\vec{\alpha})} \prod_{k=1}^{K} \vartheta_k^{\alpha_k - 1}, \quad \Delta(\vec{\alpha}) \triangleq \frac{\prod_{k=1}^{K} \Gamma(\alpha_k)}{\Gamma(\sum_{k=1}^{K} \alpha_k)}, \tag{3.16}$$

with parameters $\vec{\alpha}$ and the "Dirichlet delta function" $\Delta(\vec{\alpha})$, which we introduce for notational convenience. The function $\Delta(\vec{\alpha})$ can be seen as a K-dimensional generalisation to the beta function [Andrews et al. 1999]: $\mathrm{B}(\alpha_1, \alpha_2) \equiv \Delta(\{\alpha_1, \alpha_2\})$.[7] In an MAP or Bayesian setting, the parameters of the prior distribution are referred to as hyperparameters because the parametrise parameters.

In Fig. 3.6, the samples of a Dirichlet distribution for $K = 3$ are presented. This illustrates that the space in which multinomial parameters $\vec{\vartheta}$ are located is a simplex of $K - 1$ dimensions: $\mathbb{S}_{K-1} = \{\vec{\vartheta} : \sum_k \vartheta_k = 1, \vartheta_k \in [0, 1]\}$. This is one of the main differences to the vector-space model, where the respective space is $\mathbb{R}^K$.

In applications where differences between dimensions are neglected or no prior knowledge is available about particular dimensions, a symmetric Dirichlet distribution is used, which is defined in terms of a scalar parameter $\alpha = \sum \alpha_k / K$ and the dimension K:

$$p(\vec{\vartheta}|\alpha, K) = \mathrm{Dir}(\vec{\vartheta}|\alpha, K) \triangleq \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \prod_{k=1}^{K} \vartheta_k^{\alpha - 1} \tag{3.17}$$

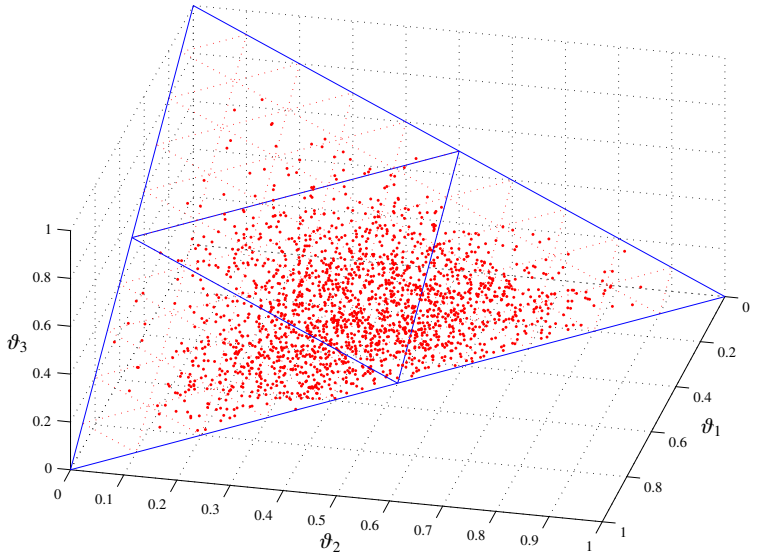
$$= \frac{1}{\Delta_K(\alpha)} \prod_{k=1}^{K} \vartheta_k^{\alpha - 1}, \quad \Delta_K(\alpha) \triangleq \frac{\Gamma(\alpha)^K}{\Gamma(K\alpha)}. \tag{3.18}$$

From a different viewpoint, various parametrisations of the Dirichlet are visualised for the two-dimensional case in Fig. 3.7, for which the Dirichlet distribution becomes identical to the beta distribution.

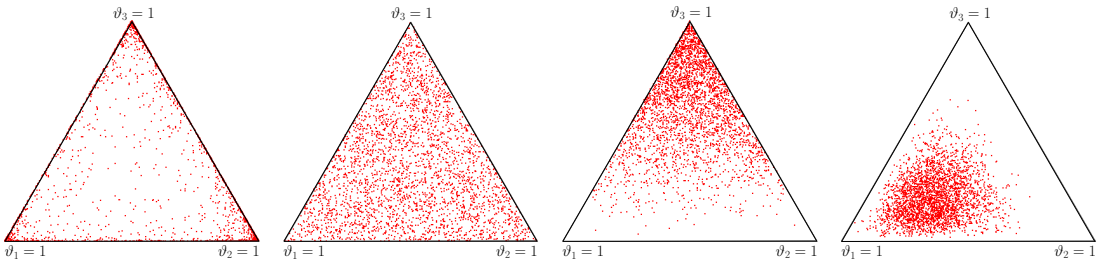
Maximum a posteriori (MAP) estimation directly extends maximum likelihood by incorporating prior belief on the parameters θ. The name MAP derives from the objective to maximise the posterior of the parameters given the data, $\mathcal{X}$:

$$\hat{\theta}^{\mathrm{MAP}} = \operatorname*{argmax}_{\theta} p(\theta \,|\, \mathcal{X}). \tag{3.19}$$

[7] The Dirichlet distribution generalises the beta distribution from 2 to K dimensions. Moreover, it can be shown that $\Delta(\vec{\alpha})$ is the Dirichlet integral of the first kind for the summation function $f(\Sigma x_i)$=1: $\Delta(\vec{\alpha}) = \int_{\Sigma x_i = 1} \prod_i^N x_i^{\alpha_i - 1} \,\mathrm{d}^N \vec{x}$, analogous to the beta integral: $\mathrm{B}(\alpha_1, \alpha_2) = \int_0^1 x^{\alpha_1 - 1}(1 - x)^{\alpha_2 - 1} \,\mathrm{d}x$.



(a) Dir(4, 4, 2)



(b) Dir(0.1, 0.1, 0.2) (c) Dir(1, 1, 1) (d) Dir(1, 1, 3) (e) Dir(8, 4, 3)

Figure 3.6: 2000 samples from Dirichlet distributions. Plot (a) shows the simplex embedded in the space $\mathbb{R}^3$, corresponding to the constraint $\sum_k \vartheta_k = 1$ for all samples $\vec{\vartheta}$.

By using Bayes' rule (3.8), this can be rewritten to:

$$\begin{aligned}
\hat{\theta}^{\text{MAP}} &= \operatorname*{argmax}_{\vartheta} \frac{p(\mathcal{X} \mid \theta) p(\theta)}{p(\mathcal{X})} \qquad \Big| \; p(\mathcal{X}) \neq f(\theta) \\
&= \operatorname*{argmax}_{\theta} p(\mathcal{X} \mid \theta) p(\theta) = \operatorname*{argmax}_{\theta} \{\mathcal{L}(\theta \mid \mathcal{X}) + \log p(\theta)\} \\
&= \operatorname*{argmax}_{\theta} \Big\{ \sum_{x \in \mathcal{X}} \log p(x \mid \theta) \; + \; \log p(\theta) \Big\}. \qquad (3.20)
\end{aligned}$$

Compared to (3.11), a prior distribution is added to the likelihood that weights the parameters by specifying how likely it is that they have a particular value. In practice, the prior $p(\theta)$ can be used to encode extra knowledge as well as to prevent overfitting by enforcing preference to simpler models, which is often referred to as Occam's razor.[8]

[8] Pluralitas non est ponenda sine necessitate = Plurality should not be posited without necessity. Occam's razor is also called the principle of parsimony.

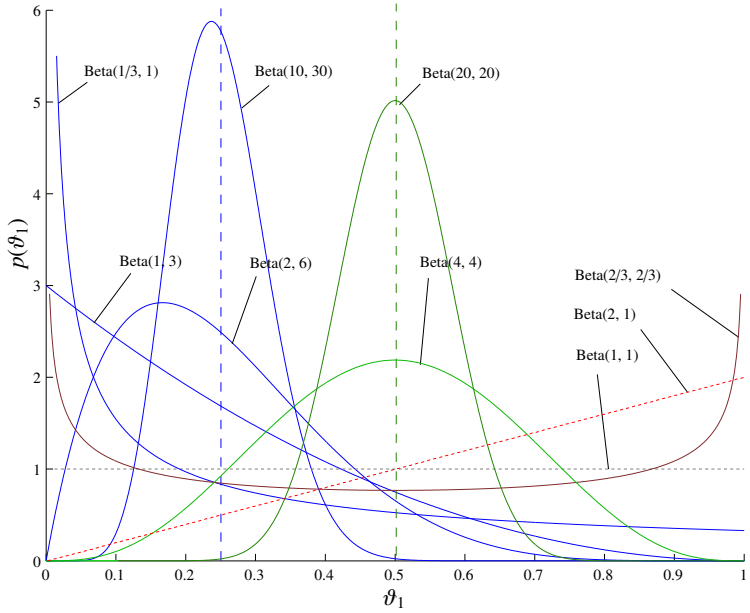


Figure 3.7: Density functions of the beta distribution (Dirichlet with $K = 2$) with different symmetric and asymmetric parameters.

The MAP estimator for the multinomial parameter $\vec{\vartheta}$ under a Dirichlet prior distribution $\mathrm{Dir}(\vec{\vartheta}|\vec{\alpha})$ and observations $\vec{x}$ then is obtained analogous to (3.14):

$$\frac{\partial}{\partial \vartheta_k}\mathcal{L}(\vec{x}\,|\,\vec{\vartheta}) + \log p(\vec{\vartheta}|\vec{\alpha}) = \frac{\partial}{\partial \vartheta_k}\sum_{k=1}^{K} n_k \log \vartheta_k + (\alpha_k - 1)\log \vartheta_k \\ - \log \Delta(\vec{\alpha}) + \lambda\Big(1 - \sum_{k=1}^{K} \vartheta_k\Big) \stackrel{!}{=} 0 \tag{3.21}$$

$$\frac{n_k}{\vartheta_k} + \frac{\alpha_k - 1}{\vartheta_k} = \lambda\,, \qquad \sum_{k=1}^{K} \frac{n_k + \alpha_k - 1}{\lambda} = 1 \tag{3.22}$$

$$\Leftrightarrow \quad \vartheta_k^{\mathrm{MAP}} = \frac{n_k + \alpha_k - 1}{N + \sum_k (\alpha_k - 1)}\,. \tag{3.23}$$

The reason why this derivation was straight-forward is the conjugate relationship between the multinomial and Dirichlet distributions.

Applying this to the observations above, we add a symmetric Dirichlet prior Dir(4) (cf. Fig. 3.7 for a graph of the two-dimensional counterpart) and obtain the MAP estimate $\vec{\vartheta}^{\mathrm{MAP}} = \{2+3, 3+3, 2+3, 1+3, 2+3, 0+3\}/(10+18) = \{.179, .214, .179, .143, .179, .107\}$. The prior belief in a fair die thus resulted in an estimate with more evenly distributed weights ϑ_k.

Posterior inference (or Bayesian inference) is used to obtain the actual *distribution* of the parameters θ given observations $\mathcal{X}$ and a prior belief on their distribution, $p(\theta)$, which may be also interpreted as how the belief changed when the data were observed. The posterior is obtained by applying Bayes' rule (3.8) to the prior and likelihood distributions.

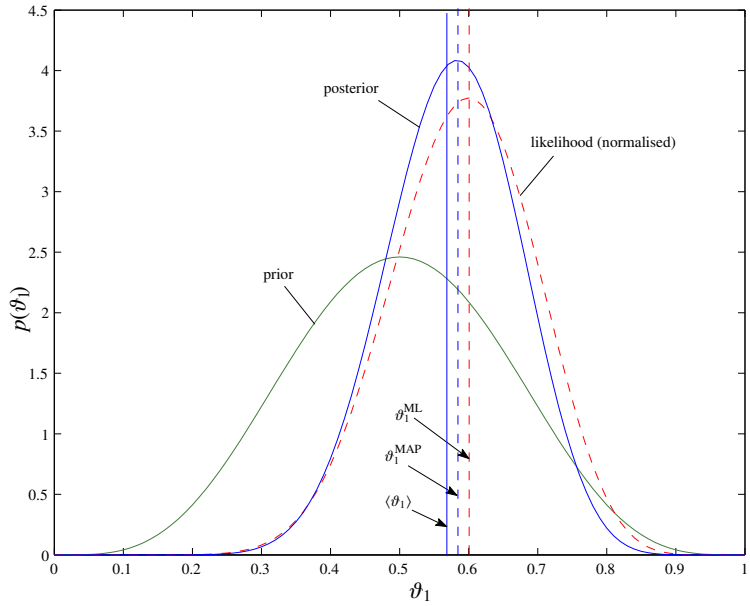


Figure 3.8: ML, MAP and posterior mean estimates of a coin experiment resulting in 12 heads and 8 tails with prior Dir(5, 5).

As we do not restrict the calculation to finding a maximum, it is necessary to calculate the normalisation term, i.e., the probability of the "evidence", $p(\mathcal{X})$, in (3.8). Its value can be expressed by the total probability w.r.t. the parameters:[9]

$$p(\mathcal{X}) = \int_{\theta\in\Theta} p(\mathcal{X}|\theta)\, p(\theta)\, \mathrm{d}\theta. \tag{3.24}$$

As new data are observed, the posterior in (3.8) automatically adapts by being added new factors (summands) into the (log) likelihood term. However, often the normalisation integral in (3.24) is the intricate part of Bayesian estimation.

Fortunately, by choosing the conjugate prior for the multinomial, similar to the MAP case it turns out that this is not a problem:

$$p(\vec{\vartheta}|\vec{x},\vec{\alpha}) = \frac{p(\vec{x}|\vec{\vartheta})p(\vec{\vartheta}|\vec{\alpha})}{p(\vec{x}|\vec{\alpha})}\,, \quad p(\vec{x}|\vec{\alpha}) = \int p(\vec{x}|\vec{\vartheta})p(\vec{\vartheta}|\vec{\alpha})\,\mathrm{d}\vec{\vartheta} \tag{3.25}$$

$$= \frac{1}{p(\vec{x}|\vec{\alpha})}\prod_{k=1}^{K}\vartheta_k^{n_k}\cdot\frac{1}{\Delta(\vec{\alpha})}\prod_{k=1}^{K}\vartheta_k^{\alpha_k-1} = \frac{1}{\Delta(\vec{n}+\vec{\alpha})}\prod_{k=1}^{K}\vartheta^{n_k+\alpha_k-1} \tag{3.26}$$

$$= \mathrm{Dir}(\vec{\vartheta}|\vec{n}+\vec{\alpha}) \tag{3.27}$$

$$\langle\vartheta_k|\vec{x},\vec{\alpha}\,\rangle = \frac{n_k+\alpha_k}{N+\sum_k\alpha_k} \tag{3.28}$$

where $\langle u\rangle$ denotes the expectation of random variable u. The quantity (3.28) is also referred to

[9] This marginalisation is why evidence is also referred to as "marginal likelihood". The integral is used here as a generalisation for continuous and discrete sample spaces, where the latter require sums.

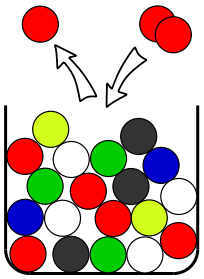


Figure 3.9: Pólya urn sampling scheme.

as posterior mean, which is the point of 50% area of the posterior whose maximum (or mode) is described by the MAP estimate (3.23). If the sums of the counts and pseudo-counts become larger, both expectation and maximum converge.

In addition to this, all other properties of the posterior distribution may be analysed to obtain information about the variance of the estimate as an indicator of its confidence, which is notably not possible with MAP and ML estimation. The variance of the Dirichlet distribution is:

$$\mathrm{Var}\{\vartheta_k \,|\, \vec{x}, \vec{\alpha}\} = \frac{\langle\vartheta_k\rangle(1 - \langle\vartheta_k\rangle)}{1 + \sum_k \alpha_k} \,. \tag{3.29}$$

Returning to the die example, we obtain the estimate of the posterior mean of $\left\langle\vec{\vartheta}\right\rangle = \{2+4, 3+4, 2+4, 1+4, 2+4, 0+4\}/(10+24) = \{.176, .206, .176, .147, .176, .118\}$ and the variance of the different components is $\{\mathrm{Var}\{\vartheta_k\}\} = \{.176\cdot.824, .206\cdot.794, .176\cdot.824, .147\cdot.853, .176\cdot.824, .118\cdot.882\}/25 = \{.076, .081, .076, .071, .076, .065\}^2$, which are given as square roots to allow easier comparison with the parameter values. Notably, the variance of the different parameters ϑ_k is dependent also on the number of times that a particular value is observed.

Estimator comparison. To visualise the "mechanism" of the different estimators, a two-dimensional example of a multinomial experiment is most illustrative, replacing the die experiment (with $K = 6$) by tossing a coin (with $K = 2$, i.e., a Bernoulli experiment). From the observation of the coin experiment of 20 tosses and results of 12 head and 8 tails, we obtain $\vec{n} = \{12, 8\}$. The parameters of the coin are estimated using the different variants introduced above. For simplicity, we define $p = \vartheta_{\text{heads}} = 1 - \vartheta_{\text{tails}}$. ML results in $p^{\text{ML}} = 0.6$, while introducing a Dirichlet prior $\mathrm{Dir}(5, 5)$ results in an MAP estimate of $p^{\text{MAP}} = 0.571$. Finally, the posterior mean becomes $\langle p\rangle = 0.567$, with variance $\mathrm{Var}\{p\} = 0.567(1 - 0.567)/11 = 0.0223 = 0.149^2$. The posterior as the full Bayesian estimate, however, is a distribution and carries information about the confidence of the estimate directly by the variance of the distribution, in this case $p(p) = \mathrm{Dir}(p \,|\, 12+5, 8+5)$. In Fig. 3.8 the three estimates are graphically compared.

Clustering property. An important property of both the MAP estimator (3.23) and posterior mean (3.28) is that they quantify a multinomial distribution whose parameters ϑ_k change their weights according to the previously observed categories k. This shows that both the MAP and Bayesian estimators (3.23) and (3.28) exhibit a clustering behaviour: The information expressed by the hyperparameters $\vec{\alpha}$ is changed by the observed category counts $\vec{n}$. The hyperparameters $\vec{\alpha}$ are therefore often called "pseudo-counts". The effect of increasing likelihood for observed data

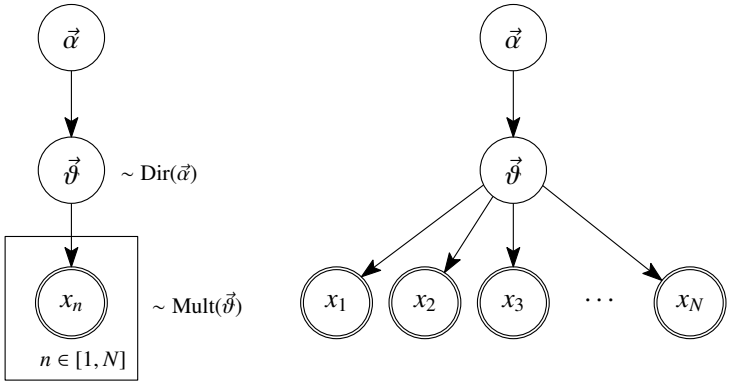


Figure 3.10: Bayesian network of a multinomial with a Dirichlet prior as in (3.28), left: plate notation, right: equivalent without plates.

values is associated with a sampling scheme called the Pólya urn: From an urn with a set of balls each having one of K colours, one ball is drawn and returned into the urn along with a second one of the same colour. This process, as shown in Fig. 3.9, is also referred to as "sampling with over-replacement" [Fisz 1963].[10]

The Dirichlet and multinomial distributions and this clustering property will be the basis for a great part of the models and inference of discrete data studied in this thesis, starting with the probabilistic approaches to latent semantics in the next sections.

3.4.3 Bayesian networks

Bayesian networks (BNs [Pearl 1985]) are a means to describe the dependencies between random variables. They can be understood as a formal graphical language to express the joint distribution of a system or phenomenon in terms of random variables and their conditional dependencies in a directed graph. BNs are a special case of graphical models, an important methodology in machine learning [Murphy 2001] that includes also undirected graphical models (Markov random fields) and mixed models [Russell & Norvig 2000]. By only considering the most relevant dependency relations, inference considerations are largely simplified – compared to assuming dependency between all variables, which is exponentially complex w.r.t. their number.

All elements of the graphical language can be seen in the Dirichlet–multinomial model shown in Section 3.4.2 whose corresponding BN is shown in Fig. 3.10. A BN forms a directed acyclic graph (DAG) with nodes that correspond to random variables and edges that correspond to conditional probability distributions, where the condition variable at the origin of an edge is called a parent node and the dependent variable at the end of the edge a child node. Bayesian networks distinguish between evidence nodes, which correspond to variables that are observed or assumed observed, and hidden nodes, which correspond to latent variables.

[10] When the ball counts are proportional to the Dirichlet parameter $\vec{\alpha}$ initially, the proportions of balls are Dirichlet-distributed after infinitely many samples.

In many models, replications of nodes exist that share parents and/or children, e.g., to account for multiple values or mixture components. Such replications can be denoted by plates, which surround the subset of nodes and have a replication count or a set declaration of the index variable at the lower right corner.

The double circle around the variable $\vec{x}=\{x_n\}$ in Fig. 3.10 denotes an evidence node, and the surrounding plate indicates the N independent identically distributed (i.i.d.) samples. The nodes $\vec{\vartheta}$ and $\vec{\alpha}$ correspond to hidden variables.

3.4.4 Conditional independence and exchangeability

The dependencies between random variables encoded in Bayesian networks can be determined from the topology of the graph. Within this topology, the relevant independence property is *conditional* independence: Two variables X and Y are conditionally independent given a condition Z, symbolically $X \perp\!\!\!\perp Y|Z$, if $p(X, Y|Z) = p(X|Z) \cdot p(Y|Z)$. A verbal explanation of conditional independence is that knowing Z, any information about the variable X does not add to the information about Y and vice versa. Here information refers to either observations or parameters.

Markov conditions. In a Bayesian network, there are two general rules for the conditional independence of a node. The first is based on the *Markov blanket*: a subgraph of the BN defined as the set of a node's parents, its children, and its children's parents (co-parents). The condition states that a node, X_i, is conditionally independent of all other nodes, $X_{\neg i}$, given its Markov blanket, $B(X_i)$: $X_i \perp\!\!\!\perp X_{\neg i}|B(X_i)$.

The second rule refers to the set of *non-descendants* of a node: In a sequence of all BN nodes that ensures no node appears before any of its parents (*topological ordering*), all predecessors of a node that are not its parents are its non-descendants. The rule states that a node, X_i, is always conditionally independent of its non-descendants, $N(X_i)$, given its parents, $P(X_i)$: $X_i \perp\!\!\!\perp N(X_i)|P(X_i)$.

Bayes ball and d-separation. To determine conditional independence between any nodes $X \perp\!\!\!\perp Y|Z$ in a BN, a straight-forward method is called "Bayes ball", which attempts to propagate a message (the "Bayes ball") from X to Y, given observations for node Z [Shachter 1988, Murphy 2001]: $X \perp\!\!\!\perp Y|Z$ is true if and only if (iff) there is no way to pass the ball from X to Y, with the rules given in Fig. 3.11: Given a Bayesian network to check for conditional independence, one identifies the patterns of the figure in that network, i.e., child, parent and transitional structures with observed or hidden nodes, and according to the blocking or passing rules tries to move between all nodes of the test network. Whenever U-turns block all paths between two nodes, then they are d-separated by the node(s) in the blocked path(s).

Summarised, the rules of Bayes ball state that child nodes block propagation iff they are hidden while parent and transitional nodes block propagation iff they are given or observed. For example, observations $\vec{x}$ and hyperparameters $\vec{\alpha}$ in Fig. 3.10 are conditionally independent given the parameters $\vec{\vartheta}$ (transitional node). The method also applies to sets of nodes $\{X_i\} \perp\!\!\!\perp \{Y_j\}|\{Z_k\}$, and conditional independence holds if all pairs (X_i, Y_j) are d-separated given the set of nodes $\{Z_k\}$, i.e., no Bayes ball path exists.

Exchangeability. An independence relation stronger than conditional independence and important in Bayesian statistics is that of exchangeability. Any finite sequence of random variables $\{X_n\}_{n=1}^{N}$ is referred to as exchangeable iff its joint distribution is invariant to any permutation

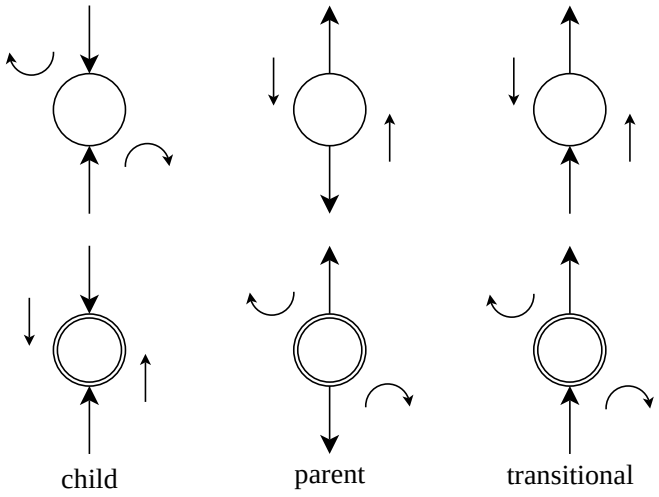


Figure 3.11: Rules for the Bayes Ball method (after [Murphy 2001]).

Perm(n) of its order: $p(\{X_n\}_{n=1}^N) = p(\{X_{\text{Perm}(n)}\}_{n=1}^N)$. For an infinite sequence, this is required of any finite subsequence, leading to infinite exchangeability.

The importance of exchangeability is due to de Finetti's theorem[11], which states that the joint distribution of an infinitely exchangeable sequence of random variables is equivalent to sampling a random parameter θ from some prior distribution and subsequently sampling i.i.d. random variables conditioned on that random parameter [Blei et al. 2003b]. The joint distribution then is $p(\{x_m\}_{m=1}^M) = \prod_{m=1}^M p(x_m|\theta)$.

In the Bayesian network graphical language, exchangeability given a parent variable is the condition to apply the plates notation, and variables can be assumed drawn i.i.d. given the parent. In Bayesian text modelling, exchangeability corresponds to the bag-of-words assumption.

3.5 Multinomial language models and mixtures

Based on the multinomial distribution, different models of language can be distinguished that have increasing expressiveness, and to locate topic models within such a systematics of language models, a brief review seems worthwhile. A statistical language model generally is a method to assign a likelihood to a given observation of words, and depending on the application, a large variety of models is possible [Manning et al. 2008]. One of the foremost usages of language models is to estimate relevance in retrieval tasks by probability distributions of a query given a document, $p(q|d)$, or vice versa, $p(d|q)$, encoding knowledge about document semantics into these distributions.

Topic models may be interpreted as a special type of language model that is based on mixture models and typically a bag-of-words assumption. The difference between these models can be explained best by the way they mix terms to produce ("generate") language as word streams.

[11] De Finetti considered binary variables, Hewitt and Savage [Hewitt & Savage 1955] generalised this to arbitrary r.v.s $X_i \in \mathcal{X}$ relevant here.

Three language models are presented in Fig. 3.12 by their Bayesian networks and associated example word distributions.

3.5.1 Multinomial model

In the simplest language model, the corpus is represented by a single multinomial over words (unigrams), here denoted z, and no local (document-specific) properties are modelled [Manning et al. 2008]. Fig. 3.12(a) and the subfigures below illustrates this. This model is useful to describe a language generally but due to missing inclusion of context it has limited expressive power. Describing the corpus as a set of word documents, $\vec{w} = \{\vec{w}_m\}_{m=1}^{M} = \{\{w_{m,n}\}_{n=1}^{N_m}\}_{m=1}^{M}$, the likelihood under a multinomial (or "unigram") model is:

$$p(\vec{w}) = \prod_m \prod_n p(w_{m,n}{=}t|z) \; . \tag{3.30}$$

Here the label z denotes a class for the data and $w_{m,n}{=}t$ denotes a term t that is observed as nth word in document m. The unigram model posits that the data are drawn i.i.d. from a single class $z = \text{const}$. In terms of a multinomial distribution, $p(w_{m,n}|z)$ is simply a multinomial parameter $\vec{\vartheta} = \{p(t|z)\}_{t=1}^{V}$ with V the size of the vocabulary. Note here the distinction between a term, t, which is part of a vocabulary, while a word, $w_{m,n}$, is part of a document observation and may instantiate a term, $w_{m,n}{=}t$.

3.5.2 Mixture of multinomials

Extending the plain multinomial model to include context information leads to a model where each document is assumed to derive from one of a set of word distributions with labels $\{z_m\}$, and multiple contexts/documents can have the same distribution $p(w|z)$. This corresponds to the cluster model in information retrieval [Manning & Schütze 1999] and is illustrated in Fig. 3.12(b) and subfigures below. That is, the corpus can be assumed to be a mixture of multinomial word (or "mixture of unigram" [Blei et al. 2003b]) distributions, and the likelihood of the corpus becomes:

$$p(\vec{w}) = \prod_m \prod_n \sum_k p(w_{m,n}{=}t|z_m{=}k)p(z_m{=}k) \; . \tag{3.31}$$

Naïve Bayes. In a supervised setting, i.e., finding the most likely word distribution for a given document, the mixture of unigrams corresponds to the Naïve Bayes (NB) classifier, and one multinomial is responsible for one class label [Nigam et al. 2000, Nigam 2001]. The idea of NB is to exploit the naïve assumption that all words $w_{m,n}$ of a document m are independent given its class $z_m{=}c$ and obtain straight-forward classification probabilities for an observed document: The class probability $p(z_m{=}c|\{w_{m,n}\}_n)$ follows from Bayes' rule, $p(z_m{=}c|\{w_{m,n}\}) \propto p(c)p(\{w_{m,n}\}_n|c) = p(c)\prod_n p(w_{m,n}|c)$ because of this naïve independence assumption. The required probabilities are easy to obtain from a labelled training set.[12]

[12] For instance, the weights in the multinomial $p(c)$ can be set proportional to the total number of words of all documents with label c, those in $p(w|c)$ proportional to the number of times word w is observed in any document labelled c. This maximum likelihood estimator is often extended to MAP.

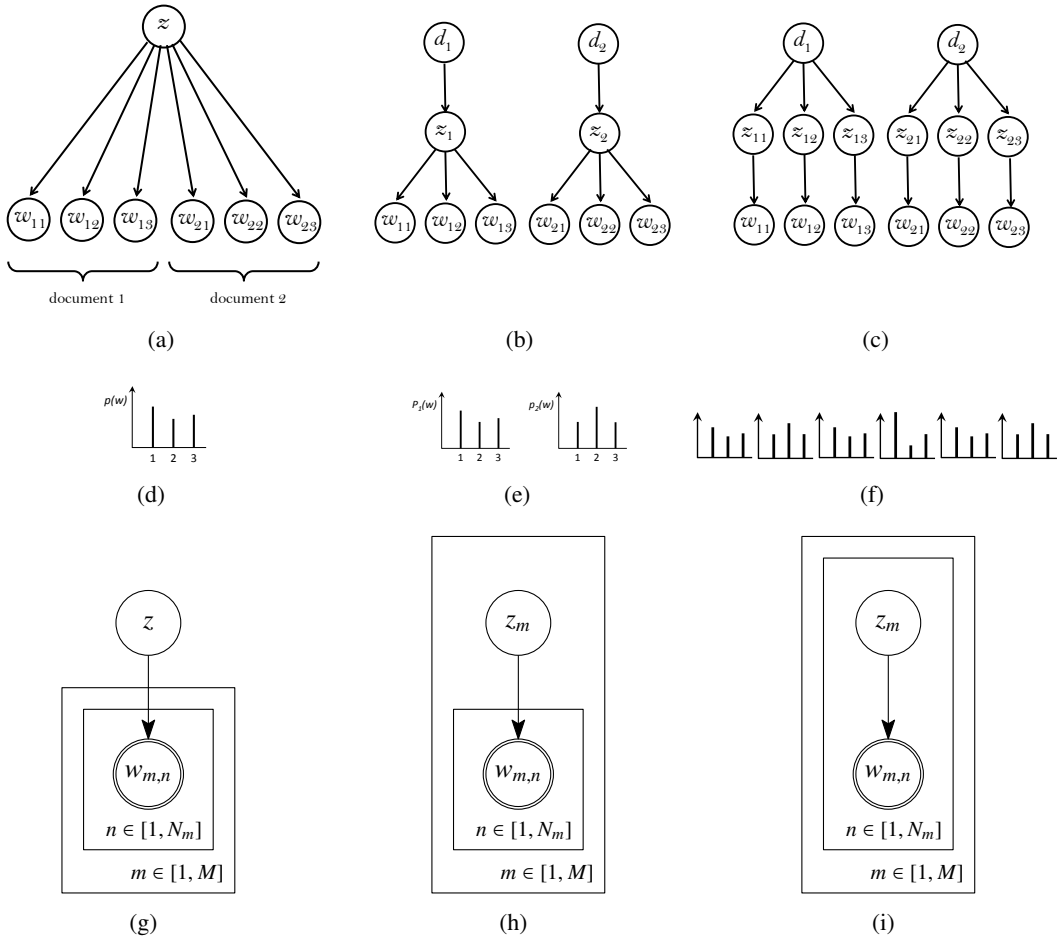


Figure 3.12: Language models: (a) unigram model, (b) mixture of unigrams model / Naïve Bayes, (c) admixture model / topic model, (d–f) corresponding example word distributions, (g–i) corresponding Bayesian networks using plate notation.

3.5.3 Multinomial admixture

Admixture is a term originally used in genetics that describes the effect of mixing between individuals from different populations, i.e., a mixture whose components are themselves mixtures of different features. Following [Pritchard et al. 2000] who proposed a Bayesian model for population genetics, we use this term to describe mixtures of mixtures. In the multinomial admixture model, a document in the corpus is assumed to derive from a set of word distributions, and each word is allowed to have another distribution, leading to a label vector $\vec{z} = \{\vec{z}_m\} = \{\{z_{m,n}\}\}$ analogous to $\vec{w}$. This is illustrated in Fig. 3.12(c) and subfigures below. The likelihood of a corpus $\vec{w}$ becomes:

$$p(\vec{w}) = \prod_m \prod_n \sum_k p(w_{m,n}{=}t|z_{m,n}{=}k)p(z_{m,n}{=}k|m) \; . \tag{3.32}$$

Note that now the probability of the label $z_{m,n}$ is conditioned on the document m. Like the mixture model, multiple documents share the same word distributions to associate their words with. Because admixture, however, leaves flexibility to assign a different label $z_{m,n}$ to every observed word, each document has a different proportion of labels, resulting in two coupled mixtures to generate a document: (1) a mixture of labels for the document and (2) a mixture of words for each label (like the unigram model).

If the word distributions contain semantically related terms, the admixture model can be used to model latent semantics, and the semantics of documents aggregate from the semantics of the word distributions they are associated with. This is the basic working principle of topic models, which may be characterised as unsupervised admixture models.[13] In a supervised setting, the aggregation of several document labels corresponds to multi-classification, i.e., relaxing the constraint that an item may only belong to a single class.

3.5.4 Probabilistic latent semantic analysis

Empirically, the SVD-based LSA approach described in Section 3.3 turned out to be effective, but the assumption of an additive Gaussian error distribution on term frequencies (due to the optimality of the SVD under the L_2-norm) implies that the SVD is suboptimal with respect to the true statistical behaviour of term frequency data [Hofmann 2001, Jansche 2003].

This finding motivated Hofmann [Hofmann 1999a; 2001] to develop a probabilistic algorithm that mimics the admixture model above to re-enact the SVD model in probabilistic terms.[14] When $\vec{w}$ is expressed as a term-document matrix and the probabilities are expressed as probabilistic matrices (row sum = 1), the structure of (3.32) corresponds to that of SVD-based LSA in (3.1): $\underline{\tilde{A}}^* = \underline{\tilde{U}}^* \underline{\tilde{S}}^* \underline{\tilde{V}}^{*\top}$ with $\tilde{a}^*_{m,t} = p(w{=}t|m)$, the probability of seeing term t on document m, $\tilde{u}^*_{k,t}\sqrt{\tilde{s}^*_k} = p(w{=}t|z{=}k)$ and $\sqrt{\tilde{s}^*_k}\,\tilde{v}^*_{m,k} = p(z{=}k|m)$ [Hofmann 2001]. Probabilistic parameter estimation thus replaces matrix decomposition.

Bayesian network. Fig. 3.13(a) shows the corresponding Bayesian network (BN). Each value $z{=}k$ describes a latent topic. Obviously, the terms in document $\vec{w}_m$ may stem from several latent

[13] As topic models form a wider set of models, also supervised settings may be implemented.

[14] The term admixture appeared only later in text modelling; the development of PLSA [Hofmann 1999a] and LDA [Blei et al. 2002] seems to have been independent of that of the Bayesian genetic admixture model [Pritchard et al. 2000].

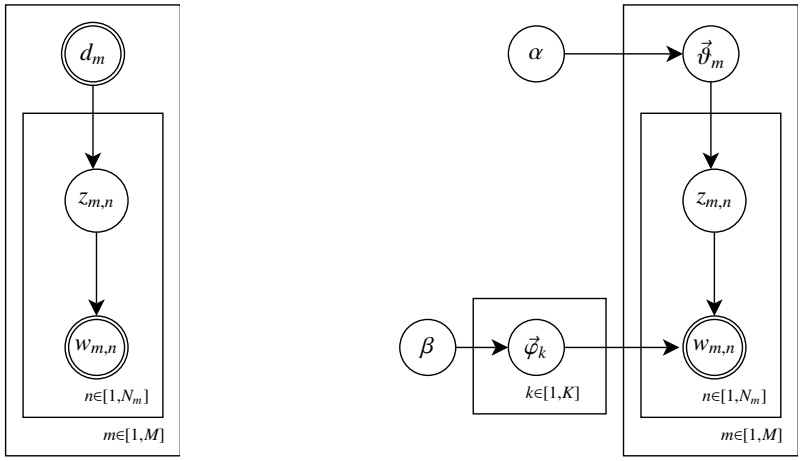


Figure 3.13: Bayesian networks of (a) PLSA and (b) LDA.

topics, and for each document there is a distribution $p(z{=}k|m)$ over the topics involved. This distribution characterises the topics in the document and can be considered a soft clustering of the documents. From a dual perspective, terms with high probabilities $p(w{=}t|z{=}k)$ with respect to a latent topic $z{=}k$ can be considered a soft clustering of terms, which, for instance, could include terms with similar meanings or synonyms. On the other hand, the same term $w{=}t$ may be associated with different topics, which may happen if it is polysemic, i.e., has different meanings.

Inference in this model is performed by maximum likelihood estimation based on the empirical multinomial distributions specific to the documents (denoted by the observed variables d_m in the Bayesian network). This means that there is no generative model to obtain these multinomials themselves. The likelihood becomes exactly that of (3.32).

Although a vast simplification, the model leads to meaningful topics, as terms that often occur together have a high probability $p(w{=}t|z{=}k)$ with respect to some topic. Hofmann has shown experimentally that PLSA achieves a higher data likelihood than LSA [Hofmann 2001] (see Section 3.7).

3.6 Generative latent semantic models

One of the major drawbacks in PLSA is that the model cannot generate unseen documents because the topics are described by empirical distributions associated with known documents. This may be alleviated by a proper generative model that furthermore allows full use of Bayesian inference, additionally reducing overfitting problems.

This section introduces the core model that achieves this task, latent Dirichlet allocation, and gives a brief overview of its extension towards non-parametric approaches. This generative approach to topic or admixture models will be the basis of subsequent research in this thesis.

3.6.1 Latent Dirichlet allocation

Latent Dirichlet allocation (LDA [Blei et al. 2003b]) extends the model of PLSA by defining the topic-specific multinomial term distributions $p(w{=}t|z{=}k)$ and document-specific mixture weights $p(z{=}k|m)$ as random variables themselves, following a fully Bayesian approach.

More specifically, LDA defines a generative model that includes Dirichlet-distributed priors over the masses of the multinomials $p(w{=}t|z{=}k)$ and $p(z{=}k|m)$ (cf. Section 3.4.2). Fig. 3.13(b) shows the corresponding Bayesian network. For the generation of document mixture weights, the multinomial $p(z{=}k|m)$, which in PLSA is an empirical distribution conditioned on the document indicator d_m, becomes a distribution $p(z{=}k|\vec{\vartheta}_m)$. This is conditioned on a parameter vector $\vec{\vartheta}_m$, which is sampled from a Dirichlet distribution $p(\vec{\vartheta}_m|\alpha)$ with hyperparameter α. Generating such a document-specific topic mixture from a prior also allows to estimate the concepts of previously unseen documents after training. This is described in Section 3.7.

In a similar manner, for $p(w{=}t|z{=}k)$ a term distribution $p(w{=}t|z{=}k, \vec{\varphi}_k)$ is introduced with a parameter vector $\vec{\varphi}_k$ for each topic $z{=}k$, sampled from a Dirichlet distribution $p(\varphi_k|\beta)$ with hyperparameter β. The complete generative model is presented in Fig. 3.14, and Fig. 3.15 gives a list of all involved quantities. With the parameters $\vartheta = \{\vec{\vartheta}_m\}$ and $\varphi = \{\vec{\varphi}_k\}$ substituting the corresponding probabilities in (3.32), the corpus likelihood becomes:

$$\begin{aligned} p(\vec{w}|\vartheta, \varphi) &= \prod_m \prod_n \sum_k p(w_{m,n}{=}t|z_{m,n}{=}k, \varphi) p(z_{m,n}{=}k|m, \vartheta) \\ &= \prod_m \prod_n \sum_k \varphi_{k,w_{m,n}} \vartheta_{m,k} \ , \end{aligned} \tag{3.33}$$

which is now a function of the parameters ϑ and φ.[15] To cover the entire generative model of LDA, the joint likelihood of all quantities may be given, conditioned on the hyperparameters, i.e., the parameters of the Dirichlet distributions:

$$\begin{aligned} p(\vec{w}, \vec{z}, \vartheta, \varphi|\alpha, \beta) = \prod_m \prod_n \mathrm{Mult}(w_{m,n}{=}t|\vec{\varphi}_k)\ \mathrm{Mult}(z_{m,n}{=}k|\vec{\vartheta}_m) \\ \cdot \prod_m \mathrm{Dir}(\vec{\vartheta}_m|\alpha) \cdot \prod_k \mathrm{Dir}(\vec{\varphi}_k|\beta) \end{aligned} \tag{3.34}$$

where α and β can be scalars or vectors.

Inference in the LDA model attempts to find the variables $\vec{z}$ and parameters $\vec{\vartheta}_m$ and $\vec{\varphi}_k$ given the observations $\vec{w}$, which extends PLSA's ML estimation of the distributions $p(w{=}t|z{=}k)$ and $p(z{=}k|d_m)$ to Bayesian inference.[16] Hyperparameters α and β are either set a priori or learnt from the observations, as well. The more expressive Bayesian model of LDA comes at the expense of numerical intractability of exact inference, and several approximate inference algorithms have been proposed, e.g., mean-field variational inference [Blei et al. 2002; 2003a, Teh et al. 2007], expectation propagation [Minka & Lafferty 2002], and Gibbs sampling [Griffiths 2002, Griffiths & Steyvers 2004, Pritchard et al. 2000].

[15] We use symbols without vector or matrix markup to refer to multi-dimensional variables generically.

[16] In fact, [Girolami & Kaban 2003] show that PLSA is an MAP estimator for LDA with "flat" Dir(1) priors.

```
// topic plate:
for all topics k ∈ [1, K] do
    sample mixture components φ⃗_k ~ Dir(β⃗);
// document plate:
for all documents m ∈ [1, M] do
    sample mixture proportion ϑ⃗_m ~ Dir(α⃗);
    sample document length N_m ~ Poiss(ξ);
    // word plate:
    for all words n ∈ [1, N_m] in document m do
        sample topic index z_{m,n} ~ Mult(ϑ⃗_m);
        sample term for word w_{m,n} ~ Mult(φ⃗_{z_{m,n}});
```

Figure 3.14: Generative model for latent Dirichlet allocation.

All of the inference approaches structurally are special cases of the expectation–maximisation (EM) algorithm [Dempster et al. 1977]. They use (3.34) as their point of departure to derive inference equations, and one can distinguish complete-model inference that learns all parameters [Blei et al. 2002; 2003a, Pritchard et al. 2000] and collapsed inference that learn only the topic associations $\vec{z}$ [Griffiths 2002, Griffiths & Steyvers 2004, Teh et al. 2007].

Collapsed inference, also referred to as Rao-Blackwellised inference [Casella & Robert 1996, Buntine & Jakulin 2005], assumes that the parameters φ and ϑ are highly correlated with the latent variable $\vec{z}$ and therefore can be integrated out and estimated from $\vec{z}$. In this case, the likelihood (3.34) simplifies. Aggregating the co-occurrences between topics and observable items (documents and terms) using $n_{m,k}$ as the number of times that topic k occurs in document m and $n_{k,t}$ the number of times that topic k occurs for term t, the joint likelihood of the model variables becomes:[17]

$$
\begin{aligned}
p(\vec{w},\vec{z}|\vec{\alpha},\vec{\beta}) &= \iint \prod_m \prod_k \vartheta_{m,k}^{n_{m,k}} \prod_t \varphi_{k,t}^{n_{k,t}} \cdot \prod_m \frac{1}{\Delta(\vec{\alpha})} \prod_k \vartheta_{m,k}^{\alpha_k-1} \cdot \prod_k \frac{1}{\Delta(\vec{\beta})} \prod_t \varphi_{k,t}^{\beta_t-1} \,\mathrm{d}\varphi\,\mathrm{d}\vartheta \\
&= \iint \prod_m \frac{1}{\Delta(\alpha)} \prod_k \vartheta_{m,k}^{n_{m,k}+\alpha_k-1} \cdot \prod_k \frac{1}{\Delta(\beta)} \prod_t \varphi_{k,t}^{n_{k,t}+\beta_t-1} \,\mathrm{d}\varphi\,\mathrm{d}\vartheta \\
&= \iint \prod_m \frac{\Delta(\vec{n}_m+\vec{\alpha})}{\Delta(\alpha)} \mathrm{Dir}(\vec{\vartheta}_m|\vec{n}_m+\vec{\alpha}) \cdot \prod_k \frac{\Delta(\vec{n}_k+\vec{\beta})}{\Delta(\vec{\beta})} \mathrm{Dir}(\vec{\varphi}_k|\vec{n}_k+\vec{\beta}) \,\mathrm{d}\vartheta\,\mathrm{d}\varphi \\
&= \prod_m \frac{\Delta(\vec{n}_m+\vec{\alpha})}{\Delta(\vec{\alpha})} \cdot \prod_k \frac{\Delta(\vec{n}_k+\vec{\beta})}{\Delta(\vec{\beta})} = p(\vec{z}|\vec{\alpha}) \cdot p(\vec{w}|\vec{z},\vec{\beta})
\end{aligned}
\tag{3.35}
$$

with the vector representations $\vec{n}_m = \{n_{m,k}\}_{k=1}^K$ and $\vec{n}_k = \{n_{k,t}\}_{t=1}^V$ and vector hyperparameters. Some of the actual inference methods will be described in later chapters and in the following, we will assume the parameters to be fitted to the observed data by one of these methods.

[17] The derivation starts with the definitions of the multinomial (3.7) and Dirichlet distributions (3.16) and pulls both together in the products over k and t. By gathering the resulting terms into $\Delta(\cdot)$ terms and Dirichlet distributions again, the integrals turn out to be over the complete Dirichlet distributions and thus vanish, leaving the ratio of $\Delta(\cdot)$ terms. Alternatively, the Dirichlet integral may be used, as described for (3.16).

M number of documents to generate (const scalar).

K number of topics / mixture components (const scalar).

V number of terms t in vocabulary (const scalar).

$\vec{\alpha}$ hyperparameter on the mixing proportions (K-vector or scalar if symmetric).

$\vec{\beta}$ hyperparameter on the mixture components (V-vector or scalar if symmetric).

$\vec{\vartheta}_m$ parameter notation for $p(z|d{=}m)$, the topic mixture proportion for document m. One proportion for each document, $\vartheta = \{\vec{\vartheta}_m\}_{m=1}^{M}$ ($M \times K$ matrix).

$\vec{\varphi}_k$ parameter notation for $p(t|z{=}k)$, the mixture component of topic k. One component for each topic, $\varphi = \{\vec{\varphi}_k\}_{k=1}^{K}$ ($K \times V$ matrix).

N_m document length (document-specific), here modelled with a Poisson distribution [Blei et al. 2002] with constant parameter ξ.

$z_{m,n}$ mixture indicator that chooses the topic for the nth word in document m.

$w_{m,n}$ term indicator for the nth word in document m.

Figure 3.15: Quantities in the model of latent Dirichlet allocation.

3.6.2 Non-parametric mixtures

One of the drawbacks of the mixture models described in the previous section is the requirement to choose the number of topics K a priori, while all other parameters and hyperparameters can be estimated. A naïve approach to this problem is to perform various model inference runs at different K and try to find the optimum K according to some quality criterion, estimating all other parameters, or setting parameters a priori and finding the optimal K as proposed in [Griffiths & Steyvers 2004]. More formally elegant, the choice of model dimensionality is the subject of non-parametric Bayesian approaches that have become a large research area in statistics and machine learning [Teh & Jordan 2009, Ghosh & Ramamoorthi 2003]. Here, the attribute "non-parametric" refers to the capability of the methods to avoid assumptions on the particular distribution of the data, in our case the dimensionality of the distribution. The predominant non-parametric prior on mixture components used in the literature is the Dirichlet process (DP) because in many respects this stochastic process is similar to the Dirichlet distribution, especially its conjugacy with the multinomial distribution. In effect, models using DP as a prior on mixture components avoid any a priori parameter choice, completely relying on the data to fit to the model.

Dirichlet process. Similar to the Dirichlet distribution, the Dirichlet process (DP [Ferguson 1973, Teh 2007]) can be understood as a distribution over distributions. To define the DP, let us consider some probability distribution $G_0 = p(x|\cdot)$ defined on a support set $S(G_0)$ as a "base distribution".

Taking the Dirichlet distribution in Fig. 3.6 as an example, this support set is the simplex defined by $\sum_k p_k = 1$. Let's further consider that the support is partitioned into different subsets $S_1, S_2, S_3, \ldots, S_K$, in the example for instance the four smaller triangles partitioning the simplex. Each of these partitions has a probability associated with it, $G_0(S_k) = p(x \in S_k|\cdot) = \int_{S_k} p(x|\cdot)\,\mathrm{d}x$, and of course all partition probabilities sum up to 1.

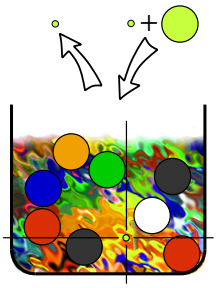


Figure 3.16: Generalised Pólya urn sampling scheme, cf. Fig. 3.9.

Formally, a Dirichlet process $\mathrm{DP}(\alpha, G_0)$, parameterised with a base distribution G_0 and a scalar parameter α, is a stochastic process that generates samples $G \sim \mathrm{DP}(\alpha, G_0)$. These distributions G have the same support set as G_0. The defining property of the Dirichlet process is that the mass of its samples G is distributed on the support $S(G)$ in such a way that one can arbitrarily partition S into subsets $\{S_k\}_{k=1}^K$ and always will obtain samples $G(S_k)$ that are jointly Dirichlet-distributed with parameters $\alpha G_0(S_k)$:

$$\{G(S_k)\}_{k=1}^K \sim \mathrm{Dir}\Big(\{\alpha G_0(S_k)\}_{k=1}^K\Big)\,. \tag{3.36}$$

For the example partition of the 3-dimensional Dirichlet in Fig. 3.6, this is:

$$\begin{aligned}\{&G(x \in S_1), G(x \in S_2), G(x \in S_3), G(x \in S_4)\} \\ &\sim \mathrm{Dir}\left(\alpha\mathrm{Dir}(x \in S_1|\vec{a}), \alpha\mathrm{Dir}(x \in S_2|\vec{a}), \alpha\mathrm{Dir}(x \in S_3|\vec{a}), \alpha\mathrm{Dir}(x \in S_4|\vec{a})\right)\end{aligned}$$

where $\vec{a} = (4, 4, 2)$.[18] This property applies to any partition of $S(G_0)$ with any finite number of elements K. Illustratively, by controlling the Dirichlet parameters in (3.36), the mass of the base distribution is imperfectly replicated in G, controlling the amount of imperfection by the precision parameter α and the "resolution" by the number of partitions K. Opposed to the Dirichlet distribution, the DP with a Dirichlet base distribution of Fig. 3.6 produces infinitely many discrete samples on the two-dimensional simplex with non-uniform weight but with the weights integrated over each partition distributed $\mathrm{Dir}(4, 4, 2)$.

The implication of (3.36) is that the DP inherits some important properties from the Dirichlet distribution. It has been shown that its samples G can be understood as a set of point masses [Sethuraman 1994], similar to the multinomials generated by the Dirichlet distribution. The difference is that G has infinitely many point masses whose locations are sampled from the base distribution G_0 as opposed to the multinomial as a sample from the Dirichlet distribution that assigns mass "only" to category labels instead of actual locations on the support $S(G)$ drawn from the base distribution.[19] Concretely, with an N-dimensional Dirichlet as a base distribution, a DP sample is a set of infinitely many ($K \to \infty$) N-dimensional multinomials.[20]

[18] It is important not to confuse the base distribution (here: $G_0 = \mathrm{Dir}(4, 4, 2)$) with the partition-generating Dirichlet valid for all DP base distributions, and not to confuse the dimensionality of the base distribution (here: 3) with the (arbitrary) number of example partitions (here: 4).

[19] When splitting the vector parameter of a Dirichlet into $\vec{\alpha} = \alpha\vec{m}$, the mean vector $\vec{m}$ may be considered the equivalent of a base distribution [Wallach 2008].

[20] The representation of finite K in (3.36) just accumulates infinitely many mass points in each partition S_k.

Moreover, the DP exhibits a posterior clustering behaviour very similar to the Dirichlet distribution (Pólya urn, cf. (3.28) and Fig. 3.9), which can be described with a "generalised Pólya urn scheme": An urn is initially filled with a liquid of infinitely many colours (=base distribution). Sampling an infinitesimal volume from the urn results in a colour value (either from a ball or from the liquid), and a ball of the sampled colour is replaced along with the sample.[21] In Fig. 3.16, this process is shown graphically.

The corresponding conditional distribution for colours/categories given prior observations is [Neal 1998]:

$$p(\tilde{z}{=}k|\vec{z},\alpha) = \begin{cases} \dfrac{n_k}{n+\alpha} & n_k > 0 \\ \dfrac{\alpha}{n+\alpha} & \text{new colour.} \end{cases} \tag{3.37}$$

Compared to the Dirichlet–multinomial distribution pair, (3.37) is the infinite limit $K \to \infty$ of the posterior (3.28) for the symmetric case $\alpha_k = 1/K$ [Neal 1998]. In this respect, $\mathrm{DP}(\alpha, G_0)$ can be considered an infinite-dimensional Dirichlet distribution with parameter αG_0, and the mass of the hyperparameter α is used to create new components in its multinomial sample G.[22] This difference to the Dirichlet (that uses α to smooth fitting to a fixed number of categories) may be seen as an alternative strategy of both models to fit to observed data with an inherent dimensionality different from the current model.

Dirichlet process mixtures. Exactly this different clustering strategy is used to estimate the number of components K in DP-based mixture models. To illustrate this, consider a fully Bayesian finite mixture that generates discrete words from parameters $\vec{\vartheta}_k$, as shown in Fig. 3.17(a). The model has a global component proportion $\vec{\pi} \sim \mathrm{Dir}(\alpha)$, which corresponds to label weights $p(w_{m,n}|z_m)$ in the mixture of unigrams model in Fig. 3.12, only with a single word per document. Components $\vec{\varphi}_k$ are drawn from a base distribution, G_0, which here is a beta distribution; see Fig. 3.17(b). Parameters $\vec{\varphi}_k \sim G_0$ are 2-dimensional multinomials. Identifying components $\vec{\varphi}_k$ from which values x_n are sampled underlies the Pólya urn scheme quantified by (3.28), and in Fig. 3.17(b), the change of the posterior over the indicator variable is shown conditioned on the observation of three samples.

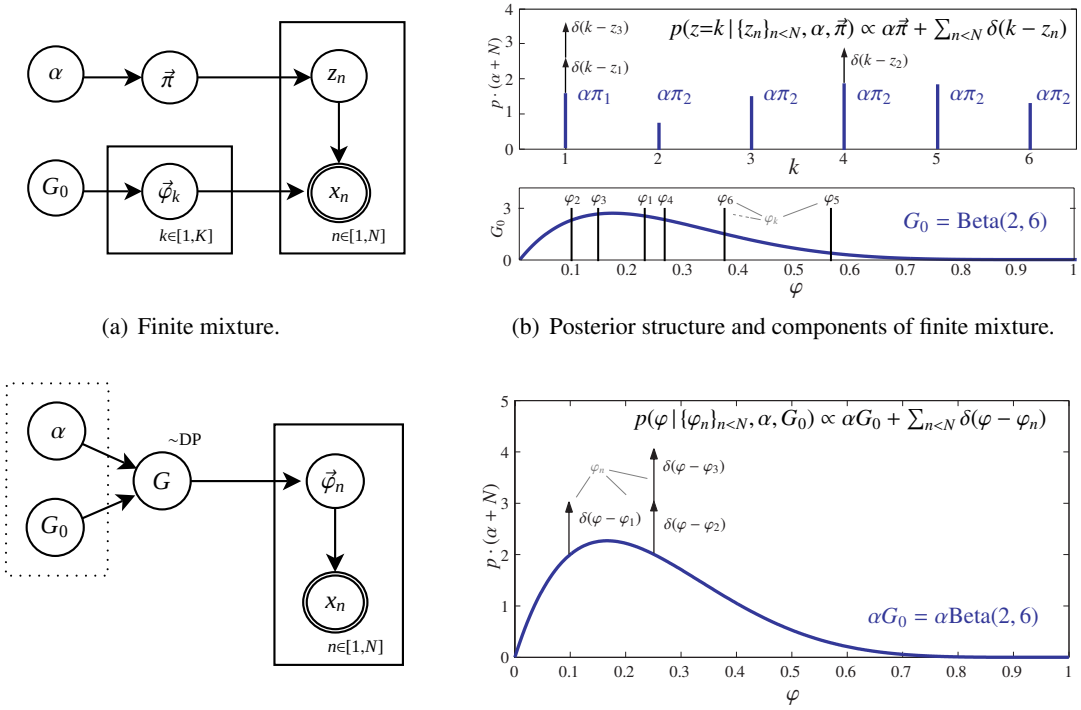
In the corresponding infinite mixture[23] model shown in Fig. 3.17(c), the DP sample G takes over the role of both the mixture proportions $\vec{\pi}$ and components $\vec{\varphi}_k$, and components are generated according to the generalised Pólya urn scheme quantified by the predictive distribution (3.37). Fig. 3.17(d) illustrates how the components are generated: This predictive distribution shows that the base distribution is augmented with the masses of the previous observations, associating non-zero probability to values on $S(G_0)$ already seen.[24] Note that (3.37) has such a simple form because G is marginalised. In practice, this representation is important and referred to as the Chinese restaurant process (CRP [Teh 2007]): A Chinese restaurant has infinitely many tables,

[21] Here the parameter α controls the initial amount of liquid.

[22] Knowing that samples G are discrete, "creating" new components in the infinite-dimensional Dirichlet distribution can be imagined as adding mass to one of infinitely many nonzero mass points not observed before, whose summed weight is α.

[23] Non-parametric methods often work with ∞ to avoid specification of parameters, but dimensions modelled infinite are finite for finitely many data. Therefore, an infinite mixture will have finite K in practice.

[24] In this figure, the δ functions are graphically added to the continuous base distribution. The value of this sum at a δ function is equal to the height or count of the stacked δ's, but equals G_0 in its immediate surroundings.



(a) Finite mixture.

(b) Posterior structure and components of finite mixture.

(c) Infinite mixture.

(d) Posterior structure of infinite mixture.

Figure 3.17: Mixture models: finite with indicator $\vec{\pi}$ and components $\vec{\varphi}_k$, infinite with DP prior on component samples $\vec{\varphi}_n$. The base distribution is Beta(2, 6) in both cases.

and customers enter the room, deciding whether to share a table k with other customers with probability n_k or choosing a new table with probability α. When sampling G, sharing a table k corresponds to drawing values from a component $\vec{\varphi}_n = \vec{\varphi}_{n-j}$ that has been seen before, and sitting at a new table to a sample from the base distribution.

Extending this infinite mixture model to LDA as an admixture model requires to sample a Dirichlet process for each document. The main problem with this is to couple these processes to share topics, i.e., $\vec{\varphi}_{m,n}$ that can be identical for words in different documents m. As DP samples G are discrete[25], the probability that two independent G_m will produce the same component $\vec{\varphi}$ goes to zero. Therefore, the DP itself needs to be extended, and several possibilities have been proposed, notably dependent DPs [MacEachern 1999] and hierarchical DPs [Teh et al. 2006]. Especially the latter allow to completely map the LDA admixture to the domain of infinite mixtures.

Hierarchical Dirichlet process. When applying the DP mixture principle to admixture, coupling between mixtures can be achieved by using a Dirichlet process sample $G_0 \sim \mathrm{DP}(\cdot)$ as a base distribution[26] of child DP samples $G_m \dim \mathrm{DP}(\alpha G_0)$, allowing them to sample subsets of the

[25]The graph in Fig. 3.17(d) can be understood as a smoothed histogram representation of the discrete infinite set of samples from the base distribution (which look just like the base distribution but are at discrete points on $S(G_0)$).

[26]Note that DP literature often makes use of measure theory, expressing the base distribution more formally as a base measure.

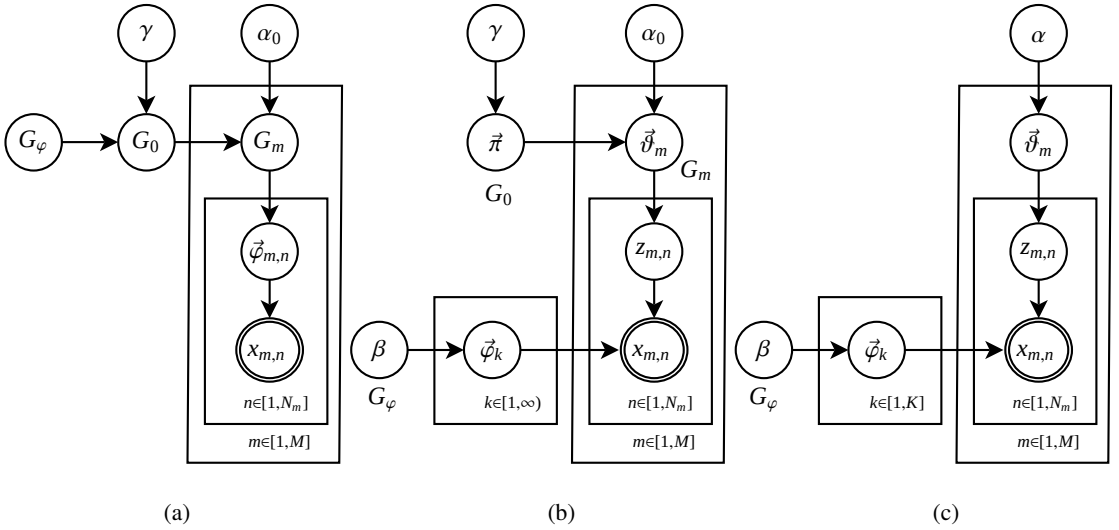
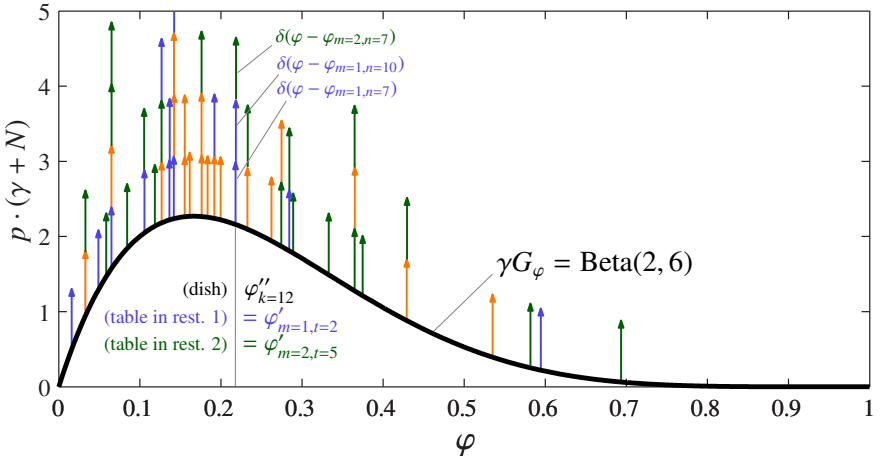


Figure 3.18: Bayesian networks of the hierachical Dirichlet process: (a) full model, (b) representation with stick-breaking prior, and (c) LDA as the finite equivalent to the HDP.

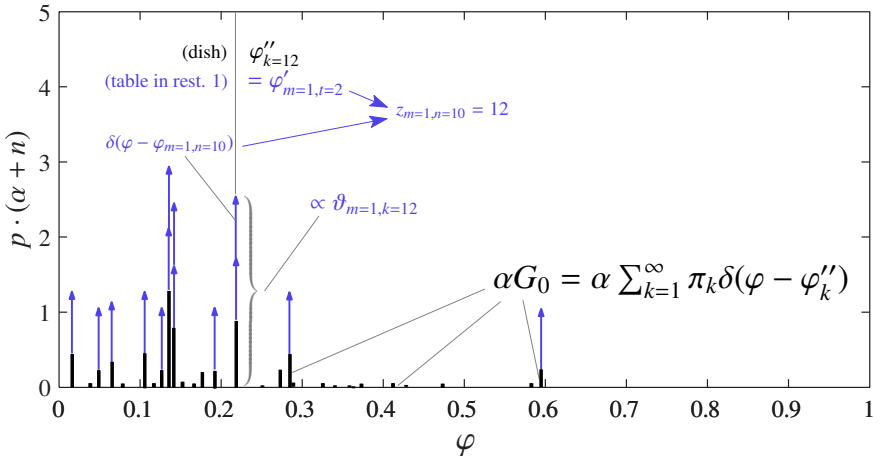
infinitely many atoms in the root DP sample. The root DP has as its base distribution that of the components, i.e., $G_\varphi = \mathrm{Dir}(\beta)$ in LDA, as in the simple infinite mixture above. Components/topics observed in several documents then are identified when the same mass points appear in the samples G_m of several documents. The resulting model is a Hierarchical Dirichlet Process (HDP [Teh et al. 2004; 2006]), and in Fig. 3.18 several representations are presented as Bayesian networks. In particular, the actual Bayesian network of the full model is given in Fig. 3.18(a), but according to the discrete nature of the sample G_0, a simplified representation may be given in Fig. 3.18(b) that uses an infinite-dimensional multinomial $\vec{\pi}$ to represent the root DP sample G_0 as the prior over document multinomials $\vec{\vartheta}_m$. These multinomials represent the document DP samples G_m that are integrated out, similar to the Dirichlet–multinomial case in Section 3.4.2. The ϑ_m can be used to explicitly sample topic indicator variables $z_{m,n}$. This representation has been often termed the stick-breaking prior (SBP [Teh 2007]) representation because a priori the point masses of $\vec{\pi}$ are distributed like the lengths of pieces broken off a stick infinitely many times with break points distributed $\mathrm{Beta}(1, \alpha)$ on the stick remaining at each step [Sethuraman 1994]. This stick-breaking process is often referred to as GEM distribution (after Griffiths–Engen–McCloskey), $\vec{\pi} \sim \mathrm{GEM}(\gamma)$.

Interestingly, both $\vec{\vartheta}_m$ and $z_{m,n}$ are equivalent to those in LDA, only that parameters $\vec{\vartheta}$ are infinite-dimensional and distributed according to $\vec{\vartheta}_m \sim \mathrm{Dir}(\alpha_0\vec{\pi})$. In this representation, the base distribution $G_\varphi = \mathrm{Dir}(\beta)$ can be explicitly indexed by the topic indicators $z_{m,n}$=k. For reference, the Bayesian network of LDA is given in Fig. 3.18(c).

In the HDP model, the DP clustering scheme still applies, but now an additional layer needs to be added to accommodate for the clustering between DPs, yielding the Chinese restaurant franchise (CRF) process [Teh et al. 2006]: Each Chinese restaurant has infinitely many tables, and on each table it serves one of infinitely many dishes that other restaurants may serve as well.



(a) Sampling tables for dishes, $\varphi'_{m,t} = \varphi''_k$, for all restaurants.



(b) Sampling customers for tables, $\varphi_{m,n} = \varphi'_{m,t}$, for restaurant $m = 1$.

Figure 3.19: Chinese restaurant franchise: HDP predictive distributions with DPs marginalised.

When entering the restaurant, not only choose customers a table (topic parameter; sample from G_φ appearing in G_0), but also whether they may have a dish popular among several restaurants (topic labels k shared among documents; samples from G_φ appearing in several G_m) or a new one that is offered globally afterwards (new label k).

In Fig. 3.19, the predictive distributions of the HDP are illustrated using the CRF representation.[27] For better readability, parameters φ are marked up with primes according to how they are indexed: by customer/token, $\varphi_{m,n}$, by table in a restaurant/document, $\varphi'_{m,t}$, and by dish/topic label, φ''_k. The location on the φ-axis identifies the index variables $z_{m,n}{=}k$ associated with the

[27] These distributions are $p(\varphi'_{m,t} \,|\, \varphi'_{<m,t}, \gamma, G_\varphi) \propto \gamma G_\varphi + \sum_k n_k \delta(\varphi - \varphi''_k)$ and $p(\varphi_{m,n} \,|\, \varphi_{m,<n}, \alpha, G_0) \propto \alpha G_0 + \sum_t n_{m,t} \delta(\varphi - \varphi'_{m,t})$, respectively, with n_k the total number of customers eating dish k and $n_{m,t}$ the number of customers eating at table m, t.

tokens. In the upper plot, the global locations of the topic multinomials φ (again: dimension 2) are sampled. It can be seen that the base distribution influences the location of the masses φ''_k and that the global topic distribution accumulates the location of tables (topic multinomials) from all documents (different colours): Tables $\varphi'_{m,t}$ from different restaurants m serve a common dish k, i.e., $\varphi'_{m,t} = \varphi''_k$. In the lower plot, the distribution for one of these restaurants/documents is shown. The base distribution αG_0 has infinitely many weights drawn from G_φ, including those located at the points of the tables in the upper plot. A subset of the infinitely many masses in G_0 will be associated to customers of each restaurant m, determining the tables t they sit on, $\varphi_{m,n} = \varphi'_{m,t}$, and, via the global association between tables and dishes, the dish k. For the 10th customer in restaurant 1, we have $\varphi_{m=1,n=10} = \varphi'_{m=1,t=2} = \varphi''_{k=12}$, which leads to $z_{m=1,n=10} = 12$. Finally, the overall mass accumulated for one table t corresponds to the (unnormalised) document–topic weight $\vartheta_{m,k}$ in the stick-breaking representation in Fig. 3.18(b), which in turn represents a point mass of the marginalised G_m.

Scope. To keep this thesis focussed, non-parametric models like the DP and HDPs will not be considered a primary research topic, but rather seen as extensions to parametric (finite) ones. Furthermore, the Dirichlet process is taken as an exemplary prior for non-parametric models, noting that its generalisation, the Pitman-Yor process or two-parameter Poisson-Dirichlet process [Pitman & Yor 1997, Buntine & Hutter 2010] is sometimes preferred because it permits a better match with the power-law distributions exhibited by language and semantic features than the DP [Teh 2006], at the cost of more complex mathematics involved.

3.7 Analysis of latent topics

Parameters of topic models such as LDA are a representation of the semantics of the corpus, its "topic structure". Because topic models usually are designed for unsupervised learning, analysis of trained parameters is one of the central tasks for their usage.

As an illustrative example, the most likely terms from topic distributions φ of a news corpus are shown in Fig. 3.20. These data were taken from [Heinrich et al. 2005b] where LDA was run on the original terms as well as on word bigrams in order to capture structures in the text beyond bag-of-words. The topic labels have been added manually after inspection.[28]

In this section, we outline methods to use the topic structure of a given corpus in order (1) to analyse their consistency qualitatively, (2) to estimate the topic structure of unseen documents ("model querying"), (3) to estimate the quality of the clustering implied by the estimated topics and (4) to measure the effectiveness of the model w.r.t. generalisation and retrieval performance.

Although mainly the LDA model is used for presentation, the discussed methods can be adapted to other models as appropriate. The presentation is restricted to generally applicable methods; many additional analysis methods will directly be motivated by the actual task that a particular model tries to solve.

[28] Automatic labelling of topics as an alternative has been proposed by, e.g., [Mei et al. 2007b].

3.7.1 Human judgement

As relevance in information retrieval and knowledge discovery tasks is a matter of human judgement, it is worthwhile to analyse the results of topic extraction the same way. This is done on the basis of the parameters $\vartheta_{m,k}$ and $\varphi_{k,t}$ for LDA (with dimensions document m, topic k and term t), which may be ordered by probability value and the corresponding items listed. Typically, probability mass in LDA's (multinomial) parameters is distributed relatively sparsely, i.e., there are only a few dimensions with significant mass for topics in $\vec{\vartheta}_m$ and for terms in $\vec{\varphi}_k$.[29]

Empirically, by far not all topics exhibit semantic coherence as well as the example topics. For instance very frequent words that do not appear in a stop list but co-occur in many documents because of their frequency may enter a common topic.

Whether to check model quality or to weed out "bad" topics, the question to answer is the semantic coherence or "interpretability" of topics, and humans will quickly notice whether the most likely words in a topic correspond to meaningful similarities between words and topics in a document to meaningful representation of its true content. To make these interpretability judgements scientifically dependable, a blind test regime has been designed in [Chang et al. 2009], which follows the spirit of tests in empirical psychology (cf. [Griffiths et al. 2007] and references).

The basic idea is to present lists of topics to test participants and let them identify "intruding" items, i.e., those that don't (or least) belong to the others. In the context of LDA, this has been proposed in two variants:

- *Word intrusion:* Test participants are presented with a randomly ordered set of six terms and asked to find the semantically least coherent term, the "intruder". Five words are the most likely of a topic distribution $\varphi_{k,t}$, the sixth one is one of low likelihood. An example is shown in Fig 3.21(a).[30]

- *Topic intrusion:* Test participants are presented with a document excerpt and four topics, three of which are highly ranked for the document $\vartheta_{m,k}$ and one is the "intruder". An example is shown in Fig 3.21(b).

To quantify word intrusion, [Chang et al. 2009] propose a model precision quantity that corresponds to the relative number of participants that agree with the model and found the intruding word, averaged over all experiments. Interestingly, they found that model precision does *not* necessarily correlate with model likelihood (also see Section 3.7.4). They conjectured that LDA trades off improved likelihood with interpretability as the number of topics increases and they become more fine-grained.

To quantify topic intrusion, [Chang et al. 2009] define the topic log odds measure: the log ratio of the probability mass $\vartheta_{m,k}$ assigned to the true intruder $k = \bar{k}$ to the probability mass of the intruder $\hat{k}$ selected by the participant, $\log \vartheta_{m,\bar{k}} / \vartheta_{m,\hat{k}}$, averaged over all experiments. Here also human judgements have been found to deviate from the model likelihood, especially if documents contain disparate themes.

[29] This may vary depending on the data, the actual model or hyperparameters; cf. Section 4.5.3.

[30] This questionnaire has been used for [Heinrich 2011b] to evaluate the topics of an expert finding model, ETT1, that is described in Chapter 10.

Topic label	Most likely words according to $\varphi_{k,t} = p(\text{word}\|\text{topic})$
Politische Parteien *political parties*	CDU *(party)* ▯ Partei *political party* ▯ Kohl *(politician)* ▯ Aufklärung *clarification* ▯ Schäuble *(politician)* ▯ Zeitung *newspaper* ▯ Union *(party)* ▯ Krise *crisis* ▯ Wahrheit *truth* ▯ Affäre *affair* ▯ Christdemokraten *christian democrats* ▯ Glaubwürdigkeit *credibility* ▯ Konsequenzen *consequences* ▯
Bundesliga *(German football league)*	FC *(club abbreviation)* ▯ SC *(club abbreviation)* ▯ München *(club city)* ▯ Borussia *(club name)* ▯ SV *(club abbreviation)* ▯ VfL *(club abbreviation)* ▯ Kickers *(club name)* ▯ SpVgg *(club abbreviation)* ▯ Uhr *o'clock* ▯ Köln *club city* ▯ Bochum *(club city)* ▯ Freiburg *(club city)* ▯ VfB *(club abbreviation)* ▯ Eintracht *(club name)* ▯ Bayern *(club name)* ▯ Hamburger *(club name)* ▯ Bayern+München *(club name)* ▯
Polizei / Unfall *police / accident*	Polizei *police* ▯ verletzt *injured* ▯ schwer *heavily* ▯ Auto *car* ▯ Unfall *accident* ▯ Fahrer *driver* ▯ Angaben *statements, information* ▯ schwer+verletzt *heavily injured* ▯ Menschen *people* ▯ Wagen *vehicle* ▯ Verletzungen *injuries* ▯ Lawine *avalanche* ▯ Mann *man* ▯ vier *four* ▯ Meter *meter* ▯ Straße *road* ▯
Tschetschenien *Chechnya*	Rebellen *insurgents* ▯ russischen *Russian (flex.)* ▯ Grosny *(Chechnyan capital)* ▯ russische *Russian (flex.)* ▯ Tschetschenien *Chechnyan* ▯ Truppen *troops* ▯ Kaukasus *Caucasus* ▯ Moskau *Moscow* ▯ Angaben *statements, information* ▯ Interfax *(Russian news agency)* ▯ tschetschenischen *Chechnyan (flex.)* ▯ Agentur *(news) agency* ▯
Politik / Hessen *politics / Hassia (a German State)*	FDP *liberal democratic party* ▯ Koch *(politician)* ▯ Hessen *(state)* ▯ CDU *party* ▯ Koalition *coalition* ▯ Gerhardt *(politician)* ▯ Wagner *(politician)* ▯ Liberalen *(party)* ▯ hessischen *Hassian (flex.)* ▯ Westerwelle *(politician)* ▯ Wolfgang *(politician, given name)* ▯ Roland+Koch *(politician, bigram)* ▯ Wolfgang+Gerhardt *(politician, bigram)* ▯
Wetter *weather*	Grad *degrees* ▯ Temperaturen *temperatures* ▯ Regen *rain* ▯ Schnee *snow* ▯ Süden *south* ▯ Norden *north* ▯ Sonne *sunshine* ▯ Wetter *weather* ▯ Wolken *clouds* ▯ Deutschland *Germany* ▯ zwischen *between* ▯ Nacht *night* ▯ Wetterdienst *weather service* ▯ Wind *wind* ▯
Politik / Kroatien *politics / Croatia*	Parlament *parliament* ▯ Partei *party* ▯ Stimmen *votes* ▯ Mehrheit *majority* ▯ Wahlen *elections* ▯ Wahl *election* ▯ Opposition *opposition* ▯ Kroatien *Croatia* ▯ Präsident *president* ▯ Parlamentswahlen *parliamentary elections* ▯ Mesic *(politician)* ▯ Abstimmung *ballot* ▯ HDZ *(a party)* ▯
Die Grünen *Green party*	Grünen *(party)* ▯ Parteitag *party congress* ▯ Atomausstieg *nuclear energy phase-out* ▯ Trittin *(politician)* ▯ Grüne *(party members, collective)* ▯ Partei *party* ▯ Trennung *separation* ▯ Mandat *mandate* ▯ Ausstieg *withdrawal, phase-out* ▯ Amt *(political) office* ▯ Roestel *(politician)* ▯ Jahren *years* ▯ Müller *(politician)* ▯ Radcke *(politician)* ▯ Koalition *coalition* ▯
Russische Politik *Russian politics*	Russland *Russia* ▯ Putin *(politician)* ▯ Moskau *Moscow* ▯ russischen *Russian (flex.)* ▯ russische *Russian (flex.)* ▯ Jelzin *(politician)* ▯ Wladimir *(politician, given name)* ▯ Tschetschenien *Chechnya* ▯ Russlands *Russia (flex.)* ▯ Wladimir+Putin *(politician, bigram)* ▯ Kreml *kremlin* ▯ Boris *(politician, given name)* ▯ Präsidenten *president (flex.)* ▯
Polizei / Schulen *police / schools*	Polizei *police* ▯ Schulen *schools* ▯ Schüler *students* ▯ Täter *offender* ▯ Polizisten *police officer* ▯ Schule *school* ▯ Tat *offence* ▯ Lehrer *teacher* ▯ erschossen *shot dead* ▯ Beamten *officers (flex.)* ▯ Mann *man* ▯ Polizist *policeman* ▯ Beamte *officer* ▯ verletzt *injured* ▯ Waffe *weapon* ▯

Figure 3.20: Selected latent topics. LDA with $K = 100$ using word unigrams and bigrams on 20k German news messages (dpa). Bar lengths for terms are proportional to $\log \varphi_{k,t}$.

Words in topic (choose worst match (A-F) in every group):		
1. A. field	2. A. images	3. A. parallel
B. teacher	B. subjects	B. processors
C. distribution	C. theorem	C. data
D. networks	D. performance	D. block
E. probability	E. face	E. processor
F. variables	F. human	F. hierarchical
4. A. cortex	5. A. examples	6. A. model
B. recognition	B. learning	B. parameters
C. image	C. teacher	C. posterior
D. set	D. student	D. data
E. neural	E. generalization	E. predictor
F. character	F. patients	F. bayesian

(a) Word intrusion

Topics for document (choose worst match with text (A-D)):

347 *A Connectionist Learning Control Architecture for Navigation.* Adaptation to their environment is a fundamental capability for living agents, from which autonomous robots could also benefit. This work proposes a connectionist architecture, DRAMA, for dynamic control and learning of autonomous robots. DRAMA stands for dynamical recurrent associative memory architecture. It is a time-delay recurrent neural network, using Hebbian update rules. It allows learning of spatio-temporal regularities...

A. robot	B. learning	C. kernel	D. units
environment	reinforcement	support	hidden
state	model	vector	network
goal	system	margin	unit
action	time	svm	input
time	sutton	function	output
agent	state	data	weights
path	barto	training	layer

(b) Topic intrusion

Figure 3.21: Coherence experiments for a machine learning corpus (NIPS proceedings). Solutions: (a): B, C, F; D, F, E; (b): C.

As a complement to this subjective test, also automatic methods of qualitative topic evaluation have been proposed, especially to solve the problem that some topics even in a well-fitted topic model indeed may be "junk" because they contain uninterpretable relations between words or high-frequency terms not in the stop list [Steyvers & Griffiths 2006]. For instance, [AlSumait et al. 2009] verify "good" topics k based on criteria like the presence of salient words in $\vec{\varphi}_k$ (re-enacting Zipf's law), a the distance between topic–word distribution $\vec{\varphi}_k$ and averaged corpus language model, and the globality of topics over corpus documents ($\vartheta_{m,k}$ should be high for only a subset of m).

Empirically, several patterns of "junk" topics can be distinguished, for which [Mimno et al. 2011] present an error typology in conjunction with an empirical study. The patterns include "chained" semantic similarity (words are pairwise similar but don't belong to a coherent theme), "intrusion" (as above), "random" (no similarity of terms in topic) and "unbalanced" (top terms are related but specificity varies strongly). Based on these, they propose a topic coherence metric, which is reviewed below.

3.7.2 Querying the model

Querying the LDA model is the operation to retrieve documents relevant to a query document. In topic models, there are two methods to perform ranking of result documents: (1) via similarity analysis and (2) via the predictive likelihood.

A query is simply a vector of words $\vec{w}'$, and we can find matches with known documents by estimating the posterior distribution of topics $\vec{z}'$ given the word vector of the query $\vec{w}'$ and a representation of the trained model parameters $\mathcal{M} = \{\vartheta, \varphi\}$: $p(\vec{z}'|\vec{w}'; \mathcal{M})$. From here onwards, we generally use a prime $\cdot'$ to denote a query. In order to find parameters specific to a query, $\vec{\vartheta}'$, we can follow the approach of [Hofmann 1999a] or [Steyvers et al. 2004] to run inference on the new document exclusively, depending on the inference method used. In Chapter 6, we will introduce concrete querying techniques generic for all topic models. Generally, the querying approach discussed here is applicable for complete collections of unknown documents $\{\vec{w}'_m\}_m$, jointly estimating the corresponding parameters $\vec{\vartheta}'_m$ for each of the documents.

Similarity ranking. In the similarity method, the topic distribution of the query document(s) is estimated and appropriate similarity measures permit ranking. As the distribution over topics $\vec{\vartheta}'_m$ in LDA now is in the same form as the elements of $\vartheta = \{\vec{\vartheta}_m\}$ that exist for documents in the training corpus, we can compare the query to the documents of the corpus. A simple measure is the Kullback–Leibler divergence [Kullback & Leibler 1951], which is defined between two discrete random variables, X and Y, as:

$$\mathrm{KL}\{X||Y\} = \sum_{n=1}^{N} p(X{=}n)\,[\log p(X{=}n) - \log p(Y{=}n)]\,. \tag{3.38}$$

The KL divergence can be interpreted as the difference between the cross entropy between X and Y, $\mathrm{H}\{X||Y\} = -\sum_n p(X{=}n) \log p(Y{=}n)$, and the entropy of X, $\mathrm{H}\{X\} = -\sum_n p(X{=}n) \log p(X{=}n)$, i.e., it is the information that knowledge of Y adds to the knowledge of X. Thus only if both distributions X and Y are equal, the KL divergence vanishes.

However, the KL divergence is not a distance measure proper because it is not symmetric. Thus alternatively, a smoothed, symmetrised extension can be used, the Jensen–Shannon divergence:

$$\mathrm{JS}\{X||Y\} = \frac{\mathrm{KL}\{X||M\} + \mathrm{KL}\{Y||M)\}}{2} \tag{3.39}$$

with the averaged variable $M = \frac{1}{2}(X + Y)$.[31] Another alternative to the KL-divergence is provided by the squared Hellinger distance that is defined as:

$$\mathrm{H}^2\{X||Y\} = \frac{1}{2}\big\|\sqrt{X} - \sqrt{Y}\,\big\|_2^2 = \frac{1}{2}\sum_{n=1}^{N}\Big(\sqrt{p(X{=}n)} - \sqrt{p(Y{=}n)}\,\Big)^2\,. \tag{3.40}$$

For its relationship to the KL-divergence, the following holds: $\mathrm{H}^2\{X, Y\} \leq 2/\ln 2 \cdot \mathrm{KL}\{Y||X\}$.

While they differ in the robustness and sensitivities to specific properties of X and Y, both the JS-divergence and the Hellinger distance have the advantage of being true distance measures, i.e., they are always non-negative, become zero if and only if $X{=}Y$, are symmetric and observe the triangle inequality, $\mathrm{D}\{X||Y\} + \mathrm{D}\{Y||Z\} \geq \mathrm{D}\{X, Z\}$. Furthermore, the values of both distances are bounded to $\mathrm{D}\{X||Y\} \leq 1$ for any pair of distributions X and Y [Lin 1991].

Specifically, the H^2-distance becomes 1 in case $p(X)$ and $p(Y)$ have disjoint support, and this boundedness is reflected also in the fact that it is the 1-complement of the Bhattacharyya coefficient (BC): $\mathrm{H}^2\{X||Y\} = 1 - \mathrm{BC}\{X||Y\}$. The BC is often used as a similarity metric on discrete distributions, for instance in classification.

Other divergence and distance measures may be applied, and [Lee 2001] gives an overview of the "classical" ones used in natural language processing.

Predictive likelihood ranking. The second approach to ranking is to calculate a predictive likelihood that the query could be generated by a document of the corpus, which is a special case

[31] The JS-divergence can be generalised beyond distribution pairs, using the centroid to determine M.

of the query-likelihood model in information retrieval [Manning & Schütze 1999]:[32]

$$p(\vec{w}'_{m'}|\vec{w}_m) = \sum_{k=1}^{K} p(\vec{w}'_{m'}|z{=}k)p(z{=}k|\vec{w}_m) \tag{3.41}$$

$$= \sum_{k=1}^{K} \frac{p(z{=}k|\vec{w}'_{m'})p(\vec{w}'_{m'})}{p(z{=}k)} p(z{=}k|\vec{w}_m) \tag{3.42}$$

$$\propto \sum_{k=1}^{K} n_k \vartheta_{m,k} \vartheta'_{m',k} \tag{3.43}$$

where n_k is the corpus-wide total of inferred word associations to topic k. Intuitively, (3.43) is a weighted scalar product between topic vectors that emphasises the match of globally strong topics. The rationale behind the approach to generate the query from documents is that the user imagines the ideal target document for an information need and tries to formulate (or generate) a query that best matches this ideal.

Retrieval. Because query results provide a ranking over the document set, querying of topic models may be used for information retrieval. This requires some additional considerations, though. By itself, the capabilities of topic models to map semantically similar items of different literal representation (synonymy) closely in topic-space and represent multiple semantics of literals (polysemy) comes at the price that results are less precise in a literal sense (while providing larger recall). Depending on the kind of relevance expected from the query results, combination of latent-topic query results with other retrieval approaches may be necessary [Wei & Croft 2006].

Another aspect of topic-based querying is that different strategies of query construction are useful. Clearly, a Boolean approach to query construction will not suffice, but rather a strategy comparable with vector-space models can be used. More specifically, for effective retrieval queries can be constructed in a way that more and more precisely narrows down the topic distribution considered relevant, which raises issues of query refinement and expansion within interactive search processes [Baeza-Yates & Ribeiro-Neto 1999].

3.7.3 Clustering

Often it is of importance to cluster documents or terms. As mentioned above, one of the properties of the LDA model is a soft clustering of the documents and of the terms of a corpus by associating them to topics. To use this clustering information requires the evaluation of similarity, and in the previous section, the similarity between a query document and the corpus documents was computed using the Kullback–Leibler divergence. This measure can be applied to the distributions of words over topics as well as to the distribution of topics over documents in general, which reveals the internal similarity pattern of the corpus according to its latent semantic structure.

In addition to determining similarities, the evaluation of clustering quality is of particular interest for topic models like LDA. In principle, evaluation can be done by subjective judgement of the estimated word and document similarities. A more objective evaluation, however, is the

[32] Here Bayes' rule is used, where the query likelihood is independent of the result document and the unconditional topic probabilities are set to $p(z{=}k) = n_k / \sum_k n_k$.

comparison of the estimated model to an a priori categorisation for a given corpus as a reference. Among the different methods to compare clusterings, we will show the variation of information distance (VI-distance) that is able to calculate the distance between soft or hard clusterings of arbitrary numbers of classes and therefore provides maximum flexibility of application.

The VI-distance measure has been proposed in [Meila 2003], and it assumes two distributions over classes for each document: $p(c{=}j|d_m)$ and $p(z{=}k|d_m)$ with class labels (or topics) $j \in [1, J]$ and $k \in [1, K]$. Averaging over the corpus yields the class probabilities $p(c{=}j) = 1/M \sum_m p(c{=}j|d_m)$ and $p(z{=}k) = 1/M \sum_m p(z{=}k|d_m)$.

Similar clusterings tend to have co-occurring pairs $(c{=}j, z{=}k)$ of high probability $p(\cdot|d_m)$. Conversely, dissimilarity corresponds to independence of the class distributions for all documents, i.e., $p(c{=}j, z{=}k) = p(c{=}j)p(z{=}k)$. To find the degree of similarity, we can now apply the Kullback–Leibler divergence between the real distribution and the distribution that assumes independence. In information theory, this corresponds to the mutual information of the random variables C and Z that describe the event of observing classes with documents in the two clusterings [Meila 2003, Heinrich et al. 2005b]:

$$\begin{aligned} \mathrm{I}\{C, Z\} &= \mathrm{KL}\{p(c, z) \| p(c)p(z)\} \\ &= \sum_{j=1}^{J} \sum_{k=1}^{K} p(c{=}j, z{=}k)[\log_2 p(c{=}j, z{=}k) - \log_2 p(c{=}j)p(z{=}k)] \end{aligned} \tag{3.44}$$

where the joint probability refers the corpus-wide average co-occurrence of class pairs in documents, $p(c{=}j, z{=}k) = \frac{1}{M} \sum_{m=1}^{M} p(c{=}j|d_m)p(z{=}k|d_m)$.

The mutual information between two random variables becomes 0 for independent variables. Furthermore, $\mathrm{I}\{C; Z\} \leq \min\{\mathrm{H}\{C\}, \mathrm{H}\{Z\}\}$ where $H\{C\} = -\sum_{j=1}^{J} p(c{=}j) \log_2 p(c{=}j)$ is the entropy of random variable C. This inequality becomes an equality $\mathrm{I}\{C, Z\}{=}\mathrm{H}\{C\}{=}\mathrm{H}\{Z\}$ if and only if the two clusterings are equal. In [Meila 2003], these properties were used to define the Variation of Information cluster distance measure:

$$\mathrm{VI}\{C, Z\} = \mathrm{H}\{C\} + \mathrm{H}\{Z\} - 2\mathrm{I}\{C; Z\}\,.$$

Like the JS-divergence and Hellinger distance, the $\mathrm{VI}\{C, Z\}$ distance is a true distance measure [Meila 2003]. Moreover, the VI distance only depends on the proportions of cluster associations with data items, i.e., it is invariant to the absolute numbers of data items.

An application of the VI distance to LDA has been shown in [Heinrich et al. 2005b], where the document–topic associations ϑ of a corpus of 20,000 news stories are compared to IPTC subject categories assigned manually to them.

3.7.4 Measuring effectiveness

In order to compare theoretical and experimental results, the effectiveness (or quality) of the models resulting from different approaches needs to be measured. A common criterion of clustering quality that does not require a priori categorisations is the likelihood of held-out data under the trained model, $p(\vec{w}'|\mathcal{M})$, i.e., the ability of a model to generalise to the unseen data.

Predictive likelihood and token perplexity. The predictive likelihood of a word vector can in principle be calculated by integrating out all parameters from the joint distribution of the word observations in a document. For LDA, the likelihood of a text document of the test corpus $p(\vec{w}'_m|\mathcal{M})$ can be directly expressed as a function of the multinomial parameters:

$$p(\vec{w}'_m|\mathcal{M}) = \prod_{n=1}^{N'_m} \sum_{k=1}^{K} p(w_n{=}t|z_n{=}k) \cdot p(z_n{=}k|d{=}m) \tag{3.45}$$

$$= \prod_{t=1}^{V} \left(\sum_{k=1}^{K} \varphi_{k,t} \cdot \vartheta_{m,k} \right)^{n_{m,t}} \tag{3.46}$$

$$\log p(\vec{w}'_m|\mathcal{M}) = \sum_{t=1}^{V} n_{m,t} \log \left(\sum_{k=1}^{K} \varphi_{k,t} \cdot \vartheta_{m,k} \right) \tag{3.47}$$

where $n_{m,t}$ is the number of times term t has been observed in document m. Note that $\vec{\vartheta}'_m$ needs to be derived by querying the model, see Section 3.7.2.

These log likelihood values are usually large negative numbers. Therefore, often perplexity is used, originally a notion from language modelling [Azzopardi et al. 2003]. Perplexity is defined as the inverse geometric mean of the token (log) likelihoods given the model $\mathcal{M}$:

$$\mathsf{P}(\vec{w}'|\mathcal{M}) = \prod_{m=1}^{M'} p(\vec{w}'_m|\mathcal{M})^{-\frac{1}{N'}} = \exp - \frac{\sum_{m=1}^{M'} \log p(\vec{w}'_m|\mathcal{M})}{\sum_{m=1}^{M'} N'_m} \,. \tag{3.48}$$

This quantity can be intuitively interpreted as the expected size of a vocabulary with uniform word distribution that the model $\mathcal{M}$ would need to generate a token of the test data. A model that better captures co-occurrences by topics requires fewer possibilities to choose tokens given their context (document etc.).

The common method to evaluate perplexity in topic models is to hold out test data from the training corpus and then test the estimated model on the held-out data.[33] Lower perplexity values indicate that the trained topics generalise well to the test documents.

Document completion. Because held-out perplexity or likelihood directly are trained on the model, it may adapt to the held-out documents while training parameters $\vec{\vartheta}'_m$ on observations $\vec{w}'_m$. This may be used to measure how low a portion of a held-out document needs to be analysed by the model during training to predict the rest of that document [Rosen-Zvi et al. 2004]. The portion of documents analysed is a free parameter, but in standard situations, a random 50% of the word

[33] This is often enhanced by cross-validation, where mutually exclusive subsets of the corpus are used as held-out data and the results averaged.

tokens are typical. The documents in held-out data are split into a set of analysed and predicted tokens: $\vec{w}'_m = \{\vec{w}^{\mathrm{a}}_m, \vec{w}^{\mathrm{p}}_m\}$. The model then is trained using the union set of training documents and analysed held-out tokens, $\vec{w}_{\mathrm{train}} = \vec{w} \cup \vec{w}^{\mathrm{a}}$, and the held-out likelihood or perplexity (see above) is determined for the predicted held-out data, $\vec{w}_{\mathrm{predict}} = \vec{w}^{\mathrm{p}}$.

Empirical likelihood. Often, if several estimates of the posterior are to be averaged, this is done using the harmonic mean [Griffiths & Steyvers 2004], which has been criticised to produce unstable results [Li & McCallum 2006, Wallach et al. 2009b]. As an alternative to harmonic mean parameter estimation, the likelihood of test documents can be computed in using an empirical likelihood estimate, which especially with more complex models produces more stable results due to its guaranteed finite variance [Li & McCallum 2006]. Such an empirical distribution of the test data set can be computed from data sampled using the generative model with the trained parameters and hyperparameters, see Fig. 3.14. For the particular case of LDA, first the corpus-wide model parameters α and φ are used to generate $\tilde{M}$ documents $\tilde{\vec{w}} = \{\tilde{\vec{w}}_{\tilde{m}}\}_{\tilde{m}=1}^{\tilde{M}}$, which are normalised to multinomial unigram models $\vec{\varrho}_{\tilde{m}}$ over the observable vocabulary. From this, the log empirical likelihood of held-out test documents $\vec{w}' = \{\vec{w}'_m\}_{m=1}^{M'}$ averaged over each sampled unigram model can be determined:

$$\log p_{\mathrm{EL}}(\vec{w}'|\mathcal{M}) \approx \sum_m \log[\tilde{M}^{-1} \sum_{\tilde{m}} \prod_n \mathrm{Mult}(w_{m,n}|\vec{\varrho}_{\tilde{m}})] \,, \tag{3.49}$$

which is a measure of the model's capability to explain tokens of unseen data. One side-effect that comes with the unconditional nature of the sampling process described above is that empirical likelihood does not give the model a chance to adapt to the test document, i.e., by seeing the first half of a test document, the second half may be already better predicted than random.

Particle filtering. From a similar motivation for a robust estimator of $p(\vec{w}'|\mathcal{M})$ derives the "left-to-right" (LTR) particle filtering method from [Wallach et al. 2009b]. This method estimates R particles each of which samples the probability of a token $p(w'_{m,n}|\mathcal{M})$ via the probability of document parts $\vec{w}'_{m,<n}$ left of it:

$$\begin{aligned} \log p(\vec{w}'|\mathcal{M}) &\approx \sum_m \sum_n \log p(w'_{m,n}|\vec{w}'_{m,<n}, \mathcal{M}) \\ &= \sum_m \sum_n \log \sum_{\vec{z}_{m,\leq n}} p(w'_{m,n}, \vec{z}'_{m,\leq n}|\vec{w}'_{m,<n}, \mathcal{M}) \,, \end{aligned} \tag{3.50}$$

which is a sum over all possible states $\{z_{m,i}{=}k\}_{i\leq n}$ that may be approximated using Gibbs sampling as described in Chapter 6 and averaged over the R sampled particles.

Topic coherence. While the previous methods use probabilistic and information-theoretic notions to define measures of model quality, one can also try to use the heuristics of human judgements to evaluate the quality of topics. This has been recently done by [Mimno et al. 2011], automating the task of effort-intensive human evaluation: They argue that cases of "chained", "intruded" and "random" topic content (as described in Section 3.7.1) may be characterised in terms of co-occurrence patterns and propose the following topic coherence measure for a topic k that is closely related to point-wise mutual information [Manning & Schütze 1999]: Given rankings $t(r)$ of the strongest R terms according to their mass in the topics distribution $\vec{\varphi}_k$, as well as the

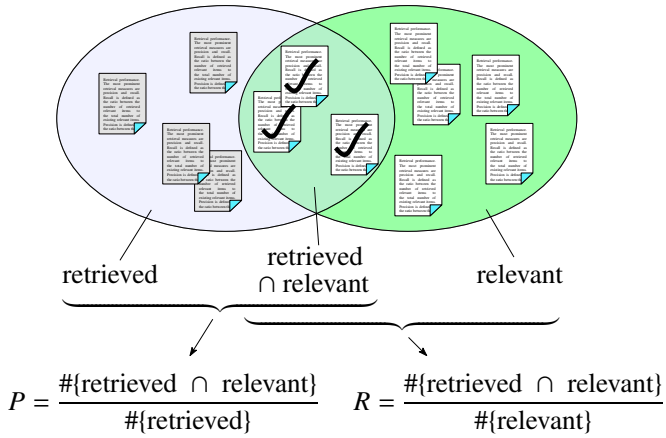


Figure 3.22: Retrieval evaluation.

document frequencies df(·) of single terms and co-occuring pairs, the coherence of the topic is estimated by:

$$\mathrm{TC}(\vec{\varphi}_k, \vec{w}, R) = \sum_{r=2}^{R} \sum_{r'=1}^{r-1} \log \frac{\mathrm{df}(t(r), t(r')) + 1}{\mathrm{df}(t(r'))} \,. \tag{3.51}$$

For this measure, a good correlation with human judgements has been shown, including topic intrusion measurements [Chang et al. 2009] discussed above, notably without performing query sampling on held-out documents and without the need for expert raters.

Retrieval performance. Other standard quality metrics view topic models as information retrieval approaches, which requires that it be possible to score items for a given query, i.e., an unknown document (see Sec. 3.7.2). The most prominent retrieval measures are precision and recall [Baeza-Yates & Ribeiro-Neto 1999], which are defined on unranked result sets. Recall is defined as the ratio between the number of retrieved relevant items and the total number of existing relevant items; see Fig. 3.22. Precision is defined as the ratio between the number of relevant items and the total of retrieved items. The goal is to maximise both, but commonly they have antagonistic behaviour, i.e., trying to increase recall will reduce precision. To compare different systems, combinations of precision P and recall R metrics have been developed, such as the F score [van Rijsbergen 1979], $F = 2PR/(P+R)$, which can also be generalised to a weighted F score, $F_\beta = (1+\beta^2)PR/(\beta^2 P + R)$ where $\beta \in [0, \infty)$ and $\beta < 1$ amplifies the importance of precision. A direct relation between precision and recall to perplexity and language models has been given in [Azzopardi et al. 2003].

Set-based measures like precision and recall require definition of a threshold score, which often is arbitrary and difficult to justify. This may be avoided by iterating the ranked list of results for all relevant documents in the corpus, truncating it at each relevant document and computing the precision at that rank, the precision at rank k, P@k. If these precisions are plotted over the recall level of each results rank (the portion of relevant documents in the corpus at the rank), a precision–recall (PR) curve is obtained whose location in the PR-space is informative of the

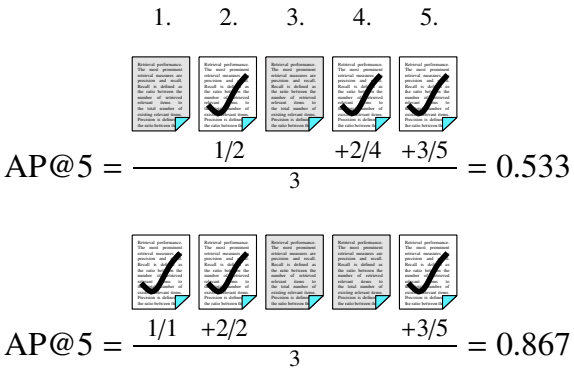


Figure 3.23: Average precision at 5 (3 relevant documents in corpus).

trade-off between both [Manning & Schütze 1999]. An alternative to iteration over all relevant documents is interpolation, commonly at 10% intervals of recall.[34]

To characterise such *PR*-curves numerically, the area under them may be estimated by using the average precision (AP) measure. In AP evaluation, the iterated precision values are averaged for all relevant documents in the corpus. Iterating up to a cutoff rank k leads to the AP at rank k measure, AP@k. Fig. 3.23 shows a numerical example, illustrating how irrelevant documents at higher ranks in the first list are penalised. By definition, AP values are normalised with the total number of relevant documents R to allow averaging between queries, and if truncated, $\min(R, k)$ is used.

Typically, a larger number of queries need to be evaluated to obtain stable, dependable results: The mean average precision (MAP) creates the mean of the AP values for different queries. A simpler alternative that empirically correlates with MAP is R-precision (RP), the precision at rank R where R is the number of relevant documents for the query. It corresponds to the break-even point in the *PR*-curve where precision equals recall.

However, one of the major issues with these evaluation measures is the availability of relevance judgements for the datasets in sufficiently many queries to estimate recall. This requires substantial human intervention. An approach to alleviate this is to perform pooling of query results, that is, running queries on a number of reference retrieval systems, merging the top results lists into a "pool" and manually selecting the items relevant for the query.

In order to deal with the recall problem or specifics of the evaluation scenario, retrieval measures may be customised by varying the averaging process in AP and MAP, the notion of recall or relevance, etc. For such tailored ranking measures based on precision averaging, see for instance [Witschel, Holz, Heinrich & Teresniak 2008] or [Antulov-Fantulin et al. 2011]. If relevance scores are graded Likert-type scales, e.g., $\{0, ..., 3\}$, or on a real interval, e.g., $[0, 1]$, the normalised discounted cumulative gain (NDCG) [Jarvelin & Kekalainen 2002] or similar measures come into play.

[34] Note that the uninterpolated version takes recall into account only implicitly – by setting the precision of irrelevant ranks to zero.

3.8 Conclusions

In this chapter, we have laid out the foundations for the contributions made in the subsequent parts of this thesis. One of the rationales in this thesis is to transfer the phenomena of latent semantic analysis to modalities other than the text content of documents. This may effectively allow tailoring of models to the assumed semantic and logic structures in the data and train them according to the co-occurrences actually found. For this purpose, in the next chapters a framework of topic models is developed including the derivation of generic inference methods.

Chapter 4

A generic approach to topic models: Networks of mixed membership

This chapter develops a generic model of topic models. To define the problem space, general characteristics for this class of models are derived, which give rise to a representation of topic models as "networks of mixed membership" (NoMMs), a domain-specific compact alternative to Bayesian networks.[1]

4.1 Introduction

In the previous chapter, the concept of latent topics has been reviewed with the example model of latent Dirichlet allocation (LDA [Blei et al. 2003b]). The idea to use co-occurrences to extract semantic relationships can be extended beyond applications where these co-occurrences refer to words in documents, extending the concept of latent topics as collections of semantically similar units beyond the rather linguistic viewpoint of the original approach. This has already been done in the literature for various applications in text mining, computer vision, bioinformatics, social network analysis and other fields. Following the idea proposed by the seminal work on LDA, which had been discussed in the previous chapter, these topic models also exploit the conjugacy of Dirichlet and multinomial/discrete distributions to learn discrete latent variables from discrete co-occurrence data (e.g., [Steyvers et al. 2004, Li & McCallum 2006, Mimno et al. 2007]) or from the co-occurrence of discrete and continuous features (e.g., [Barnard et al. 2003]).

However, what is missing in previous work is a clear statement of what constitutes this larger class of topic models and what properties this implies that may simplify construction and analysis of models and their implementations. This missing "framework" is the main goal of this chapter.

One method to find common properties in topic models derives from the observation that all applications may be seen as a form of "higher-order mixture models". Generally, mixture models [McLachlan & Peel 2000, Titterington et al. 1985, Everitt & Hand 1981] are a powerful tool to model complex probabilistic distributions by convex sums of component densities, $p(x) = \sum_k p(z{=}k)p(x|\vartheta_k)$, where z is an index variable that indicates which component k the observation x originates from. Among the large class of such models, mixture models with discrete component

[1] A major portion of this chapter corresponds to the first part of the paper [Heinrich 2009a].

densities $p(x|\vartheta_k)$ are of particular interest because in this case the component densities can serve as weighting functions for other mixtures, which themselves can have again discrete or non-discrete component densities. This fact makes it possible to construct models that consist of cascades or even networks of coupled discrete mixtures as generative structure underlying one or more observable mixtures with arbitrary (e.g., discrete or Gaussian) component densities.

Such coupling of mixtures can be considered a defining characteristic of topic models. Via the interrelation of the latent variables across different mixture levels, structures assumed in the data can be accounted for in specialised topic models, which renders the topic model approach a powerful and flexible framework. But as stated above, the published work on topic models only defines this framework implicitly; authors tend to analyse and derive probabilistic properties and inference algorithms on a model-specific basis, typically deriving models from scratch or by differences to particular prior work. Although, on the other hand, frameworks for automatic inference in (more general) Bayesian networks exist that are in principle capable of handling topic models as special cases (e.g., WinBUGS [Lunn et al. 2000], HBC [Daumé 2007], AutoBayes [Gray et al. 2002], or VIBES [Winn 2004]), the generality of this software makes it difficult (1) to gain insights from the result of the automatic inference derivation process, and (2) to make performance improvements that may be possible for more restricted model structures, which is desirable especially because topic models may have scalability issues. Such improvements may be based on the recent advances in massively parallel hardware, along with general-purpose programming platforms like OpenCL [Khronos OpenCL Working Group 2008], or heterogeneous computing architectures including specialised FPGA processor designs, along with programming interfaces like the hArtes toolchain [Rashid et al. 2009, Heinrich et al. 2011]. For such high-performance computing architectures, a generic approach to topic model inference may permit reuse of highly optimised kernels across models and therefore to focus optimisation effort.

Chapter outline. Apart from theoretical interest, these practical considerations motivate a closer look on topic models with the intent to characterise their properties in a generic manner. Specifically, we characterise topic models by their common structures in Section 4.2. Motivated by this general characterisation, we propose a specialised representation of topic models in Section 4.3: networks of mixed membership (NoMMs), for which example models are discussed in Section 4.4. In Section 4.5, general properties of the new representation are derived, and expressing topic models as NoMMs, the inference problem is formulated in Section 4.6 as a basis for the subsequent chapters.

4.2 Generalising topic models

In this section, we present a generic characterisation to topic models. As a basis for the following derivations, consider an arbitrary Bayesian network (BN, see Section 3.4.3) with variables $U_n \in U$. Its likelihood can be generally formulated as:

$$p(U) = \prod_n p(U_n \,|\, \mathrm{pa}(U_n)) \tag{4.1}$$

where the operator $\mathrm{pa}(U_n)$ refers to the set of parents of some BN node that belongs to variable U_n. The joint probability of the model is the product of all variables given their dependencies (i.e., parent nodes).

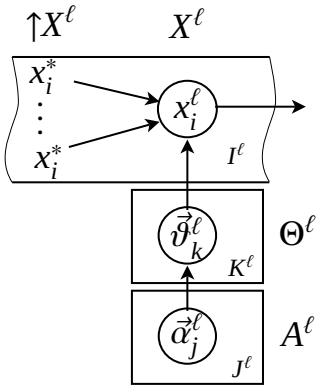


Figure 4.1: Bayesian network of a single mixture level. Superscript $\cdot^*$ used as placeholder for multiple parent variables.

Characteristics. As has been outlined in Section 4.1, the first notable characteristic of topic models is their use of the conjugate Dirichlet and multinomial/discrete distributions. Focussing on discrete observations, such models can be structured entirely into "mixture levels", each of which consists of a set of multinomial components $\vec{\vartheta}_k \in \Theta \triangleq \{\vec{\vartheta}_k\}_{k=1}^K$ that are themselves drawn from Dirichlet priors with some set of hyperparameters $\vec{\alpha}_j \in A \triangleq \{\vec{\alpha}_j\}_{j=1}^J$. Based on one or more discrete values from parent nodes in the BN, a component k among the multinomial mixture is chosen and a discrete value x_i sampled from it, which is part of the observation sequence $X \triangleq \{x_i\}_{i\in I}$ with index sequence I. The corresponding generative process for one mixture level can be summarised as:

$$\begin{aligned} x_i \,|\, \vec{\vartheta}_k, k{=}g(\uparrow x_i, i) &\sim \mathrm{Mult}(x_i|\Theta, \uparrow x_i) \\ \vec{\vartheta}_k \,|\, \vec{\alpha}_j, j{=}f(\uparrow X) &\sim \mathrm{Dir}(\vec{\vartheta}_k|A, \uparrow X) \end{aligned} \tag{4.2}$$

where the component index k is some function of the incoming discrete values or their indices that maps to components of the local mixture level. For this, the parent variable operator $\uparrow x_i$ is introduced that collects all parent variables of x_i (excluding parameters: $\uparrow X \triangleq \mathrm{pa}(X)\backslash\Theta$), and the component selection function can consequently be expressed as $k = g(\uparrow x_i, i)$. Hyperparameter indices j can be chosen either to be global for all k, i.e., $j \equiv 1$, or, similarly to the component indices, assigned to a group of components with some grouping function $j = f(\uparrow X)$.[2] This grouping can be used to model clustering among components (see, e.g., [Shafiei & Milios 2006, Li & McCallum 2006]).

The generative process in (4.2) reveals the second characteristic of topic models: Mixture levels are solely connected via discrete parent variables ($\uparrow X$), which ensures a simple form of the joint likelihood of the model.[3] Based on (4.1), the complete topic model can be constructed from

[2] For j, the requirement exists that $\uparrow x$ must be known when sampling $\vec{\vartheta}_k$ for the particular level according to the generative process in (4.2). This is symbolised by making it dependent on the complete set of parent variables, $\uparrow X$.

[3] With the hyperparameters dependent on $j = f(\cdot)$, formally there is an additional dependency between mixture levels, but this is dropped by assuming the set A known; common EM-type inference methods estimate hyperparameters independently inside their M-step; see Section 4.6.

i^ℓ, I^ℓ	sequence index and sequence range at level ℓ.
k^ℓ, K^ℓ	index and total number of mixture components.
j^ℓ, J^ℓ	index and total number of parameter groups.
$\vec{\alpha}^\ell_j, A^\ell$	hyperparameter vector and set of all hyperparameters on level (α can be scalar if $j \equiv 1$).
$\vec{\vartheta}_k, \Theta^\ell$	multinomial component parameters and set of all parameters on the level.
x^ℓ_i	variable token at index i on level.
$x^*_i, \uparrow X^\ell$	here: any parent variable token at index i and set of all parent variables of level.

Figure 4.2: Quantities in generic mixture levels, cf. Fig. 3.15.

the mixture levels $\ell \in L$, yielding the likelihood:

$$p(X, \Theta|A) = \prod_{\ell \in L} p(X^\ell, \Theta^\ell \,|\, A^\ell; \uparrow X^\ell) \tag{4.3}$$

$$= \prod_{\ell \in L} \left[\prod_{i \in I} \mathrm{Mult}(x_i \,|\, \Theta, \uparrow x_i) \prod_{k=1}^{K} \mathrm{Dir}(\vec{\vartheta}_k \,|\, A, \uparrow X) \right]^{[\ell]} . \tag{4.4}$$

For simplicity, we mark up variables specific to a level with a superscript ℓ, and use brackets $[\cdot]^{[\ell]}$ to group variables into a level. For entire equations we specify the relevant level in the text. Without mark-up or explanations, symbols are assumed to be model-wide sets of variables X, parameters Θ, hyperparameters A, etc.

The structure of the joint likelihood as given in (4.4) is interesting: Multinomial observations factorise over the sequence of tokens generated by the model, and the Dirichlet priors factorise between the components. The partial Bayesian network equivalent to a single mixture level that corresponds to this factor structure is depicted in Fig. 4.1. A summary of the quantities involved is given in Fig. 4.2.[4]

Mixture level likelihood. Due to the conjugacy of the Dirichlet and multinomial distributions, the inner terms of (4.4) can be simplified further after a transformation from tokens with index $i \in I$ (part of a sequence) to counts over component dimensions with index over $t \in [1, T]$ (part of a "vocabulary"), each specific to a mixture level ℓ. For every ℓ, the following holds for the total count of "co-occurrences" between outcomes $x_i{=}t$ and mixture components $k{=}g(\uparrow x_i, i)$ responsible for them:

$$n_{k,t} = \sum_{i \in I} \delta(k - g(\uparrow x_i, i))\, \delta(t - x_i) \tag{4.5}$$

where $\delta(x)$ is the delta function, $\delta(x) = \{1 \text{ if } x{=}0, 0 \text{ otherwise}\}$. Using these co-occurrence counts, the likelihood of one mixture level becomes:

$$p(X^\ell, \Theta^\ell \,|\, A^\ell; \uparrow X^\ell) = \left[\prod_{k=1}^{K} \frac{1}{\Delta(\vec{\alpha}_j)} \prod_{t=1}^{T} \vartheta_{k,t}^{n_{k,t}+\alpha_{j,t}-1} \right]^{[\ell]} \tag{4.6}$$

[4] For a comprehensive list of symbols and notation used in the thesis, please see Appendix A.

where $\Delta(\vec{\alpha})$ is the partition function (or normalisation constant) of the Dirichlet distribution as defined in (3.16).

Mixture level variants. The topic model framework is not restricted to the Dirichlet–multinomial type of mixture levels that provide depict the typical case. Several possibilities exist to extend the framework and plug as levels into (4.3):

- *Symmetric hyperparameters* are a common variant to the standard vectors, see, e.g., the original LDA model [Blei et al. 2003b]. The Dirichlet partition function simplifies to $\Delta_T(a)$ as defined in (3.18).
- *Observed parameters* introduce known mixture proportions or fixed observations like labels (see, e.g., the author–topic model [Steyvers et al. 2004]), which leads to

$$p(X|\Theta; \uparrow X) = \prod_i \text{Mult}(x_i|\Theta, \uparrow x_i) = \prod_{k,t} \vartheta_{k,t}^{n_{k,t}} \tag{4.7}$$

  for a level, i.e., the Dirichlet vanishes in (4.3).
- *Non-Dirichlet priors* like logistic–normal allow a more flexible combination of topics in particular mixture levels, as in the correlated topic model [Blei & Lafferty 2007].
- *Non-discrete observation components* use, e.g., Gaussian distributions in the final mixture level (e.g., in Corr-LDA [Barnard et al. 2003]).
- *Infinite mixtures* allow model adaptation to data dimensionalities and typically use Dirichlet process (DP) mixtures or generalisations [Teh & Jordan 2009]. Mixture interrelations in topic models are handled typically using hierarchical DPs [Teh et al. 2006], as outlined in Section 3.6.2.

To stay focussed, we restrict ourselves to the first two variants mentioned above, which covers a large set of models in the literature. Beyond this, in Chapter 5 a systematic discussion of mixture node variants is put in context to more general model structures.

4.3 Networks of mixed membership

The dependency structure characteristic for topic models exhibited by (4.3) gives rise to the idea of a domain-specific representation and associated graphical notation. In such a representation, the structures in the models may become more explicit than for instance in Bayesian networks where interrelations between mixtures may be occluded in complex network structures. Maintaining correspondence with the specific structure of the likelihood may moreover help simplify derivation of other model properties.

To obtain such a representation, we use the two characteristics discussed in Section 4.2: Dirichlet–multinomial mixture levels and the connections between levels via discrete variables, which can be seen as nodes and edges in a new network structure. This leads to a representation of topic models as "mixture networks" or "networks of mixed membership", abbreviated NoMM.[5]

[5] In the first publication about the representation, [Heinrich 2009a], the term "mixture network" was introduced. With more insight, "network of mixed membership" seems to capture the idea more explicitly.

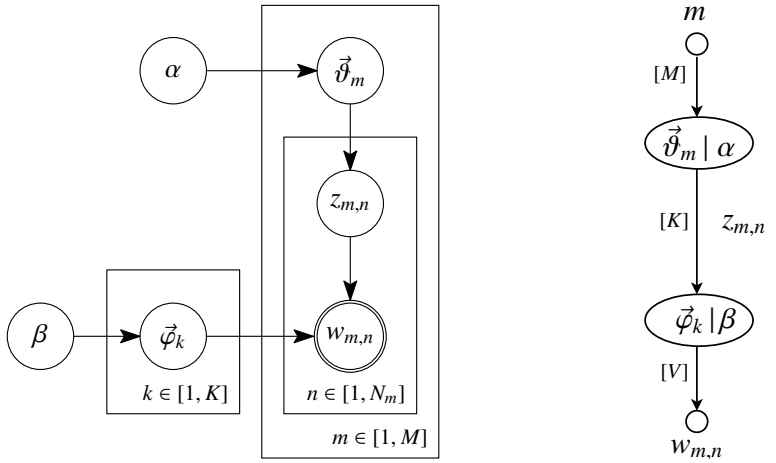


Figure 4.3: Latent Dirichlet allocation: (a) Bayesian network and (b) NoMM.

Definition. A network of mixed membership (NoMM) is a digraph $G(N, E)$. In its node set, N, an inner node represents a mixture level as described above, i.e., a sampling operation from a mixture component, and a terminal node represents a discrete observable value. Each edge in the edge set, $E : N \times N$, is directed and transmits discrete values from its parent to its child node where they control selection of component indices.

Graphical representation. In order to represent NoMMs, a dedicated graphical notation is proposed to facilitate design and analysis of model structures. Following the definition above, in Fig. 4.3, the different parts of the graphical representation are given for the example of the LDA model introduced in Section 3.6.1 along with the Bayesian network: The model has two mixture levels: a document–topic mixture $\vec{\vartheta}_m|\alpha$ and a topic–term mixture $\vec{\varphi}_k|\beta$. This is reflected in the two NoMM nodes on the right. Note that the structural parts of the NoMM correspond to the generative process defined in (4.2), and with respect to the partial BN in Fig. 4.1, the corresponding NoMM of a single mixture level is shown in Fig. 4.4.

Opposed to BNs that visualise dependencies between random variables and express data sizes (plate notation), NoMMs focus on the interrelations between discrete mixtures. In the example LDA, these are document–topic, $\vec{\vartheta}_m|\alpha$, and topic–word, $\vec{\varphi}_k|\beta$, distributions. They specifically show the dimensions of samples that "run" along edges, in LDA: $[M]$, $[K]$ and $[V]$, which are shorthands for $[1, M]$, etc.[6] The sequence of sampling a particular variable is encoded in subscripts, in the example: $\vec{\vartheta}_m$, $\vec{\varphi}_k$ and $w_{m,n}$ referring to document-, topic- and word-wise sampling schedules, respectively. Importantly, the NoMM representation strictly distinguishes model variables (that grow with the data) and model parameters (that control variable generation), representing them as edges and nodes, respectively. This is rooted in the causality assumed by generative processes: Parameters are sampled prior to variables (cf. (4.2)). In connection to this, two types of BN plates may be distinguished: On one hand, "sequence plates" run over the data points and correspond to NoMM sequence indices i^ℓ as part of the data "streamed" along the edges, x_i^ℓ.

[6] Viewing NoMMs as systems, mixture nodes can be interpreted as system blocks or filters and edges as signals.

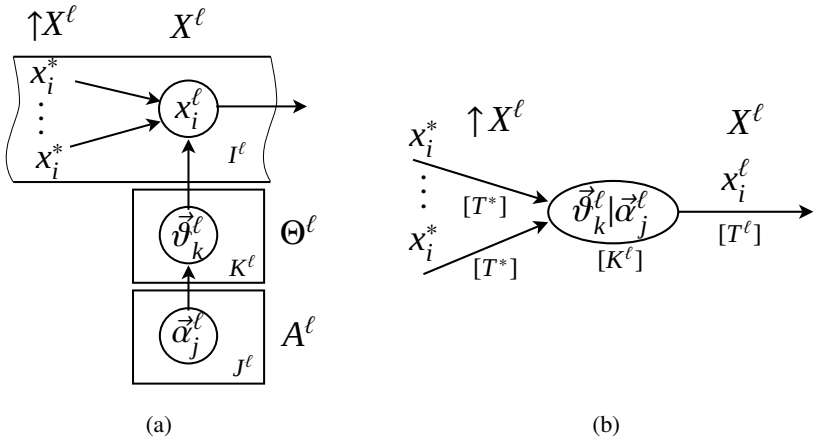


Figure 4.4: BN and NoMM notation for one mixture level.

On the other, "component plates" index mixture components. In NoMMs, they correspond to the indices k^ℓ in nodes, which depend on incoming information as arguments of component selection functions, $k^\ell = g^\ell(\uparrow x_i^\ell, i^\ell)$. The correspondence between BNs and NoMMs in Fig. 4.4 illustrates this.

In order to represent NoMMs within text or in place of equations, a simplified notation may be used. Non-terminal standard nodes are represented by parentheses, e.g., $(\vartheta_k \,|\, \alpha)$, terminal nodes just by the variable they represent, e.g., t or $w_{m,n}$, and edges are given as arrows as in the full graphical notation, e.g., $\xrightarrow[[K]]{z_{m,n}=k}$. The corresponding notation for the LDA model is therefore:

$$m \xrightarrow[[M]]{m} (\vec{\vartheta}_m \,|\, \alpha) \xrightarrow[[K]]{z_{m,n}=k} (\vec{\varphi}_k \,|\, \beta) \xrightarrow[[V]]{w_{m,n}=t} t \quad \{M, N_m\}\,, \tag{4.8}$$

which reads as follows: For a document m, a parameter $\vec{\vartheta}_m$ is chosen and generates a topic $z_{m,n}$ of vocabulary size K that indexes (as index variable k) a parameter $\vec{\varphi}_k$, which in turn generates a word $w_{m,n}$ of vocabulary size V. Eventually, a term t is observed. Optionally, the range of the sequence of the variables may be given, in the current case $\{m, n : m \in [1, M], n \in [1, N_m]\}$, for which the shorthand $\{M, N_m\}$ is used. Note that in this process, the parameters ϑ and φ are assumed to be known, but in effect they are adapted to the data, as will be presented in Part II of this thesis.

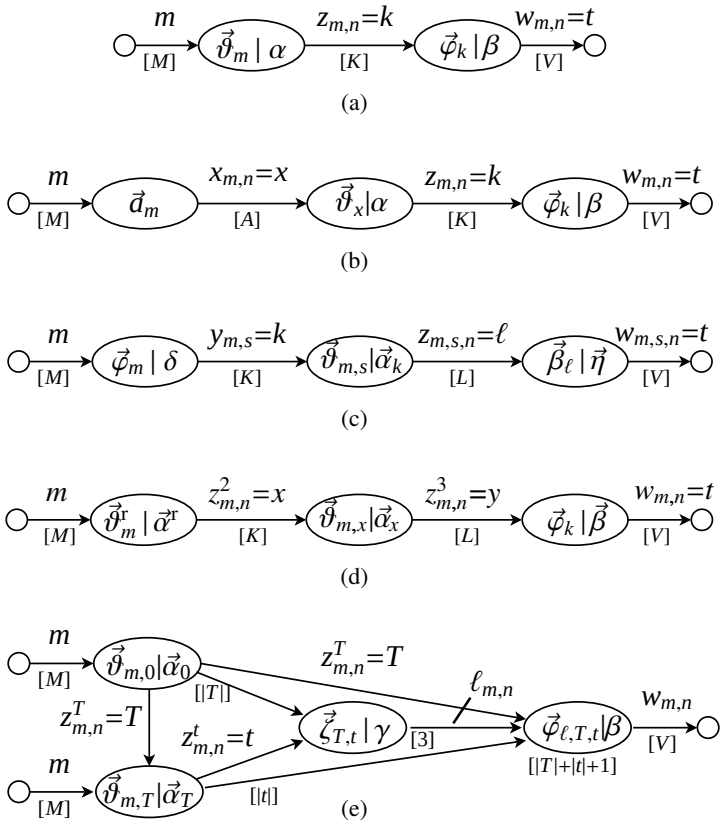


Figure 4.5: Models from the literature in NoMM notation: (a) latent Dirichlet allocation (LDA [Blei et al. 2003b], mixtures: document–topic $\vec{\vartheta}_m$, topic–term $\vec{\varphi}_k$), (b) author–topic model (ATM [Steyvers et al. 2004], additional mixture: observed document–author $\vec{d}_m$), (c) latent Dirichlet co-clustering model (LDCC [Shafiei & Milios 2006]: $\vec{\vartheta}_{m,s}$ are segment topics, adding a logical structure level between words and documents), (d) 4-level pachinko allocation (PAM4 [Li & McCallum 2006]: models semantic structure with a hierarchy of topics $\vec{\vartheta}_m$, $\vec{\vartheta}_{m,x}$, $\vec{\vartheta}_y$), (e) hierarchical pachinko allocation (hPAM [Li et al. 2007a]: also topic hierarchy; complex mixture structure).

4.4 Example models

In illustrate the applicability of the NoMM representation, the NoMM diagrams of some topic models from the literature are drawn in Fig. 4.5. Fig. 4.5(a) shows the NoMM of the model of latent Dirichlet allocation (LDA [Blei et al. 2003b]) as explained above. The extension of this model gives illustrative insight into the organisation of mixtures to account for logical and semantic structure assumed in the data.

Author–topic model. A straight-forward extension to the plain LDA model is the author–topic model (ATM [Steyvers et al. 2004]) shown in Fig. 4.6(b). The ATM models the topic association with authors using three mixture levels. Its BN is shown in Fig. 4.6(b). The levels correspond to

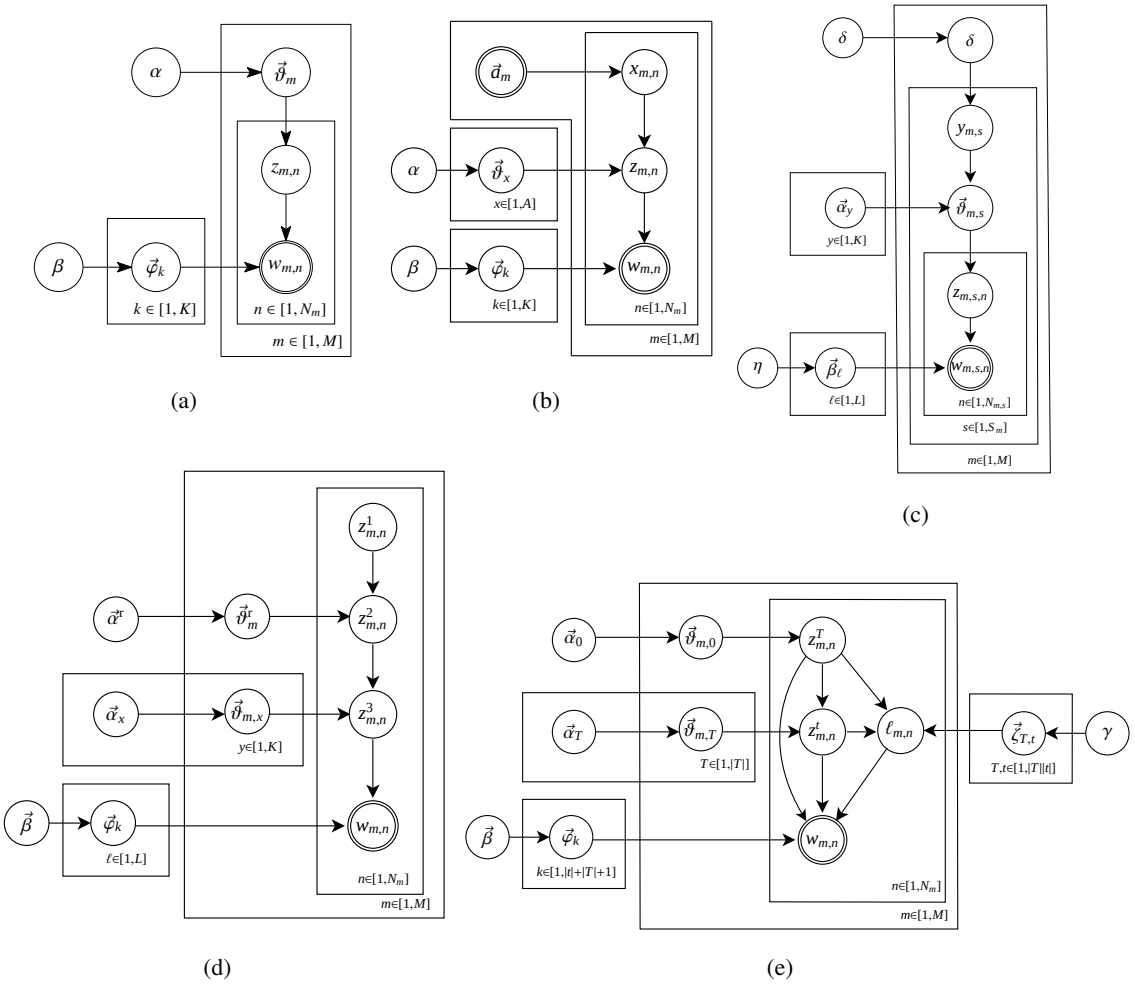


Figure 4.6: Bayesian networks for (a) LDA, (b) ATM, (c) LDCC, (d) 4-level PAM and (e) hPAM1 corresponding to NoMMs in Fig. 4.5.

distributions between documents and authors, ($\vec{a}_m$), authors and topics, ($\vec{\vartheta}_x|\alpha$), and topics and words/terms, ($\vec{\varphi}_k|\beta$). An important feature of this model is the observed parameter a_m, which in the corresponding leftmost (inner) node omits a hyperparameter (see mixture level variants in Section 4.2).

Co-clustering. The model of latent Dirichlet co-clustering (LDCC [Shafiei & Milios 2006]) in Fig. 4.5(c) uses a sequence of document segments, (m, s), and a sub-sequence of words in each segment, (m, s, n), to infer an additional logical layer of topics from the data: For each section s, a topic distribution $\vec{\vartheta}_{m,s}$ can be inferred, and document topic distributions $\vec{\varphi}_m$ index word-topics $z_{m,s,n}$ indirectly via section topics $y_{m,s}$, allowing a finer-grained handling of topic structure across documents with component selection function $g(\uparrow z_{m,s,n}, (m, s, n)) = (m, s)$. Furthermore, segment topics $\vec{\vartheta}_{m,s}$ are coupled across documents via the topic-hyperparameters $\vec{\alpha}_y$ with a component group function $f(\uparrow z) = k$.

Pachinko allocation. Another multi-level NoMM is the class of pachinko allocation models (PAM), of which the four-level variant (PAM4) is depicted in Fig. 4.5(d), as described in [Li & McCallum 2006]. For each word, a path through a topic hierarchy is sampled consisting of the indicators (z^1, z^2, z^3) where z^1=1 provides the root of the tree, associated with LDA-type document topics $\vec{\vartheta}^r_m$, and based on its sample, a document- and topic-dependent level $\vec{\vartheta}_{m,x}$ is sampled, $(g(\uparrow z^3_{m,n}, (m,n)) = (m,x))$, finally indexing word-topics $\vec{\vartheta}_y$. The topics on these levels may be referred to as super-topics and sub-topics, respectively. Similar to the LDCC model, component grouping is used: $f(\uparrow z^3) = x$. In Fig. 4.7, the process of data generation through the NoMM is illustrated.

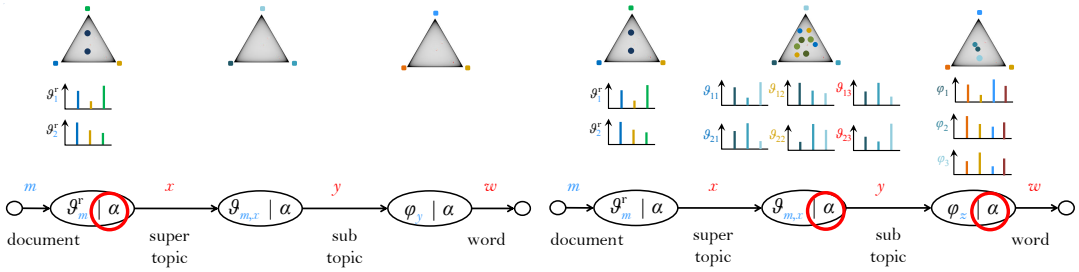
Hierarchical pachinko allocation [Mimno et al. 2007] (hPAM) as shown in Fig. 4.5(e) is an example of a more complex model that allows a hierarchy of topic–term distributions: As in PAM, the topic hierarchy consists of document-specific root- and super-, as well as global sub-topics, but each node k in the hierarchy is associated with a topic–term distribution $\vec{\varphi}_k$, and for each word $w_{m,n}$, a complete topic path (root–super–sub) is sampled along with a level $\ell_{m,n}$ from $\vec{\zeta}_{T,t}$ specific to super- and sub-topics (hPAM1 in [Mimno et al. 2007]). The topic sample on level $\ell_{m,n}$ selects from the set $k = \{1, 1+T, 1+|T|+t\}$ the component $\vec{\varphi}_k$ that finally generates the word conditioned on super- and sub-topics.

Although with a different goal in mind, the concept of pachinko allocation models is closely related to the approach pursued with NoMMs because it allows to connect different levels of mixtures with great flexibility. In fact, NoMMs can be considered a generalisation of PAMs that allows free interconnection of nodes in general DAG structures with different types of mixture levels (observed, unobserved parameters) and with observable variables (edges or parameters) at arbitrary points in the network. In effect, this covers a much larger range of real-world models. The most flexible flavour of the PAM concept, component-dependent subtrees mentioned in [Li & McCallum 2006], may be realised with NoMMs by appropriate choice of index selection functions, $g^\ell(\uparrow x^\ell_i, i^\ell)$.

In Chapter 5, more examples from the literature are given within a larger typology of topic model structures.

4.5 General properties

With NoMMs as a general perspective on topic models, there are specific properties of the NoMM nodes imposed by their interrelations via edges, and it is of interest to analyse how statistical dependency, the effect of parameters, hyperparameters, and count variables function in NoMMs generically.



(a) Sample document distributions for $m = 1, 2$.

(b) Sample all other component distributions.

(c) Document $m = 1$: select component ϑ^{r}_1.

(d) Sample from $\vartheta^{\mathrm{r}}_1 \rightarrow$ supertopic $x = 3$.

(e) Select component ϑ_{13} ($m = 1, x = 3$).

(f) Sample from $\vartheta_{13} \rightarrow$ subtopic $y = 2$.

(g) Select component φ_2.

(h) Sample from $\varphi_2 \rightarrow$ word $w = 1$.

Figure 4.7: Animating the NoMM process for pachinko allocation (PAM4).

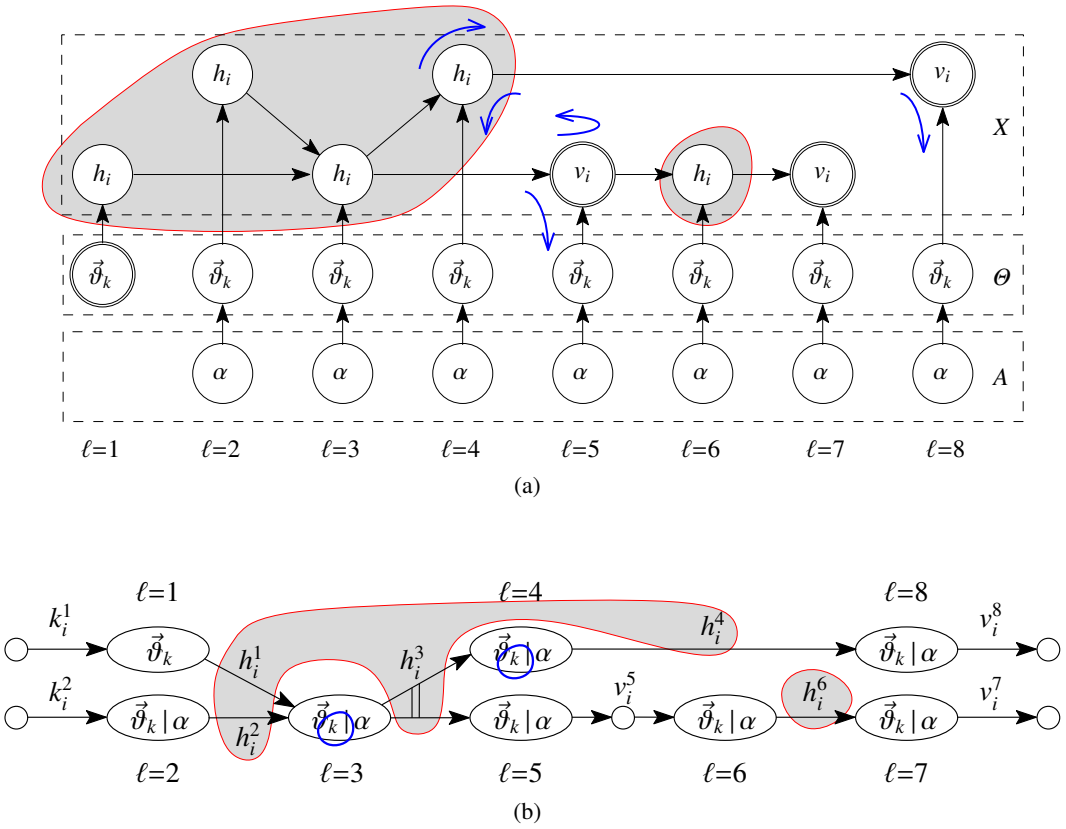


Figure 4.8: Explaining dependencies: (a) example Bayesian network (no plates shown), (b) equivalent NoMM.

4.5.1 Dependencies

In Bayesian networks, as outlined in Section 3.4.4, the rules of d-separation allow identification of conditional independence between partial networks given observations. The question is how these rules need to be transformed to apply to NoMM structures. To investigate this, a hypothetical example topic model is used, as presented in the Bayesian network in Fig. 4.8(a) that spans over eight mixture levels denoted ℓ=1, ..., 8,[7] covering various aspects encountered in topic models: observed parameters on level ℓ=1, multiple inputs and outputs on level 3, observed output edge in levels 5, 7 and 8. In Fig. 4.8(b), the equivalent NoMM structure is shown.

Remembering that mixture levels are connected solely via indicator variables, for the purpose of considering dependencies between levels, we can study the dependencies of latent variables h^ℓ separately. Generally we can assume $\Theta^\ell \perp\!\!\!\perp \Theta^j | \{H, V\}$ for $j \neq \ell$, i.e., parameters are dependent across levels only via the variables they generate according to the BN. On the other hand, variables depend on the parameters of each level ℓ are dependent: $X^\ell \not\perp\!\!\!\perp \Theta^\ell \not\perp\!\!\!\perp A^\ell$. Therefore, parameters Θ^ℓ and A^ℓ "inherit" dependencies between levels from their associated variables X^ℓ. Moreover,

[7] Levels correspond to "columns" of variables with a level index at the bottom: ℓ=1, ℓ=2, etc.

parameters generating visible variables V^ℓ are conditionally dependent on the latent variables $H^{\uparrow\ell}$ of parent levels, given the observations V^ℓ; cf. the blue Bayes ball arrows on levels 5 and 8 in Fig. 4.8(a).

Considering the latent variables h_i in the BN in the figure, it is easily verified that those of levels 1 to 4 are not only conditionally dependent but also marginally (i.e., unconditionally). The visible latent variable on level 6 blocks propagation of the dependency to the right, d-separating levels 5 and 6. This is caused solely by the visibility status of level 5, notably not by the particular way how the latent variables are connected (as long as there is an observed BN child node; cf. level 3) or whether they have observed parameters (cf. level 1). Any inference algorithm for the example network may therefore work on the two independent variable sets $\ell \in \{1, 2, 3, 4\}$ and $\ell \in \{6\}$, with the observations of adjacent levels ($\ell \in \{5, 8\}$ and $\ell \in \{7\}$) influencing the behaviour. Note that the independence would be broken if for instance level 6 fed the output level 8.

These observations may be transferred to NoMMs by considering the NoMM in Fig. 4.8(b), which is analogous to the example model. The structure of the latent variables is mapped from the Bayesian network to the edges of the NoMM, with the Dirichlet branches of the Bayesian network collapsed into the mixture nodes and terminal nodes added for observations. The output of level 3 has two connected edges h_i^3. By analogy with the respective BN structures, dependencies are determined by the indicator variables of the node parameters $\vec{\vartheta}_k$ as these are the places where incoming information is processed in a mixture level. Consequently, all hidden mixture edges are dependent that are connected via nodes with latent variables in their component selection function $k = g(\uparrow\cdot)$. In Fig. 4.8(b), this is illustrated by enclosed component indices. Analogously this also applies to hyperparameter grouping functions $j = f(\uparrow\cdot)$. However, in the latter case the dependency is weaker and in many EM-type inference algorithms hyperparameters are estimated in a separate M-step, which in practice leads to dropping this dependency in the E-step (as is done for instance in LDCC [Shafiei & Milios 2006]).

In summary, the dependency rules for NoMMs can be formulated as follows:

1. A child edge x_i^ℓ of a node $(\vec{\vartheta}_k^\ell \,|\, \vec{\alpha}_j^\ell)$ depends on all parent edges $\uparrow x_i^\ell$ whose values are used in the node's component selection functions, either for parameters, $k = g(\uparrow\cdot)$, or for hyperparameters, $j = f(\uparrow\cdot)$, if they are not inferred in a separate M-step.

2. Multiple child edges $x_{1,i}^\ell, \ldots, x_{u,i}^\ell$ and multiple parent edges $\uparrow x_i^\ell$ are conditionally dependent given node parameters Θ if they are drawn jointly, i.e., with the same sequence index i. Different tokens of any edge are conditionally independent given the parameters.[8]

3. Hidden parameters in NoMM nodes are dependent on hyperparameters and on the union set of parent and child edges.[9]

[8] This is valid under the weak assumption that component selectors contain all parent edges but not different tokens, as for instance in $k = x_i^{\mathrm{a}} + x_{i-1}^{\mathrm{b}}$.

[9] This follows from the observation made above that parameters Θ and hyperparameters A are dependent on the latent variables on a node level and that parameters of nodes with observed output are dependent on their generating incoming edges (=BN parent variables).

4.5.2 Parameters and counts

The parameters of a model define its actual behaviour. By way of (4.6), the count variables $n^{\ell}_{k,t}$ are directly connected with them: It is easy to see that the term $\vartheta^{n_{k,t}+\alpha}_{k,t}$ central to the likelihood increases if the $\vartheta_{k,t}$ are highest for those $n_{k,t}$ that are highest as well: Both are bounded (ϑ by $\sum_t \vartheta_{k,t} = 1$ and $n_{k,t}$ by $\sum_{k,t} n_{k,t} = W$, cf. (4.5)) and need to be "economical" with their mass or counts. Thus, parameters and counts qualitatively have the same influence on the model behaviour: The more mass is accumulated in particular dimensions, the more one component in a node prefers it over the other dimensions, promoting clustering. Because there are several components that may promote different dimensions, and multiple clusters interact, mixing between different variables (with different degrees of freedom) is accomplished: For example, with the values of $w_{m,n}$=t observed and corresponding $n_{k,t}$ in the output level, a "likely" model will have some components k that contain those t co-occuring often in different contexts. Consequently, the respective $\vartheta_{k,t}$ will accumulate mass.

4.5.3 Hyperparameters

Dirichlet hyperparameters generally have the effect to smooth out the influence of data on the model, as can be seen again in the term $\vartheta^{n_{k,t}+\alpha}_{k,t}$ in (4.6). The effect may be best illustrated for the model of LDA. Reducing the smoothing effect of hyperparameters by lowering the values of α and β will result in more decisive topic associations, thus ϑ and φ will become sparser. Sparsity of φ, controlled by β, means that the model prefers to assign few terms to each topic, which again may influence the number of topics that the model assumes to be inherent in the data. This is related to how "similar" words need to be (that is, how often they need to co-occur across different contexts[10]) to find themselves assigned to the same topic. For sparse topics, the model will fit better to the data if K is set higher because the model is reluctant to assign several topics to a given term. This is one reason why in non-parametric models (cf. Section 3.6.2), the dimensionality of a hidden edge (topic count) strongly depends on the hyperparameters. Sparsity of ϑ in the LDA example, controlled by α, means that the model prefers to characterise documents by few topics.

As the relationship between hyperparameters, topic number and model behaviour is a mutual one, it can be used for synthesis of models with specific properties, as well as for analysis of features inherent in the data. On the other hand, learning hyperparameters from the data can be used to increase model quality (w.r.t. to the objective of the estimation method), given the number of topics. Furthermore, hyperparameter estimates may reveal specific properties of the data set modelled. In LDA, the estimate for α is an indicator of how different documents are in terms of their (latent) semantics, and the estimate for β suggests how large the groups of commonly co-occurring words are. However, the interpretation of estimated hyperparameters is not always straight-forward, and the influence of specific constellations of document content has not yet been thoroughly investigated in the literature beyond work like [Asuncion et al. 2009] that determine the dependence of hyperparameter behaviour on different inference methods or [Wallach et al. 2009a] who work on more complex Dirichlet-based priors (see Section 5.2).

[10] Latent topics often result from higher-order co-occurrence, i.e., t_1 co-occurring with t_2 that co-occurs with t_3 represents a second-order co-occurrence between t_1 and t_3, and so on [Kontostathis & Pottenger 2006].

4.6 Posterior inference

Inference in the context of NoMMs refers to finding the parameters Θ and hyperparameters A given the observations. With the model variables X divided into sets of visible (observed) and hidden (latent) variables, $X = \{V, H\}$, this is typically a two-part process of (1) Bayesian inference for the posterior distribution,

$$p(H, \Theta|V, A) = \frac{p(V, H, \Theta|A)}{p(V|A)} , \tag{4.9}$$

and (2) estimation of the hyperparameters, for which ML or MAP estimators are commonly sufficient because of the simpler search space.

As in many latent-variable models, determining the posterior (4.9) is generally intractable in NoMMs due to excessive dependencies between the latent variables H and parameters Θ in the marginal likelihood for the observations V in the denominator,

$$p(V|A) = \sum_H \int p(V, H, \Theta|A) \, \mathrm{d}\Theta . \tag{4.10}$$

Knowing from (4.4) that the form of the integrand is a product of Dirichlet and multinomial distributions over all mixture levels, the complexity can be re-enacted: For every combination of hidden variables H (one set of hidden variables $\{h_i^\ell\}_\ell$ for every token index i in the data), a summand is necessary. This leads to an exponential number of terms, which already for the simple model of LDA is on the order of K^W with W the number of tokens and K the dimension of the (single) hidden variable.

To circumvent this intractability, approximate inference methods have been proposed, for topic models including mean-field variational Bayes [Blei et al. 2003b], collapsed variational Bayes [Teh et al. 2007], expectation propagation [Minka & Lafferty 2002] and collapsed Gibbs sampling [Griffiths & Steyvers 2004].

For our purposes, a method is needed that has feasible complexity with reasonable accuracy even when it comes to modelling dependencies between variables. The full factorisation of variational mean-field distributions may be adverse for model fitting [Dueck & Frey 2004], and structured approaches become complicated quickly [Winn 2004]. Expectation propagation on the other side has not been commonly used with more complex topic models, which may be due to its higher computational demand as shown in [Griffiths & Steyvers 2004] or its sensitivity to starting values [Minka & Lafferty 2002].[11] Thus, Gibbs sampling appears to be the most straight-forward method for a formulation of approximate inference for NoMMs and will be discussed in Chapter 6. Moreover, it will be investigated in Chapter 7 how variational inference in generic topic models compares to this.

[11] Recently, an accelerated method using expectation propagation has been proposed [Seeger & Nickisch 2011].

4.7 Conclusions

In this chapter, a generic approach to topic models has been developed based on their interpretation as "higher-order" mixtures. The relation of this mixture-network approach to existing models in the literature was presented and it was shown how the scope of models may be extended by introducing variants to the Dirichlet–multinomial distribution pair. Furthermore, a graphical notation to represent the structure of topic models was proposed, which may be seen as a domain-specific, more compact alternative to Bayesian networks.

The benefit of representing topic models as NoMMs is that the common properties of a wide variety of models can be made explicit and used to develop inference methods that may be re-used across models. This closes an apparent gap in the literature that so far considered topic models individually or with reference to particular basis models. In the next chapter, NoMMs are analysed with respect to how they map to models in the literature, developing a system of classification of such models. In Part II of this thesis, the posterior inference problem described in Section 4.6 will be investigated using approximate inference techniques.

Chapter 5

A typology of NoMM structures

The generic approach to topic models may help simplify derivation of new models. However, its potential advantages are not limited to that. Based on the state of the art in topic modelling, this chapter presents a typology of NoMM structures that serves as a classification of state-of-the-art models and as the basis for a "library" of building blocks for new models.[1]

5.1 Introduction

Besides exploring the range of NoMM structures in more detail than in Chapter 4, this chapter may be seen as an overview of the state of the art of topic models. However, it does not follow the strategy pursued in typical reviews in the literature that are organised by (and often restricted to) particular applications. In contrast, it uses the compact NoMM representation to study previous work from the viewpoint of model structure, noting that many of these model structures can be applied across application domains. The resulting typology of node and edge structures is intended both to classify important prior work and to explore the capabilities of the meta-model defined by NoMMs.

As a large body of literature exists on models that may be considered topic models and fit into the framework of NoMMs, representative examples are chosen rather than an exhaustive enumeration. With this thesis focussing on applications in community knowledge discovery, such models are prioritised that have been or may be readily applied to community scenarios, for instance such that correspond to those outlined in Chapter 2.

Chapter outline. The typology makes use of the coarse-grained "structural primitives" as a primary classification criterion, looking first at node structures in Section 5.2, then at structures where information branches from a node to several children in Section 5.3, which leads to specific edge types, and further at structures where information from several mixture node parents is merged in Section 5.4, which leads to specific ways of component selection in a mixture node. Finally, the specifics of non-parametric models are analysed in Section 5.5, which is done in a separate section because of the various implications of such an extension. The types of structures are labelled according to a simple nomenclature: the class of structure, followed by a running

[1] The systematics of this typology have been published as part of [Heinrich 2011b].

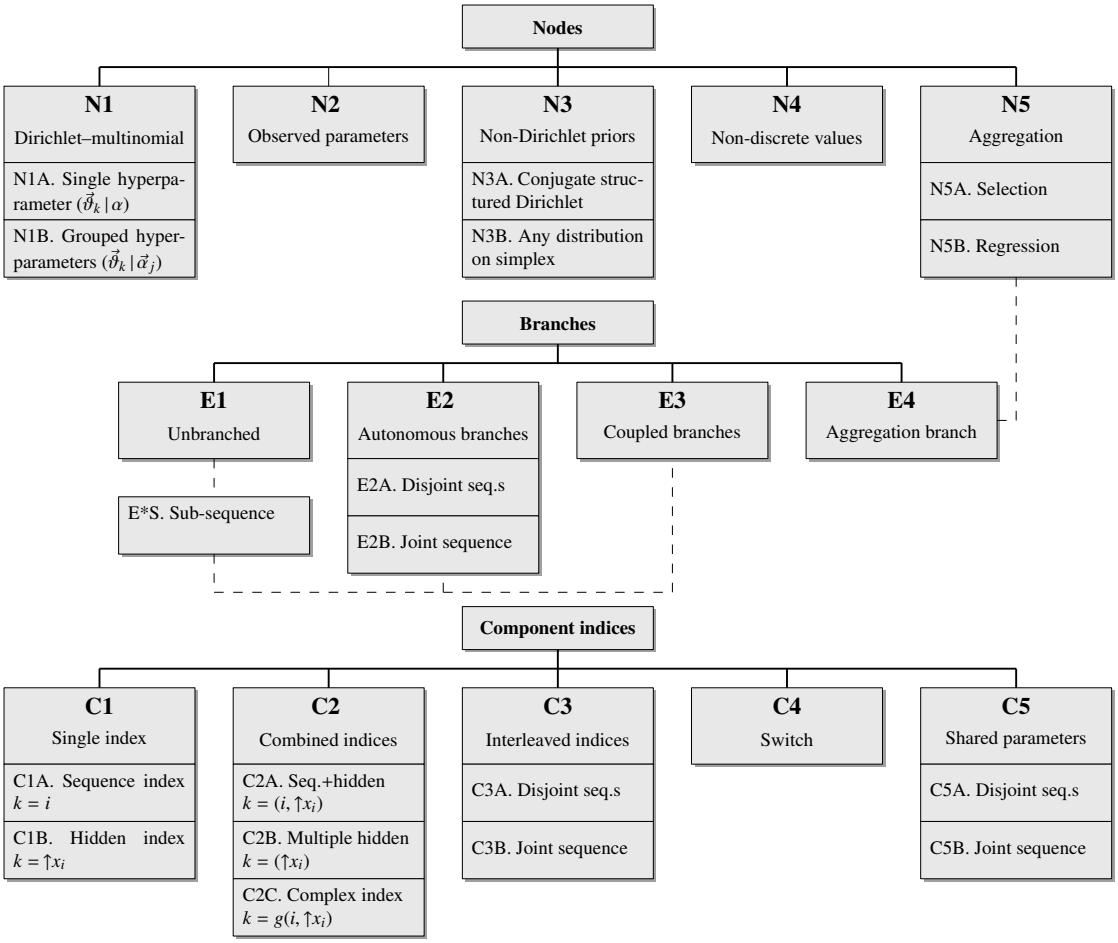


Figure 5.1: Overview of the NoMM structure typology.

number. By developing this typology, more specific node and edge structures are introduced based on the NoMM representation presented in Section 4.3. An overview of the typology is given in Fig. 5.1.

5.2 Mixture node structures

With nodes as the first structure class, we deepen our analysis of NoMM level variants outlined in Chapter 4. We consider mixture node structures for the case of a single input and output, extending this to more complex network topologies afterwards. We analyse nodes together with an outgoing edge (mixture levels), which corresponds to a group of dependent nodes in the equivalent Bayesian network consisting of the set of multinomial variables (outgoing edge) and its Dirichlet prior with parameters (node), cf. Fig. 4.8. Not considering any specific topologies that nodes are embedded in, different node types can be distinguished, starting with the standard case of Dirichlet–multinomial nodes with a hidden output, such with observed output, such with observed parameters, and then varying the prior and output distributions.

All of these types have in common that they generate exchangeable values on outgoing edges, and the co-occurrence between input and output "signals" affects the parameters, according to the clustering property of the Dirichlet or other clustering properties for alternative distributions.

5.2.1 N1: Dirichlet–multinomial nodes

Dirichlet–multinomial nodes are the standard case in admixture models. To each data group (document etc.) in an admixture model, one or more components are associated. Ordinary mixture models on the other hand are covered by the special case of a single component $\vec{\vartheta}$ (cf. Section 4.3), from which document-specific labels z_m are sampled, both using N1 nodes:

$$1 \rightarrow \boxed{(\vec{\vartheta}\,|\,\alpha)}_{\text{N1}} \xrightarrow[[K]]{z_m=k} \boxed{(\vec{\varphi}_k\,|\,\beta)}_{\text{N1}} \xrightarrow[[V]]{w_{m,n}} t\ . \tag{5.1}$$

This corresponds to the multinomial mixture model in Section 3.5.2.[2] In LDA [Blei et al. 2003b] as the basic Bayesian admixture model, the left N1 node is extended to document-specific components and per-word topics:[3]

$$m \xrightarrow[[M]]{m} \boxed{(\vec{\vartheta}_m\,|\,\alpha)}_{\text{N1}} \xrightarrow[[K]]{z_{m,n}=k} \boxed{(\vec{\varphi}_k\,|\,\beta)}_{\text{N1}} \xrightarrow[[V]]{w_{m,n}} t\ . \tag{5.2}$$

Hyperparameters in N1 nodes can be scalars or vectors. If set a priori, scalar hyperparameters can be used to control the dispersion of groups into topics (cf. Section 4.5.3), and vector hyperparameters additionally to influence the weighting of the different dimensions of the output edge. If estimated, the ratio of vector parameters re-enacts that of the mean ratio between dimensions of the different components generated using these parameters. Because hyperparameters again influence the parameters, grouping components into sets, each of which with a different vector hyperparameter, allows an additional dimension of clustering of components, especially if the groups are learnt from the data. An example for this has been given with the PAM4 model in Fig. 4.5(d). N1 nodes with parameter grouping can be referred to as a subtype N1B, whereas those with a single hyperparameter (scalar or vector) are named N1A.

Another variant is an empty hyperparameter, $(\vec{\vartheta}_m\,|\,)$, which indicates a maximum likelihood or other prior-less estimator for the parameter. In contrast, a $\cdot$ as hyperparameter, $(\vec{\vartheta}\,|\,\cdot)$, assumes Bayesian inference with some unspecified hyperparameter.

Nodes with observed edges are used to represent known data. In this typology, we do not consider terminal nodes themselves, which have only indicator character, but the closest non-terminal nodes along with their terminal edges.

Observed input edges typically correspond to those plates in the corresponding BN that cover sequences of exchangeable data points or groups (e.g., word and document plates), and NoMM nodes with the incoming edges indexing components define clusterings of groups in the

[2] Note that word-wise sampling of topic labels, $z_{m,n}$, would not work in (5.1) as the context information is lost if every word in different documents is drawn from the same global distribution $\vec{\vartheta}$.

[3] Here the context is given by the document-specific topic distributions, and $z_{m,n}$ can cluster differently for each document.

observed data. For the LDA model for example, the input edge m indexes components $\vec{\vartheta}_m$ as document-specific distributions over topics.

Observed output edges represent the actually observed data, and co-occurrences in these are "transported" into the parents nodes of such edges. Considering this for LDA, as shown in (5.2), the terminal nodes correspond to the document index m and the term t observed for word $w_{m,n}$. The grouping by documents in the left N1 node (with output sequence (m, n)) defines the context that clusterings are "looked at" by the model, which are contained in the parameters $\vec{\varphi}_k$.

5.2.2 N2: Nodes with observed parameters

Nodes with observed parameters contain fixed multinomial distributions that generate edge sequences. They can be used to introduce distributions over observable labels. As parameters are fixed, they have no hyperparameter given. An example is the author–topic model [Rosen-Zvi et al. 2004] where for each word in a document m an author is chosen from the multinomial over authors of the document $\vec{a}_m$:

$$m \xrightarrow[[M]]{m} \underset{\text{N2}}{(\vec{a}_m)} \xrightarrow[[A]]{x_{m,n}=x} (\vec{\vartheta}_x \,|\, \alpha) \underset{\text{N1}}{} \xrightarrow[[K]]{z_{m,n}=k} \underset{\text{N1}}{(\vec{\varphi}_k \,|\, \beta)} \xrightarrow[[V]]{w_{m,n}} t \,. \tag{5.3}$$

The notation used here is to drop the hyperparameter in the node, $(\vec{a}_m)$. Cf. the previous section for notations $(\vec{\vartheta}\,|\,)$ and $(\vec{\vartheta}\,|\,\cdot)$.

Alternatively, N2 nodes can be introduced in any context where the ratio between two or more influences is to be controlled by a parameter. Notably, observed parameters do not block dependency between parents and children (see Section 4.5), as this is solely based on the component selection function of the nodes, whatever their type. Observed-parameter nodes rather force their output to their parameters without adapting them.

5.2.3 N3: Nodes with non-Dirichlet prior

Despite all the advantages of the Dirichlet distribution as a prior on the multinomial topic distributions described in Chapter 3, it has some properties that are unintended for some applications: Although samples from a Dirichlet with vector parameters $\vec{\alpha}$ can be controlled to have a vector mean that is $m_k = \alpha_k / \sum_k \alpha_k$ per component, the precision $s = \sum_k \alpha_k$ is global for all components. Furthermore, correlation between dimensions cannot be controlled; in fact, via the summing constraint $\sum_t \vartheta_{k,t} = 1$, Dirichlet samples have a mutual weak negative correlation between dimensions. Correlation between admixture topics in an LDA-like model therefore is not captured with Dirichlet priors [Blei & Lafferty 2007]. To alleviate this, alternative prior distributions may be employed on the multinomial parameters. In principle, all distributions on the simplex may be used, but two classes of distributions have been shown to be particularly useful: the logistic-normal and structured variants of the Dirichlet.

Logistic-normal. An important example of such an approach is the correlated topic model, CTM, with a logistic-normal distribution [Blei & Lafferty 2007] as a prior. The logistic-normal (or "log-normal") distribution [Aitchison & Shen 1980] is a multivariate Gaussian whose samples $x \sim \mathcal{N}(\vec{\mu}, \Sigma)$ are mapped to the simplex $\sum y_k = 1$ via the multinomial logistic transform (a.k.a.

softmax neural activation function [Sutton & Barto 1998]):

$$y_k = \frac{\exp x_k}{\sum_k \exp x_k} \,, \tag{5.4}$$

and the NoMM of the CTM is (with $\vartheta_{m,k}{=}y_k$):

$$m \xrightarrow[[M]]{m} \boxed{(\vec{\vartheta}_m \,|\, \vec{\mu}, \underline{\Sigma})}_{\substack{\text{logistic } \mathcal{N} \\ \text{N3}}} \xrightarrow[[K]]{z_{m,n}=k} (\vec{\varphi}_k \,|\, \beta)_{\text{N1}} \xrightarrow[[V]]{w_{m,n}} t \,. \tag{5.5}$$

The properties of log-normal samples can be controlled via the mean $\vec{\mu}$ and covariance matrix $\underline{\Sigma}$ in a much more flexible manner than would be possible with the Dirichlet prior, so in effect the correlation structure between topics may be captured and further a better perplexity reduction obtained on text data [Blei & Lafferty 2007] than in LDA with an N1 node for ϑ. Nodes with non-conjugate priors (like the logistic-normal distribution) are classified as N3B.

Structured Dirichlet. Other approaches to Dirichlet prior variants use the Dirichlet distribution in a modified, "restructured" form and preserve the conjugacy with multinomial latent model variables. They are classified as type N3A.

As a trivial case, a mixture of Dirichlet distributions may be used. In fact, this results in a regular NoMM with an additional N1 level that samples the indices of the hyperparameters, and the mean of the resulting Dirichlet samples could be taken as $\vec{\vartheta}_m$:

$$1 \to \boxed{(\vec{\pi} \,|\, \eta)_{\text{N1}} \xrightarrow[[C]]{c_{m,j}=c} (\vec{\vartheta}_m \,|\, \vec{\alpha}_c)_{\substack{\vec{\vartheta}_m=\langle\vec{\vartheta}_{m,j}\rangle \\ \text{N3}}}} \xrightarrow[[K]]{\dots} \,. \tag{5.6}$$

A similar approach with one sample per component ($j = 1$) has been used to model word burstiness in the DCM-LDA model by [Madsen et al. 2005, Doyle & Elkan 2009]. Here the Mixture of Dirichlet is used as a Dirichlet-compound multinomial (DCM) prior by integrating out the parameter $\vec{\pi}$. Because the DCM distribution is not in the exponential family and thus non-conjugate with the multinomial, an exponential-family approximation has been proposed: the EDCM distribution [Elkan 2006].

In [Wallach 2008, Wallach et al. 2009a] the Dirichlets in mixture nodes are given a prior that itself has a Dirichlet distribution over the mean and a precision parameter, i.e., $\vec{\alpha}' = \alpha\vec{m}$ and $\vec{m} \sim \text{Dir}(\gamma)$. By conditioning the mean $\vec{m}$ of the prior parameter on specific contexts of words, clustering of components is demonstrated similar to the vector hyperparameter discussed in type N1. To illustrate this, [Wallach 2008] uses an LDA-like model where the N1 output node has parameters φ of dimension $V(V-1)$ over all word bigrams $(w_{m,n-1}, w_{m,n})$ in the vocabulary, and the prior parameter means $\vec{m}_t$ are conditioned on the current word $w_{m,n}{=}t$, thus clustering all

components of $\varphi_{k,t,*}$ belonging to term t:

$$m \xrightarrow[[M]]{m} (\vec{\vartheta}_m \,|\, \alpha) \xrightarrow[[K]]{z_{m,n}=k} \boxed{(\vec{\varphi}_{k,t} \,|\, \beta \vec{m}_t)_{\vec{m}_t \sim \mathrm{Dir}(\alpha_m)}} \xrightarrow[[V]]{w_{m,n}} t\,. \tag{5.7}$$

N1 N3

In effect, this model allows to capture both local (bigrams) and document-wide dependencies in a single model.

Another variant is the usage of Dirichlet tree [Minka 1999] and forest distributions, which has been shown to encode prior knowledge on the contents of LDA topics [Andrzejewski et al. 2009] in a model similar to (5.7). The Dirichlet tree is a distribution that generates multinomials by a stochastic process that traverses a tree, sampling a Dirichlet over the branches of each tree node and generating the multinomial masses at its leaves [Minka 1999]. Notably, the Dirichlet tree also retains conjugacy to the multinomial distribution. By appropriate parametrisation of its parameters, multinomials with dependent dimensions can be generated (each inner node creates a dependency). Consequently, prior knowledge on similar terms can be incorporated by creating a Dirichlet tree with similar terms in the hierarchy. To repel terms, [Andrzejewski et al. 2009] use a mixture of Dirichlet trees (Dirichlet forest) with a component for each mutually exclusive term subset.

For completeness, it should be noted here that indeed every topic model with additional mixture levels compared to the two-level model LDA may be considered a two-level model with a structured, optionally multi-level Dirichlet prior. In the PAM4 model for example, this prior would simply correspond to the document–supertopic and supertopic–subtopic mixtures combined (see Fig. 4.5(d) and its description in Chapter 4). However, parameter estimation of prior Dirichlet levels is then regarded as optimisation of hyperparameters and the approaches proposed for NoMM inference are left unused.

Furthermore, non-parametric priors based on the Dirichlet process in principle also fit into the non-Dirichlet prior class, but while model changes with the above variants are local to NoMM nodes, introduction of non-parametric methods requires a more profound change of structure, which is discussed in Section 5.5.

5.2.4 N4: Nodes with non-discrete components

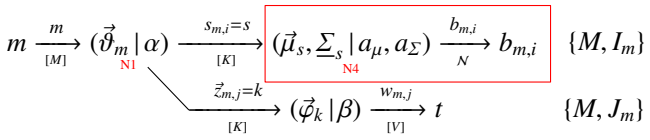
As a variant of type N1 with an observed edge, component distributions other than the multinomial can be given to model non-discrete observations. In this framework, the “classical” Gaussian mixture model [McLachlan & Peel 2000] is an important example of this node type. It uses a global mixing weights distribution $\vec{\pi}$:

$$1 \longrightarrow (\vec{\pi} \,|\, \alpha) \xrightarrow[[K]]{z_n=k} \boxed{(\vec{\mu}_k, \Sigma_k \,|\, \cdot) \xrightarrow[\mathcal{N}]{x_n} x} \quad . \tag{5.8}$$

N1 N4

An example in the area of topic modelling is the Gaussian–multimodal LDA (GM-LDA) model in [Blei & Jordan 2003] and its extension Corr-LDA [Barnard et al. 2003] where mixtures of Gaussian components are observed. The relevant edge $b_{m,i}$ is added an $\mathcal{N}$ as range specifier

signifying a normal distribution, and GM-LDA becomes:



(5.9)

where the Gaussian components have prior distributions with hyperparameters a_μ and a_Σ: Analogous to the Dirichlet–multinomial case, the conjugate priors for the Gaussian mean and covariance can be used, which are Gaussian and inverse-Wishart distributions for the multivariate case. This example also shows branching, which is discussed in Section 5.3.

By connecting both discrete and Gaussian observations, it is possible to detect correlations between caption terms and physical observations, in this case image features. The authors showed that with such a model it is possible to predict the caption text of unseen images with some confidence [Blei & Jordan 2003].

Generally, this node type is frequently seen in media applications where audio, image or video features are modelled as mixtures of Gaussians, see for instance models for image analysis [Wang & Grimson 2007], computer vision [Cao & Fei-Fei 2007, Sudderth 2006] or music analysis [Hu & Saul 2009].

5.2.5 N5: Aggregation nodes

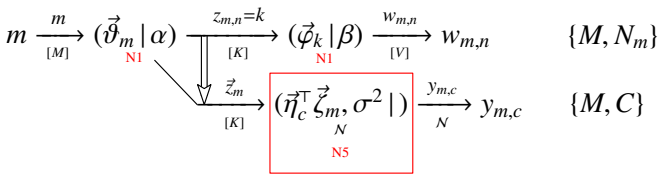
Typically, the sequence index of mixture nodes either is the same at input and output, e.g., $\xrightarrow{x_{m,n}} (\vec{\vartheta}_x \,|\, \alpha) \xrightarrow{y_{m,n}}$, or there are several output samples for an input, e.g., $\xrightarrow{x_m} (\vec{\vartheta}_x \,|\, \alpha) \xrightarrow{y_{m,n}}$. The third possibility, $\xrightarrow{x_{m,n}} (\vec{\vartheta}_x \,|\, \alpha) \xrightarrow{y_m}$, is the case of aggregation. In other words, aggregation is to generate a single variable y_m from multiple inputs $x_{m,n}$. In topic models, the predominant case for such a setting is a single label of an item to be generated from on multiple topic associations, which can lead for instance to classifier models. There are at least two approaches that such a setting may be realised: (1) stochastically, one value (or more) may be selected from the set of inputs $\{x_{m,n}\}_n$, and (2) deterministically, a regression function may be used whose response is the output.

Selection. Randomly selecting an element from a given set of input values is the simplest approach to aggregation nodes, type N5A. This sampling may be done using a symmetric fixed multinomial over token indices. When the multinomial parameter is hidden, it may be estimated from the data, e.g., using a Dirichlet prior: $\vec{\vartheta} \sim \mathrm{Dir}(\eta)$ with the clustering behaviour of (3.28). As an example model, the supervised topic model in [Bundschus et al. 2009] uses a fixed distribution over all topic labels of a document $\{z_{m,n}\}_n$ to sample a class label $\tilde{z}_m$.[4] The NoMM of this example is presented with the discussion of aggregation edges in Section 5.3, type E4.

Regression. A prominent example of deterministic aggregation is to use a regression function, which has been proposed in the supervised LDA model [Blei & McAuliffe 2007] and is classified as type N5B. Supervised LDA aggregates a feature sequence to a supervised output with Gaussian

[4] One may also interpret this as sampling from an empirical distribution of the document topics.

response based on regression of the topic samples:



(5.10)

where $\zeta_{m,k} = 1/N \sum_n \delta(z_{m,n} - k)$ are the hidden empirical frequencies of the document topics and c is the class label. That is, the response is generated from a Gaussian with a class-dependent regression function $\vec{\eta}_c$ weighting the strength of the different topics $\vec{\zeta}_m$ for a label supervised quantity c according to the rest of the data. [Blei & McAuliffe 2007] give substitutions for the unbounded real response with other response types, using generalised linear models, and empirically prove that supervised LDA works well in classification tasks, outperforming the approach of classifying LDA topic parameters with support vector machines [Blei et al. 2003b]. As an alternative variant of the unbounded Gaussian response, the logistic transform (5.4) may be used, leading to a response on the unit interval, $y \in [0, 1]$.

5.3 Mixture branching and edges

Mixture branching allows linking of several observations or other effects to a common cause. Among the different possibilities outlined in Section 5.3, autonomous branches are the most common approach and modelled by a NoMM node connected to more than one child node. Autonomous branching can be considered an essential design instrument for NoMMs that generate data of different modalities. Modality here refers to some type of data, such as text, class labels, authorship, references, etc., and especially in community scenarios, multiple data modalities are common, as has been discussed in Chapter 2.

In general, at branching points latent variables that are generated from the same parameters in the parent node of the branch are made dependent, which will cause the parameters to adapt to co-occurrences between the different modalities. Here, the manner in which different edges depend on each other allows control over how dependencies in the multimodal observations are digested by the model. This is why the study of mixture branching is a study of mixture edges at the same time, and in the next paragraphs, different edge/branching cases are investigated, starting with the "trivial" case of unbranched edges and then considering the different types of dependency between edges.

5.3.1 E1: Unbranched edges

Unbranched edges couple the mixtures of their parent and child nodes, allowing to bring together the different clusterings between respective inputs and outputs. Observed edges are a special case that simply indicate that their values are actually known a priori, and beside data contained in observed parameters, they are the common method to introduce evidence to the model (see node type N1). All models in Chapter 4 except for hPAM exclusively rely on E1 edges, and the way LDA works is most illustrative: The topics in the first node index the components in the second node (non-terminal nodes counted from the root), coupling both mixtures. Similarly, the PAM model as shown in Fig. 4.5(d) uses supertopics that mix with subtopics in the first and second node from the root.

Sub-sequence and super-sequence edges. A special case of an unbranched edge is an edge that uses a sub-sequence of some hidden dependent edge. A good example is the edge $z_{m,s,n}$ in the LDCC model [Shafiei & Milios 2006] in Fig. 4.6(c), which is a sub-sequence of $y_{m,s}$: For every segment of a document (m, s), a sequence of words $w_{m,s,n}$ is generated, along with topics $z_{m,s,n}$.

Because sub-sequences always appear in relation to some other edge, this is not an ordinary structure type. In fact, sub-sequences may appear with other edge types and are classified using a suffix S(e) to any of the edge types (E1 etc.) where e is an edge or set of edges with super-sequences. In the example in LDCC, $z_{m,s,n}$ is E1S($y_{m,s}$).

5.3.2 E2: Autonomous branches

If a node has several children and the edges draw separate samples from this common parent node, then they are dependent via its (hidden) parameter that adapts to the sequences of both hidden edges. In the literature, this approach has been chosen for several models that model correspondence between discrete data modalities, such as multi-multinomial LDA (MM-LDA) [Ramage et al. 2009]:

$$m \xrightarrow[[M]]{m} (\vec{\vartheta}_m \,|\, \alpha) \underset{\text{N1}}{} \begin{cases} \xrightarrow[[K]]{x_{m,i}=x} (\vec{\varphi}_x \,|\, \beta) \underset{\text{N1}}{} \xrightarrow[[V]]{w_{m,i}} w_{m,i} \\ \xrightarrow[[K]]{y_{m,j}=y} (\vec{\psi}_k \,|\, \beta) \underset{\text{N1}}{} \xrightarrow[[T]]{t_{m,j}} t_{m,j} \end{cases} . \tag{5.11}$$

This typus of model re-appears in various applications that use data from various corresponding sources of evidence, for instance the multimodal topic model MoM-LDA [Blei & Jordan 2003] that generates topics for every modality involved, and the conditionally independent LDA model as an "entity–topic model" [Newman et al. 2006a] that generates word and entity topics independently given a common document–topic node. Generally, effects are modelled that correlate within the same group of data (e.g., document).[5]

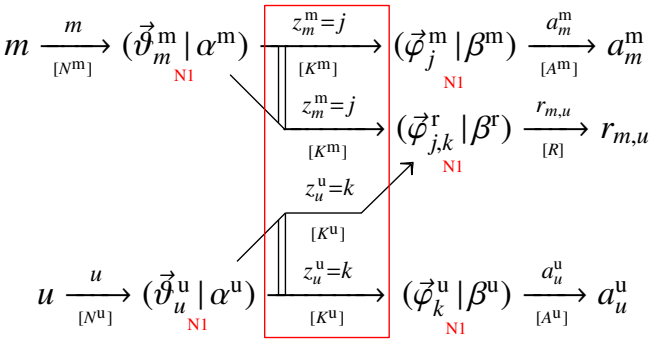
Notably, although both branches are called "autonomous", the respective sub-models are not statistically independent if their values are jointly sampled from the parent node, that is, if both branches share the same sequence. If they don't, the samples for E2 edges become

[5] E2 edge sets of size N are similar to sub-sequences of size N with the parent edge as super-sequence.

conditionally independent given the parent node parameters. In the typology, E2 branches with disjoint sequences can be classified as E2A sub-type, with joint sequence as E2B.

5.3.3 E3: Coupled branches

In the case of coupled branches, edges share a single hidden token sequence, creating a stronger dependency between the child nodes than in the case of type E2: Each child node in E3 branches is indexed with the same topic value, thus correlation between particular tokens is modelled. This type of branching structure has been adopted for instance by the hidden relational model (HRM) in [Xu et al. 2006]:

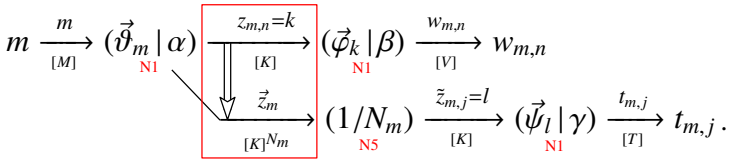


(5.12)

where the connected edges have identical sequences (and show identical variables). In [Xu et al. 2006] a framework of applying HRMs to relational database schemata is described. The hidden relational model shown in (5.12) is designed for the scenario of ratings $r_{m,u}$ between N^{u} users (actors) and N^{m} movies (media) set in context of user and movie attributes a^{u} and a^{m}. Given these data, ratings of unknown users may be learnt from the adapted parameters, which in effect is a form of statistical relational learning.

5.3.4 E4: Aggregation branches

Aggregation branches can be considered a variant to coupled branches, type E3. But now one of the edges aggregates the others, for instance to provide input sets appropriate for aggregating nodes (type N6). An example is the supervised LDA model in [Blei & McAuliffe 2007] (5.10) using a regression node, or the topic-tag model in [Bundschus et al. 2009] based on supervised LDA where tag topics $\tilde{z}_{m,j}$ are drawn from the set of sampled document topics $\vec{z}_m$:



(5.13)

Both models adapt to data given as document labels or response values, yielding topic models for classification scenarios. The E4 edge is commonly seen with the N5 node, see Section 5.2.5 and Appendix C.3, for more details.

5.4 Mixture merging and component selection

Mixture merging allows linking of one effect to several possible causes (according to the generative process of the model), and it is modelled by connecting a child node to more than one parent node, dual to branching structures. Merging mixtures makes it possible to integrate several types of influence that observations may be drawn from, generally with more differences than when generating them from a single parent node (=mixture model) whose components may account for different influences. In the NoMM representation, merging inputs corresponds to designing appropriate component selection functions $k = g(\uparrow x_i, i)$, which have a decisive influence on model behaviour. This section analyses mixture merging by studying different possibilities of component selection, starting with the trivial case of single-index components and continuing with different forms of combining components in the child node parameters.

5.4.1 C1: Single-index components

Components in a mixture node that have only a single index do not actually merge mixtures, but are the standard case of component selection function. An important distinction in this case is that between observed and unobserved indexes because this imperatively influences how the model can adapt to the data.

Sequence nodes. In any node with a component selection function that only depends on an observed index, components can be associated to data a priori, and $k = g(i)$. In the case of LDA, this corresponds to choosing a separate document–topic distribution $\vec{\vartheta}_m$ for each document m in the node $(\vartheta \mid \alpha)$, which is used to create co-occurrence contexts for the latent semantic analysis performed with the model. Because in virtually all cases of observed component selection, the actual index is that of a sequence element (document m, user u in data sequences), these nodes may be called "sequence nodes", as already discussed in Chapter 4. In typical (but not all) topic models, sequence nodes are found as direct children of root terminal nodes because they provide the grouping contexts that are required to form (ad-)mixture.

Topic nodes. A component selection function that depends on a hidden input variable creates a dependency to its parent edge, with a component selection function of $k = g(\uparrow x_i)$. In the LDA model as an illustrative example, the hidden input selection of the N1 node $(\varphi \mid \beta)$ with observed output in (5.2) allows to link the observed distribution of terms in topics to that of topics in documents, which is ensured by the N1 sequence node $(\vartheta \mid \alpha)$ as described above. In typical (but not all) topic models, topic nodes are found specifically as direct parents of output terminal nodes, linking observations to topics that group different contexts (grouping topics for documents in $(\vartheta \mid \alpha)$ linked with grouping all documents in $(\varphi \mid \beta)$ in LDA).

For the typology, component selection functions leading to sequence nodes, $k = i$, are considered a sub-type C1A, and such leading to topic nodes, $k = x_i^{\ell-1}$, sub-type C1B. As will be shown in Chapter 6, the distinction between sequence and topic nodes also has important consequences for querying in NoMM models.

5.4.2 C2: Combined component indices

The simplest way to join several discrete inputs is to set up one component for each combination of inputs. This assumes exchangeability of all component combinations given the indicator variables.

The simplest case for such as structure is to combine a C1A observed sequence index and a C1B hidden index, $k = (i, x_i^{\ell-1})$, leading to sub-type C2A. An example of this is the second level $(\vec{\vartheta}_{m,x} \,|\, \alpha_x)$ of the PAM4 model in 4.5(d) described in Chapter 4, which allows to cluster subtopic associations with terms (y, t) by documents, leading to super-topics that identify these clusters. Extending the distinction between sequence and topic nodes, this node in PAM4 can be considered a sequence node because there is an individual set of parameters for elements of the sequence (here: documents).

Combining the values of several incoming hidden edges (i.e., several C1B indices) to a multi-dimensional component index, $k = \uparrow x^i$ leads to a combined hidden index, which is considered the C2B sub-type. An example for such a structure that fully factors several hidden input edges is the multi-LDA model [Porteous et al. 2008a]:

$$\begin{array}{l} m \xrightarrow[[M]]{m} (\vec{\vartheta}_{\underset{\text{N1}}{m}} \,|\, \alpha) \\ u \xrightarrow[[U]]{u} (\vec{\zeta}_{\underset{\text{N1}}{u}} \,|\, \alpha) \end{array} \boxed{\begin{array}{l} \xrightarrow[[K]]{x_{u,m}=x} \\ \xrightarrow[[K]]{y_{u,m}=y} \end{array}} (\vec{\varphi}_{\underset{\text{N1}}{x,y}} \,|\, \beta) \xrightarrow[[R]]{r_{u,m}} r_{u,m} \,. \tag{5.14}$$

In this model (the two-branch case Bi-LDA is shown here), users u and movies m form the rows and columns of a rating matrix $\underline{r}$, and the model allows to predict empty ratings from the known ones. Notably, the factoring of component indexes also can be interpreted as a matrix factorisation approach.

Another example of mixture merging with separable components is the hidden relational model in (5.12), where the central node $(\varphi^{\mathrm{r}}_{j,k} \,|\, \beta^{\mathrm{r}})$ merges both inputs from user and movie topics.

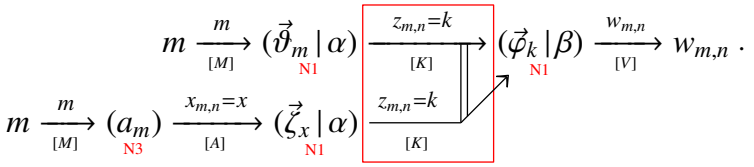
Extending this, C2 structures allow arbitrary design of complex component selection functions. The unrestricted case of component selection, $k = g(\uparrow x^{\ell}, i)$, is denoted as sub-type C2C. For instance, the selection function may map several edge values to a smaller number of components than there are edge value combinations. An example of this is the hierarchical PAM model in Fig. 4.5(e) described in Chapter 4.

5.4.3 C3: Interleaved component indices

As a variant to independent edges indexing components in a node, input edges may be merged for different reasons. One is to reduce computation and data requirements, as C2 components typically strongly increase the number of components in the node with C2 inputs. Another reason is to obtain a tool that forces child components to be directly dependent on both parents.

One way to obtain such a structure is to actually mix the influence of the incoming edges, which is proposed here as a novel NoMM structure. This mixing may be done by sampling both the two incoming edges through the node with the C3 component selector. Opposed to for

instance forcing both inputs to have equal values, this mixing allows to retain a generative model proper.[6] A straight-forward example for such a structure is a model that generates author- and document-specific topic distributions as a mixture of LDA [Blei et al. 2003b] and the author–topic model (ATM) [Rosen-Zvi et al. 2004]:



(5.15)

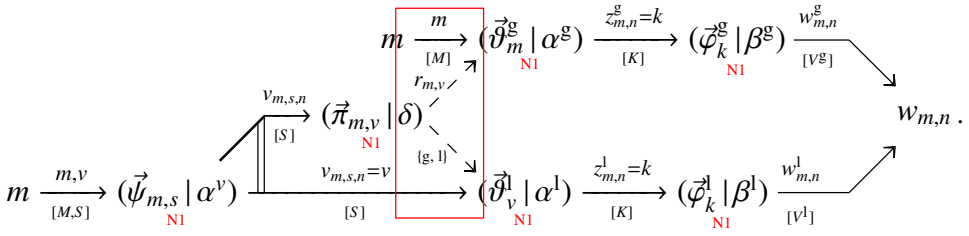
The C3 structure in this novel model "interleaves" the topics learnt from the contexts created by documents and those from contexts created by authorship. The latter additionally needs to choose among the different authors of a document according to the author–topic model. By using the same component index dimension for both incoming branches, a set of global topics φ is obtained that is compatible with both document topics ϑ and author topics ζ.

Similar to what had been said about E2 branch structures, where it matters whether the branches use the same token sequence, in the C3 structure one may also distinguish between the incoming edges having the same sequence (in which case two equal values $w_{m,n}$ are drawn jointly) and using disjoint incoming sequences. In the typology, C3 structures with disjoint sequences are classified as C3A sub-type, such with a joint sequence as C3B.

5.4.4 C4: Switches

The influence of parent sub-models may also be combined using switches as component selectors. So far, all mixture nodes have transported their output to their children that used it to index their components directly. Opposed to this, a switching input does not choose a component but chooses an edge from a set that is then used to index the component.[7]

An example for this is the multi-grain topic model in [Titov & McDonald 2008] where an input edge controls the use of local or global topics, i.e., enables one of two mixnet branches, one superscripted g that learns topics from global information, and one superscripted ℓ that adapts from local information (a token window). Switching edges are denoted by the set of branch values in braces $\{\cdot\}$ and dashed arrows:



(5.16)

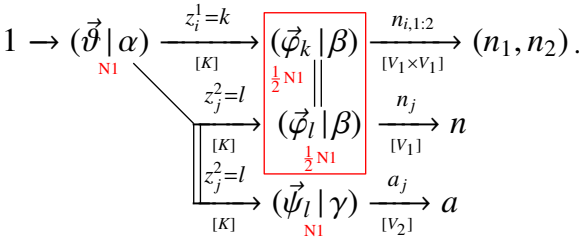
[6] Forcing equal inputs requires a generative process that is somewhat exotic: Drawing until both inputs are equal.

[7] In principle, this may be achieved with specific C2C component selection functions. However, literature does not explicitly discuss the fundamental difference between variables indexing components and such indexing edges, despite their fundamental differences.

In [Newman et al. 2006a], a similar approach is taken for the "Switch-LDA" model, where according to a switching node with binomial distribution and beta prior (i.e., non-multinomial node type N4 but still conjugate), either a word or a named entity is sampled for each word in a document.

5.4.5 C5: Coupled-node components

Analogous to edges coupled via shared sequences, nodes may be coupled via shared parameters. This way, several nodes have their own separate input and output edges but share parameters and hyperparameters, which strongly couples their parallel in- and outgoing edge sequences. Because several node inputs are merged (although across nodes), this structure is considered a type of mixture merging. An example of such a structure is the "simple relational component model" (SRCM) in [Sinkkonen et al. 2008] that models two types of co-occurrence: one between nodes in a graph $(n_{i,1}, n_{i,2})$, that is, the model generates edges, and one between nodes and their attributes (n_j, a_j):



(5.17)

Because the nodes n_1, n_2 (incident on the edge) and n (co-occurring with the attribute a) are drawn from the same distribution but these pairs stem from different draws from the topic–node distribution φ (one for edge i, one for node attribute j), two mixture nodes appear in the model that share parameters. Note that the connection between nodes sharing parameters is not a NoMM edge, as there are no discrete values transmitted; it rather indicates the identity of the parameters for both nodes.

In [Sinkkonen et al. 2008], the SRCM model has been compared to HRM (5.12) in Section 5.3, which uses type E3, yielding slightly less accuracy but at far less computational complexity.[8] Similar to the HRM model discussed above, SRCM is a special case of a more generic framework that covers multimodal data with several types of co-occurrence that may be represented as a graph (or optionally as a hyper-graph because co-occurrences may be defined between more than two items). Each data modality then receives its own topic-specific modality distribution, and for every co-occurrence type that a modality is part of, a mixture node is used that shares its parameters with all other mixture nodes that cover this modality across co-occurrence types.

As a variant of sharing parameters (which implies shared hyperparameters), nodes may be coupled by sharing their hyperparameters but generate their individual components from them.[9]

As with E2 and C3 structures, it matters whether the shared nodes are sampled from jointly or in different sequences. In the typology, C5 structures with disjoint sequences are classified as C5A sub-type, such with joint sequence as C5B.

[8] This comparison was made based on non-parametric variants of the models.

[9] Note that this requires that all hyperparameter indices be known before the components are sampled according to the model's generative process.

5.5 Towards non-parametric extensions

All structures presented in the typology above require an a priori choice of the edge dimensions and consequently component counts (or information about the inherent dimensionality in the data). In order to be able to estimate these from the data and fit models to data without any a priori parameter assumption, this section studies how the typology may be extended to incorporate non-parametric methods into NoMMs.

As a representative of a wider set of non-parametric priors, the Dirichlet process (DP) is considered, which has been introduced in Section 3.6.2. The interesting aspect about the DP is its clustering behaviour, which is similar to that of the Dirichlet distribution. With $G \sim \mathrm{DP}(\alpha, G_0)$ with concentration parameter α and base distribution G_0, given previous samples $\{\vec{\varphi}_i\}_{i=1}^n$, the likelihood that a new sample $\vec{\varphi}_{n+1}$ repeats some sample $\vec{\varphi}_k$ already seen as $\vec{\varphi}_i{=}\vec{\varphi}_k$ with $i \in [1, n]$ is proportional to the number n_k of samples previously associated with $\vec{\varphi}_k$, and the likelihood to sample a $\vec{\varphi}_k$ previously unseen (i.e., $n_k = 0$) is proportional to the concentration parameter α (cf. Fig. 3.17(d)):[10]

$$p(\vec{\varphi}_{n+1} \mid \{\vec{\varphi}_i\}_{i=1}^n, G_0, \alpha) = \frac{1}{n+\alpha}\left(\sum_{k=1}^{\infty} n_k \delta(\vec{\varphi}_{n+1} - \vec{\varphi}_k) + \alpha G_0\right) . \tag{5.18}$$

That is, with the DP prior and its clustering scheme, a well-defined component generation process is available that is the key to estimate the dimensionality in NoMMs.

However, using this generation process comes at the price of some intricacies, which is why the models are treated separately from other structures here. In the next section, we will apply the DP to NoMMs with simple infinite mixtures and will subsequently extend this to admixtures. Finally, we will give a brief overview on how the different NoMM structure types can be transformed into non-parametric structures. Among the different representations of the DP discussed in Section 3.6.2, we will use the SBP representation (5.18) as it allows to retain a large part of the structure of the equivalent finite models and is mostly compatible with the typology of NoMM structures. This way, NoMMs can be used as model representation that "overloads" both parametric and non-parametric versions of corresponding models, like LDA overloading HDP.

In a nutshell, the Dirichlet process produces discrete samples $G \sim \mathrm{DP}(\alpha, G_0)$ that contain infinitely many point masses whose locations are sampled from some base distribution G_0 and whose weights are distributed according to the stick-breaking process (SBP, also known as GEM distribution), parametrised by α [Sethuraman 1994, Ishwaran & James 2001]:

$$G(\vec{\varphi}) = \sum_{k=1}^{\infty} \pi_k \delta(\vec{\varphi} - \vec{\varphi}_k), \quad \vec{\varphi}_k \sim G_0,\ \vec{\pi} \sim \mathrm{GEM}(\alpha) . \tag{5.19}$$

For a Dirichlet base distribution, $G_0 = \mathrm{Dir}(\beta)$, this is a set of infinitely many finite-dimensional multinomial parameters $\vec{\varphi}_k \sim \mathrm{Dir}(\beta)$ in a finite-dimensional space.

[10] This representation integrates out the actual G.

5.5.1 I1: Mixtures

Mixtures based on non-parametric priors allow estimation of the data dimensionality but do avoid coupling of several mixture levels. A prominent example of such an approach is the infinite Gaussian mixture model [Rasmussen 2000], a fully non-parametric model (i.e., one that does not require parameters to set a priori) where the data points cluster around an infinite number of Gaussian components chosen from a global weight distribution that itself is a draw from a DP:

$$1 \rightarrow \boxed{(\vec{\pi}\,|\,\gamma) \underset{\vec{\pi}\sim\mathrm{GEM}(\gamma)}{\xrightarrow[\text{I1,N3}]{z_n=k}} (\vec{\mu}_k, \underset{\text{N4}}{\underline{\Sigma}_k}\,|\,\theta)} \xrightarrow[\mathcal{N}]{x_n} x\ . \tag{5.20}$$

This infinite mixture is the non-parametric equivalent to (5.8) in Section 5.2, and its type is N3 because of the non-Dirichlet prior. Here $\vec{\pi}$ is the parameter from the SBP representation of the DP, while γ is its precision or concentration parameter and θ is the set of parameters of the base distribution that generates the $\vec{\mu}$ and $\underline{\Sigma}$ for each component. Also, the z_n are introduced as auxiliary variables.

For topic models that create discrete observations, such an approach may be used as well, and in an LDA-like model, the document-specific topic distributions $\vec{\vartheta}_m$ may be generated according to the DP and thus re-used across documents, i.e., documents cluster via the possibility of shared $\vec{\vartheta}_m = \vec{\vartheta}_i$ with $i \in [1, m)$. This has been achieved with the Dirichlet-enhanced LSA model [Yu et al. 2006]:

$$m \xrightarrow[[M]]{m} \boxed{(\vec{\vartheta}_m\,|\,G) \underset{G\sim\mathrm{DP}(\gamma,G_0)}{}_{\text{I1,N3}}} \xrightarrow[[K]]{z_{m,n}=k} (\vec{\varphi}_k\,|\,\beta) \xrightarrow[[V]]{w_{m,n}} t\ . \tag{5.21}$$

Here, the topic proportions $\vec{\vartheta}_m$ are not document-specific but rather shared between similar documents. In the SBP representation of the DP, this can be expressed analogous to (5.20):

$$1 \rightarrow \boxed{(\vec{\pi}\,|\,\gamma) \underset{\vec{\pi}\sim\mathrm{GEM}(\gamma)}{\xrightarrow[\text{I1,N3}]{c_m=c}} (\vec{\vartheta}_c\,|\,\alpha) \underset{\vec{\vartheta}_c\sim G_0}{}_{\text{I1,N1}}} \xrightarrow[[K]]{z_{m,n}=k} (\vec{\varphi}_k\,|\,\beta) \xrightarrow[[V]]{w_{m,n}} t \tag{5.22}$$

where the first node is of type N3 (multinomial with non-Dirichlet prior) and creates a set of infinitely many point masses that are sampled from to create the clusters for the $\vec{\vartheta}_m$=$\vec{\vartheta}_c$. Furthermore, the second node is a regular N1 node that generates components $\vec{\vartheta}_c \sim \mathrm{Dir}(\alpha)$ as draws from G_0.

5.5.2 I2: Admixtures

In an ideal world, NoMMs could simply be extended to estimate their dimensionality using infinite-dimensional edges, and a non-parametric LDA NoMM would simply look like:

$$m \xrightarrow[[M]]{m} (\vec{\vartheta}_m \,|\, \alpha) \xrightarrow[[K\to\infty]]{z_{m,n}=k} (\vec{\varphi}_k \,|\, \beta) \xrightarrow[[V]]{w_{m,n}} t \ .$$

Unfortunately, as soon as there are weights local to a part of the data set, which implies admixture as is common in topic models, things become more complicated. The standard approach to extend the DP for this case is the hierarchical DP (HDP [Teh et al. 2006]) discussed in Section 3.6.2: A hierarchy of DPs where the root DP G_0 generates DPs G_i along the token sequence (typically document-specific, i.e., $G_i \equiv G_m$). The root DP has the component distribution G_φ as base distribution, and in a non-parametric LDA model, this is $G_\varphi = \mathrm{Dir}(\beta)$, i.e., the finite prior of the second node of LDA. This root DP generates components $\vec{\varphi}_k \sim \mathrm{Dir}(\beta)$ and weights according the stick-breaking process and the clustering scheme outlined in (5.18). Each G_m then samples from G_0, obtaining a subset of components $\vec{\varphi}_k$ from the global set of point masses that G_0 consists of, cf. Figs. 3.18 and 3.19. From the G_m, parameters $\vec{\varphi}_{m,n}{=}\vec{\varphi}_k$ are sampled that observations are generated from directly without an auxiliary variable $z_{m,n}$.

Because discrete topic indicators $z_{m,n}{=}k$ of the components are not part of this model, its structure can only be represented as a "degenerate" NoMM with a non-discrete edge for the infinitely many components $\vec{\varphi}_{m,n}$ on the $(V-1)$-dimensional simplex:

$$m \xrightarrow[[M]]{m} \boxed{(G_m \,|\, \alpha_0, G_0) \underset{G_m\sim\mathrm{DP}(\alpha_0 G_0);\, G_0\sim\mathrm{DP}(\gamma G_\varphi)}{\xrightarrow[\mathrm{S}_{V-1}]{\vec{\varphi}_{m,n}=\vec{\varphi}_k}} (\vec{\varphi}_k \,|\, G_\varphi)}_{\substack{\text{I2,N3,N4} \\ \qquad\qquad \text{I2,N1}}} \xrightarrow[[V]]{w_{m,n}} t \ . \tag{5.23}$$

Stick-breaking representation. As a relief to this somewhat awkward construction, the NoMM may also be created from an SBP representation directly. According to the transformed Bayesian network in Fig. 3.18(b) and due to the derivations in [Teh et al. 2006], this yields:[11]

$$m \xrightarrow[[M]]{m} \boxed{(\vec{\vartheta}_m \,|\, \alpha_0\vec{\pi}) \underset{\vec{\pi}\sim\mathrm{GEM}(\gamma)}{\xrightarrow[[K\to\infty]]{z_{m,n}=k}} (\vec{\varphi}_k \,|\, \beta)}_{\substack{\text{I2,N3} \\ \qquad\qquad \text{I2,N1}}} \xrightarrow[[V]]{w_{m,n}} t \ . \tag{5.24}$$

This model augments the finite LDA model (5.2) by a specific hyperparameter $\alpha_0\vec{\pi}$ that represents the root DP and the concentration parameter α_0 but otherwise retains much of the finite structure.[12] Specifically, the document-specific topic distributions $\vec{\vartheta}_m$ are the representations of

[11] The variables used here re-enact those used to explain LDA; [Teh et al. 2006] uses $\vec{\beta}$ for $\vec{\pi}$ and $\vec{\pi}_j$ for $\vec{\vartheta}_m$ with j as document index.

[12] Note that this fits an N3 node.

Type	Name	Approaches, restrictions
N1	Hidden parameters	Restrictions: No vector hyperparameters as in finite model: HDP generation process replaces weighting of components by the component weights $\vec{\pi}$ and $\vec{\vartheta}_m$ in the SBP representation. Solutions: Advanced approaches like dependent DPs [MacEachern 1999]. Examples: [Wang et al. 2009b, Li et al. 2007b] (the latter re-designs the PAM model for non-parametric inference). [Huang & Renals 2008] (HDP is added an additional level that allows clustering of document-specific G_m into (observed) roles by using $G_r \sim \mathrm{DP}(\alpha_0, G_0)$ and $G_m \sim \mathrm{DP}(\alpha_1, G_r)$).
N2	Observed parameters	Restrictions: Fixed dimension of output, like in the author–topic model. Another approach is to convert N2 nodes to branched observed egdes, e.g., adding author labels as output edge to a HDP-like model in a non-parametric author–topic model in [Wallach 2008].
N3	Non-Dirichlet priors	The base distribution G_φ may be used but will lead to more complex inference, as the conjugacy relation between the Dirichlet and discrete node output is lost. Usage with alternative prior distributions is possible for child nodes of the DP root, e.g., in a structure like $(\vec{\vartheta} \mid \alpha_0 \vec{\pi}) \xrightarrow[[K\to\infty]]{z_{m,n}=k}$ $(\vec{\varphi} \mid \theta)$ with some non-Dirichlet component distribution $p(\vec{\varphi}\mid\theta)$.
N4	Non-discrete components	The base distribution G_φ will become non-Dirichlet, possibly the conjugate prior of the component distribution. Examples: Infinite Gaussian mixture model in (5.20) and several models in the computer vision area, e.g., [Sudderth 2006].
N5/E4	Aggregation nodes/branches	Restrictions: N5 nodes need to be modified to cope with changing numbers of possible topics at the input, but especially for the sampling case as in (5.13), this is straight-forward.
E1	Unbranched edges	Standard case with HDP models.
E2,E3	Branched edges	Issue: For each branch a separate base distribution must exist. Exception: hybrid models where one branch is a finite model and the other one fed by the HDP. Workarounds: Using the product of branch distributions as base distribution, e.g., [Wallach 2008], and using the SBP representation with infinite-dimensional $\vec{\pi}$ indexing several branches, e.g., infinite hidden relational model [Xu et al. 2006] (no admixture).
C1	Single component selectors	Restrictions: The provider of the base distribution must be a topic node. All parents in the HDP hierarchy should be sequence nodes.
C2	Combined indices	Examples: [Porteous et al. 2008a] propose a multi-HDP that merges two roots into the same base distribution. Infinite hidden relational model [Xu et al. 2006] (not considering admixture and thus using flat DPs instead of hierarchies).
C3	Interleaved indices	Issue: No direct solution for DPs because each of the source DPs creates its own set of discrete probability masses (the elements of $\vec{\pi} \sim \mathrm{GEM}(\gamma)$ above), which (almost surely) is disjoint from each other. May be emulated by C4 switches.
C4	Switching inputs	Branches may be considered separately with the counts of sampled tokens split between sub-models.
C5	Coupled-node components	Issue see C3. Limited example: Simple relational component model [Sinkkonen et al. 2008] (no admixture and C5 in finite nodes).

Figure 5.2: Towards non-parametric approaches to NoMM structures.

the document DPs G_m, i.e., are multinomials indexing infinitely many topic distributions $\vec{\varphi}_k$ in the second node of (5.24) that otherwise is identical to (5.2).

Having a representation of a non-parametric LDA model, the question arises how this generalises to other NoMM structures.

5.5.3 Non-parametric typology

In principle, each of the different types considered in Sections 5.2 to 5.4 may be subject to a non-parametric extension. However, from a first analysis HDP structures introduce some constraints that reduce the practical scope:

- All dependent latent variables (infinite edges) need to be part of the HDP hierarchy, and child DPs in this hierarchy form subsets of atoms of their parents.
- There is no control over the hyperparameters as an additional grouping dimension, i.e., N1B nodes are not directly possible. In such a case, alternative means of coupling DPs may be considered, including the dependent DP [MacEachern 1999].
- For mixture merging, no direct counterpart exists for DPs because each of the source DPs creates its own set of discrete probability masses (the elements of $\vec{\pi} \sim \text{GEM}(\gamma)$ above), which are disjoint from each other with probability 1.
- Sub-sequences of samples, E*S, as described in Section 5.3.1, require extensions to the DP, such as those shown by [Du et al. 2010].

Based on these considerations, Fig. 5.2 summarises possible extensions of the different structure types in the typology to non-parametric models (basically structured variants of the HDP model) and their restrictions. These extensions are expected to provide an interesting alternative to many of the structures presented previously, with many of the finite structures having a non-parametric counterpart.

NoMMs then become polymorphic in the sense that a given NoMM structure is equivalent to the Bayesian network of the parametric as well as the non-parametric specialisation where overloading of structures may be based on the stick-breaking representation of the DP.

In practice, however, the derivation of the corresponding inference approaches that estimate the posterior distribution (4.9) will create the actual challenges, and it is not guaranteed that viable or scalable solutions exist for all structures. In order to maintain the scope in this thesis, solving this is left to future work.

5.6 Conclusions

In this chapter, the structure of topic models has been analysed from the perspective of NoMMs. Generally, it turns out that models in the literature can be considered re-combinations of a relatively small set of basic substructures. Their properties may be identified and used to construct models in the future in a more systematic manner. Although the set of structures covered in this chapter is not considered a complete one, it fully covers the majority of the example models mentioned in this thesis.

A more detailed approach to using these structures in design will be given in Chapter 9 after inference is investigated in Part II of this thesis.

Part II

Inference

The Inference part develops numerical methods to train and test the generic models developed in the Analysis part. Chapter 6 derives a generic form of the collapsed Gibbs sampler and a method to quickly create implementations of these samplers. Complementing this, Chapter 7 develops a generic approach to variational inference for generic topic models. Finally, to address scalability problems, fast variants of the Gibbs sampling algorithms are studied in Chapter 8 based on acceleration of the serial sampling process, its parallelisms as well as independence assumptions.

Main contributions:

- Gibbs sampling for generic topic models → Chapter 6
- Gibbs meta-sampler via code generation → Chapter 6
- Variational inference for generic topic models → Chapter 7
- Generic serial and parallel accelerators for Gibbs sampling → Chapter 8

Chapter 6

Gibbs sampling in NoMMs

This chapter contributes a generic derivation of Gibbs sampling for topic models, which are here represented as NoMMs. In addition to deriving Gibbs full conditional distributions, formulations for predictive inference and convergence monitoring are given, all valid across NoMM structures. These findings are the basis for a Gibbs sampling tool that is able to produce model-specific algorithm source code based on an easy-to-use model specification language.[1]

6.1 Introduction

In Chapter 4, the inference problem of topic models has been formulated, and several approximate inference methodologies have been discussed as possible solutions. For a generic inference approach, a good method has feasible complexity with reasonable accuracy even when it comes to modelling dependencies between variables. With topic models specifically, Gibbs sampling has proven feasible (see, e.g., the PAM model [Li & McCallum 2006]) and will therefore be the first inference approach investigated for NoMMs.

Gibbs sampling [Geman & Geman 1984] is an approximative inference method suited for models where the marginals of the posterior can be expressed in closed form (even if the actual posterior cannot), in particular for high-dimensional discrete models. As a Markov-chain Monte Carlo (MCMC) method, Gibbs sampling uses a Markov chain that upon convergence approximately generates samples according to the posterior distribution. By sampling one dimension of the posterior at a time, Gibbs sampling avoids computationally complex Metropolis-Hastings acceptance calculations [Gilks et al. 1996]. Using notation from Chapter 4, one iteration in Gibbs sampling corresponds to sampling dependent hidden variables h_i for each data token v_i from the full conditional distribution, $h_i \sim p(h_i|H_{\neg i}, V, \Theta, A)$, where $\cdot_{\neg i}$ refers to the complete set of tokens except i. Analogously, $\vec{\vartheta}_k$ must be sampled in such an approach [Pritchard et al. 2000].

With topic models, it has been shown, however, that collapsed approaches to Gibbs sampling lead to particularly good convergence behaviour [Griffiths & Steyvers 2004]. In collapsed Gibbs sampling, the parameters Θ are integrated out [Liu 1994], and for topic models, the superior performance is attributed to removal of dependencies between parameters and variables assumed independent for the learning process.

[1] The main part of this chapter has been published in the paper [Heinrich 2009a].

The posterior considered for collapsed Gibbs sampling is:

$$p(H|V,A) = \int p(H,\Theta|V,A)\,\mathrm{d}\Theta\ . \tag{6.1}$$

The Markov state of the Gibbs sampler then reduces to H, and the resulting NoMM inference approach can be considered a form of stochastic EM algorithm [Jank 2005] that trains the latent variables H in its E-step and hyperparameters A in its M-step.

Chapter outline. In the remainder of this chapter, the Gibbs sampler will be derived generically in Section 6.2, and in Section 6.3 the resulting generic full conditionals will be discussed, followed by the associated parameter estimation methods in Section 6.4. A generalised method for predictive inference is presented in Section 6.5. Based on the findings of the previous sections, a method is presented in Section 6.6 to automatically generate source code for Gibbs sampling algorithms, and experimental results are outlined in Section 6.7.

6.2 Generic Gibbs sampling

In this section, we derive a formulation for generic Gibbs sampling. To keep the derivation compact, we assume the models to be restricted to Dirichlet–multinomial nodes with arbitrary hyperparameters or observed components.[2]

Taking the notation introduced in the Chapter 4, to sample from posteriors of NoMMs, for each independent latent variable H^ℓ (generic variables, complete sequence: upper case) with tokens $h_i^\ell \in H^\ell \triangleq \{h_{i'}^\ell\}_{i'\in I^\ell}$ (tokens: lower case; convention: $h_i^\ell \equiv h_{i^\ell}^\ell$ unless otherwise noted), a separate full conditional distribution $p(h_i^\ell|H_{\neg i}^\ell, H^{\neg\ell}, V, A)$ must be formulated for each token $h_i^\ell \in H^\ell$, with $\cdot^{\neg\ell}$ used analogous to $\cdot_{\neg i}$ to exclude NoMM levels. Typically, however, several hidden variables are dependent and need to be drawn as a block. Therefore, with dependency groups denoted by H^d with $H^\ell \subseteq H^d \subseteq H$ as sequences of groups of dependent tokens h_i^d, the full conditionals sought are: $p(h_i^d|H_{\neg i}^d, H^{\neg d}, V, A)$ for each group d and each token $i = i^d$. Remember that subscripts refer to sequence indices and superscripts to levels. Furthermore, note that h_i^d is a configuration of hidden variables that corresponds to a unique combination of components k^ℓ and outputs t^ℓ of the mixture levels involved.

To find the full conditional distributions, we start from the joint likelihood, (4.3), and for a collapsed approach integrate out its parameters via Dirichlet integrals:[3]

$$\begin{aligned} p(V,H|A) &= \prod_{\ell\in L}\left[\int \prod_{k=1}^{K} \frac{1}{\Delta(\vec{\alpha}_j)} \prod_{t=1}^{T} \vartheta_{k,t}^{n_{k,t}+\alpha_{j,t}-1}\,\mathrm{d}\Theta\right]^{[\ell]} \\ &= \prod_{\ell\in L}\left[\prod_{k=1}^{K} \frac{\Delta(\vec{n}_k+\vec{\alpha}_j)}{\Delta(\vec{\alpha}_j)}\right]^{[\ell]} \end{aligned} \tag{6.2}$$

where the level-specific $\vec{n}_k$ are vectors of "co-occurrence" counts $n_{k,t}$.

[2] The derivation follows [Heinrich 2009a] and covers structure classes N1–2, E1–3 and C1–2, with other types handled in Chapter 9.

[3] Alternatively using the definition of the Dirichlet distribution directly, the integrand at level ℓ becomes $\left[\prod_i \mathrm{Mult}(x_i|\Theta,k)\prod_k \mathrm{Dir}(\vec{\vartheta}_k|\alpha)\right]^\ell = \left[\prod_k \frac{\Delta(\{n_{k,t}\}+\alpha)}{\Delta(\alpha)}\mathrm{Dir}(\vec{\vartheta}_k|\{n_{k,t}\}+\alpha)\right]^\ell$, and all $\mathrm{Dir}(\cdot)$ terms integrate to 1.

This equation shows that the joint likelihood of the model variables is a product of ratios of Dirichlet partition functions for each component on each individual mixture level in the model. Interestingly, using the identity $\Gamma(a+n) = \Gamma(a)\prod_{c=0}^{n-1}(a+c)$ with real $a > 0$ and integer $n \geq 0$, we obtain a ratio of finite product sequences:

$$\frac{\Delta(\vec{a}+\vec{n})}{\Delta(\vec{a})} = \frac{\prod_{t=1}^{T}\prod_{c=0}^{n_t-1}(a_t+c)}{\prod_{c=0}^{[\sum_t n_t]-1}([\sum_{t=1}^{T} a_t]+c)}\,, \tag{6.3}$$

which for a unit difference in a single element u, $\Delta(\vec{a}+\delta(t-u))/\Delta(\vec{a})$, reduces to $a_u/\sum_t a_t$. Note that with (6.3), we can alternatively expand (6.2) into products without any special functions, which comes at the cost of obtaining denominator terms in (6.2) specific to components k.

The next step to obtain full conditionals is to determine dependent edges $H^d \subseteq H$ as subsets of the full edge set. As explained in Section 4.5, in NoMMs we can identify dependent hidden edges by finding subgraphs that extend through nodes whose component selection function, $g(\uparrow x_i, i)$, contains the respective hidden edges. Furthermore, edges that belong to different sequences i^ℓ are sampled in different dependency groups because there is no joint observation between tokens in the model. In the examples given in Section 4.2, ATM, PAM and hPAM models have dependent edges; LDCC does not because the segment and word topics are defined on different sequences, (m, s) and (m, s, n), and hidden variables are only connected via hyperparameters (assumed given in the full conditional).

Moreover, edges of nodes adjacent to subgraph H^d but independent of H^d are collected in a set $S^d \subset \{V, H\}$ with token sets s_i^d. We use the notation $\cdot^{\neg s}$ to denote the exclusion of S^d. With these definitions, full conditional distributions can be derived generically by applying the chain rule:

$$\begin{aligned}
p(h_i^d|H_{\neg i}^d, H^{\neg d}, V, A) &= \frac{p(h_i^d, s_i^d|H_{\neg i}^d, S_{\neg i}^d, H^{\neg d,\neg s}, V^{\neg s}, A)}{p(s_i^d|H_{\neg i}^d, S_{\neg i}^d, H^{\neg d,\neg s}, V^{\neg s}, A)} \\
&\propto p(h_i^d, s_i^d|H_{\neg i}^d, S_{\neg i}^d, H^{\neg d,\neg s}, V^{\neg s}, A) \\
&= \frac{p(H, V|A)}{p(H_{\neg i}^d, S_{\neg i}^d, H^{\neg d,\neg s}, V^{\neg s}|A)} \\
&= \prod_{\ell\in\{H^d,S^d\}}\left[\prod_{k=1}^{K}\frac{\Delta(\vec{n}_k+\vec{\alpha}_j)}{\Delta(\vec{n}_{k,\neg i^d}+\vec{\alpha}_j)}\right]^{[\ell]}\,.
\end{aligned} \tag{6.4}$$

This equation is illustrative for the functioning of the Gibbs updates. The hidden and visible values of the level set S^d are assumed known and control how the hidden values in the dependent levels H^d cluster. Over the training time, this converges to represent a stationary distribution of the Markov chain.

6.3 Generic full conditionals

In (6.4), all terms except those with a count difference between numerator and denominator cancel out. The remainder of terms can be simplified by applying (6.3) with $\vec{a} = \vec{n}^{\ell}_{k^{\ell},\neg i^{d}} + \vec{\alpha}^{\ell}_{j}$, and the resulting full conditional becomes a product of the following form if all mixture levels $\in \{H^d, S^d\}$ exclude only a single token with $\neg i^d$:

$$p(h^d_i | H^d_{\neg i}, H^{\neg d}, V, A) \propto \prod_{\ell \in \{H^d, S^d\}} \left[\frac{n_{k,t,\neg i^d} + \alpha_{j,t}}{\sum_{t=1}^{T} n_{k,t,\neg i^d} + \alpha_{j,t}} \right]^{[\ell]} . \tag{6.5}$$

The factors in (6.5) can be interpreted as posterior means of Dirichlet distributions with hyper-parameters $\vec{\alpha}_j$ and observation counts $\vec{n}_{k,\neg i^d}$, $\langle \mathrm{Dir}(\cdot | \vec{n}_{k,\neg i^d} + \vec{\alpha}_j) \rangle$ on level ℓ. Although this form of full conditional factors is prevalent in a majority of topic models, with the scope of models considered in this chapter alternative forms are possible:

1. If $g(\uparrow x_i, i)$ contains no hidden edges, the denominator can be omitted (e.g., nodes with m as only component index). This corresponds to component selection type C1A in Chapter 5.

2. If one index i^d of a mixture edge at the input of a node corresponds to an entire subsequence of i^{ℓ}, $\neg i^d$ excludes more than one token in the factor denominator in (6.4) (e.g., in LDCC, section topics $y_{m,s}$ aggregate word topic sequences $\{z_{m,s,n}\}_n$, or in the mixture model/Naïve Bayes classifier in Section 3.5.2, a document topic z_m aggregates multiple words of that document, $w_{m,n}$), which yields a factor analogous to (6.3):

$$\left[\frac{\prod_{t=1}^{T} \prod_{c=0}^{n_{k,t}-1} (c + \alpha_{j,t})}{\prod_{c=0}^{[\sum_{t=1}^{T} n_{k,t}]-1} (c + \sum_{t=1}^{T} \alpha_{j,t})} \right]^{[\ell]} . \tag{6.6}$$

3. Finally, mixture levels with observed parameters, node type N2, have components $\vec{\vartheta}_k$ as factors. In this case, few non-zero elements in $\vec{\vartheta}_k$ indicate the use of sparse representations, while symmetric non-zero values cancel out.

6.4 Parameter estimation

Generally, estimation of parameters and hyperparameters is part of an M-step dual to the Gibbs E-step in a stochastic EM procedure. It can be performed on a per-node basis in NoMMs.

6.4.1 Hyperparameters

In many topic models, hyperparameters are of decisive importance, e.g., to model data dispersion or to couple component groups. As there is no closed-form solution for estimation of Dirichlet parameters from count data, iterative or sampling-based approaches are commonly employed.

Estimation. Extending results from [Minka 2000] yields the following fixed-point iterations for node-specific standard and symmetric Dirichlet distributions that result in maximum likelihood estimates:[4]

$$\alpha_{j,t} \leftarrow \alpha_{j,t} \frac{\left(\sum_{\{k:f(k)=j\}} \Psi(n_{k,t} + \alpha_{j,t})\right) - K_j \Psi(\alpha_{j,t})}{\left[\sum_{\{k:f(k)=j\}} \Psi(\sum_{t=1}^{T} n_{k,t} + \alpha_{j,t})\right] - K_j \Psi(\sum_{t=1}^{T} \alpha_{j,t})} \,, \tag{6.7}$$

$$\alpha \leftarrow \alpha \frac{\left(\sum_{k=1}^{K} \sum_{t=1}^{T} \Psi(n_{k,t} + \alpha)\right) - KT\Psi(\alpha)}{T\left[\left(\sum_{k=1}^{K} \Psi([\sum_{t=1}^{T} n_{k,t}] + T\alpha)\right) - K\Psi(T\alpha)\right]} \,. \tag{6.8}$$

where $\Psi(x) = \mathrm{d}/\mathrm{d}x \log \Gamma(x)$ is the digamma function and level indicators ℓ are omitted. For the case $j \not\equiv 1$ we use $f(\uparrow X) = f(k)$ for notational simplicity. Each $\alpha_{j,t}$ then is estimated from K_j components for each of the J component groups. Estimators are initialised with a coarse-grained heuristic or a previous estimate and converge within few iterations.

Alternatively to this method, in [Minka 2000] various other approaches are given, and [Wallach 2008] gives extensions and references to alternative methods.

Sampling. Opposed to deterministic estimation, different sampling methods exist for Dirichlet hyperparameters. This allows to include the hyperparameters in a fully Bayesian setting. Specific methods include adaptive rejection sampling (ARS, [Gilks & Wild 1992]) based on the posterior $p(\alpha|X) \propto p(X|\alpha)p(\alpha|\theta_\alpha)$ where θ_α is the set of "hyper-prior" parameters and X the set of relevant visible and hidden variables. Although the Dirichlet distribution does not have a conjugate prior, the Gamma distribution exhibits valuable properties for a prior distribution and is adopted in many Bayesian settings. Beside the use of ARS that is able to sample from any distribution whose log is concave, which is fulfilled for $p(X|\alpha)\,\mathrm{Gam}(\alpha|\theta_\alpha)$. Other methods that allow to sample from standard distributions include slice sampling [Neal 2003](which samples uniformly in the "area" under the density [Wallach 2008]) as well as a method using auxiliary variables to simplify the distribution into standard distributions [Newman et al. 2009], adapting [Escobar & West 1995].

[4] Augmentation to MAP estimators with gamma prior $\mathrm{Gam}(a|\varkappa, \tau), a = \{\alpha_{j,t}, \alpha\}$ is possible by additions to the numerator and denominator: $a \leftarrow aP/Q \Rightarrow (aP + \varkappa - 1)/(Q + \tau^{-1})$.

6.4.2 Component parameters

Estimation of component parameters Θ is possible directly from the statistics of the collapsed state H and hyperparameters A. According to the model, for each level ℓ the parameters are distributed according to a Dirichlet posterior distribution conditioned on the counts $\vec{n}_k$: $\vec{\vartheta}_k \sim \mathrm{Dir}(\vec{\vartheta}_k|\vec{\alpha}_j, \vec{n}_k) = \mathrm{Dir}(\vec{\vartheta}_k|\vec{\alpha}_j + \vec{n}_k)$. A straight-forward estimate for the parameters is the posterior mean, which with (3.28) leads to the point estimate:

$$\vartheta_{k,t} = \frac{n_{k,t} + \alpha_{j,t}}{\sum_{t=1}^{T} n_{k,t} + \alpha_{j,t}} \tag{6.9}$$

where $\alpha_{j,t} \equiv \alpha$ for the symmetric case. Usually several samples $H^{(r)}, r \in [1, R]$ are taken from the stationary Markov chain with a sampling lag in between to ensure decorrelation. Finally parameters are averaged: $\vec{\vartheta}_k \approx R^{-1} \sum_r \vec{\vartheta}_k^{(r)}$.

This averaging process is used also for post-hoc Bayesian inference, i.e., working with the parameters as the posterior distributions beyond mere point estimates. To consider the variance of the parameters explicitly, a given post-hoc function of the parameters is averaged over different samples from the model. For instance, in order to estimate the distribution of some variable η conditioned on the observations V via Bayesian inference, $p(\eta|V)$, the model parameters Θ must be marginalised (cf. Section 3.4.2). This is done using the values of $\vec{\vartheta}_k$ sampled from the original Markov chain: $p(\eta|V) = \int p(\eta|\vec{\vartheta}_k) p(\vec{\vartheta}_k|V)\, \mathrm{d}\vec{\vartheta}_k \approx R^{-1} \sum_r p(\eta|\vec{\vartheta}_k^{(r)}) p(\vec{\vartheta}_k^{(r)}|V)$.[5]

6.5 Predictive inference

This section discusses predictive inference on the models. First, the information necessary to perform predictive inference is identified. Subsequently, this is applied to estimation of model quality and convergence indicators for the Gibbs sampler, applying and generalising the methods described in Section 3.7 from LDA to NoMMs.

6.5.1 Model parameters

In many applications, it is necessary to predict the topics of some query data set V' given the model $\mathcal{M}$ trained on the observations V. Regarding the information required to represent the model $\mathcal{M}$, two different types of node can be distinguished:[6]

- *Topic nodes*, $\ell \in L^*$, represent mixtures whose components are not specific to documents, i.e., $g(\uparrow x_i, i) \equiv g(\uparrow x_i)$, and $\mathcal{M}$ contains their parameters $\Theta^* = \{\Theta^\ell\}_{\ell \in L^*}$,
- *Sequence nodes*, $\ell \in L'$, represent mixtures specific to documents, and $\mathcal{M}$ contains their hyperparameters $A' = \{A^\ell\}_{\ell \in L'}$ that allow to find parameters $\Theta' = \{\Theta^\ell\}_{\ell \in L'}$.

Thus we can define $\mathcal{M} \triangleq \{\Theta^*, A'\}$, and finding the association of unseen data V' with a state H' can be achieved using Gibbs sampling with a predictive full conditional analogous to (6.4).

[5] A more in-depth discussion about different averaging approaches and their implications on the confidence and variance of the parameter samples is beyond scope. See for instance [Wallach et al. 2009b] for pointers on this.

[6] Note that this corresponds to the sequence and component plates discussed in Section 4.3.

However, now it is possible (1) to treat parameters of topic nodes Θ^* as observed and (2) to restrict sampling to the query state H' without M-step updates, which both accelerates convergence of H' compared to H:

$$p(h_i'^d | H_{\neg i}'^d, H'^{\neg d}, V', \mathcal{M}) \propto \prod_{\ell \in \{S^{*d}, H^{*d}\}} \left[\vartheta_{k,t}\right]^{[\ell]} \cdot \prod_{\ell \in \{S'^d, H'^d\}} \left[\prod_{k=1}^{K} \frac{\Delta(\vec{n}_k + \vec{\alpha}_j)}{\Delta(\vec{n}_{k,\neg i^d} + \vec{\alpha}_j)}\right]^{[\ell]} . \tag{6.10}$$

With this equation, all findings on generic full conditionals that were derived from the analogous (6.4) can be reused, including (6.5) and (6.6). Parameters can be estimated again using (6.9).

6.5.2 Convergence monitoring and model quality

Gibbs sampling and other MCMC methods pose the general problem to determine when their Markov chain reaches a stationary state that allows to sample from the posterior distribution. With standard convergence diagnostics [Robert & Casella 2004] difficult to apply to the high-dimensional discrete problem at hand, an alternative approach is to use some measure of model quality that reaches an optimum at convergence. Because of its wide applicability and frequent use of similar approaches in topic model evaluation, the likelihood of held-out test data given the trained model $\mathcal{M}$ (as defined in Section 6.5) has been chosen as a quality measure whose generalisation can be outlined as follows:

- For each sequence node of the network, the hidden state H' is trained on test data $V' = \{v_i'\}_i$ and hyperparameters A' according to (6.10), resulting in predictive parameters $\Theta' = \{\Theta'^{\ell}\}_\ell$.
- For each test-data token v_i', the likelihood given parameters $\{\Theta', \Theta^*\}$ is calculated:[7]

$$p(v_i' \,|\, \Theta', \Theta^*) = \sum_{h_i'} \prod_{\ell \in L} \left[\vartheta_{k,t}\right]^{[\ell]} \tag{6.11}$$

  where the sum over h_i' refers to marginalisation of all hidden variables. To calculate (6.11) efficiently, the NoMM is traversed level by level according to its generative process, multiplying the respective level parameters (elements of Θ'^{ℓ} or $\Theta^{*\ell}$) and summing over values of latent variables $h_i'^{\ell}$ not indexing components k^{ℓ} of child levels. Further, duplicate v_i' have identical likelihood.

- The log likelihood of held-out test documents is accumulated from the token likelihoods: $\mathcal{L}(V') = \sum_i \log p(v_i' \,|\, \Theta', \Theta^*)$.

Perplexity. As a variant, the test-set likelihood $\mathcal{L}(V')$ can be exponentiated and normalised with the number of tokens in the test data W' to obtain the perplexity: $\mathcal{P}(V') = \exp(-\mathcal{L}(V')/W')$, i.e., the inverse geometric mean of the token likelihoods. Both $\mathcal{L}(V')$ and $\mathcal{P}(V')$ are measures of how well a model is able to explain unseen data. Specifically, perplexity can be intuitively interpreted

[7] Exceptions: (6.11) excludes the case of sub-sequences that occur if a token i^{ℓ} at node input corresponds to multiple tokens $j \in i^{\ell}$ at its output, as described for E2 and E*S edge structures in Chapter 5. In this case, the parameter $\vartheta_{k,t}$ can be redefined as a product of sub-token likelihoods: $\vartheta_{k,t} = \prod_{j \in i^d} \vartheta_{k,t_j}$. This reduces to the plain $\vartheta_{k,t}$ if there is no subsequence, i.e. $j \equiv 1$, and expands to a hierarchy of products for recursive sub-sequences.

as the expected size of a vocabulary with uniform word distribution that the model would need to generate a token of the test data. A model that better captures co-occurrences in the data requires fewer possibilities to choose tokens given their context (document, etc.). Due to the stochastic nature of the states H and H', values of $\mathcal{L}(V')$ and $\mathcal{P}(V')$ are not strictly monotonic over iterations. Thus, convergence of their moving-average process can be used as indicator of Markov chain stationarity.

Document completion. Because held-out perplexity or likelihood directly are trained on the model, the model may adapt to the held-out documents while training parameters $\vec{\vartheta}'_m$ on observations V'. This may be used to measure how low a portion of a held-out document needs to be analysed by the model during training to predict the rest of that document. This has been shown by [Rosen-Zvi et al. 2004] for the author–topic model, demonstrating its superiority over LDA for small portions of documents to train $\vec{\vartheta}'_m$. The portion of documents analysed is a free parameter, but in standard situations, random 50% of the word tokens are typical, with all of the author information retained for training and testing in the author–topic model case.

Because the ability to generalise to new documents and use local information for adaptation (from partial documents) is generally desired, such a prediction method is useful for monitoring and quality measurements in general, as well. This leads to separating the tokens i into a set of analysed held-out tokens V'_{a} and predicted ones V'_{p}, where visible modalities that the model is conditioned on, e.g., authorship in the author–topic model above, may be retained in both V'_{p} and V'_{a}. The model then is trained using the union set of training documents and analysed held-out tokens, $V_{\mathrm{train}} = V \cup V'_{\mathrm{a}}$ and the held-out likelihood is run on the predicted held-out data, $V'_{\mathrm{predict}} = V'_{\mathrm{p}}$ using (6.11). Note that this bypasses (6.10) altogether, as held-out parameters $\vec{\vartheta}'_m$ may be learnt using the training equation, (6.4).

Training-set likelihood and "pseudo-perplexity". Analogous to measures on held-out data, in many practical cases it is possible to perform intermediate monitoring steps using the likelihood of the training data itself, which is proposed here. Because no additional estimation of held-out data topics needs to be computed, this measurement is rather efficient compared to test-set likelihood and "standard" perplexity. Using training-set likelihood to compute a perplexity measure as above may then be called "pseudo-perplexity" to distinguish it from the perplexity proper.

The information gained from the pseudo-perplexity is the extent to which the model was able to model the training data. It does not allow to draw conclusions on the ability to generalise to unseen data. However, as long as no overfitting occurs, the difference between both test and training likelihood are low.

Conversely, this relation makes pseudo-perplexity a novel indicator of overfitting: If the difference between perplexity and pseudo-perplexity values remains low, overfitting may be excluded because the model is able to predict training as well as test data similarly.

Empirical likelihood. As an alternative measure of model quality, an empirical test-set likelihood similar to [Li & McCallum 2006] can be used. In contrast to $\mathcal{L}$, Θ' are not trained using the model but a number of $\tilde{M}$ sequence-node parameter samples $\Theta'_{\tilde{m}}$ are sampled unconditionally from $\mathrm{Dir}(A')$. This makes the measure independent of the held-out document currently learnt, making the measure independent of adaptation effects. In the second step, the multiplication–marginalisation process outlined for the test-set likelihood is applied similarly: A mixture of $\tilde{M}$ unigram models $\vec{\varphi}_{\tilde{m}}$ is generated: multinomials over the observed vocabulary T^{ℓ} on each

Algorithm `genericGibbs(`V, V'`)`
Input: training and test observations V, V'
Global data: level-specific dimensions $K^H = \{K^\ell\}_{\ell \in H}$, $T^H = \{T^\ell\}_{\ell \in H}$, selection functions f and g, count statistics $N^\ell = [\{\vec{n}_k\}_{k=1}^K]^\ell$, $N^\ell \in N$ and their sums $\Sigma^\ell = [\{\sum_x n_{k,t}\}_{k=1}^K]^\ell$, $\Sigma^\ell \in \Sigma$ for each node with hidden parameters, memory for full conditional array $p(h_i^d|\cdot)$, likelihood $\mathcal{L}$
Output: topic associations H, parameters Θ and hyperparameters A
`// initialise`
for all nodes ℓ in topological order **do**
  random initialise hidden sequences $h_i^\ell \sim \text{Mult}(1/T^\ell)$, update counts N^ℓ and Σ^ℓ
`// Gibbs EM over burn-in period and sampling period`
while not (converged/burn-in completed and R samples taken) **do**
  `// stochastic E step to sample collapsed state`
  for all dependency groups $H^d \subseteq H$ **do**
    for all joint tokens $h_i^d \in H^d$ **do**
      decrement counts N^d and sums Σ^d according to current state h_i^d
      assemble array for $p(h_i^d|H_{\neg i}^d, H^{\neg d}, V)$ acc. to (6.5)
      sample new state $h_i^d \sim p(h_i^d|H_{\neg i}^d, H^{\neg d}, V)$
      increment counts N^d and sums Σ^d according to changed state h_i^d
  `// M step to estimate parameters`
  for all nodes ℓ **do**
    update hyperparameters A^ℓ acc. to Eqs. 6.7 and 6.8
  for all nodes ℓ **do**
    find parameters Θ^ℓ according to (6.9)
  `// optionally: monitor convergence using test data likelihood`
  $\mathcal{L} \leftarrow$ **call** `testLik(`Θ, A, V'`)` using Eqs. 6.9–6.11
  if ($\mathcal{L}$ converged/burn-in completed and L sampling iterations since last read out) **then**
    `// different parameter read-outs are averaged`
    $\bar{\Theta} \leftarrow \bar{\Theta} + \Theta$
`// Complete parameter average`
$\Theta = \bar{\Theta}/R$

Figure 6.1: Generic Gibbs sampling algorithm.

leaf level. Conditioned on these multinomials $\vec{\varrho}_{\tilde{m}}$, then the log empirical likelihood of held-out test documents $V' = \{v_i'\}_i$ with $i = (m, n)$ is determined: $\mathcal{L}_e = \sum_m \log(\tilde{M}^{-1} \sum_{\tilde{m}} \prod_n \varrho_{\tilde{m}, v'_{m,n}})$. The stochastic nature of states H and H' as well as the additional empirical sampling process again lead to non-monotonous increase of the empirical likelihood values over iterations.

Monitoring. Alternatively to using a quality metric to monitor the arrival at a stationary distribution, the "distance" between multiple parallel Markov chains may be used [Brooks & Gelman 1998], which multiplies the computational load for a model, though. Some type of distance may be obtained by comparing the inter-chain variance to the chain variance within the chains [Gelman & Rubin 1992].

Pre-defined burn-in period. As a simplistic variant of convergence management, it is possible to define a "reasonably" large number of "burn-in" samples and then assume convergence. The question for the "reasonably large" number is answered by prior tests with comparable data sets and model structures, monitoring convergence with one of the above methods. Despite the simplicity of the method, it has high relevance in practice.

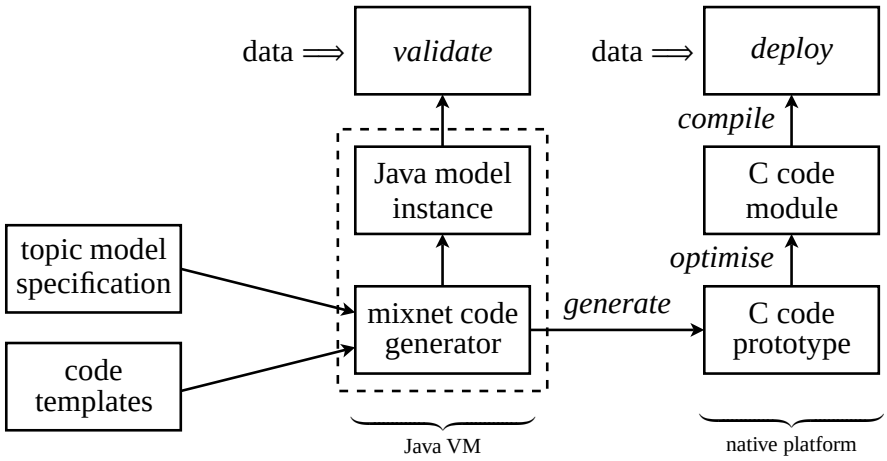


Figure 6.2: NoMM Gibbs sampler development workflow.

6.6 A Gibbs meta-sampler

Gathering all parts of the Gibbs sampler, a generic algorithm can be constructed, as outlined in Fig. 6.1. Across different NoMM models, the design follows a stochastic EM approach that after initialisation loops over alternating sampling (E) and hyperparameter estimation (M) steps until convergence, after which samples can be drawn from the simulated posterior.

Within the algorithm, the coherence of (6.5)–(6.11) across models leads to the conclusion that Gibbs sampler implementations can be achieved based on a small number of computation kernels. Few reusable kernels are desirable when targeting architectures that require high optimisation effort. In this section, a proof-of-concept implementation of a NoMM Gibbs sampler generator is outlined that, although it targets a CPU-based architecture, may be a basis for topic model implementation on massively parallel and FPGA-based architectures. For such architectures, this "Gibbs meta-sampler" may help reduce the algorithm-specific optimisation effort.

6.6.1 Workflow

The implementation of NoMM Gibbs samplers is based on a multi-stage workflow that allows construction of software modules with increasing levels of optimisation. This approach intends to keep the interface for the researcher simple while retaining flexibility with respect to target architectures. An overview of the workflow is given in Fig. 6.2.

The process starts with a NoMM specification, for which a domain-specific language [Stahl & Voelter 2006] has been developed that looks as in the example shown in Fig. 6.3. This NoMM script is fed to a Java-based NoMM code generator that has two modes of operation: (a) simulation of a Java-based instance of the Gibbs sampling class directly from the NoMM script, e.g., for model validation purposes, and (b) generation of the full sampler implementation as Java or C source code. The generated code can include user-defined templates and is ready for further optimisation and integration before it is compiled and deployed on the native computing platform.

```
mixnet = HPAM2

description:
    Hierarchical PAM model 2 (HPAM2)

nodes:
    # "active" K are X-1 and Y-1
    doc-suptop   = theta  : M, X         | alpha  : X
    top-subtop   = thetax : M, X, Y      | alphax : X, Y
    hiertop-word = phi    : 1 + X + Y, V | beta   : 1 : fixed

sequences:
    words = w : m, n : M, w[m].length : W

edges:
words = w ::
    document = m : M
    suptopic = x : X
    subtopic = y : Y
    word     = w : V

network:
    m    >>  theta  | alpha      >>  x
    m,x  >>  thetax | alphax[x]  >>  y
    x,y  >>  phi[k]              >>  w
    # java code to assign k
    k : {
        if (x==0) { k = 0; }
        else if (y==0) k = 1 + x;
        else k = 1 + X + y;
    }.
```

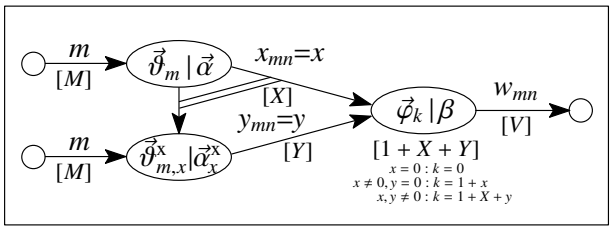


Figure 6.3: Example NoMM specification: hPAM2 [Li et al. 2007a], script and corresponding graphical notation (inset).

6.6.2 A NoMM scripting language

The NoMM scripting language may be considered a serialisation of the NoMM representation introduced in Chapter 4. Its central constituents can be recognised from its sections in the example script Fig. 6.3.[8] After definition of a model name, declarations of the different model parts are given, then the network structure itself is defined. The declarations in the script refer to both variable names to be generated in the final program code, as well as to the structure that the inference equations are derived from.

In particular, nodes are referenced by their parameter names and initially declared with hyperparameters and dimensions. Node dimensions have one or more variables, and the rightmost one equals the dimension of the outgoing edge, whereas all others refer to the set of component indexes (e.g., `thetax : M, X, Y` is an array of $M \times X$ components of dimension Y).

Sequences are declared by an edge variable (e.g., `words = w`) and associated index variables (e.g., `m,n`), along with the respective ranges and total token counts. Edges are then defined grouped by sequences they "run in" (e.g., `words = w ::`. For each edge, the variable and range are given (e.g., `x : X`).

[8] The generated model in Java can be found at `http://arbylon.net/resources.html`.

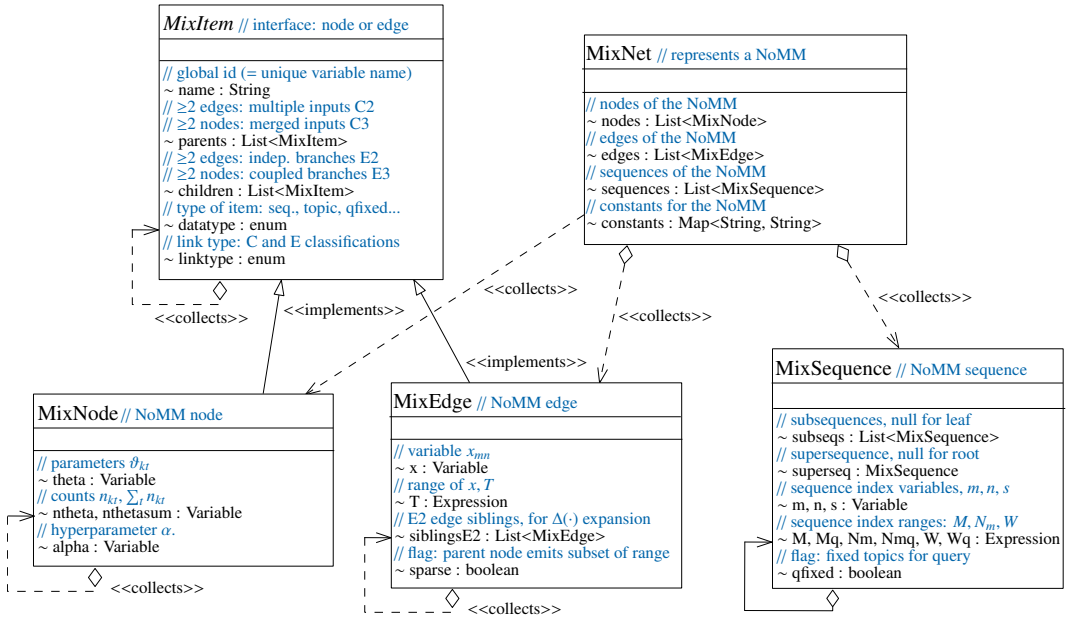


Figure 6.4: Simplified Java class diagram to model NoMMs in the Gibbs meta-sampler.

Finally, the network is specified node by node, with its input and output edges. Hyperparameters and indices are optional, as long as they have been declared in nodes and component indices are standard array indices (e.g., `m,x >> thetax[m,x]` and `m,x >> thetax` are equivalent in Fig. 6.3), but specific indexing is defined separately (e.g., `x,y >> phi[k]` and `k : {...}.` in Fig. 6.3).

This scripting approach allows full control over variable naming, including human-readable descriptions for all quantities. By injecting and commenting this information at the right places in the source code, algorithm readability is kept at a maximum.

A detailed description of the modelling language is given in Appendix C.2.2, including the Backus–Naur form (BNF) of the NoMM scripting language.

6.6.3 Meta-sampler design

The code generator of the "Gibbs meta-sampler" follows three design decisions: (1) to keep the structure of the program and program generators similar, which simplifies updates and extensions considerably, (2) to keep string manipulations central and exploit the possibilities of dedicated code parsers and regular expressions, and (3) to use the idea of "facets", i.e., functions that the code generator queries on the objects representing NoMM nodes, edges etc. and is returned code that can be emitted by a code renderer. A facet is designed to return different code for different callers.

Using these design decisions, the meta-sampler consists of five conceptual parts:

- *Data structure:* A NoMM is modelled as an object (of class MixNet) that links to collections of its structural parts: node, edge and sequence objects. Interrelations and aggregations are modelled using fields. The data structure is shown in Fig. 6.4 as a class diagram.

Function	Parameters/Return	Usage
main	(unused)	Driver with standard parameters and corpus (test and training set for all sequences including labels) according to model spec. Construct sampler, init main kernel, run a query process to obtain perplexity (initq, runq, ppx), run main sampler (run), rerun query process to analyse perplexity reduction, C: destroy sampler.
C: create, Java: <constructor>	Hidden dimensions, observed parameters and data, random number generator	Set up data and random numbers, allocate sampling space, determine sequence token counts, C: allocate all program variables
C: destroy	void	C: deallocate all program variables
init	void	Allocate and initialise all arrays for the main Gibbs sampler
initq	void	Allocate and initialise all arrays for the query Gibbs sampler
run	number of iterations	Main Gibbs sampler kernel, samples the training set through all sequences with standard sequence nesting. Implements (6.4)
runq	number of iterations	Query Gibbs sampler kernel, estimates topic-node parameters and samples the test set with standard sequence nesting. Implements (6.10)
estAlpha	void	Estimate hyperparameters during run. Implements (6.7) and (6.8)
ppx	return perplexity	Estimate sequence-node parameters from previous runq execution, calculate model perplexity. Implements (6.11)
checkState	return ok or error	Check consistency of hyperparameters, count variables and edge values at run-time. Auxiliary debugging function (optional).

Figure 6.5: Functions in the generated code.

- *Language parser:* The language parser creates a NoMM object graph from the NoMM scripting language.
- *Classifier:* The graph classifier detects the different NoMM structures defined in Chapter 5 from the NoMM object graph, checking structural constraints and missing information.
- *Inference engine:* The inference engine creates internal representations of the update and likelihood equations. For this, it calls facet methods in the structure classes (for the NoMM itself and NoMM nodes, edges and sequences) to convert the abstract structural description into concrete code representations. Because of the close resemblance between NoMM structure and inference equations, the inference engine can use traversals on the NoMM structures to infer the equations to implement.
- *Code renderer:* A code renderer transforms the facets of the NoMM objects into actual source code.

From the perspective of the typical approach to domain-specific languages [Stahl & Voelter 2006], the NoMM data structure takes the role of an abstract syntax tree (AST), a tree structure from

Model	Hidden vars.	Structures	Reference	Final LoCC	Man. adjustment / LoCC	Remark
LDA	1	—	Fig. 4.5(a)	378	0	—
ATM	2	N2	Fig. 4.5(b)	454	0	sparse parameters $\vec{a}_m$
LDCC	2	N1B, E1S	Fig. 4.5(c)	542	sequence nesting / 151	E1S = sub-seq. for segments
PAM4	2	C2A	Fig. 4.5(d)	521	0	C2A = C1A + C1B
PAM5	3	C2A	Fig. 4.5(d)	766	0	PAM4 + additional level
hPAM1	3	C2C, E3	Fig. 4.5(e)	786	querying / 118	ℓ not to be sampled in query
hPAM2	2	C2C, E3	Fig. 6.3	653	0	—
sLDA	1	N5+E4	(5.10)	523	aggregation branch / 131	regression library, App. C.3
MM-LDA	2	E2	(5.11)	441	0	—
Bi-LDA	2	C2B	(5.14)	437	sequence order / 41	indep. sequences
SRCM	2	C5B	(5.17)	502	sequence order / 44	graph data

Figure 6.6: Results of the code generator. LoCC refers to "lines of commented code", not including specialised libraries. Trivial structures N1, N2, E1 and C1 not mentioned.

which the source code can directly be generated by traversal. An explicit AST does, however, never have to be created in the Gibbs meta-sampler, as the inference engine directly transforms the NoMM representation into source code snippets.[9]

6.6.4 Generated algorithms

For a given NoMM script, the meta-sampler implements the generic algorithm as a Java class or C module that can be run instantly. On source code level, the generic algorithm (see Fig. 6.1) is structured into a small number of functions that cover all training and testing tasks on the model for standard settings. An overview is given in Fig. 6.5.

Important data structures in the generated code include the Markov state H, its count statistics as well as the arrays for multinomial sampling from the full conditional. The main computation kernels are those for full conditionals, (6.5) (including filling of the multinomial masses of $p(h_i^d|\cdot)$), for parameter estimation, (6.7)–(6.9), as well as for convergence monitoring (see Section 6.5.2).

6.7 Code generation results

In order to validate the code generator, it is tested on the models referenced in Chapter 4 and on various of those discussed in Chapter 5. Most of the structures covering Dirichlet–multinomial nodes (N1–N3), associated edge (E1–E3) and component index structures (C1–C5) work out of the box, including freely definable component selectors in Java of C (as in the C2B structure for the hPAM2 model in Fig. 6.3).

[9] For more details on the implementation, the concept of facet functions and options of the Gibbs meta-sampler, please refer to Appendix C.2.

```
/** run the main Gibbs sampling kernel */
public void run(int niter) {

// iteration loop
for (int iter = 0; iter < niter; iter++) {

// major loop, sequence [m][n]
for (int m = 0; m < M; m++) {
    // component selectors
    int mxsel = -1;
    int mxjsel = -1;
    int ksel = -1;

    // minor loop, sequence [m][n]
    for (int n = 0; n < w[m].length; n++) {
        double psum;
        double u;
        // decrement counts
        nmx[m][x[m][n]]--;
        mxsel = X * m + x[m][n];
        nmxy[mxsel][y[m][n]]--;
        nmxysum[mxsel]--;
        if (x[m][n] == 0)
            ksel = 0;
        else if (y[m][n] == 0)
            ksel = 1 + x[m][n];
        else
            ksel = 1 + X + y[m][n];
        nkw[ksel][w[m][n]]--;
        nkwsum[ksel]--;

        // compute weights
        /* p(x_{m,n} \eq x, y_{m,n} \eq y ... (LaTeX omitted) */
        psum = 0;
        int hx = -1;
        int hy = -1;
        // hidden edge
        for (hx = 0; hx < X; hx++) {
            // hidden edge
            for (hy = 0; hy < Y; hy++) {
                mxsel = X * m + hx;
                mxjsel = hx;
                if (hx == 0)
                    ksel = 0;
                else if (hy == 0)
                    ksel = 1 + hx;
                else
                    ksel = 1 + X + hy;
                pp[hx][hy] = (nmx[m][hx] + alpha[hx])
                        * (nmxy[mxsel][hy] + alphax[mxjsel][hy])
                        / (nmxysum[mxsel] + alphaxsum[mxjsel])
                        * (nkw[ksel][w[m][n]] + beta)
                        / (nkwsum[ksel] + betasum);
                psum += pp[hx][hy];
            } // for h
        } // for h

        // sample topics
        u = rand.nextDouble() * psum;
        psum = 0;
        SAMPLED:
        // each edge value
        for (hx = 0; hx < X; hx++) {
            // each edge value
            for (hy = 0; hy < Y; hy++) {
                psum += pp[hx][hy];
                if (u <= psum)
                    break SAMPLED;
            } // h
        } // h

        // assign topics
        x[m][n] = hx;
        y[m][n] = hy;

        // increment counts
        nmx[m][x[m][n]]++;
        mxsel = X * m + x[m][n];
        nmxy[mxsel][y[m][n]]++;
        nmxysum[mxsel]++;
        if (x[m][n] == 0)
            ksel = 0;
        else if (y[m][n] == 0)
            ksel = 1 + x[m][n];
        else
            ksel = 1 + X + y[m][n];
        nkw[ksel][w[m][n]]++;
        nkwsum[ksel]++;
    } // for n
} // for m

// estimate hyperparameters
estAlpha();

} // for iter
} // run()
```

Figure 6.7: Generated Gibbs kernel for hPAM2 model in Fig. 6.3.

However, more complex node structures (N4–N6) and interleaving sequences can be intricate to transform for the generator, and for such structures, the generation process is reduced to a basis model with the structures covered by the generator and then complemented by a manual implementation step for the remaining parts.

In Fig. 6.6, an overview of some generated test models from the literature is presented, showing the structures involved and the corresponding volume of the generated source code.[10] The LoCC counts are based on the Java versions because modifications in Java were easier to accomplish than in C. For generated C code, LoCC counts increase by roughly 5%, with the program structure being identical (cf. Fig. 6.5).

An example of generated code is illustrated in Fig. 6.7. The figure shows the sampling kernel of the hPAM2 model that is generated using the script in Fig. 6.3.[11] Compared to results of many other code generators, the Java code of this Gibbs kernel is easily readable and annotated to be extended or adjusted. It can be easily seen how the different variable structures in Fig. 6.3 are

[10] Some generated algorithms and NoMM scripts may be found at `http://arbylon.net/resources.html`.

[11] In Appendix C.1, details on multinomial sampling are given, which may help understand this source code.

converted to Java variables and update equations, for instance the full conditional on lines 46ff that implements (6.5), based on the hidden variables `hx` and `hy` (`x` and `y` with local prefix `h`), selectors `mxsel`, `mxjsel` and `ksel` (local suffix: `sel`) and observed data `w[m][n]`.

As discussed above, for some models the current generator only created an incomplete model and adjustments had to be done to make up for the model structures not covered. The amount of adjusted code is given in the table along with what had to be changed. The lines of change were calculated by the number of statements of the Unix `diff` utility to delete, modify or add lines. These adjustments are mostly due to complex interaction between different sequences (LDCC, SRCM, Bi-LDA) and structures not implemented in the generator.

Of course, it is worthwhile to consider extending the current meta-sampler implementation to support all of those models automatically. However, as the state-of-the-art advances, new approaches and extensions are expected that need to be manually added to the more "standard" structures generated out of the box. Therefore, the implementation favours a limited scope of generated structures with high maintainability thanks to good readability and expressive code structuring over modelling completeness that can at best be asymptotic.

6.8 Conclusions

In this chapter, Gibbs sampling full conditionals were derived based on the NoMM representation developed in Chapter 4 and relying on the typology made in Chapter 5. As a result, a generic Gibbs sampling algorithm was formulated and a "meta-Gibbs sampler" implementation was created, based on code generation for specific models.

With the results of this chapter, it is now possible (1) to determine the Gibbs full conditionals of NoMMs of a wide variety of topic models (cf. Chapter 5), and (2) to automatically implement a large portion of them using a simple domain-specific modelling language. By automatically generating Gibbs samplers, including convergence monitoring and querying routines, not only is development effort reduced, but also sources of errors may be limited to uncertain modelling assumptions and extensions of the generated code. Consequently, the results of this chapter serve as the basis for implementations in most of the remainder of this thesis.

Based on the generic formulation of the Gibbs sampler, questions of computing performance may be addressed by extending the meta-sampler to generate code with optimisations in order to improve scalability issues common with topic models. Such optimised code is often much more complex in structure as the corresponding unoptimised algorithm, and the advantages of automatic code generation in a meta-sampler may be leveraged even more effectively. Some work in this direction will be presented in Chapter 8 where generic approaches to scalable sampling algorithms are studied. Meanwhile, the next chapter will investigate an alternative to Gibbs sampling, variational inference, for the purpose of comparison and to demonstrate that the utility of the NoMM representation is not limited to collapsed Gibbs sampling.

Chapter 7

Variational inference in NoMMs

In this chapter, an alternative approach to inference in NoMMs is derived using variational inference. Opposed to Gibbs sampling, variational inference has not been widely used for more complex topic models. Central to the approach is the usage of a "topic field", a multi-way array of likelihoods for all latent configurations of a set of hidden variables that explicitly models their dependencies.[1]

7.1 Introduction

While Gibbs sampling as a Markov-chain Monte Carlo technique uses a stochastic approach to finding optima in the model parameter space, variational inference is a deterministic method. This allows, for instance, better convergence detection.

Re-using the notation from Section 4.6, variational inference [Beal 2003] (also: variational Bayes, ensemble learning) is an approximative inference technique that relaxes the structure of the posterior $p(H, \Theta|V)$ by a simpler variational distribution $q(H, \Theta|\Psi, \Xi)$ conditioned on sets of free variational parameters Ψ and Ξ to be estimated in lieu of H and Θ.

To understand the variational inference approach, it is necessary to recall and extend some quantities from Chapter 3. The expectation of a random variable (r.v.) X, $\langle X \rangle$, may be taken w.r.t. to a distribution $q(X)$ other than its actual distribution $p(X)$: $\langle X \rangle_{q(X)} = \int x\, q(X{=}x)\, \mathrm{d}x$ is the expectation of X w.r.t. distribution $q(X)$. Furthermore, recall the entropy of r.v. X, $\mathrm{H}\{X\} = -\int p(X{=}x) \log p(X{=}x)\, \mathrm{d}x$ and the Kullback–Leibler divergence between r.v.s X and Y, $\mathrm{KL}\{X||Y\} = \int p(X{=}x) \log p(X{=}x)/p(Y{=}x)\, \mathrm{d}x$.

Given these definitions, the variational inference approach is based on the finding that minimisation of the Kullback-Leibler divergence of the variational distribution $q(H, \Theta|\Psi, \Xi)$ to the true posterior $p(H, \Theta|V)$ is equivalent to maximisation of a lower bound on the log marginal likelihood, $\log p(V)$:

[1] The content of this chapter is based on the results in the paper [Heinrich & Goesele 2009].

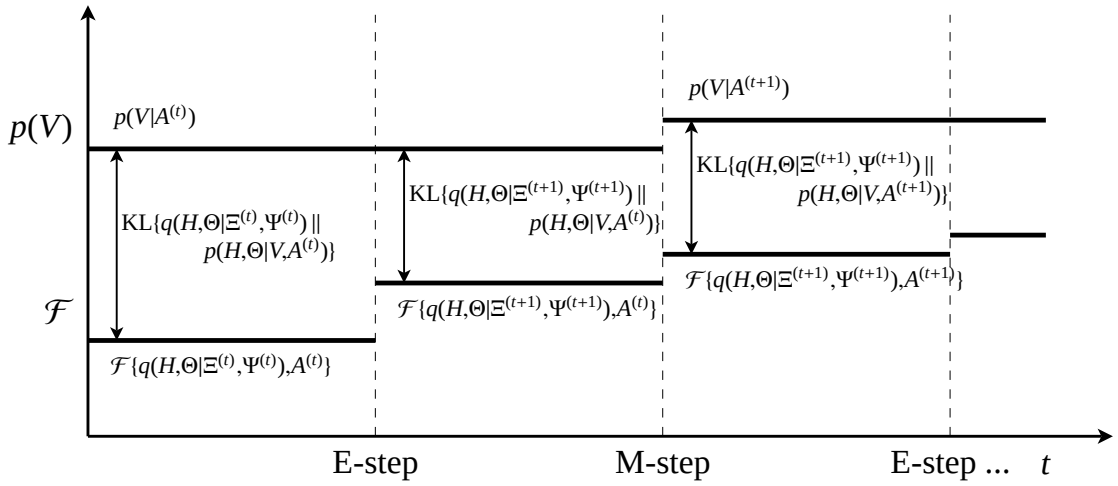


Figure 7.1: Variational inference: increasing the lower bound $\mathcal{F}$.

$$\log p(V) = \log \sum_H \int \frac{q(H,\Theta)}{q(H,\Theta)} p(V,H,\Theta)\,\mathrm{d}\vartheta \tag{7.1}$$

$$\geq \sum_H \int q(H,\Theta) \log \frac{p(V,H,\Theta)}{q(H,\Theta)}\,\mathrm{d}\vartheta \tag{7.2}$$

$$= \left\langle \log p(V) \right\rangle_{q(H,\Theta)} + \left\langle \log p(H,\Theta|V) \right\rangle_{q(H,\Theta)} - \left\langle \log q(H,\Theta) \right\rangle_{q(H,\Theta)} \tag{7.3}$$

$$= \log p(V) - \mathrm{KL}\{q(H,\Theta) \parallel p(H,\Theta|V)\} \tag{7.4}$$

$$= \left\langle \log p(V,H,\Theta) \right\rangle_{q(H,\Theta)} + \mathrm{H}\{q(H,\Theta)\} \triangleq \mathcal{F}\{q(H,\Theta)\} \tag{7.5}$$

where $\mathcal{F}\{q(H,\Theta)\}$ is the (negative) variational free energy – the quantity to be optimised.

Variational inference thus transforms the original problem of assigning hidden variables to observations into an optimisation problem that may be solved using an EM-like algorithm that alternates between an E-step that maximises $\mathcal{F}$ w.r.t. the variational parameters to pull the lower bound towards the marginal likelihood and an M-step that maximises $\mathcal{F}$ w.r.t. the true parameters to raise the marginal likelihood. By appropriate choice of the variational distribution q, this can be formulated to be computationally tractable. In Fig. 7.1, the increase of the lower variational bound in the variational inference algorithm is illustrated.

Chapter outline. In the following, the principle of variational inference is applied to NoMMs. In Section 7.2, a mean-field approach to variational inference is proposed and the free energy for generic NoMM structures derived. The resulting update equations and generic algorithm are then presented in Sections 7.3 and 7.4, respectively. Finally, an experimental evaluation of the approach in Section 7.5 compares performance with Gibbs sampling.

7.2 The "topic field"

Mean-field approximation. Following the variational mean-field approach that uses unconstrained variational distributions on the model parameters [Ghahramani & Beal 2000, Beal 2003], in the LDA model the variational distribution consists of fully factorised Dirichlet and multinomial distributions [Blei et al. 2003b]:[2]

$$q(\vec{z},\beta,\vartheta|\varphi,\lambda,\gamma) = \prod_{m=1}^{M}\prod_{n=1}^{N_m} \mathrm{Mult}(z_{m,n}|\vec{\varphi}_{m,n}) \cdot \prod_{k=1}^{K} \mathrm{Dir}(\vec{\beta}_k|\vec{\lambda}_k) \prod_{m=1}^{M} \mathrm{Dir}(\vec{\vartheta}_m|\vec{\gamma}_m) \; . \tag{7.6}$$

In [Blei et al. 2003b], this approach proved very successful, which raises the question how it can be transferred to more generic NoMM structures. Our approach is to view (7.6) as a special case of a more generic variational structure that captures dependencies $\uparrow X$ between multiple hidden mixture levels and includes LDA for the case of one hidden level ($H = \{\vec{z}\}$):

$$q(H,\Theta|\Psi,\Xi) = \prod_{\ell\in H}\left[\prod_i \mathrm{Mult}(x_i|\vec{\psi}_i,\uparrow x_i)\right]^{[\ell]} \prod_{\ell\in L}\left[\prod_k \mathrm{Dir}(\vec{\vartheta}_k|\vec{\xi}_k,\uparrow X)\right]^{[\ell]} \quad , \tag{7.7}$$

where $\ell \in H$ refers to all levels that produce hidden variables.

Assumptions. In the following, we assume that the indicator i is identical for all levels ℓ, e.g., words in documents $i^\ell = i \equiv (m,n)$. Furthermore, tokens i in the corpus can be grouped into terms v and (observable) document-specific term frequencies $n_{m,v}$ introduced. We use the shorthand $u = (m,v)$ to refer to specific unique tokens or document–term pairs. From the perspective of the typology developed in Chapter 5, we limit ourselves to node types N1 and N2, edge types E1–E3 as well as component index types C1 and C2.

Topic field. The dependency between mixture levels, $\uparrow x_u^\ell$, can be expressed by the likelihood of a particular configuration of hidden variables $\vec{x}_u{=}\vec{t} \triangleq \{x_u^\ell{=}t^\ell\}_{\ell\in H}$ under the variational distribution: $\psi_{u,\vec{t}} = q(\vec{x}_u{=}\vec{t}|\,\Psi)$. The complete structure $\underline{\psi}_u$ (the joint distribution over all $\ell \in H$ with $\Psi = \{\underline{\psi}_u\}_{\forall u}$) is a multi-way array of likelihoods for all latent configurations of token u with as many index dimensions as there are dependent latent variables. For instance, Fig. 4.5(a) reveals that LDA has one hidden variable with dimension K while PAM4 has two with dimensions $K \times L$. Because of its interpretation as a mean field of topic states in the model, we refer to $\underline{\psi}_u$ as a "topic field" (in underline notation to denote its multidimensional character).

We further define $\psi_{u,k,t}^\ell$ as the likelihood of configuration (k^ℓ, t^ℓ) for document–term pair u. This "marginal" of $\underline{\psi}_u$ depends on the mappings between parent variables $\uparrow x_u$ and components k on each level. To obtain $\psi_{u,k,t}^\ell$, the topic field $\underline{\psi}_u$ is summed over all descendant paths that $x_u{=}t$ causes and the ancestor paths that can cause component indices $k = g(\uparrow x_u, u)$ on level ℓ according to the model:

$$\psi_{u,k,t}^\ell = \sum_{\{\vec{t}_\mathrm{A}^\ell,\vec{t}_\mathrm{D}^\ell\}} \psi_{u;(\vec{t}_\mathrm{A}^\ell,k^\ell,t^\ell,\vec{t}_\mathrm{D}^\ell)} \; ; \quad \vec{t}_\mathrm{A}^\ell = \text{path causing } k^\ell \; , \vec{t}_\mathrm{D}^\ell = \text{path caused by } t^\ell \; . \tag{7.8}$$

[2] In [Blei et al. 2003b] this refers to the smoothed version; it is described in more detail in [Blei et al. 2003a].

Descendant paths $\vec{t}^{\ell}_{\mathrm{D}}$ of t^{ℓ} are obtained via recursion of $k = g(\uparrow x^{d}_{u}, u)$ over ℓ's descendant levels d. Assuming bijective $g(\cdot)$ as in the topic models in Fig. 4.5, the ancestor paths $\vec{t}^{\ell}_{\mathrm{A}}$ that correspond to components in parents leading to k^{ℓ} are obtained via $(\uparrow x^{a}_{u}, u) = g^{-1}(k)$ on ℓ's ancestor levels a recursively. Each pair $(\vec{t}^{\ell}_{\mathrm{A}}, \vec{t}^{\ell}_{\mathrm{D}})$ corresponds to one element in $\underline{\psi}_{u}$ per (k^{ℓ}, t^{ℓ}) at index vector $\vec{t} = (\vec{t}^{\ell}_{\mathrm{A}}, k^{\ell}, t^{\ell}, \vec{t}^{\ell}_{\mathrm{D}})$.

Free energy. Transforming (4.3), (7.7) and (7.8) and plugging them into (7.5), the free energy of the generic model becomes:

$$\begin{aligned}\mathcal{F} = \sum_{\ell\in L}\Bigg[\sum_{k}\log\Delta(\vec{\xi}_{k}) - \log\Delta(\vec{\alpha}_{j}) + \sum_{t}\Big(\Big(\textstyle\sum_{u} n_{u}\psi_{u,k,t}\Big) + \alpha_{j,t} - \xi_{k,t}\Big)\cdot\mu_{t}(\vec{\xi}_{k})\Bigg]^{[\ell]} \\ - \sum_{u} n_{u}\sum_{\vec{t}}\psi_{u,\vec{t}}\log\psi_{u,\vec{t}} = \sum_{\ell\in L}\mathcal{F}^{\ell} + \mathrm{H}\{\Psi\}\,,\end{aligned} \tag{7.9}$$

where $\mu_{t}(\vec{\xi}) \triangleq \Psi(\xi_{t}) - \Psi(\sum_{t}\xi_{t}) = \left\langle \log\vec{\vartheta}|\vec{\xi}\right\rangle_{\mathrm{Dir}(\vec{\vartheta}|\vec{\xi})} = \nabla_{t}\log\Delta(\vec{\xi})$, and $\Psi(\xi) \triangleq \mathrm{d}/\mathrm{d}x\log\Gamma(\xi)$ is the digamma function.[3]

7.3 Variational update equations

Based on the free energy, the actual update equations for the E-step and M-step can be derived generically by differentiation w.r.t. the variational parameters.

Variational E-steps. In the E-step of each model, the variational distributions for the joint multinomial $\underline{\psi}_{u}$ for each token (its topic field) and the Dirichlet parameters $\vec{\xi}^{\ell}_{k}$ on each level need to be estimated. The updates can be derived from the generic (7.9) by setting derivatives with respect to the variational parameters to zero, which yields:[4]

$$\psi_{u,\vec{t}} \propto \exp\left(\textstyle\sum_{\ell\in L}\left[\mu_{t}(\vec{\xi}_{k})\right]^{[\ell]}\right)\,, \tag{7.10}$$

$$\xi^{\ell}_{k,t} = \left[\Big(\textstyle\sum_{u} n_{u}\psi_{u,k,t}\Big) + \alpha_{j,t}\right]^{[\ell]} \tag{7.11}$$

where the sum $\sum_{u} n_{u}\psi^{\ell}_{u,k,t}$ for level ℓ can be interpreted as the expected counts $\langle n^{\ell}_{k,t}\rangle_{q}$ of co-occurrence of the value pair (k^{ℓ}, t^{ℓ}). The result in (7.10) and (7.11) perfectly generalises that for LDA in [Blei et al. 2003a].

M-steps. In the M-step of each model, the Dirichlet hyperparameters $\vec{\alpha}^{\ell}_{j}$ (or scalar α^{ℓ}) are calculated from the variational expectations of the log model parameters $\left\langle \log\vartheta_{k,t}\right\rangle_{q} = \mu_{t}(\vec{\xi}_{k})$, which as in the Gibbs sampler in Chapter 6 can be done locally for each mixture level because (7.11) has no reference to $\vec{\alpha}^{\ell}_{j}$ across levels.

Each estimator for $\vec{\alpha}_{j}$ (omitting level ℓ) should "see" only the expected parameters $\mu_{t}(\vec{\xi}_{k})$ of the K_{j} components associated with its group $j = f(k)$. We assume that components be associated

[3] Note the distinction between the function $\Psi(\cdot)$ and quantity Ψ. The connection between $\Psi(\cdot)$ and the expectation of a log multinomial w.r.t. $\mathrm{Dir}(\cdot)$ is explained in Appendix B.

[4] In (7.10) we assume that $t^{\ell}{=}v$ on final mixture level(s) ("leaves"), which ties observed terms v to the latent structure. For "root" levels where component indices are observed, $\mu_{t}(\vec{\xi}_{k})$ in (7.10) can be replaced by $\Psi(\xi_{k,t})$.

Algorithm `genericVEM(V)`
Input: training observations V as count statistics $\{n_u\}_u$, initialised variational parameters Ξ, topic field Ψ, hyperparameters A
Global data: level-specific dimensions $K^H = \{K^\ell\}_{\ell \in H}$, $T^H = \{T^\ell\}_{\ell \in H}$, selection functions f and g, variational free energy $\mathcal{F}$
Output: Parameters Θ and hyperparameters A
`// Main variational EM loop`
repeat
  `// Repeat E-step loop until convergence w.r.t. variational parameters`
  while free energy $\mathcal{F}$ (7.9) not converged **do**
    for each observed unique token u **do**
      for each configuration $\vec{r}$ **do**
        Compute var. multinomial $\psi_{u,\vec{r}}$ (7.10) or (7.14) left.
      for each (k,t) on each level ℓ **do**
        Compute var. Dirichlet parameters $\xi^\ell_{k,t}$ based on topic field marginals $\psi^\ell_{u,k,t}$ in (7.8) and (7.11), which can be done differentially: $\xi^\ell_{k,t} \leftarrow \xi^\ell_{k,t} + n_u \Delta\psi^\ell_{u,k,t}$ with $\Delta\psi^\ell_{u,k,t}$ the change of $\psi^\ell_{u,k,t}$.
  `// Perform M-step`
  for each j on each level ℓ **do**
    Compute hyperparameter $\alpha^\ell_{j,t}$ (7.12) or (7.13), inner iteration loop over t.
  for each (k,t) in point-estimated nodes ℓ **do**
    Estimate $\vartheta^\ell_{k,t}$ using (7.14) right.
until free energy $\mathcal{F}$ (7.9) converged between E-steps
for each level ℓ **do**
  Compute parameters Θ^ℓ from variational Ξ^ℓ using Dirichlet from (7.7).

Figure 7.2: Generic variational inference algorithm.

a priori (e.g., PAM4 in Fig. 4.5(c) has $\vec{\vartheta}_{m,x} \sim \mathrm{Dir}(\vec{\alpha}_x)$) and K_j is known. Then the Dirichlet ML parameter estimation procedure given in [Blei et al. 2003b, Minka 2000] can be used in modified form. It is based on Newton's method with the Dirichlet log likelihood function f as well as its gradient and Hessian elements g_t and h_{tu}:

$$
\begin{aligned}
f(\vec{\alpha}_j) &= -K_j \log \Delta(\vec{\alpha}_j) + \textstyle\sum_t (\alpha_{j,t} - 1) \sum_{\{k:f(k)=j\}} \mu_t(\vec{\xi}_k) \\
g_t(\vec{\alpha}_j) &= -K_j \, \mu_t(\vec{\alpha}_j) + \textstyle\sum_{\{k:f(k)=j\}} \mu_t(\vec{\xi}_k) \\
h_{tu}(\vec{\alpha}_j) &= -K_j \Psi'(\textstyle\sum_s \alpha_{j,s}) + \delta(t-u) K_j \Psi'(\alpha_{j,t}) = z + \delta(t-u) h_{tt} \\
\alpha_{j,t} &\leftarrow \alpha_{j,t} - (\underline{H}^{-1} \vec{g})_t = \alpha_{j,t} - h_{tt}^{-1} \left(g_t - (\textstyle\sum_s g_s h_{ss}^{-1}) \Big/ (z^{-1} + \textstyle\sum_s h_{ss}^{-1}) \right) .
\end{aligned}
\tag{7.12}
$$

Scalar α (without grouping) is found accordingly via the symmetric Dirichlet ML estimator:

$$
\begin{aligned}
f &= -K[T \log \Gamma(\alpha) - \log \Gamma(T\alpha)] + (\alpha - 1) s_\alpha , \quad s_\alpha = \textstyle\sum_{k=1}^K \sum_{t=1}^T \mu_t(\vec{\xi}_k) \\
g &= KT[\Psi(T\alpha) - \Psi(\alpha) + s_\alpha] , \quad h = KT[T\Psi'(T\alpha) - \Psi'(\alpha)] \\
\alpha &\leftarrow \alpha - g h^{-1} .
\end{aligned}
\tag{7.13}
$$

Model:		LDA			ATM			PAM		
Dimensions {A,B}:		$K = \{25, 100\}$			$K = \{25, 100\}$			$(K, L) = \{(5, 10), (25, 25)\}$		
Method:		GS	VB_{ML}	VB	GS	VB_{ML}	VB	GS	VB_{ML}	VB
Convergence time [h]	A	0.39	0.83	0.91	0.73	1.62	1.79	0.5	1.25	1.27
	B	1.92	3.75	4.29	3.66	7.59	8.1	5.61	14.86	16.06
Iteration time [sec]	A	4.01	157.3	164.2	6.89	254.3	257.8	5.44	205.1	207.9
	B	16.11	643.3	671.0	29.95	1139.2	1166.9	53.15	2058.2	2065.1
Iterations	A	350	19	20	380	23	25	330	22	22
	B	430	21	23	440	24	25	380	26	28
Perplexity	A	1787.7	1918.5	1906.0	1860.4	1935.2	1922.8	2053.8	2103.0	2115.1
	B	1613.9	1677.6	1660.2	1630.6	1704.0	1701.9	1909.2	1980.5	1972.6

Figure 7.3: Results of variational and Gibbs experiments.

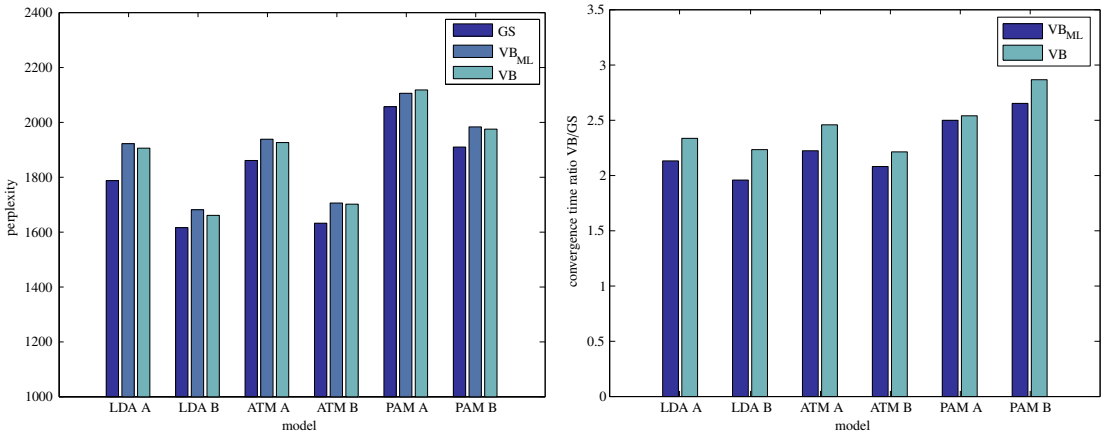
Variants. As an alternative to Bayesian estimation of all mixture level parameters, for some mixture levels ML point estimates may be used that are computationally less expensive (e.g., "unsmoothed" LDA [Blei et al. 2003b]). By applying ML only to levels without document-specific components (topic nodes), the generative process for unseen documents is retained. The E-step with ML levels has a simplified form of (7.10), and ML parameters ϑ^c are estimated in the M-step (instead of hyperparameters):

$$\psi_{u,\vec{t}} \propto \exp\left(\sum_{\ell \in L \setminus c} \left[\mu_t(\vec{\xi}_k)\right]^{[\ell]}\right) \cdot \vartheta^c_{k,t}\,, \quad \vartheta^c_{k,t} = \frac{\langle n_{k,t}\rangle_q}{\langle n_k\rangle_q} \propto \textstyle\sum_u n_u \psi^c_{u,k,t}\,. \tag{7.14}$$

Moreover, as an extension according to the typology in Chapter 5, it is straightforward to introduce observed parameters (type N2) that for instance can represent labels, as in the author–topic model, cf. Fig. 4.5. In the free energy in (7.9), the term with $\mu_t(\vec{\xi}_k)$ is replaced by $(\sum_u n_u \psi_{u,k,t}) \log \vartheta_{k,t}$, and consequently, (7.10) takes the form of (7.14) (left), as well.

7.4 Algorithm structure

The complete variational EM algorithm alternates between the variational E-step and M-step until the variational free energy $\mathcal{F}$ converges at an optimum. At convergence, the estimated document and topic multinomials can be obtained via the variational expectation $\log \hat{\vartheta}_{k,t} = \mu_t(\vec{\xi}_k)$. Initialisation plays an important role to avoid local optima, and a common approach is to initialise topic distributions with observed data: For each topic a small number of documents is selected whose words create an initial perturbation in Ξ via (7.11). Possibly several such initialisations can be used concurrently. In Fig. 7.2, the actual variational EM loop is outlined in its generic form.



(a) Perplexity.

(b) Convergence time ratio against GS.

Figure 7.4: Comparing perplexity and convergence time for LDA, ATM and PAM4.

In practice, similar to [Blei et al. 2003a], this algorithm can be modified by separating levels with document-specific variational parameters $\Xi^{\ell,m}$ (NoMM sequence nodes) and such with corpus-wide parameters $\Xi^{\ell,*}$ (NoMM topic nodes). This allows a separate E-step loop for each document m that updates $\underline{\psi}_u$ and $\Xi^{\ell,m}$ with $\Xi^{\ell,*}$ fixed. Parameters $\Xi^{\ell,*}$ are updated afterwards from changes $\Delta\psi^{\ell}_{u,k,t}$ accumulated in the document-specific loops, and their contribution added to the free energy $\mathcal{F}$.

7.5 Experimental study

This section presents empirical results achieved with concrete realisations of the generic algorithm in Section 7.4.

Setting. As evaluation models, some NoMMs from Fig. 4.5 have been selected: LDA, ATM and PAM4, and two versions of each were investigated: an unsmoothed version that performs ML estimation of the final mixture level (using (7.14)) and a smoothed version that places variational distributions over all parameters (using (7.10)).

Except for the component grouping in PAM4 ($\vec{\vartheta}_{m,x}$ have vector hyperparameter $\vec{\alpha}_x$), scalar hyperparameters were used. As a base-line, Gibbs sampling implementations of the corresponding models were used (based on the results of Chapter 6). Two criteria are immediately useful: the ability to generalise to test data V' given the model parameters Θ, and the convergence time (assuming single-threaded operation). For the first criterion, because of its frequent usage with topic models we use the perplexity, the inverse geometric mean of the likelihood of test data tokens given the model, as defined in Chapter 6. The log likelihood of test tokens $\log p(v'_u|\Theta)$, with $v'_u \in V'$, is obtained by (1) running inference with the training data and first half of the test documents, (2) running the inference algorithms on the second half of the data documents, which yields Ξ' and consequently Θ', and (3) marginalising all hidden variables h'_u in the likelihood $p(v'_u|h'_u, \Theta') = \prod_{\ell\in L}\left[\vartheta_{k,t}\right]^{[\ell]}$.[5]

[5] ATM perplexity was conditioned on the known authorship labels, see $\vec{a}_m$ in Figs. 4.5(b) and 4.6(b).

The experiments were performed on the NIPS corpus with $W = 2301375$ word tokens in $M = 1740$ documents (174 held-out) and $A = 2037$ authors. For details, see Appendix D.2.[6]

Results. The results of the experiments are shown in Fig. 7.3 with summary comparisons in Fig. 7.4. It turns out that generally the variational algorithms were able to achieve perplexity reductions roughly in the range of their Gibbs counterparts, which verifies the approach taken. Furthermore, the full variational Bayes (VB) approaches tend to yield slightly improved perplexity reductions compared to the ML versions. Yet the variational results were consistently weaker compared to the Gibbs baselines. This may be due to adverse initialisation of variational distributions, causing algorithms to become trapped at local optima.

It may alternatively be a systematic issue due to the correlation between Ψ and Ξ assumed independent in (7.7), a fact that has motivated the collapsed variant of variational Bayes in [Teh et al. 2007].

Considering the second evaluation criterion, the results show that the current VB implementations generally converge less than half as fast as the corresponding Gibbs samplers, as shown in Fig. 7.4(b). This is why more work needs to be undertaken in the direction of code optimisation, including parallelisation for multikernel CPUs.

7.6 Related work

The proposed representation of topic models as NoMMs makes work on variational inference in discrete DAG models relevant: In [Beal & Ghahramani 2006], a variational approach for structure learning in DAGs is provided with an alternative derivation based on exponential families leading to a structure similar to the topic field. They do not discuss mapping of components or hyperparameters and restrict their implementations to structure learning in graphs bipartite between hidden and observed nodes. Furthermore, [Li & McCallum 2006] present their pachinko allocation models as DAGs, but formulate inference based on Gibbs sampling. In contrast to this, the novelty of the work presented here is that it unifies variational inference of topic models using the NoMM representation and including labels, the option of point estimates and component grouping for variational Bayes, giving empirical results for real-world topic models.

7.7 Conclusions

In this chapter, variational inference algorithms for a large class of topic models have been derived analogously to the Gibbs sampling approach in the previous chapter. Based on a "topic field", a variant of the mean-field method of variational inference, variational update equations could be obtained in a generic way. These equations are the basis for an algorithm that can be applied to a range of topic models directly.

We have applied the algorithm to a couple of example models, verifying the general applicability of the approach. So far, especially more complex topic models have predominantly used inference based on Gibbs sampling. Therefore, this chapter is a step towards exploring the possibility of variational approaches in this context. However, what can be drawn as a conclusion from

[6] The experimental system was a Thinkpad T43 notebook (for details, see System 1 in Appendix D.1).

the experimental results, more work remains to be done in order to make variational algorithms as effective and efficient as their Gibbs counterparts.

Based on these results, here the decision is taken to choose Gibbs sampling as the primary inference method to be used in subsequent chapters of this thesis, continuing with an investigation of methods to speed up generic Gibbs samplers developed in Chapter 6.

Chapter 8

Scalable sampling for NoMMs

Based on the generic model of topic models and its Gibbs sampling formulation developed in Chapters 4 and 6, fast sampling schemes are introduced using serial and parallel methodologies. Furthermore, independent sampling of dependency groups is analysed for its convergence behaviour. The speed-ups achieved allow usage of models, especially with several dependent topics, for a much wider range of applications because scalability issues with existing implementations can be reduced.

8.1 Introduction

While the generic formulation of topic models as NoMMs and their associated inference methods may be considered a simplifying step, the real benefits of the NoMM representation and its associated code generation approach become clear as soon as the algorithms become more complex. Such an increase of complexity is typical when scalability considerations come into play. For topic models, this is especially indicated because their Gibbs samplers have a time and memory complexity that grows with the product of the number of topics on each of the dependent hidden variables in the model, T^ℓ, and that of the data points, W: $O(W \prod_\ell T^\ell)$. For the special case of LDA, several approaches have been published that allow accelerated processing, using both serial and parallel techniques, and the purpose of this chapter is to find similar and novel approaches for generic models.

LDA inference. As a prerequisite to study generic NoMMs, the Gibbs full conditional distribution of LDA is reviewed, as presented in Chapter 6. Recall the NoMM structure of LDA:

$$m \xrightarrow[[M]]{m} (\vec{\vartheta}_m \,|\, \alpha) \xrightarrow[[K]]{z_{m,n}=k} (\vec{\varphi}_k \,|\, \beta) \xrightarrow[[V]]{w_{m,n}=t} t \quad \{M, N_m\} \tag{8.1}$$

where m is the document index, n the word index, $z_{m,n}$ is the hidden topic and $w_{m,n}$ the observed word (term t), with corresponding parameters $\vartheta, \varphi, \alpha, \beta$, edge ranges M, K, V and sequence range

$\{M, N_m\}$. The single hidden dependency group is $H^d = \{\vec{z}\}$, and using (6.5) on (8.1) yields:[1]

$$p(z_{m,n}{=}k|w_{m,n}{=}t, \vec{z}_{\neg m,n}, \vec{w}_{\neg m,n}, \alpha, \beta) = \frac{1}{Z_{m,n}}(n_{m,k}^{\neg m,n} + \alpha)\frac{n_{k,t}^{\neg m,n} + \beta}{n_k^{\neg m,n} + V\beta} \tag{8.2}$$

where $n_{m,k}$ is the number of times that topic k is associated to document m, $n_{k,t}$ the number of times term t is associated with topic k, $n_k = \sum_t n_{k,t}$, and $\neg m, n$ means exclusion of observation (m, n). Moreover, the $\propto$ sign in (6.5) has been expanded to an equality with an explicit normalisation constant $Z_{m,n}$ ("partition function"), which is the sum of all weights:

$$Z_{m,n} = \sum_k p(z_{m,n}{=}k|w_{m,n}{=}t, \vec{z}_{\neg m,n}, \vec{w}_{\neg m,n}, \alpha, \beta) \ . \tag{8.3}$$

Notable properties. Once per iteration, the standard Gibbs sampler draws $k \sim p(z_{m,n}{=}k|\cdot)$ over the complete corpus, $\{m, n : m \leq M, n \leq N_m\}$. For this process, three observations can be made:

1. *Weight normalisation:* The normalisation constant $Z_{m,n}$ requires the full set of weights to be computed for $k \in [1, K]$. This constant is different for every token (m, n).

2. *Distribution sparsity:* Typically, the weights of the full conditional are concentrated at a few entries, with other $p(z_{m,n}{=}k|\cdot)$ being close to 0.

3. *Topic globality:* The counts $n_{k,t}$ and n_k are global to the complete corpus. This means that locally updating a document-specific topic $z_{m,n}$ influences the Gibbs full conditional weights of all other documents.

With respect to fast sampling schemes, these properties are decisive: The normalisation constant needs to be handled efficiently when speeding up the algorithm serially, for which topic concentration may be beneficial, while the globality of topics needs to be handled efficiently when accelerating the algorithm using parallelisation.

Another candidate for speed-up relevant for models with more than one dependent hidden variable is what we may call "independent sampling": It may be considered to resolve the dependency between variables and reduce the complexity from $O(W \prod_\ell T^\ell)$ to $O(W \sum_\ell T^\ell)$, which is significantly smaller. The question is, however, to what extent such a simplification will lead to local optima or overfitting phenomena and therefore the trained models will not reach qualities (as defined in Section 6.5.2) as high as their counterparts that heed dependencies during inference.

Chapter objectives and outline. This chapter will analyse different methods to speed up complex NoMM models by serial, parallel and independent sampling techniques, which partly applies existing work to more complex models and partly proposes new approaches, contributing to the application of NoMMs in domains with large data volumes or to make more tractable complex model structures that so far have been prohibitive in terms computational demand, even with approximate inference like Gibbs sampling.

Specifically, in Section 8.2, serial methods are investigated, proposing a generic form of a sampling method that approximates the normalisation constant of the Gibbs full conditional ($Z_{m,n}$

[1] As mentioned in alternative 1 for (6.5), the denominator of the $n_{m,k}$ term can be omitted for observed indices like document m.

in LDA) for NoMMs. In Section 8.3, parallel approaches to NoMM collapsed Gibbs samplers are discussed and a particular method is proposed. Subsequently, the influence of variable (NoMM edge) dependencies is analysed in Section 8.4. Finally, in Section 8.5 experiments on different acceleration techniques for some example NoMMs are undertaken.

8.2 Serial fast sampling

Serial methods for fast sampling speed up the algorithm itself without using parallelisms. This may include implementation aspects, such as loop unrolling or inlining of methods, but the most effective serial methods address the problem from a mathematical point of view. Such mathematically oriented methods have been proposed for LDA based on the mechanism of discrete sampling with the inversion method [Gentle 2003].[2] In SparseLDA [Yao et al. 2009], the structure of the LDA sampling distribution is mathematically divided into intervals responsible for different parts of the sampling mass, and only a part of the sampling weights needs to be calculated to obtain a sample if the partial masses are cached and sorted intelligently.[3] This is equivalent to keeping track of the partition function. To reduce the extra amount of data needed to keep the sorting structures in memory, a specific data structure has been proposed for SparseLDA that employs a portion of the bits in the count statistics array elements to encode their order.

Another approach, FastLDA [Porteous et al. 2008b], sorts the sampling masses directly, and when iterating through the sampling space the method avoids explicit computation of the expensive partition function. Instead, an upper bound on it is used that can be found efficiently.

Although SparseLDA reportedly performs better than FastLDA for the LDA model [Yao et al. 2009], it may not directly be applied to more complex models because it is based on the specific algebraic structure of the sampling distribution of LDA that becomes much more complex if generalised. FastLDA, on the other hand, directly retains the structure of the original sampling distribution and adds norms and sorting information in a more modular manner. It may further be combined with the order encoding of SparseLDA. We therefore focus on a "bound-based" method that generalises FastLDA.

8.2.1 Bound-based sampling

When sampling from a multinomial distribution, the most common approach is the inversion method: First a cumulative mass function (cmf) is assembled, and in a second step a uniform random value is "thrown" at the inverse of the cmf – hence the name. The final step is to search the component whose height in the cmf (or interval in the inverse) was hit by the random value. When sampling from distributions with a high number of components, the computation of the partition function (normalising constant) is a considerable effort, and consequently fast serial methods will be effective if they simplify its construction.

The method published in [Porteous et al. 2008b] avoids explicit calculation of the partition function by approximating it by an upper bound. As the sampling iterates through the components

[2] For a more complete description of discrete sampling methods, see Appendix C.1.

[3] The full conditional of LDA can be rewritten as: $p(z{=}k|\cdot) \propto (n_{m,k} + \alpha)\frac{n_{k,t}+\beta}{n_k+V\beta} = \frac{\alpha\beta+\beta n_{m,k}+(n_{m,k}+\alpha)n_{k,t}}{n_k+V\beta}$, resulting in three probability bins that need to be re-computed only in specific cases: on hyperparameters updates, once every document for $n_{m,k} > 0$ and once every document and term for $n_{k,t} > 0$. The third term commonly is the dominant one, and sorting by descending weight avoids the majority of iteration steps through k.

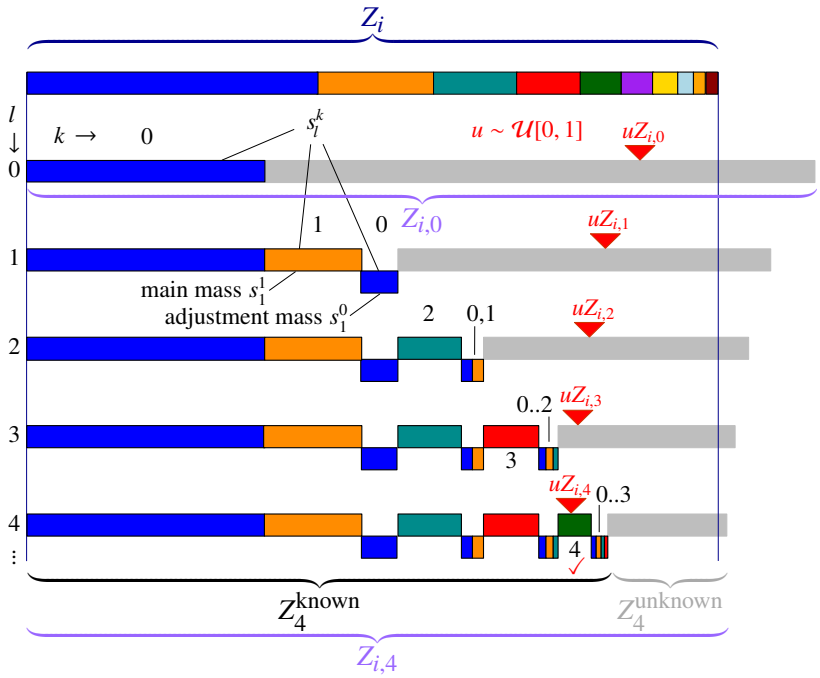


Figure 8.1: Overview of the bound-based sampling scheme.

k upwards and calculates their unnormalised masses, more and more of the true partition function is known, and at the ultimate element the upper bound converges to the true normalisation value of the sampling distribution. Although all intermediate search steps are approximations, the search process can be truncated as soon as the segment corresponding to the uniform random value is found, notably without ever calculating the full partition function but relying on its bound that makes sure that no component is assigned more mass than it would have in the true distribution. This shortcutting directly translates into a speed-up of the algorithm.

Rewriting (8.2), with $i = (m, n)$ for simplicity, the mass of component k for token i in the probability mass function (pmf) of the LDA model is:

$$\begin{aligned} p(z_i{=}k|\cdot) &= Z_i^{-1} \cdot (n_{m,k}^{\neg i} + \alpha) \cdot (n_{k,t}^{\neg i} + \beta) \cdot (n_k^{\neg i} + \beta V)^{-1} \\ &= Z_i^{-1} \cdot a_{m,k} \cdot b_{k,t} \cdot c_k = Z_i^{-1} \cdot w_{i,k} \end{aligned} \tag{8.4}$$

where $w_{i,k}$ is the unnormalised probability mass of $p(z_i{=}k|\cdot)$.

In principle, any bound on the partition function (and an associated mathematical inequality) can be used, but viability depends on three criteria: (1) The bound should be on a sum of product terms (to accommodate the structure of the full conditional), (2) given the knowledge on a portion of the terms, the bound should be close to the real partition function, and (3) the bound should be computable efficiently.

Although in principle other bounds could be used, [Porteous et al. 2008b] use Hölder's inequality, which turns out to match all criteria. Collecting the respective terms of the pmf of

LDA into vectors over k, $\vec{a}_m$, $\vec{b}_t$ and $\vec{c}$, and using Hölder's inequality in its generalised form,

$$\|\prod_j \vec{x}^j\|_r \leq \prod_j \|\vec{x}^j\|_{p_j} \text{ where } \sum_j p_j^{-1} = r^{-1} \tag{8.5}$$

with $r = 1$, the product of norms over these vectors is an upper bound on the partition function Z_i:

$$Z_i = \sum_{k=1}^{K} w_{i,k} \leq \|\vec{a}_m\|_{p_a} \cdot \|\vec{b}_t\|_{p_b} \cdot \|\vec{c}\|_{p_c} \tag{8.6}$$

for any set of norm orders $\vec{p} = \{p_a, p_b, p_c\}$ that fulfils $\|\vec{p}^{-1}\|_1 = 1$, including $\vec{p} = \{2, 2, \infty\}$, $\vec{p} = \{2, 4, 4\}$ and $\vec{p} = \{3, 3, 3\}$. Because this inequality also holds for subsets of elements in the norm (components k), the elements can be replaced iteratively by the true component masses. The sampling mass s_l for a topic l is:

$$s_l = w_{i,l} \cdot Z_{i,l}^{-1} \text{ with } Z_{i,l} = \underbrace{\sum_{k \leq l} w_{i,k}}_{Z_l^{\text{known}}} + \underbrace{\|\vec{a}_{m,k>l}\|_{p_a} \cdot \|\vec{b}_{k>l,t}\|_{p_b} \cdot \|\vec{c}_{k>l}\|_{p_c}}_{Z_l^{\text{unknown}}} . \tag{8.7}$$

Note that for a topic l, all $w_{i,k}$ with $k \leq l$ are known already and that $Z_{i,l} \geq Z_i$. Because masses s_l are normalised based on an inexact bound $Z_{i,l}$, their values need to be adjusted: After improving the bound to a new $Z_{i,l}$, all previous topic masses $k < l$ are complemented by masses s_l^k with the previous normalisation $Z_{i,l-1}$ replaced by $Z_{i,l}$:

$$s_l^k = w_{i,k} \cdot (Z_{i,l}^{-1} - Z_{i,l-1}^{-1}) \tag{8.8}$$

with $Z_{i,0} = 0$. Correspondingly, a sequence of partial mass segments is iterated that in its entirety exactly corresponds to the true normalised distribution, i.e., $\sum_l s_l^k = p(z_i{=}k|\cdot)$. The sequence alternates between calculating "main masses" $s_l^l \equiv s_l$ and "adjustment masses" s_l^k (contributing to topics $k < l$) until the cumulated segments are large enough to be hit by the random value, $u \sim \mathcal{U}[0, 1]$:[4]

$$s_0^0 \;|\; s_1^1 \;|\; s_1^0 \;|\; s_2^2 \;|\; s_2^0 \;|\; s_2^1 \;|\; s_3^3 \;|\; s_3^0 \;|\; s_3^1 \;|\; s_3^2 \;|\; \cdots$$

To eventually minimise the number of segments to be calculated for a complete sampling step, components are sorted in descending order of their masses $w_{i,k}$, giving the first topics iterated the highest probability of being hit. As observed in Section 8.1, the masses in typical topic distributions are similar to power-law distributions where few topics accumulate a major portion of the probability mass.

The complete process of sampling and approximating the normalising constant Z_i is depicted in Fig. 8.1. In this graphical representation, Z_l^{known} (coloured bins) stands for the weights already calculated, and Z_l^{unknown} (rightmost, borderless grey bins) for those that are approximated by the bound. For all refinement steps $l \in [1, K]$, the approximated $Z_{i,l} = Z_l^{\text{known}} + Z_l^{\text{unknown}}$ (cf. (8.7)) are guaranteed to be at least as large as the real normalising constant, Z_i (shown on top). Note also the adjustment of the sampled value $uZ_{i,l}$ through these refinements.

[4] Compared to [Porteous et al. 2008b], this re-arranges the sampling order to sample main masses before the corresponding adjustment masses.

8.2.2 Generic scalable serial sampling

In more complex topic models, the sampling space of dependent variables can be expected to be even sparser than in the relatively simple LDA model invesigated by [Porteous et al. 2008b]. Therefore, the idea of bound-based fast sampling promises to generalise well.

As the generic full conditional equation, (6.4), suggests, topic models with Dirichlet–multinomial distribution pairs can be written as products of different sampling terms. Per NoMM level, the product structure can be expressed as some function $q^\ell(k,t)$:[5]

$$q^\ell(k,t) = a^\ell_{k,t} \cdot c^\ell_k \text{ where } a^\ell_{k,t} = \left[n_{k,t,\neg i^d} + \alpha_{j,t}\right]^{[\ell]} \text{ and } (c^\ell_k)^{-1} = \left[\textstyle\sum_{t=1}^{T} n_{k,t,\neg i^d} + \alpha_{j,t}\right]^{[\ell]}, \quad (8.9)$$

with (6.5) and alternatives applying accordingly: c^ℓ_k is dropped for observed component indices k^ℓ_i, and $q^\ell(k,t) = \vartheta^\ell_{k,t}$ for observed parameters. The complete sampling distribution of the generic model then becomes:

$$p(h^d_i{=}\vec{t}\,|\cdot) = Z_i^{-1} \cdot \textstyle\prod_{\ell \in d} q^\ell(k,t) = Z_i^{-1} \cdot w^d_{i,\vec{t}}, \quad (8.10)$$

with $w^d_{i,\vec{t}}$ the product over a model-dependent number of terms $a^\ell_{k,t}$ and c^ℓ_k for all ℓ and $\vec{t}$ now the combination of topic values for the different dependent mixture levels (cf. the topic field index $\vec{t}$ introduced in Chapter 7). In fact, this vector can be represented linearly by nesting the different dimensions similar to the linear indexing of a multi-dimensional array. This linearisation allows to directly use the iteration over l of the bound-based sampling approach to multidimensional sampling spaces of the dependent latent token set h^d_i, thus retaining the sequence of partial masses s^k_l and s^l_l.

Another difference to the bound-based sampler for LDA is the scope of norms. To have the norms represent a bound on the joint sampling space of several hidden variables, they need to range over the complete sampling space. For instance, two terms $a_{m,k}a_{k,l}$ in the full conditional with a sampling space (and norm space) (k,l) ranging over $K \times L$ elements require norms $\|\{a_{m,k}\}_{l=1}^{L}\|_p^p = L\|a_{m,k}\|_p^p$ and $\|a_{m,k,l}\|_q^q$. Correspondingly, each element of $a_{m,k}$ is used L times when a Gibbs sampling distribution is calculated.

For the generalised serial sampling approach, there are two major questions: (1) Which combination of norms are computationally efficient and provide an efficient approximation for a given model? And: (2) How may components along the linearised sampling space, $w^\ell_{i,l}$, with l as a function of $\vec{t}$, be ordered to minimise the number of calculations?

Norm orders. Selection of norm orders is dictated mainly by Hölder's inequality, (8.5), and its computational consequences: The reciprocal sum is bound to be 1. Therefore, the more complex the model becomes, the higher orders of finite norms need to be taken, e.g., for a Gibbs full conditional with five terms (from three mixture levels like PAM4), such sets could be $\vec{p} = \{5;5,5;5,5\}$ or $\vec{p} = \{3;3,\infty;3,\infty\}$ where each element refers to the numerator and denominator norm order p_{a^ℓ} and p_{c^ℓ} of each node, respectively, and nodes are separated by semicolons for clarity.

Norms of high orders, like the 5-norm, appear prohibitive in terms of computational effort because not only must the norms be taken themselves but updated whenever a topic changes for a

[5] The notation $q^\ell(k,t)$ refers to k^ℓ and t^ℓ, respectively.

token. Furthermore, [Porteous et al. 2008b] could show for the case of LDA that the bound for the symmetric norm orders $(3;3,3)$ is consistently looser than that for the $(2;2,\infty)$. This motivates the usage of infinite norms also in multi-variable models.

Component sorting. With respect to component orders for the fast sampling process, the fact is used that the joint sampling space over dependent hidden variables h_i^d reflects the dependency structure between hidden variables best even if other terms are omitted. Thus one possible solution is to sort the linearly indexed space of topic combinations to be sampled. This is an alternative to more complex hierarchical sorting schemes.

To facilitate inner-loop computation, the usage of a single node's count statistics is considered, resulting in sorting over a significant part of the weights, rather than their entirety. That is, sorting will remain incomplete but with a tendency to catch high weights early, thus still potentially accelerating the algorithm. Otherwise, sorting over multiple dependent hidden variables would require updating products of several nodes, e.g., $\{(n_{x,y}+\alpha)(n_{y,z}+\beta)(n_{z,w}+\gamma)\}_{y,z}$ for hidden variables y and z instead of $\{n_{y,z}+\beta\}_{x,y}$. It then would be necessary to recalculate and cache the extra product.

Fast sampling state. In summary, a generic approach to fast sampling can be given by maintaining a "fast sampling state" F in addition to the regular Gibbs sampling state H (and the associated count vectors). This fast state F contains all information necessary to run the actual algorithm with the norm information necessary for bound-based sampling:

- Finite norms $\|\vec{a}^\ell\|_{p_{a^\ell}}$ and/or $\|\vec{c}^\ell\|_{p_{c^\ell}}$. It is simpler to omit the root operation in the values stored in order to be able to easier update them, i.e., to keep $\|\vec{a}^\ell\|_{p_{a^\ell}}^{p_{a^\ell}}$ etc. For simplicity, we refer to this as norm, nevertheless. Norms for a node are taken over all information hidden in the current dependency group, and there is one norm value stored for each possible value of an observed edge incident on the node. For instance, the node $(\vec{\vartheta}_{m,x}\,|\,\cdot)$ in the PAM4 model in Fig. 4.6(d) has a norm for each document m, each of which ranges over all supertopics x (as part of component index k^ℓ) and subtopics y (output value t^ℓ).

- Infinite norms $\max_t a_{k,t}^\ell$ and $\min_k c_k^\ell$. With the ordering described above, it is sufficient to track the first or last elements in an index array that keeps the ordering.

- The sampling order of the joint hidden variables. One ordering index per dependency group is used.

A node thus may have attached to it a finite norm, or rather a vector whose elements are the norms over hidden information for each observable configuration the node "sees" and up to two indices that keep the ordering of its hidden state: one for sorting weights and one for tracking infinite norms.

Algorithm. The algorithm directly extends that given in Fig. 6.1 for the "plain" generic Gibbs sampler. Now, initially each node in the NoMM is added fast-sampling information, which is initialised to the full norm values, therefore requiring to iterate the sample space of each node once. Furthermore, sort indices are created. When the algorithm is run, the $\neg i$ terms in the Gibbs sampling process need to be re-enacted by the norms for every token and every iteration, which requires one or two updates to all norms per iteration over tokens. Norm increments may

Algorithm `genericFastGibbs(`V, V'`)`
Input: training and test observations V, V'
Global data: level-specific dimensions $K^H = \{K^\ell\}_{\ell \in H}$, $T^H = \{T^\ell\}_{\ell \in H}$, selection functions f and g, count statistics $N^\ell = [\{\vec{n}_k\}_{k=1}^K]^\ell$, $N^\ell \in N$ and their sums $\Sigma^\ell = [\{\sum_x n_{k,t}\}_{k=1}^K]^\ell$, $\Sigma^\ell \in \Sigma$ for each node with hidden parameters, memory for full conditional array $p(h_i^d|\cdot)$, likelihood $\mathcal{L}$
Output: topic associations H, parameters Θ and hyperparameters A

```
// initialise regular state
```
for all nodes ℓ **do**
  random initialise hidden sequences $h_i^\ell \sim \text{Mult}(1/T^\ell)$
  update counts N^ℓ and Σ^ℓ

```
// initialise fast state
```
for all nodes ℓ **do**
  initialise norms $\|\vec{a}^\ell\|_{p_{a^\ell}}^{p_{a^\ell}}$ and $\|\vec{c}^\ell\|_{p_{c^\ell}}^{p_{a^\ell}}$ acc. to (8.6)
  sort components if infinity norm (minimum or maximum in elements of lower bound)

```
// Fast Gibbs EM over burn-in period and sampling period
```
while not (converged and R samples taken) **do**
  `// stochastic E step to sample collapsed state`
  for all dependency groups $H^d \subseteq H$ **do**
    for all joint tokens $h_i^d \in H^d$ **do**
      for all nodes $\ell \in d$ **do**
        `// regular sample decrement`
        decrement counts N^ℓ and sums Σ^ℓ according to current state h_i^ℓ
        `// fast sample decrement,` $\Delta x = -1$
        decrement norms $\|\vec{a}^\ell\|_{p_{a^\ell}}^{p_{a^\ell}}$ and $\|\vec{c}^\ell\|_{p_{c^\ell}}^{p_{a^\ell}}$ and save values in $\hat{a}^\ell$ and $\hat{c}^\ell$ (8.11)
        update sort indices as appropriate
      `// sample new` $h_i^d \sim p(h_i^d|H_{\neg i}^d, H^{\neg d}, V)$ `(Fig. 8.3, no need for explicit array` $p(h_i^d|\cdot)$`)`
      call `boundBasedSampler` $(N^\ell, \Sigma^\ell, \|\vec{a}^\ell\|_{p_{a^\ell}}^{p_{a^\ell}}, \|\vec{c}^\ell\|_{p_{c^\ell}}^{p_{a^\ell}} \forall \ell$, indices)
      for all nodes $\ell \in d$ **do**
        `// regular sample increment`
        increment counts N^ℓ and sums Σ^ℓ according to changed state h_i^ℓ
        `// fast sample increment,` $\Delta x = 1$
        increment norms $\|\vec{a}^\ell\|_{p_{a^\ell}}^{p_{a^\ell}}$ and $\|\vec{c}^\ell\|_{p_{c^\ell}}^{p_{c^\ell}}$ (8.11) or reassign $\hat{a}^\ell$ and $\hat{c}^\ell$ if $(k,t)^\ell$ unchanged
        update sort indices as appropriate
      increment counts N^d and sums Σ^d according to changed state h_i^d
  `// M step to estimate parameters`
  for all nodes ℓ **do**
    update hyperparameters A^ℓ
    update norms $\|\vec{a}^\ell\|_{p_{a^\ell}}^{p_{a^\ell}}$ and $\|\vec{c}^\ell\|_{p_{c^\ell}}^{p_{a^\ell}}$ ((8.11), $\Delta x = \Delta\alpha^\ell$)
    update component ordering if dependent on (vector) $\vec{\alpha}^\ell$
  `// optional convergence monitoring and parameter read-out`
  for all nodes ℓ **do**
    find parameters Θ^ℓ
  `// optionally: monitor convergence using test data likelihood`
  $\mathcal{L} \leftarrow$ **call** `testLik(`Θ, A, V'`)`
  if $\mathcal{L}$ converged and L sampling iterations since last read out **then**
    `// different parameter read outs are averaged`
    $\bar{\Theta} \leftarrow \bar{\Theta} + \Theta$

```
// Complete parameter average
```
$\Theta = \bar{\Theta}/R$

Figure 8.2: Generic fast Gibbs sampling algorithm.

Algorithm `boundBasedSampler`($N^\ell, \Sigma^\ell, \|\vec{a}^\ell\|_{p_{a^\ell}}^{p_{a^\ell}}, \|\vec{c}^\ell\|_{p_{c^\ell}}^{p_{a^\ell}} \forall \ell$, indices)

Input: regular and fast sampling states for token i: counts N^ℓ, Σ^ℓ, norms $\|\vec{a}^\ell\|_{p_{a^\ell}}^{p_{a^\ell}}$, $\|\vec{c}^\ell\|_{p_{c^\ell}}^{p_{a^\ell}}$, and indices

Output: sample h^d

for all nodes ℓ in dependency group d **do**
  `// save finite norms; assumed here for` a^ℓ`, infinite for` c^ℓ
  $\tilde{a}^\ell = \|\vec{a}_{\cdot}^\ell\|_{p_{a^\ell}}^{p_{a^\ell}}$

`// sample uniform`
$u \sim \mathcal{U}[0, 1]$
`// loop through sorted joint topic combinations` l
for topics $l \in [0, L)$ **do**
  `// compute partial terms and mass; index(`l`) maps` l `to original topic(s)`
  for all nodes ℓ in dependency group d **do**
    `// example for standard case in (8.9) and` $(k,t)^\ell$ `both hidden`
    $q^\ell(l) = a_l^\ell c_l^\ell$ with $a_l^\ell = \left[n_{\text{index}(l)} + \alpha_{j(\text{index}(l))}\right]^{[\ell]}$ and $(c_l^\ell)^{-1} = \left[\sum_{t=1}^{T} n_{\text{index}(l)} + \alpha_{j(\text{index}(l))}\right]^{[\ell]}$
    `// remove now-known element` $q(l)$ `from finite norms (keep` ∞ `norms)`
    $\tilde{a}^\ell = \tilde{a}^\ell - (a_l^\ell)^{p_{a^\ell}}$
  `// cumulated mass of known elements,` Z_l^{known}`, and bound on unknown element mass,` Z_l^{unknown}
  $Z_l^{\text{known}} = \prod_\ell q^\ell(l) + (l > 0)\ ?\ Z_{l-1}^{\text{known}} : 0$
  $Z_l^{\text{unknown}} = \prod_\ell (\tilde{a}^\ell)^{1/p_{a^\ell}} \cdot (\prod_\ell \min c_{\cdot}^\ell)^{-1}$
  `// partition bound` Z_l `adds known and unknown elements, cf. Fig. 8.1`
  $Z_l = Z_l^{\text{known}} + Z_l^{\text{unknown}}$
  `// if` u `in main or adjustment mass segments` s_l^k `with` $k \le l$
  if $u \le Z_l^{\text{known}} Z_l^{-1}$ **then**
    `// if adjustment mass` s_l^k `with` $k < l$ `(none for` $l = 0$`)`
    if $l > 0$ and $u \le Z_l^{\text{known}} Z_{l-1}^{-1}$ **then**
      `// offset of segments up to` s_{l-1}^{l-1}
      $u = u - Z_{l-1}^{\text{known}} Z_{l-1}^{-1}$
      `// adjustment factor`
      $v = (Z_l^{-1} - Z_{l-1}^{-1})^{-1}$
      for previous topics $k \in [0, l)$ **do**
        `//` $uv = (uZ_{l-1} - Z_{l-1}^{\text{known}}) Z_l (Z_{l-1} - Z_l)^{-1}$
        if $uv \le Z_k^{\text{known}}$ **then**
          `// adjustment mass` s_l^k
          $h_i^d = \text{index}(k)$
    else
      `// main segment` s_l^l
      $h_i^d = \text{index}(l)$

Figure 8.3: Fast bound-based Gibbs sampling kernel.

be formulated for the case that the underlying element k has already been incremented by a difference value Δx. For low orders p, computations are reduced by binomial expansions:[6]

$$\begin{aligned}
\Delta\|\vec{x}\|_p^p &= -(x_k - \Delta x)^p + x_k^p = \textstyle\sum_{q=0}^{p} \binom{p}{q} x_k^{p-q} (\Delta x)^q \\
\Delta\|\vec{x}\|_1^1 &= -(x_k - \Delta x) + x_k = \Delta x \overset{\Delta x=\pm 1}{=} \pm 1 \\
\Delta\|\vec{x}\|_2^2 &= -(x_k - \Delta x)^2 + x_k^2 = \Delta x(2x_k - \Delta x) \overset{\Delta x=\pm 1}{=} \pm 2x_k - 1 \\
\Delta\|\vec{x}\|_3^3 &= -(x_k - \Delta x)^3 + x_k^3 \overset{\Delta x=\pm 1}{=} -3x_k(1 \mp x_k) \pm 1 \\
\Delta\|\vec{x}\|_4^4 &= -(x_k - \Delta x)^4 + x_k^4 \overset{\Delta x=\pm 1}{=} \mp 4x_k(x_k(x_k \pm \tfrac{3}{2}) + 1) - 1 \\
\Delta\|\vec{x}\|_5^5 &= -(x_k - \Delta x)^5 + x_k^5 \overset{\Delta x=\pm 1}{=} \pm 5x_k(x_k(x_k(x_k \mp 2) + 2) \mp 1) \pm 1 \,.
\end{aligned} \tag{8.11}$$

Analogous to the norms, component orderings need to be constantly synchronised with the updates of the counts and hyperparameters.

Putting it all together, the generic serial fast sampling algorithm may be summarised as given in Fig. 8.2, with the actual sampling kernel presented in Fig. 8.3. This sampling kernel is a generalised variant of the one published by [Porteous et al. 2008b], using a different sequence of main and adjustment masses to sample weights first that are expected to be larger on average and making consequent use of the norm increments of (8.11).

8.3 Parallel fast sampling

Opposed to their serial counterparts, parallel accelerated sampling methods make use of specific parallel computing architectures, a wide variety of which exists that may be classified by the type of operations concurrently executed [Flynn 1972, Tanenbaum 2006]: MIMD (multiple instruction, multiple data) approaches allow several instructions at a time to be executed on several chunks of data, i.e., threads are independent of each other in terms of operations, whereas SIMD (same instruction, multiple data) architectures constrain this by requiring the same instructions to be executed on the different chunks of data. Moreover, different communication models can be distinguished for parallel architectures: While messaging allows loose coupling of a large number of computers that may be distant to each other, shared memory tends to have higher transmission bandwidths. For each of these combinations of architectural approaches, existing algorithms are suited to different degrees, and beside the systematic potential of an algorithm to be parallelised, the effectiveness of parallel approaches strongly depends on the architecture used, with its specific granularity of parallelism, starting from computing grids suited for less communication and MIMD approaches and ending at graphics processors (GPUs) with strict SIMD architecture and shared memory. At the center of this continuum, multi-core PC architectures can be found that support MIMD and partly SIMD operations (e.g., SSE instruction sets on Intel processors for parallel multiplication, etc.).

In the following, a number of approaches to handle the parallelisation of NoMM Gibbs samplers are discussed, starting with the parallelisation of the exact sampler, which raises some

[6] The form given here minimises the number of multiplications. For denominator norms, binomial expansions result in infinite series and are avoided for precision reasons.

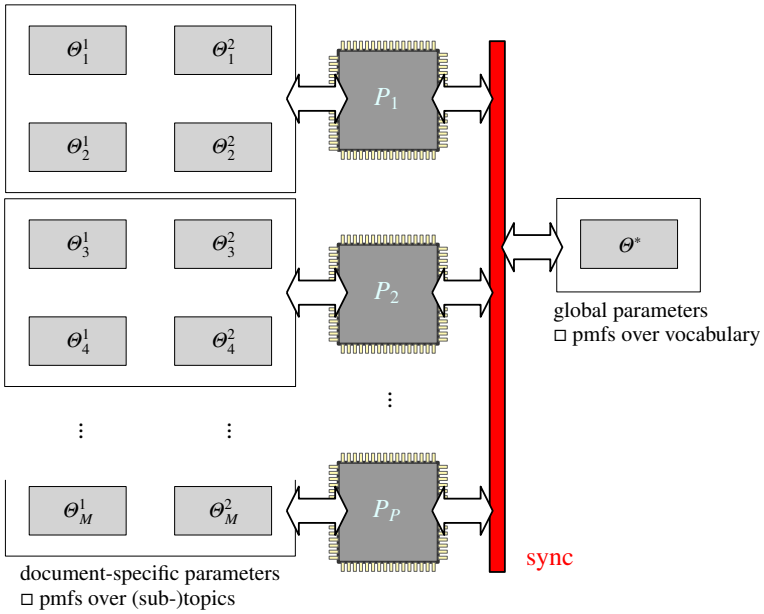


Figure 8.4: Processor communication for parallel sampling (example: PAM4).

synchronisation issues, and then several approaches that approximate the sampling "state" of the serial Gibbs sampler. All of these approaches are suited for MIMD with shared memory, while we will note which algorithms extend to distributed (grid) architectures.

8.3.1 Exact sampling with naïve synchronisation

As pointed out in Section 8.1, the second node $(\varphi\,|\beta)$ in the LDA model (8.1) is global to all documents, and its representation in the collapsed Markov state, $(n_{k,t}+\beta)/(n_k+\beta V)$ in (8.2), needs to be updated from all documents. Such global nodes are essential to topic models because they cover the co-occurrence between different contexts, and, following Section 5.4, we can refer to them as "topic nodes", Θ^*, to distinguish them from "sequence nodes", Θ_m, nodes that are attached to particular parts of the token sequence, m, such as documents. This situation is typical in collapsed Gibbs samplers for NoMMs. As an example, the communication between different parts of the model parameters (or their associated counts, respectively) in the PAM4 model (see Fig. 4.5(d)) is shown in Fig. 8.4.

The most straight-forward approach for parallel acceleration is to synchronise on the accesses to Θ^*, that is, to force atomic updates to all global counts $n^*_{k,t}$, for which different synchronisation methods may be used, such as locks, semaphores or atomic variables [Mattson et al. 2004]. In NoMM notation, for LDA this may be represented as:

$$m \xrightarrow[[M]]{m} (\vartheta^p_m\,|\,\alpha) \xrightarrow[[K]]{z_{m,n}=k} (\vec{\varphi}_k\,|\beta) \xrightarrow[[V]]{w_{m,n}} t\,, \tag{8.12}$$

where superscript $\cdot^p$ designates a quantity that is computed on processors with index p (for parameters $\vec{\vartheta}^p_k$, this refers to the associated count variables $\vec{n}_k$ and n_k, cf. (6.9)).

In effect, this method will be highly dependent on the capabilities of the computing platform to handle concurrent write accesses to memory, in this case that of the global variables associated to $\vec{\varphi}_k$, and it is clear that this can only work if parallel processing is MIMD with shared memory and high-bandwidth memory communication.

The advantage of such an approach, however, is that the models will be numerically very similar to their serial counterparts. The fact that identity of random number generation between serial and parallel versions[7] is difficult to accomplish is not a problem in practice because despite different random numbers per sample the model assumptions are identical, even though with identical random seeds different parameters are likely to be found.

Because we work with the full information of the state of the Markov chain and updates are almost identical compared to the serial version, we refer to this first approach to as *full-state exact* (FSE) parallelisation (or, depending on context, synchronisation). In Section 8.3.4, we will describe a version of the approach generalised to NoMMs.

8.3.2 Approximate sampling with full state

Direct synchronisation includes a systematic performance loss compared to unsynchronised approaches. Therefore, changes to the algorithm may be considered that resolve synchronisation points and may allow distribution of computation with reduced communication. In this context, especially the work of [Newman et al. 2006b] can be named, which contributes an approach to speed up the LDA Gibbs sampler on a computing cluster, following the MIMD concurrency model without the need for locally shared memory: approximate distributed LDA (AD-LDA).

AD-LDA. For P distributed processors, the AD-LDA approach uses P different LDA models that are run on subsets of the document set, and each processor starts with a topic count matrix, $n^p_{k,t}$, that is equal to the global $n_{k,t}$, but updates only $n^p_{k,t}$ as a local approximation of $n_{k,t}$. After each Gibbs sampling iteration, the total difference of the local $n^p_{k,t}$ is merged centrally into $n_{k,t}$ and re-distributed, which is also referred to as reduction.

In NoMM notation, this may be represented as:

$$m \xrightarrow[[M]]{m} (\vartheta^p_m \,|\, \alpha) \xrightarrow[[K]]{z_{m,n}=k} (\vec{\varphi}^p_k \,|\, \beta) \xrightarrow[[V]]{w_{m,n}} t \;. \qquad (8.13)$$
$$\vec{\varphi}_k = \langle \vec{\varphi}^p_k \rangle$$

Below a node in this notation, some rule for merging distributed states etc. may be given, as is shown under the $(\varphi_k \,|\, \beta)$ node: To obtain $\vec{\varphi}_k$ in the reduction step, the distributed $\vec{\varphi}^p_k$ are averaged with the expectation operator.

During this thesis, an approach very similar to AD-LDA has been developed independently of [Newman et al. 2006b], where the main difference is to keep the global $n_{k,t}$ but to store $\Delta n_{k,t}$ locally, which shifts some of the merging operations into the parallelised stage and was motivated

[7] This means, for instance, to draw $u = 0.51327737 \sim \mathcal{U}[0, 1]$ for $z_{103,38}$ in both a serial and parallel version of an LDA implementation. In pseudo-random number generators used in sampling, the sequence of draws $u \sim \mathcal{U}[0, 1]$ only depends on a seed value, making random sequences predictable and experiments repeatable. Thus, although a Gibbs sampler is seen as a random procedure, it is deterministic from the viewpoint of the random-number generator.

rather by the avoidance of synchronisation than distribution. We will refer to both approaches as AD-LDA, as their implementations and properties are almost identical if used on a shared-memory system.

As a statistically more principled variant, [Newman et al. 2006b] present a hierarchical model, "hierarchical distributed" LDA, HD-LDA, that has P LDA sub-models whose φ^p is sampled from a Dirichlet with parameters $\beta_k \psi_{k,t}$:

$$m \xrightarrow[[M]]{m} (\vartheta^p_m \,|\, \alpha) \xrightarrow[[K]]{z_{m,n}=k} \underset{\beta_k \sim \mathrm{Gam}(a,b);\ \vec{\psi}_k \sim \mathrm{Dir}(\gamma)}{(\vec{\varphi}^p_k \,|\, \beta_k \vec{\psi}_k)} \xrightarrow[[V]]{w_{m,n}} t\,. \tag{8.14}$$

Note that here $\vec{\beta}$ is a vector over topics k rather than terms t as in LDA. In the merging step, both $\vec{\beta}$ and ψ are estimated according to the generative models $\vec{\psi}_k \sim \mathrm{Dir}(\gamma)$ and $\beta_k \sim \mathrm{Gam}(a, b)$ (the gamma distribution) with careful setting of hyperprior parameters a, b and γ. Although conceptually simpler and an approximation (which Gibbs sampling is already by itself), AD-LDA has been reported to perform very close to HD-LDA in terms of perplexity reduction and retrieval metrics while being computationally more efficient.

Compared to Section 8.3.1, the AD-LDA approach still uses the full information of the Markov chain but approximates the true state by local copies. Therefore, we will refer to this approach as *full-state approximate* (FSA) parallelisation (or synchronisation) when we generalise it in Section 8.3.4.

Disjoint tokens. As a variant to FSA parallelisation, synchronisation may also be handled by managing the accesses to $n_{k,t}$ within the algorithm itself. Such an approach has been proposed for an SIMD architecture in [Yan et al. 2009]. Because m and $w_{m,n}{=}t$ are observed, concurrent access to $n_{k,t}$ is avoided by not only splitting documents m across processors p, formally into subsets $m \in M^p$, but also splitting the observed vocabulary into subsets of terms $t \in V^p$ a priori. Then a pair of document and term subsets is handled on each processor p, $\{m, t : m \in M^p, t \in V^q\}$ where q is used to permute the associations between document and term subsets: $q = (p + d) \bmod P$ with re-permutations according to $d \in [1, P]$ such that all their combinations can be sampled successively.

The approach uses the same approximation of n^p_k as AD-LDA but handles $n_{k,t}$ exactly. Such an algorithm is particularly suited to SIMD approaches with limited shared memory and the constraint to perform identical operations in the different processors, as shown by [Yan et al. 2009] on GPUs.

8.3.3 Approximate sampling with split state

Under a regime like MCMC that is already an approximation, one may argue that the approximative nature of its actual realisation like the thread-specific $n^p_{k,t}$ in the simple parallel method described above (a "second-order" approximation) is of little significance, as long as its empirical results are sufficiently good. Of course, this viewpoint is debatable since Gibbs sampling and MCMC have strong theoretical underpinnings [Liu 2001, Geman & Geman 1984]. Nevertheless, in practical situations the question is justified whether other approximations may even simplify the parallelisms and lead to similarly good results. One idea for such an approach is inspired by the way variational methods are realised, notably the original inference method in [Blei et al. 2003b]:

Here, the (global) topic-term distributions are represented by a set of counts, similar to that of the collapsed Gibbs sampler, but updated in an M-step separate from the E-step that performs the update of the document-topic distributions and document-wise term-topic distributions (also cf. Chapter 7):

$$m \xrightarrow[[M]]{m} (\vartheta_m^p \,|\, \alpha) \xrightarrow[[K]]{z_{m,n}=k} \underset{\vec{\varphi}_k \sim \mathrm{Dir}(\langle \vec{n}_k^p \rangle + \beta)}{(\vec{\varphi}_k \,|\, \cdot)} \xrightarrow[[V]]{w_{m,n}} t \; . \tag{8.15}$$

While this approach is in line with the theory of variational EM[8], including point estimates of parameters in the M-step, for the Gibbs sampler it is actually a sacrifice of its purely Bayesian character for the sake of a clear parallelism and potentially accelerated E-step computations: Topic counts are used to estimate parameter φ, which remains constant over an iteration, and the Gibbs sampler is used to fit the document-specific counts to φ. For this, a full conditional distribution similar to that of a query sampler is used, applying (6.10) to (8.1) but now conditioned on φ as the current state estimate during training:[9]

$$p(z_{m,n}{=}k \,|\, w_{m,n}{=}t, \vec{z}_{m,\neg n}, \vec{w}_{m,\neg n}, \varphi, \alpha) \;\propto (n_{m,k} + \alpha)\, \varphi_{k,t} \; . \tag{8.16}$$

In the newly introduced M-step, the parameter φ is re-estimated from the token sequence by a corpus-wide reduction:

$$\varphi_{k,t} \propto \beta + n_{k,t} = \beta + \sum_m \sum_n \delta(z_{m,n} - k)\, \delta(w_{m,n} - t) \; , \tag{8.17}$$

which applies (4.5) and (6.9). An interesting aspect of this is the similarity to the topic field in Chapter 7 and specifically (7.11): The variational topic field ψ that represents a distribution over possible values of latent variables is now replaced by the respective topic count that represents samples of such values drawn from the full conditional.

By "splitting" its state into a document-specific collapsed part (corresponding to $\vec{z}_{m,\neg n}$) and a global part that uses a point estimate of the parameter (corresponding to φ), the algorithm uses only part of the information available in the system and therefore only will achieve a more coarse-grained approximation than the AD-LDA approach. Such a "split-state" Gibbs sampler might be more prone to local optima (one of the drawbacks of point estimates) because it is less capable to cross weak regions. However, it has a much more straight-forward parallelism and a smaller computational footprint because the update equation does only contain the actual (uncollapsed) parameter by shifting a part of the computational load to the typically much faster reduction step.

The sequence of updates can be done analogous to the variational approach in an EM-like schedule:

- (E) Running a full iteration through the corpus using (8.16) conditioned on φ and
- (M) Updating hyperparameters and estimating φ using (8.17).

[8] Note that this approach has also been adopted for the variational generalised model in Chapter 7.

[9] The full conditional is now conditioned only on the current document $\vec{w}_m$ and associated topics $\vec{z}_m$ with word n removed, the rest of the information is contained in the estimate of φ.

In the next section, we will generalise this approach as the *split-state approximate* (SSA) parallelisation (or synchronisation).

8.3.4 Generic parallel scalable sampling

In this section, we discuss the extension of the LDA-based parallelisation approaches discussed above to more generic model structures. In principle, such generalisations may be achieved as follows: In a given model (NoMM), all sequence nodes are distributed between sub-processes, creating nodes Θ^p_m, and the different techniques for global parameters in topic nodes Θ^* can be handled using the different synchronisation techniques introduced above:

- *Full-state exact (FSE):* Generally, as models become more complex in terms of per-sample computations, the amount of time needed for synchronisation will decrease relatively to that for computations. In NoMMs with more than one dependent hidden variable, the adverse effects of standard synchronisation are therefore expected to drop, as they do for larger numbers of topics in the case of AD-LDA and distributed approaches. Synchronisation is on all accesses to the count statistics of topic nodes, like $n_{k,t}$ and n_k in LDA.

- *Full-state approximation (FSA):* Extending AD-LDA to full-state approximate samplers ("AD-NoMMs") is proposed as follows: All sequences that exist in a given model are distributed over processors, along with the counts of the sequence nodes. All counts for topic nodes are copied to processors, summing them to the global distributions after each iteration in an M-step reduction, node by node.

- *Split-state approximation (SSA):* Similar to the full-state approximate model, an M-step reduction is introduced that re-estimates parameters according to the local states. For the E-step, in (6.4) the observed parameter case is used for all topic nodes, which in fact is equivalent to the approach usually taken for querying topic models.

  The full conditional for split-state sampling in NoMMs is identical to the predictive full conditional (6.10), but now conditioned on the current state estimate $\hat{\mathcal{M}} \triangleq \{\hat{\Theta}^*, A\}$:

$$p(h_i'^d | H'^d_{\neg i}, H'^{\neg d}, V, \hat{\mathcal{M}}) \propto \prod_{\ell \in \{S^{*d}, H^{*d}\}} \left[\hat{\vartheta}_{k,t}\right]^{[\ell]} \cdot \prod_{\ell \in \{S'^d, H'^d\}} \left[\prod_{k=1}^{K} \frac{\Delta(\vec{n}_k + \vec{\alpha}_j)}{\Delta(\vec{n}_{k,\neg i'^d} + \vec{\alpha}_j)}\right]^{[\ell]} \tag{8.18}$$

  with H'^d denoting NoMM levels with sequence nodes and other notational conventions as described in Chapter 6. The estimate for the topic node parameters can then be obtained using the expectation of the Dirichlet priors, analogous to (6.9) or (8.17), respectively.

8.4 Independent sampling

One of the main reasons of poor scalability in NoMMs is the requirement to compute and sample over the complete set of combinations of dependent hidden variables in the inner loop of the sampling algorithm. This results in time and memory complexity that grows with the product of the number of topics, $\prod_{\ell \in H^d} T^\ell$. To resolve this problem, one may draw inspiration from mean-field methods that resolve dependencies to simplify variational inference (cf. Chapter 7). The question is, however, what impact this resolution of dependent hidden variables has on the general optimisation behaviour of the Gibbs samplers. It is known that variational approaches achieve inferior results compared to Gibbs sampling, which may be due to resolving the dependencies between model variables [Blei et al. 2002]. A reasonable hypothesis to explain this is that the algorithms will be more likely to get stuck in local optima and therefore will produce inferior results to the full block samplers.

However, each iteration will have significantly lower computational requirements because each dimension will simply add to the size of the total sampling space instead of multiplying, turning the time complexity from $\prod_{\ell \in H^d} T^\ell$ to $\sum_{\ell \in H^d} T^\ell$ and the memory complexity to $\max T^\ell$, an enormous reduction.

For example, considering PAM4 with two hidden variables and dimensions $\{20, 20\}$, the effective sampling space for the block sampler is $20 \cdot 20 = 400$ while that of the independent samplers is $20 + 20 = 40$, a ratio of 10. Considering the number of floating point operations (flops), for the block sampler with (6.5) we have $(3(2[+] + 1[/]) + 2[\times]) \cdot (20 \cdot 20) = 4.4$ kflops vs. $(2(2[+] + 1[/]) + 1[\times]) \cdot (20 + 20) = 280$ flops in the group of independent samplers (all per iteration and token), a factor of more than 15.7. This figure is of course theoretical because it ignores integer increments, memory access etc. However, it shows the potential of trading more iterations for significant reduction in per-iteration flops.

Another great advantage of the approach to sample hidden variables independently is that the overall Gibbs sampler can be separated into several Gibbs samplers with a single hidden edge and its incident mixture nodes, which especially in unbranched models (E1 and C1 NoMM structure types) is identical to the LDA Gibbs sampler. Not only simplifies this the algorithm significantly, but also it is possible to directly reuse findings that have been made for LDA regarding serial and parallel sampling approaches rather than generalising them, which results in higher complexity.

8.5 Experimental study

In order to determine the potential of scalable NoMM Gibbs sampling methods, various experiments have been performed. The goal in particular was to determine to what extent:

1. The results of some of the LDA-based methods are transferrable to more generic NoMM structures, (a) for bound-based sampling, (b) for parallel sampling,

2. The proposed split-state and independent sampling approaches are alternatives in terms of training performance and model performance quality,

3. Serial, parallel and independent acceleration methods may be used in combination.

Symbol	Description
s	Serial acceleration
p	Parallel acceleration
i	Independent samplers
a	Full-state exact sampler (FSE)
b	Split-state sampler (SSA)
c	Full-state approximate sampler (FSA)

Figure 8.5: Shorthands for implementations.

Models. As models for our study, we use direct extensions of LDA, so the difference to the base-line becomes directly visible and the same data set may be used. In addition to LDA, the following models have been considered: four-level PAM (PAM4), featuring a C2A component selector (see Chapter 5) and two dependent hidden edges, as well as hierarchical PAM with a C2C complex component selector function (hPAM2).

Furthermore, to evaluate the influence of topic nodes a model is introduced that extends LDA by an additional topic node, the LDA-E3 model: On an E3 branch, 2 words per topic token $z_{m,n}$ are output from two topic nodes $(\vec{\varphi}^1_k|\beta)$ and $(\vec{\varphi}^2_k|\beta)$ that both need to be synchronised:[10]

$$m \xrightarrow[[M]]{m} (\vec{\vartheta}_m \,|\, \alpha) \begin{array}{l} \xrightarrow[[K]]{z_{m,n}=k} (\vec{\varphi}^1_k \,|\, \beta) \xrightarrow[[V]]{w^1_{m,n}} w^1_{m,n} \\ \xrightarrow[[K]]{z_{m,n}=k} (\vec{\varphi}^2_k \,|\, \beta) \xrightarrow[[V]]{w^2_{m,n}} w^2_{m,n} \,. \end{array} \tag{8.19}$$

Implementations. To implement the models for this study, the Gibbs meta-sampler of Chapter 6 has been extended: Parallel sampling schemes are now created in a fully automatic fashion for all synchronisation strategies considered: FSE, FSA and SSA. The generation of bound-based algorithms has been realised by letting the meta-sampler emit comments in the source code as well as placeholders for variable declarations and operations for the fast sampling state described in Section 8.2.2. This largely simplifies implementation of serial acceleration while keeping the meta-sampler at reasonable complexity. Finally, for NoMMs with multiple hidden edges, generation of independent samplers according to Section 8.4 has been realised.

With this extended meta-sampler, algorithms have been generated and appropriately adjusted. As in Chapter 6, Java was used as an output language because more comfortable support libraries are available. With each reasonable combination of acceleration methods implemented separately, the total number of Gibbs sampling algorithms amounts to 38.

To simplify presentation, implementations have been named according to their serial acceleration, parallel acceleration and independent sampling characteristics, for which symbols s, p, and i are used, respectively. For parallel acceleration, the three synchronisation strategies have been named a–c. Fig. 8.5 summarises these conventions, which come in handy as soon as combinations of the acceleration methods are investigated.

[10] The use of the LDA-E3 model is limited to measuring scalability, and no practical application is envisioned. An E3 branch was chosen because this allows extension of LDA to two topic nodes without additional hidden variables.

Data. As a representative data set, the NIPS corpus is chosen, which has been used in much literature on topic models. The corpus consists of $M = 1740$ documents with a vocabulary of $V = 13649$ terms and $W = 2301375$ word tokens. For verification, experiments have been repeated with two other corpora for a random subset of the settings used on NIPS. Mostly the length of the documents and vocabulary size are varied. As a control data set, the ACL Anthology is used (also described in Chapter 2) with $M = 14291$ research articles, $V = 49953$ and $W = 24.5$M. For details on these data sets, see Appendix D.2.

Metrics. We are interested in several aspects of accelerated sampling approaches, for which the following metrics have been chosen:

- *Computational performance* of models is measured by both the time to complete training and the relative speed-up. The latter is defined as the ratio between the time required for a task on the target system divided by time for the same task on a baseline system. For this chapter, speed-up is defined relative to the execution of the model on a system with a single processor.[11]

- *Model quality* is measured by the held-out perplexity as well as pseudo-perplexity, with the difference between both as a measure of model overfitting to training data (cf. Chapter 6). This difference is of interest especially in the context of split-state sampling and independent sampling.

- *Distribution sparsity.* In order to understand the degree of concentration of the latent-variable distributions in NoMMs that are important prerequisites for acceleration, we assume that the parameters are distributed with a power law (following assumed degree distributions of the observed data). As a straight-forward and illustrative parameter that is strongly dependent on the concentration of a distribution, we choose the relative weight of the highest bin, $\hat{\vartheta}_{k,t} = \max_t \vartheta_{k,t}$. This measure shows the extent of "decision" for a particular topic. Empirically, it appears to be more sensitive than the entropy of the parameters, $\mathrm{H}\{\vec{\vartheta}^{\ell}_k\} = -\sum_t \vartheta^{\ell}_{k,t} \log_2 \vartheta^{\ell}_{k,t}$, and strongly correlates with the quantiles of the distribution, i.e., the number of reversely-ranked bins that are required to cover a certain portion of the probability mass.

Experiments have been carried out by running the different models in different parametrisations and acceleration methods in "batch mode". Based on the resulting measurement data in Figs. 8.6–8.11 below, different analyses were undertaken to empirically answer the questions formulated above.

Regarding consistency across data sets, it turned out that results generalise to the random test sample taken with the control data (ACL Anthology), so that it is safe to assume that the results for the NIPS corpus are representative.

In the following, we discuss results, starting with an analysis of LDA with combinations of serial and parallel acceleration methods in Section 8.5.1 and the generalisation of serial acceleration to more than a single hidden edge in Section 8.5.2. This is followed by an analysis of different parallelisation strategies on LDA and the more complex models in Section 8.5.3. Finally, we have a look at independent sampling in Section 8.5.4, as well as a proof-of-concept study aiming at optimal combination of fast sampling methods in Section 8.5.5.

[11] The experimental system was a quad-core PC (for details see System 2 in Appendix D.1).

8.5.1 LDA acceleration

We analysed the speed-up of LDA in different constellations. Not only was it of interest to compare the algorithm implementation with that of [Porteous et al. 2008b], but also to contribute a parallelised version. Using FSE synchronisation in Section 8.3.1 (method a), we aim at an answer to question 3 above.

As algorithms are numerically equivalent and therefore converge in the same number of iterations, we only need to look at the iteration times to determine the effective speed-up for convergence. An illustrative result is shown in Fig. 8.6 for the LDA model for $K = 500$, a relatively high number of topics, and fixed hyperparameters. In the baseline algorithm, each iteration requires around 17 seconds for the whole corpus.

Looking at the speed-up for serial acceleration, the result is to a great extent consistent with the literature: [Porteous et al. 2008b] report on speed-ups of around 8 for similar values while the value for the experiment is around 9.[12] An interesting aspect in this context is the increase of speed-up as iterations increase and distributions become sparser and the model becomes better in terms of perplexity.

We use the statistics of the largest weights in the Gibbs full conditionals of the NoMM nodes to monitor sparsity, and results are shown in Fig. 8.7(a). As φ and ϑ deviate from the evenly distributed initial weights, and $\hat{\varphi}_k$ and $\hat{\vartheta}_m$ start to increase, the iteration times for the fast samplers decrease, which is visible in Fig. 8.6. As can be seen, the distributions of topics settle for maximum weights rather quickly, becoming more pronounced in later iterations.[13] For the bound-based sampler, iteration times seem to gain especially from the first changes of the weights, as they stabilise at around iteration 40.

For parallel acceleration using the exact algorithm on a quad processor machine, speed-up of 3.91 or 97.75% of linear speed-up is achieved.[14] The speed-up for the methods combined was measured to be around 30 against the ordinary LDA Gibbs sampler and around 7 against the parallelised algorithm. In effect, the combination of serial and parallel methods indeed works while the speed-up drops by about 23%, which may be attributed to a higher likelihood of collisions in the synchronisation because of fewer computations per sample. Nevertheless, the combination of serial and parallel methods is highly effective when considering computation scalability.

Repeating the experiments for LDA for other numbers of topics leads to smaller speed-ups, as is presented in Fig. 8.8(a) and (b). Clearly, scalable sampling as implemented here leverages resources best if used on large dimensionalities of hidden variables. For serial acceleration, the number of saved topic weight calculations diminishes and the more complex norm calculation for the bound-based sampler takes its toll. For parallel sampling, with fewer topics to place sampling results into, different threads are more likely to collide during accesses to $n_{k,t}$, resulting in waiting time. Generally, the results for the combined methods appear consistent with those for $K = 500$.

[12] In [Porteous et al. 2008b], the same raw data set was used, but possibly with different pre-processing, resulting in different effective word counts and vocabulary size and consequently co-occurrence statistics. Furthermore, slightly different sorting of sampling weights has been used.

[13] The plot in Fig. 8.7 for $K = 100$ is qualitatively equivalent to one matching $K = 500$ in Fig. 8.6.

[14] Note that in practice the design of the memory hierarchy may lead to super-linear speed-up ("cache effect").

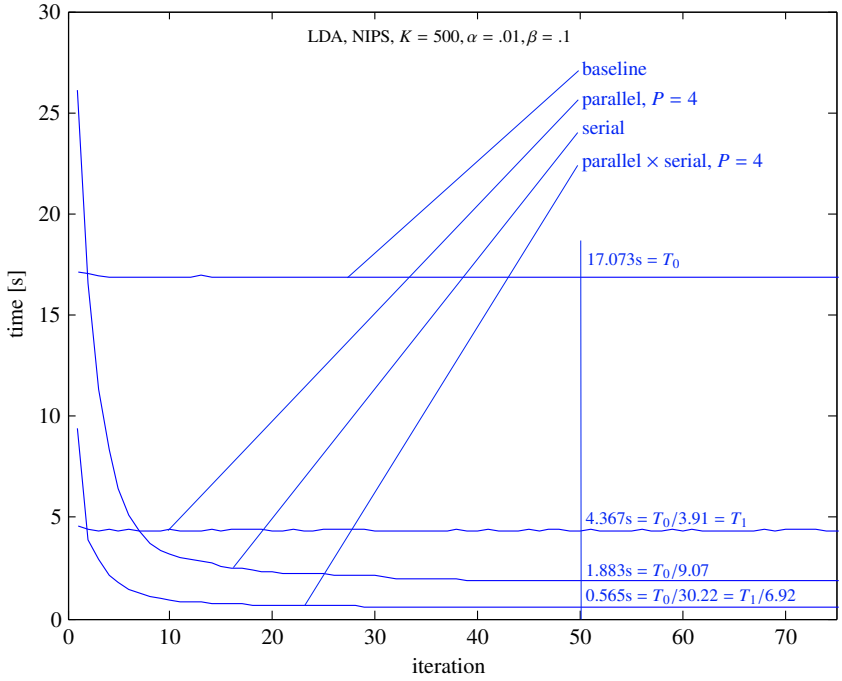


Figure 8.6: Iteration time in serial and parallel sampling for LDA with $K = 500$. Speed-ups are given as fractions of reference times T_0 for serial and T_1 for parallel operation at iteration 50.

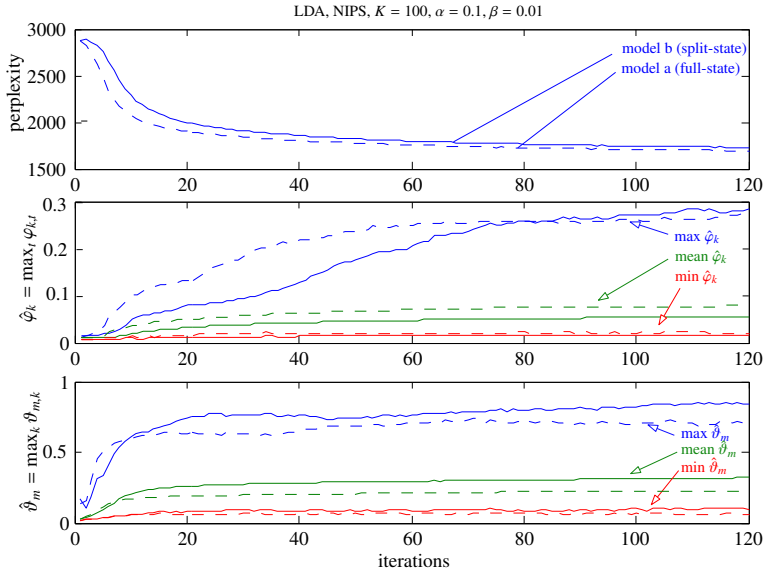
8.5.2 Generalised bound-based sampling

We tested the approach of generalising bound-based sampling to several dependent hidden variables/edges as described in Section 8.2.2. An appropriate model for two variables is the four-level PAM model. We looked at two different sets of norm orders, $(3; 3, \infty; 3, \infty)$ and $(5; 5, 5; 5, 5)$. The results for this and different model dimensions are presented in Figs. 8.8(c) and (d). Although these data suggest that a speed-up is possible, it is much less significant than for a single variable as with LDA above. Using the 3-norm clearly results in faster computation than using the 5-norm because fewer multiplications are required to update norms. The resulting speed-up is above 1 but never exceeds 1.8. Indeed it seems that for larger models the performance even decreases, which may be due to the bound being too loose, effectively requiring more weights to be computed per sample with larger dimensions.

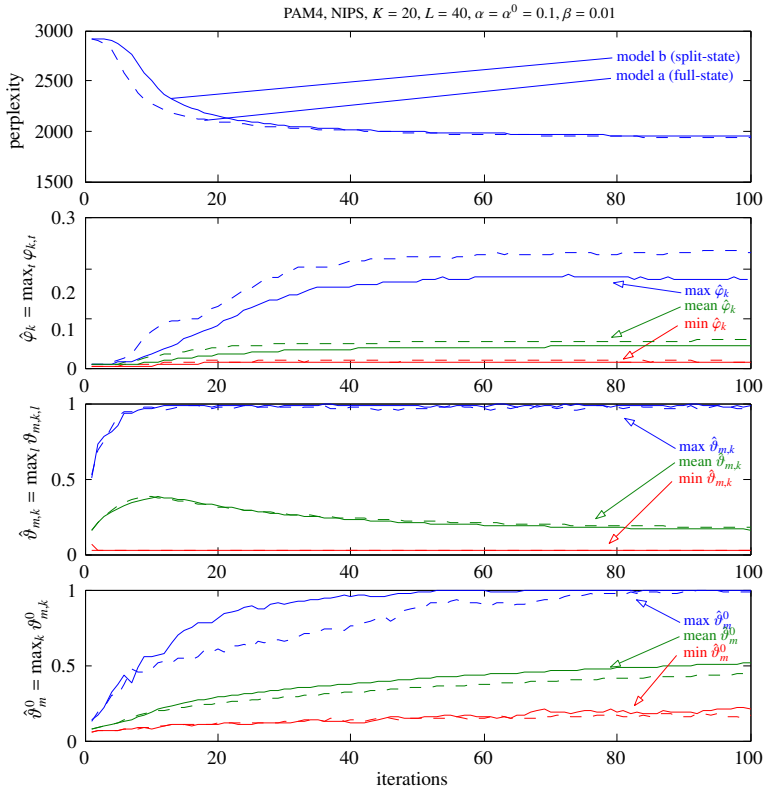
To verify this, a look at the parameter sparsity as presented in Fig. 8.7(b) is illustrative: In the middle node, some maxima $\hat{\vartheta}_{m,k}$ quickly increase to occupy in effect the complete mass of their component. With a norm that is based on the minimum of values in the infinite denominator norm, especially for highly concentrated components this bound will grossly overestimate the true value of the normalisation function Z.

For the 5-norm, the data suggest that although the bound seems to be tighter, its computation is too expensive to be leveraged at lower dimensions. For $K = L = 10$, speed-up is even below 1.

In summary, the results for two variables are disappointing, leaving a solution for efficient serial acceleration an open question. Because even higher norm orders are required for more dependent variables, no better results are expected there.

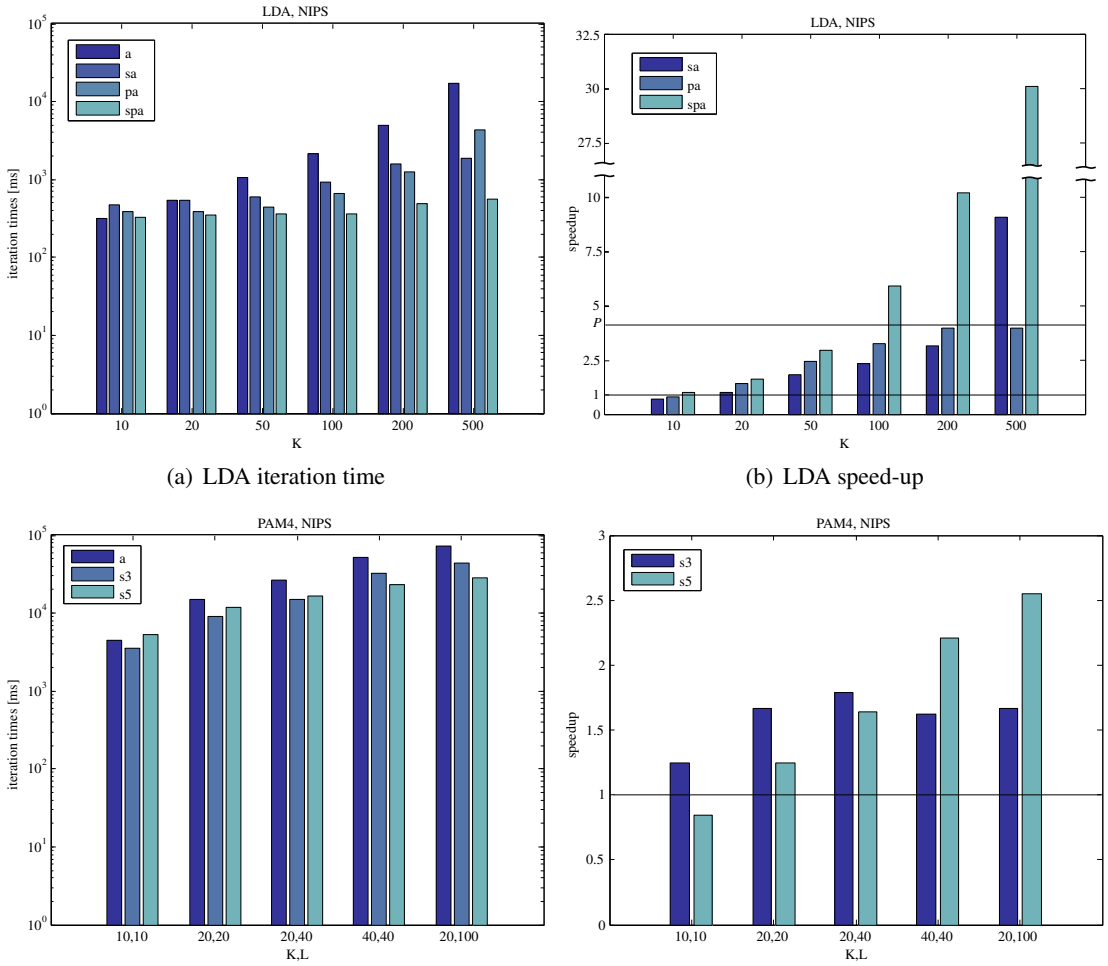


(a) LDA



(b) PAM4

Figure 8.7: Maximum weights of parameters and perplexity over iterations, for LDA with $K = 100$ and PAM4 with $K = 20, L = 40$. Full-state (dashed) and split-state approaches.



(a) LDA iteration time

(b) LDA speed-up

(c) PAM4 iteration time

(d) PAM4 speed-up

Figure 8.8: Timing results of bound-based acceleration for example models LDA and PAM4. For PAM4, s3 refers to $(3; 3, \infty; 3, \infty)$, and s5 to $(5; 5, 5; 5, 5)$.

8.5.3 Parallel acceleration

The results for the different parallelisation methods analysed are presented in the bar plots in Fig. 8.9 for LDA and LDA-E3 models and Fig. 8.10 for the pachinko allocation models. In each group of bars, columns a, pa, pb and pc correspond to the serial version, FSE, SSA and FSA parallelisations, respectively. For each model, both the iteration times and speed-ups are given, and comparisons are easiest based on the latter.

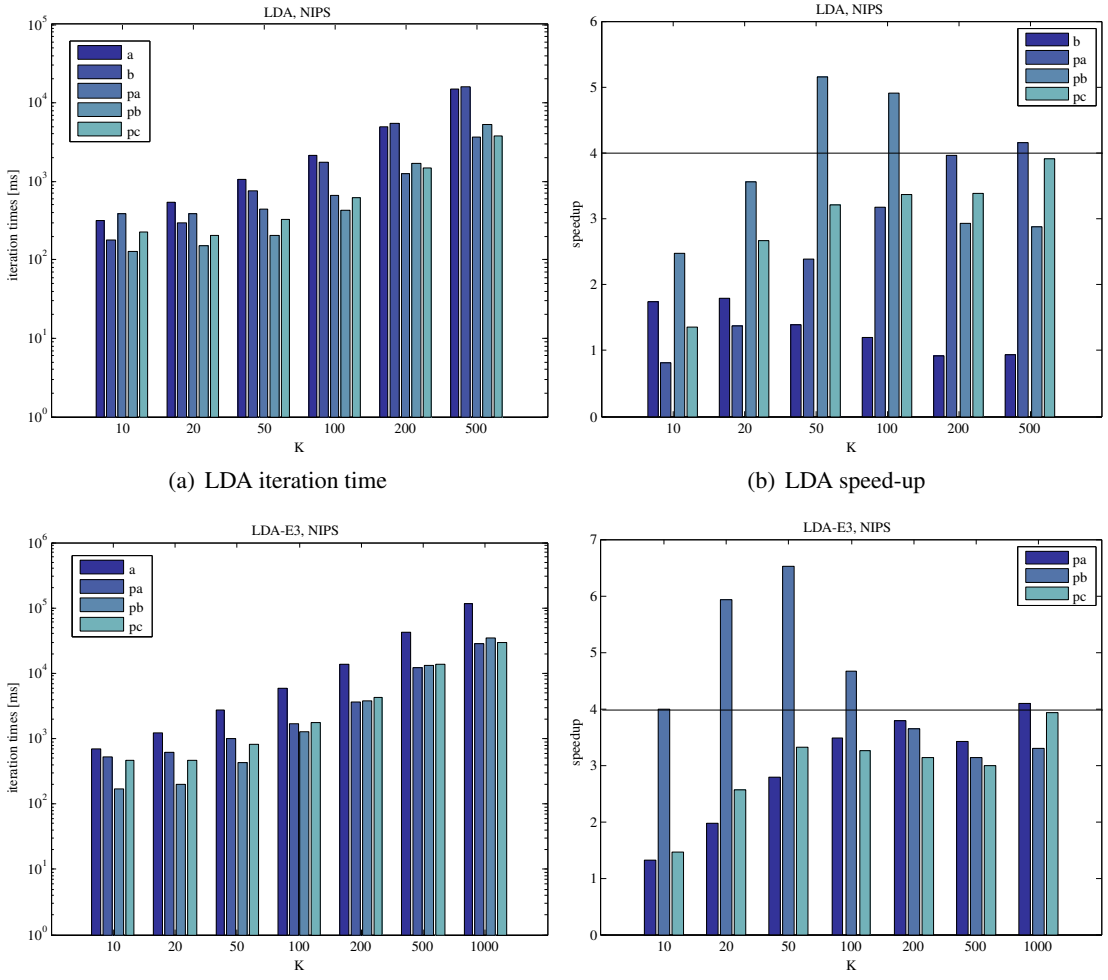
Full-state parallelisation. We first look at the full-state parallelisations FSE and FSA in LDA, LDA-E3 and PAM4. In Figs. 8.9(b) and (d), as well as Fig. 8.10(b) top, the relevant columns are pa and pc, which are compared to column a in the speed-ups. It turns out that both FSA and FSE methods are roughly equivalent in terms of speed-up when dimensions become larger, and at high dimensions reach linear speed-up or are partly even above this, for which architectural effects (caching) can be made responsible. For lower dimensions, the avoidance of synchronisation points in the FSA sampler (columns pc) seems to outweigh the extra time for the reduction step after each iteration, and the FSA sampler (columns pc) performs better than its FSE counterpart (columns pa).

Split-state parallelisation. To analyse split-state parallelisation, we first have a look at its effect on computational load without parallelisation. Looking at LDA in Fig. 8.9(b), columns b, surprisingly the simpler inference equation due to split-state sampling does not decrease iteration times for all model dimensions: While at $K = 20$ using a split state allows speed-ups of almost 2, this figure decreases until at $K = 200$ it even slows down the sampler compared to the baseline model.

In the parallel case in LDA (columns pb), this behaviour is re-enacted. While at $K = 50$ and $K = 100$ a super-linear speed-up is achieved, the parallelisation speed-up quickly decreases for higher values. For the LDA-E3 model, a similar behaviour (again, column pb) may be observed, however with the speed-up more pronounced ($K = \{20, 50\}$).

A different situation arises in the pachinko models with more than one dependent variable. Looking at PAM4 and hPAM2 in Fig. 8.10, it seems that the split-state versions of the samplers (columns pb, upper speed-up diagrams) reach consistently better speed-ups than the full-state versions. On second thought, this is not a surprise as these models typically are run on fewer (per-edge) dimensions than LDA-like models because these dimensions multiply in the total sampling space, which is reflected in the choices of dimensions used in the experiments.

Model quality. With the speed-ups of the parallel models largely as expected, the question is whether the approximations FSA and SSA are comparable in model quality at convergence to the FSE method (and consequently the serial counterpart). For this, we use the perplexity values at convergence, complemented by pseudo-perplexity to see how much better the parameters fit to the training set than the test set. The results for this are presented in Fig. 8.11, where again colums pa, pb and pc are of interest. As with speed-up, for FSE and FSA models values seem to be rather similar for both perplexity and pseudo-perplexity (shown as line marks partitioning the colums). This suggests that the approximation in the FSA synchronisation has no significant influence on the sampling process. Generally, the relative changes of the counts during one iteration in each of the P copies of the global counts are only significant in the first few iterations, which seems to have no adverse effect on convergence, however.



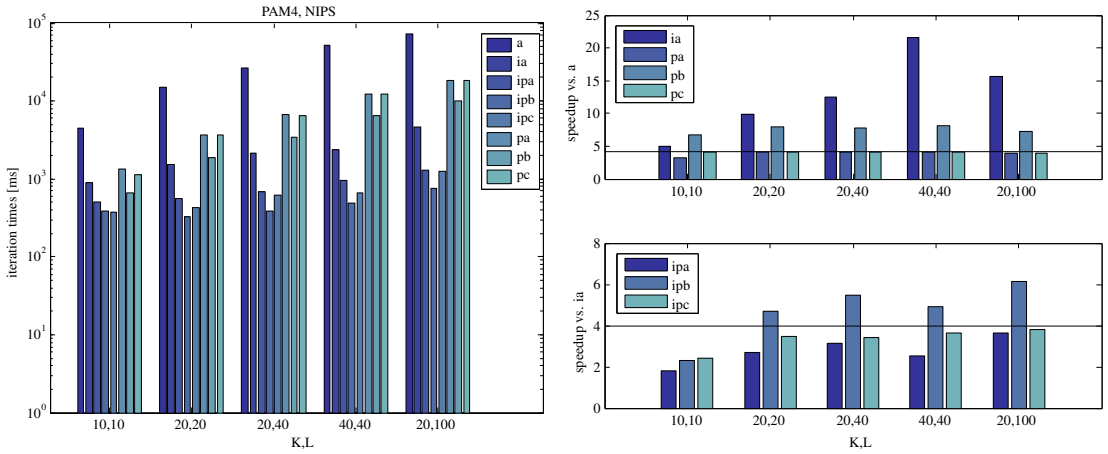
(a) LDA iteration time

(b) LDA speed-up

(c) LDA-E3 iteration time

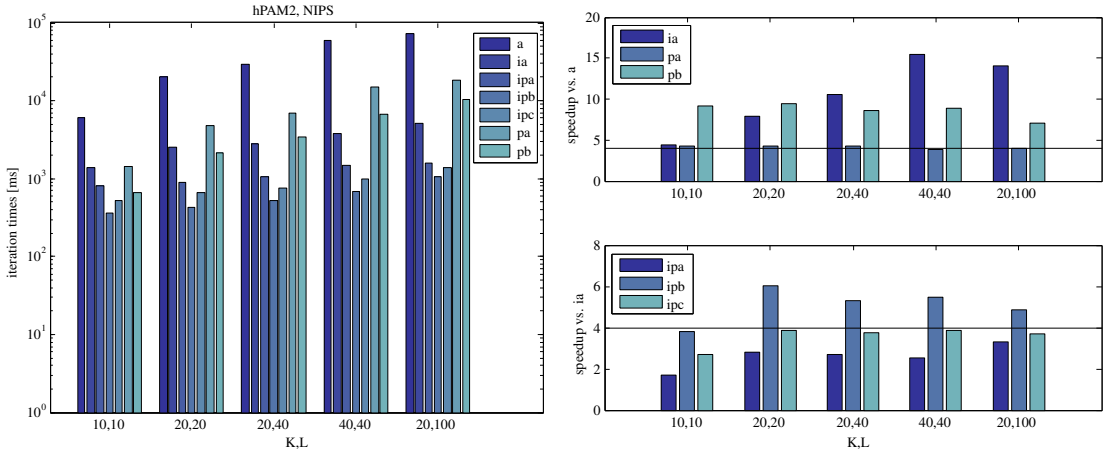
(d) LDA-E3 speed-up

Figure 8.9: Timing results for parallel LDA and LDA-E3 models: Speed-up is against the serial sampler (variant a with $P = 1$).



(a) PAM4 iteration time

(b) PAM4 speed-up



(c) hPAM2 iteration time

(d) hPAM2 speed-up

Figure 8.10: Timing results and convergence for parallel pachinko allocation models. Speed-ups for independent samplers are given separately in the lower diagrams in (b) and (d).

Regarding split-state sampling (columns pb), on the other hand, a difference may be observed consistently in all models, with the difference for LDA being the largest: Perplexity values seem to be higher than for the FSE versions while pseudo-perplexity tends to be below that of the exact version. This suggests that this approximation influences the training process much more than the partial approximation in the FSA method. In Fig. 8.7(a), the sparsity of the parameters for the LDA model has been tracked, and it can be seen that the trajectories of the maxima on each level are different between the split-state and full-state versions: The global parameter maxima, $\hat{\varphi}_k$, seem to change slower over iterations, which is plausible because they are updated only in steps, not continuously. Conversely, the document-wise parameter maxima $\hat{\vartheta}_m$ change faster. This seems to lead to topic configurations that are slightly less favoured by the held-out likelihood of LDA.

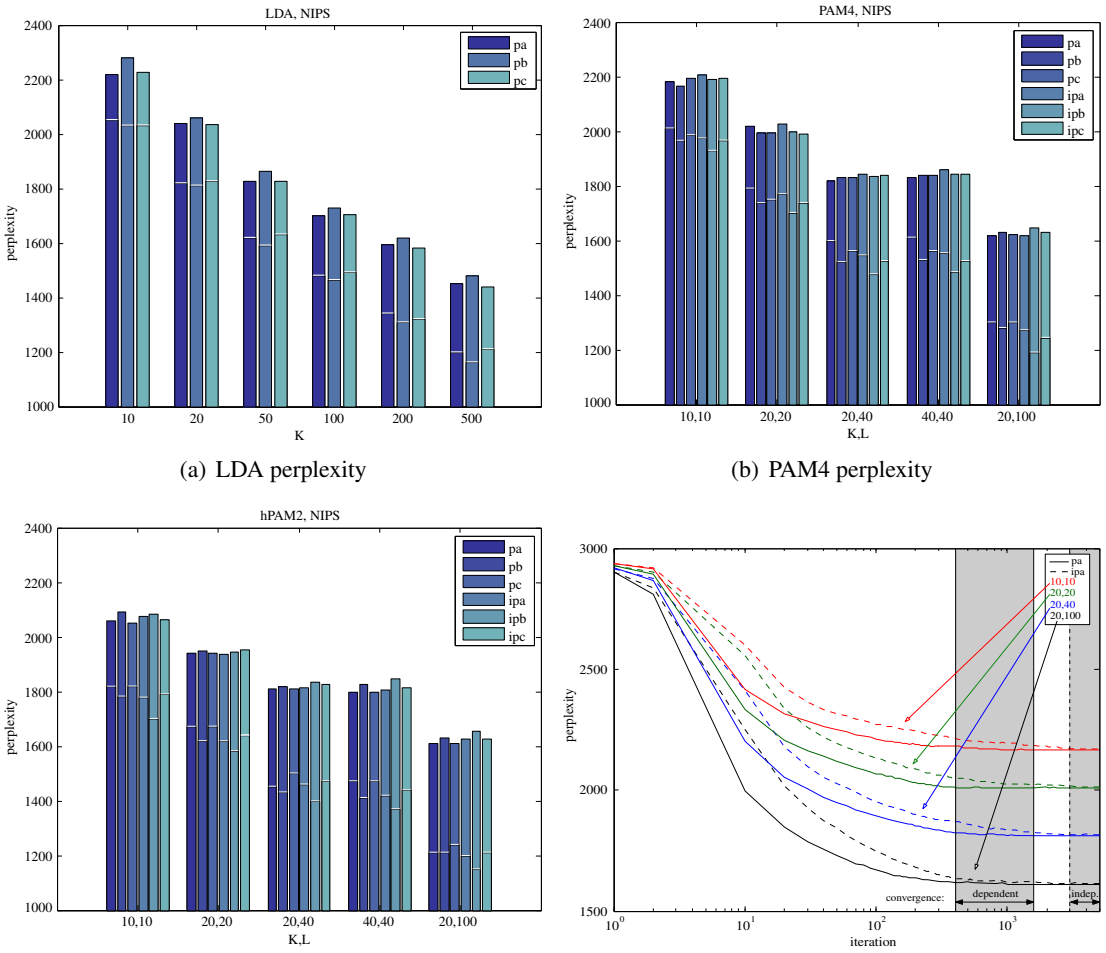
For more complex models, this effect seems to be weaker. Looking at the perplexity results for PAM4 and hPAM2 in Fig. 8.11(b) and (c), the perplexity disadvantage of split-state models sometimes even vanishes. However, pseudo-perplexity seems to be lower by a small margin. Looking at the evolution of the parameter maxima over iterations as presented in Fig. 8.7(b), in effect a similar behaviour can be seen for PAM4 as for LDA for the parameters $\hat{\vartheta}^0_m$ and $\hat{\varphi}_k$. However, for the central node $\hat{\vartheta}_{m,k}$, the approximation seems to have no effect. This indicates that with multiple-variable models split-state sampling leads to slightly different but not necessarily worse topics. Combined with the favourable results in speed-ups described above, split-state sampling may indeed be a good acceleration method in this case.

But also for LDA, the speed advantages for dimensions from $K = 50$ to 100 may be leveraged, with a perplexity disadvantage which is less than the difference between exact Gibbs sampling and variational methods as described in Chapter 7.

Convergence iterations. So far, all models in all variants have been analysed in terms of relative speed-up against a baseline. This speed-up of course only translates to real time savings in model training if convergence is reached in comparable numbers of iterations. As outlined in Chapter 6, the stochastic nature of the Gibbs sampler makes convergence monitoring a difficult task and there exists no absolute certainty that the algorithm has converged. Consequently, values for the analysed algorithms can only be coarse-grained estimates, using perplexity as a convergence indicator.

Comparing multiple sampler runs with different random seeds, it turned out that regardless of model order and parametrisation, convergence of the dependent Gibbs samplers was roughly the same for LDA, PAM4 and hPAM2 model. All experiments converged within the first 400–1500 iterations, and we let the algorithm run for 1500 iterations before the final parameters are sampled from the Markov state. The region of convergence is relatively wide because even for the same parametrisation, the algorithms may need a different number of iterations to converge. Fig. 8.10(d) presents the perplexity trajectories over iterations for the PAM4 model, which is analogous for all models analysed.[15]

[15]The LDA-E3 model is out of competition here because it was designed to measure the effect of two topic nodes on speed-up for different synchronisation methods rather than to provide favourable perplexity reduction. It converges already at iterations below 300, but to higher perplexities compared to LDA or pachinko allocation.



(a) LDA perplexity

(b) PAM4 perplexity

(c) hPAM2 perplexity

(d) PAM4 perplexity over time

Figure 8.11: Perplexity results and iterations to convergence for split-state and independent samplers: LDA, PAM4, hPAM2. The horizontal marks on the bars indicate pseudo-perplexity.

8.5.4 Independent sampling

As a final experiment, we analysed the effect of sampling each dependent latent variable on its own in the multi-variable models PAM4 and hPAM2.

Regarding speed-up, the results are presented in Figs. 8.10(b) and (d), columns ia, ipa, ipb and ipc: The measured speed-ups of the models are given in columns ia in Figs. 8.10(b) and (d) top against the standard model a. Based on the ia models, the parallelisations are given in the corresponding lower subplots, and it can be seen that here the theoretical speed-up of $P = 4$ is not as closely approached in the independent samplers. This is due to the significantly smaller computational footprint in the inner loop, which increases the portion of synchronisation effort per iteration and thus slows down the parallel algorithm. As for low dimensions in the standard parallel samplers, FSA synchronisation performs significantly better, avoiding synchronisation in the loops but requiring a reduction step.

While speed-up results were no surprise, the perplexity values indeed depict one. Looking at Fig. 8.11(b) and (c), columns ipa, ipb and ipc, the perplexity difference to the models that explicitly sample variables jointly to accommodate for their statistical dependency (colums pa, pb and pc) was insignificant. Similar to the split-state sampler, the pseudo-perplexity values went down slightly. However, the important finding is that held-out data were explained by the trained parameters with the same quality as using dependent sampling.

Convergence iterations. The close resemblance of perplexity reduction between dependent and independent samplers comes at a price: The effect of independent sampling is basically a slowed down convergence in terms of iterations. This is illustrated in Fig. 8.10(d) for the PAM4 model but equivalent for hPAM2. In the figure, the trajectories of perplexity measurements are presented over the iterations. Looking at the dashed trajectories for independent samplers reveals a resemblance to the standard ones. The convergence region for the independent samplers is shifted to the range of 3000 to 5000 iterations. According to Fig. 8.10(b), for a PAM4 model with $K = 20$ and $L = 100$, the resulting time speed-up between the parallel independent sampler (ipa) and its dependent counterpart (pa) is therefore on the order of:

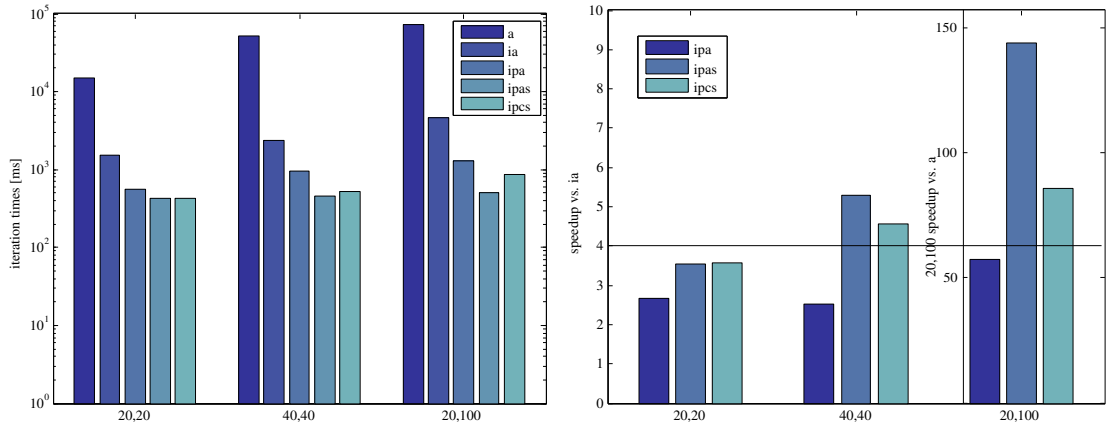
$$\frac{\text{time(pa)}}{\text{time(ipa)}} = \frac{\text{speedup(ipa}\,|\,\text{ia)}}{\text{iterations(ipa)}} \times \frac{\text{iterations(pa)}}{\text{speedup(pa}\,|\,\text{a)}} \times \text{speedup(ia}\,|\,\text{a)} = \frac{3.6}{5000} \times \frac{1500}{3.9} \times 16 = 4.43\,.$$

8.5.5 Towards an optimum

In order to run inference with the fastest possible speed-up, in the wake of the experiments we neglected the minor differences between dependent and independent samplers and combined the bound-based sampling schemes with appropriate parallelisms and independent sampling. Such sampling reuses the fast sampling state of an LDA-like model, and only 2-norms are necessary (cf. Section 8.5.1).

This approach notably also applies to branching and merging structures, because then 2-norms are set over partial products that share the hidden variable currently sampled. For instance, an E3 structure $q(j,k)q(k,s)q(k,t)$ will use the two 2-norms over k, $\|a(j,k)\|_2$ and $\|a(k,s)a(k,t)\|_2$, and the denominator norm $\|c^{-1}(k)\|_\infty$.[16]

[16]Note, however, that nodes $q(x,y)$ with both variables x and y hidden require two norms to be stored and updated, one to sample their parent edge(s) and one for the child edge. This is the case for instance in the middle node $q(x,y)$ in



(a) PAM4 iteration time (b) PAM4 speed-up

Figure 8.12: Timing results for combined independent/parallel/bound-based (ips) samplers: Example model PAM4.

Results for a proof-of-concept realisation of a fast serial–parallel PAM4 model are presented in Fig. 8.12. Especially with higher model dimensions, speed-ups are significant. In the case of PAM4 with $K = 20$ and $L = 100$, the best model reached a speed-up of more than 140 against the naïve serial version of the PAM4 Gibbs sampler, while that against the parallel version ipa it is 2.5. In this respect, the usage of bound-based methods contributes to scalability of more complex topic models much better than the proposed generalisation to higher norms.

Alternatively, the use of single hidden variables allows application of the SparseLDA approach by [Yao et al. 2009] mentioned earlier in place of the bound-based method. Given that [Yao et al. 2009] report a speed-up of 2 against FastLDA, this may result also in acceleration of the method generalised for NoMMs and may be combined with independent and parallel acceleration methods. However, implementation and experimental verification of this is considered an exercise for the future. The foundation for such a presumedly optimally fast class of algorithms has clearly been laid out here by the modularisation of the sampling distribution that enables application of the method of various NoMM structures.

8.6 Conclusions

In this chapter, methods have been shown that improve the performance of the Gibbs sampler for NoMMs. While prior work has focussed on the basic topic model LDA, the present contribution is to analyse such methods also for more complex models and provide new methods.

In particular, when models become more complex, it has been found that parallelisms for LDA generalise well, both for exact and approximate parallelisations, which makes for instance results from [Newman et al. 2006a] applicable for a wide range of models.

Furthermore, especially with two (or more) dependent parameters, significant speed-ups could be demonstrated using the split-state approximation proposed here that leaves a part of parameters constant over iterations and estimates them separately in a reduction step.

PAM4, which uses a norm $\|a(x, y)\|_2$ over x to sample x and one over y to sample y.

Another important result obtained empirically is that independently sampling variables that are dependent according to the model results is almost equivalent perplexity reductions compared to the case with full dependencies sampled. For lower model dimensions, this reduces convergence time for NoMM Gibbs samplers with multiple variables significantly.

The equivalence of independent sampling allows to compensate the weak generalisation behaviour found for the bound-based serial acceleration method for multiple variables: In combination with independent sampling, high speed-ups could be achieved with bound-based sampling for the model PAM4 with two hidden variables, in principle allowing application of a modified FastLDA or SparseLDA method [Porteous et al. 2008b, Yao et al. 2009] to arbitrary NoMMs with Dirichlet–multinomial structures. Generally, different combinations of independent, serial and parallel acceleration methods have been found to work well, especially for higher model dimensions.

Beyond the models analysed here, results are expected to generalise to other structures as the number of dependent variables (or NoMM edges) appears to be the critical property and methods have been proposed generically across NoMM structures. Thus, the results of this chapter widen the scope that especially more complex NoMMs can be used in, and on larger data volumes.

Part III

Application

The Application part, Chapters 9 and 10, uses the results of Modelling and Inference parts for topic modelling in community data. In particular, Chapter 9 derives a design method that is based on the modularity of NoMM representations. Chapter 10 then illustrates the method in practice for a real-world scenario, also experimentally validating the designed models.

Main contributions:

- Topic model analysis by NoMMs decomposition → Chapter 9
- Topic model design using NoMMs → Chapter 9
- Expert–tag–topic models and retrieval → Chapter 10

Chapter 9

Towards model design using NoMMs

This chapter investigates how topic models may be constructed using the advantages of the NoMM representation. In particular, a design method is proposed that is based on assembling models from NoMM sub-structures, associating them with structures in the data. Using the numerical properties of Gibbs full conditionals and data likelihood as predictors of model behaviour, models can be constructed in a more controlled way than using today's approach.[1]

9.1 Introduction

Discrete data of virtual communities and other fields, from bioinformatics to computer vision, have often been associated with latent-variable models, and many topic models have empirically led to robust results in the presence of sparsity and noise (cf. models in Chapter 5). Typically, in the literature such models are designed by assuming generative processes to originate the observations to be modelled. For example, each word in the LDA model is thought to be generated by sampling a topic indicator from a document-specific topic multinomial and an observable word from a topic-specific vocabulary multinomial.

While the viewpoint of generative processes is intuitive in the sense of explaining models on a high level, the connection to their inference equations is not directly obvious. Rather, in traditional topic model development the inference equations are derived from scratch, typically based on the Bayesian network. Depending on the model size, this is a more or less complex calculation, often with some portion of tedious steps and sometimes filling page-long appendices in publications (cf. [Shafiei & Milios 2006] and Appendix E for example derivations).

A direct mapping between model structure and inference equations may therefore be beneficial. In addition to short-cutting inference derivation, having available the numerical model properties during model construction may provide insights into the model behaviour and allow more informed design decisions. The modeller may see for instance how the model will learn from a particular configuration of model variables (e.g., a topic co-occurring with a word in LDA increasing the Gibbs sampling weight of the topic) or detect scalability issues in the update equations.

To create a connection between model structure and inference equations, the generic model formulations in Part II of this thesis are instrumental. In principle, such a connection exists

[1] A concise version of this chapter has been published as part of [Heinrich 2011b].

for the standard representations of topic models, Bayesian networks (BN). The mapping task is facilitated, however, by exploiting the abstractions provided by the NoMM representation. As discussed in Chapters 4, 6 and 7, NoMM model structures have straight-forward correspondences with the structures of inference equations and the data likelihood.

In this chapter, we will therefore use the explicit mapping between NoMM structure and probabilistic properties to draft a novel design method for topic models. This method may be seen as a step towards recommending to developers *what* they can input into the implementation workflow presented in Section 6.6, given a particular modelling task. It completes the tool-set for generic topic modelling that this thesis aims at.

A desirable feature of the envisioned design method is to work additively, i.e., by constructing models from sub-structures, such as those identified in Chapter 5. The designer then can assemble a model from a "library" of sub-structures to an overall model, taking care of numerical model properties in each step.

Chapter outline. Following the additive approach envisioned, in Section 9.2 we study how NoMM structures and their numerical properties decompose into sub-structures. We revisit the particular sub-structures from Chapter 5 and present their Gibbs update and likelihood functions on a modular basis in Section 9.3, leading to the NoMM structure "library" and different methods to incorporate evidence in models. Based on this library, the actual design method is drafted in Section 9.4.

9.2 NoMM numerical decomposition

A prerequisite to the envisioned design approach is to determine how the numerical properties of NoMMs may be deconstructed into smaller parts, and in particular we are interested in the inference equations. Although such a decomposition may in principle be applied to the generic variational Bayes algorithm from Chapter 7 as well, we focus on the collapsed Gibbs sampler in Chapter 6 because its full conditionals have relatively simple numeric forms that may bear some intuitive meaning for prediction of model behaviour and because of the superior empirical results achieved in the previous chapters. Apart from this, the decomposition behaviour of the data likelihood is of interest, as it is closely related to the objective of the learning process.

Notation review. For the following considerations, recall the notation introduced in Chapter 4: Let upper-case symbols denote sets of their lower-case counterparts introduced above. That is, Θ, A, and X correspond to all component parameters, hyperparameters and variables of a given model. If a superscript $\cdot^{\ell}$ is given or within brackets, $[\cdot]^{[\ell]}$, symbols are specific to a level, including any indices and function arguments. Among variables, X, we furthermore distinguish the sets of hidden and visible variables, H and V. For the purpose of illustrating the decomposition of models, we restrict ourselves to the "standard" node types with Dirichlet–multinomial parameters as well as observed parameters, N1 and N2, along with all common E and C types.

9.2.1 Full conditionals

As outlined in Chapter 6, model training in the collapsed Gibbs sampler is based on full conditional distributions, $p(h_i|V, H_{\neg i}, A)$. Full conditionals are the marginals of the posterior distribution, $p(H, \Theta|V, A)$, w.r.t. hidden variables at single data points, i, given all other information and parameters Θ integrated out. Index $\neg i$ denotes exclusion of i. The set of these distributions forms the transition matrix of a Markov chain, and repeatedly sampling through all i over time leads to a stationary state that simulates the true posterior.

Full conditionals may therefore be seen as a low-dimensional representation of the true posterior – a desirable quantity for prediction of model behaviour, as it represents the model parameters as a function of the observations.

In NoMMs, full conditionals have the following form, as shown in Chapter 6:

$$p(h_i^d \mid V, H_{\neg i^d}, A) \propto \prod_\ell \left[\prod_k \frac{\Delta(\vec{n}_k + \alpha)}{\Delta(\vec{n}_{k,\neg i^d} + \alpha)} \right]^{[\ell]} \tag{9.1}$$

where d represents a group of levels with mutually dependent variables, $\vec{n}_k^\ell = \{n_{k,t}\}_t^\ell$ is the "co-occurrence" count vector between component index k^ℓ and node output values t^ℓ and α^ℓ represents all possible variants of hyperparameter, scalar, vector or grouped. Note that h_i^d are dependent hidden variables for an observation v_i across different levels $\ell \in d$ (all other levels vanish). The sequence index i^d can be different for different h_i^d, but we'll use i for simplicity.

To illustrate the principle that underlies (9.1), recall from Section 6.3 that its factors reduce to quotients of sums if the exclusion of the current sample (with $\neg i$) from the vectors corresponds to a unit difference between numerator and denonimator:

$$p(h_i \mid V, H_{\neg i}, A) \propto \frac{n^a_{k,t,\neg i} + \alpha^a}{\sum_t n^a_{k,t,\neg i} + \alpha^a} \cdot \frac{n^b_{k,t,\neg i} + \alpha^b}{\sum_t n^b_{k,t,\neg i} + \alpha^b} \cdots , \tag{9.2}$$

that is, the normalised and smoothed co-occurrence counts reinforce the respective sampling weights in a "rich-get-richer" manner, which is known from the Pólya urn sampling scheme associated with the Dirichlet–multinomial distribution pair (see Section 3.4.2).

Partial posteriors at mixture level: q-terms. To facilitate the following discussions, we define a shorthand for the factors in the generic full conditional (9.1), similar to the one in Chapter 8:[2]

$$q_i^\ell(k, t \mid \alpha) \triangleq \left[\frac{\Delta(\vec{n}_k + \alpha)}{\Delta(\vec{n}_{k,\neg i} + \alpha)} \right]^{[\ell]} \overset{\text{case of (9.2)}}{=} \left[\frac{n_{k,t,\neg i} + \alpha}{\sum_t n_{k,t,\neg i} + \alpha} \right]^{[\ell]} , \tag{9.3}$$

emphasising the interrelation of indices that is expected to play a vital role in designing models, basically encoding the Pólya urn scheme and the posterior behaviour of a single N1 node. From a mathematical point of view, the main purpose of "q-terms" is to determine which elements carry an excluded item $\neg i$ in (9.1) and therefore don't cancel out when resolving the $\Delta(\cdot)$ functions. In effect, they are a key to facilitate model design.

[2] More formally: $q_i^\ell(k, t \mid \alpha) \triangleq [p(x_i{=}t \mid k{=}g(\uparrow x_i, i), X_{\neg i}, A)]^{[\ell]}$. The dependence on $X^\ell_{\neg i}$ is due to integrating out parameters Θ^ℓ in the collapsed Gibbs sampling case; it is in effect the dependence on the count statistics $n^\ell_{k,t,\neg i}$.

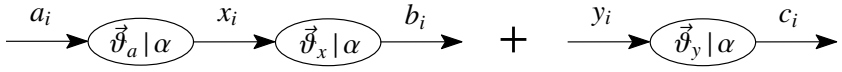


Figure 9.1: Example structures for NoMM composition.

To efficiently work with q-terms, we avoid notational clutter by omitting every argument that is clear from context, leading to a simple standard form $q(k, t)$ that represents a full-conditional term with implicit notation of level ℓ, sequence index i and hyperparameter α.

As their "input", q-terms always take a single mixture component, which is given by the first argument, e.g., k above, or (x, y) if there is a combination of two parent edges x and y. Considering the "output", the standard case is that a single edge value is drawn, e.g., t from above. This expands to the term on the right-hand side of (9.3).

For cases with several values at the output of a q-term, we introduce an "edge combination" operator, $\oplus$: Indexes combined with $\oplus$ refer to sums of the count vectors identified by the arguments. For instance, $q(k, t \oplus u)$ will contain $\Delta(\vec{n}^{(t)}_{k,\neg i} + \vec{n}^{(u)}_{k,\neg i} + \alpha)$ in its denominator, with $\vec{n}^{(t)}_{k,\neg i} = \{n_{k,t,\neg i}\}_t$, etc. By convention, if there is ambiguity, superscripts in parentheses identify the edges that the count variables belong to, so in $\vec{n}^{(t)}_{k,\neg i} + \vec{n}^{(u)}_{k,\neg i}$ the effect of edges t and u is combined. In (9.3), a quotient with two tokens excluded in the denominator expands to two quotients:

$$q(k, t \oplus u) = \frac{\Delta(\vec{n}^{(t)}_{k} + \vec{n}^{(u)}_{k} + \alpha)}{\Delta(\vec{n}^{(t)}_{k,\neg i} + \vec{n}^{(u)}_{k,\neg i} + \alpha)} = \frac{n^{(t)}_{k,t,\neg i} + n^{(u)}_{k,t} + \alpha}{n^{(t)}_{k,\neg i} + n^{(u)}_{k} + \alpha} \cdot \frac{n^{(t)}_{k,u} + n^{(u)}_{k,u,\neg i} + \alpha + \delta(t-u)}{n^{(t)}_{k} + n^{(u)}_{k,\neg i} + \alpha + 1} \tag{9.4}$$

where $\delta(t-u)$ is the Kronecker delta and $n^{(t)}_{k}$ is the sum of elements $n^{(t)}_{k,t}$, etc.

This expansion follows (6.6). In general, any variable combined with $\oplus$ leads to a factor in (9.4) if it is decremented for token i. Variables not decremented simply add their counts without leading to additional factors. Such undecremented variables are symbolised using a tilde, $\tilde{\cdot}$. For instance: $q(k, t \oplus \tilde{u})$ only expands to the first term on the right of (9.4).

Model decomposition. Using the q-terms, the full conditional (9.1) may be partitioned into level-wise factors:[3]

$$p(h_i | V, H_{\neg i}, A) \propto \prod_{\ell} q^{\ell}(k, t) \tag{9.5}$$

where each k^{ℓ} and t^{ℓ} may represent several joint values in the set of hidden variables h_i, corresponding to the NoMM topology: k^{ℓ} may be a function of multiple parent edges and sequence values (C2 type), and t^{ℓ} may represent the joint draw of child edges (E2 type).

An important prerequisite for the modularisation in (9.5) is that the factors $q^{\ell}(k, t)$ are effectively the same regardless of which part of k^{ℓ} or t^{ℓ} is among the hidden variables to be sampled, h_i.[4] The structure of (9.5) therefore enables us to indeed look at sub-structures separately

[3] In (9.5), the product over components is omitted compared to (9.1) because h_i implies a particular configuration of k^{ℓ} and t^{ℓ}, and every other k^{ℓ} leads to a factor of 1.

[4] This is due to the fact that (1) q-terms are conditional probability distributions, that (2) variables are conditionally independent for different tokens $i \neq i'$ and (3) because of the $\propto$ relation in (9.5): Any full conditional distribution (q-term) with two hidden variables, $p(x_i, y_i | X_{\neg i}, Y_{\neg i}) = p(X, Y)/p(X_{\neg i}, Y_{\neg i})$, is proportional to a distribution with only

and use the same q-terms regardless of how they appear within the overall model. Moreover, we may create sub-structures of more than a single q-term, $w(h_i^c|\cdot) = \prod_{\ell \in c} q^\ell(k,t)$. The corresponding composition rule for "w-terms" then is:

$$p(h_i|\cdot) \propto \prod_c w(h_i^c|\cdot) = \prod_c \prod_{\ell \in c} q^\ell(k,t)\,. \tag{9.6}$$

For example, composing a NoMM with hidden variables $\{x, y\}$ from the two structures in Fig. 9.1 requires setting $y_i \equiv b_i$ and then multiplying the sub-terms $w(x|a,b) = q(a,x)q(x,b)$ and $w(y|c) = q(y,c)$, leading to $w(x,y|a,c) = q(a,x)q(x,y)q(y,c)$. Alternatively, the second sub-structure may be inserted at the center of the first one, which requires splitting up edge x_i into a left and a right part and setting $y_i \equiv x_i^{\mathrm{L}}$ and $x_i^{\mathrm{R}} \equiv c_i$. This yields $w(y,x|a,b) = q(a,y)q(y,x)q(x,b)$.

9.2.2 Data likelihood

The likelihood of the data under the set of trained model parameters is an interesting descriptive property because it is closely related to the objective function that is optimised during inference. Furthermore, it may be seen as an indicator of expected model performance (which here refers to the best numerical result the model can in principle achieve).

The likelihood of observations under the trained model is mainly a function of the node parameters, $\Theta^\ell = \{\{\vartheta^\ell_{k,t}\}_t\}_k$, which for the Gibbs sampler can be modelled as the expectations of the Dirichlet priors given the co-occurrences: $\vartheta_{k,t} \propto n_{k,t} + \alpha_t$. As shown in Chapter 6, the likelihood may be generically expressed as:[5]

$$p(v_i|\Theta) = \sum_{h_i} \prod_\ell \vartheta^\ell_{k,t} \tag{9.7}$$

where the summation over h_i refers to all configurations of values of the dependent hidden variables that lead to different k^ℓ and t^ℓ to index the parameters. Note the close similarity of the forms of (9.5) and (9.7), even though the free variables are disjoint: hidden tokens h_i in the full conditional vs. observations v_i in the likelihood.

Model decomposition. Similar to the full conditional, the data likelihood may be partitioned into sub-structures. From (9.7), it may be inferred that the inner terms of the likelihood of the complete model factor into that of sub-models. If two dependent sub-structures are joined, the marginal sums $\sum_h$ need to be taken care of, which is done by summing over hidden variables that connect the sub-structures. Again considering the sub-structures in Fig. 9.1 as an example and setting $y_i \equiv b_i$, the likelihood for the joint model becomes: $p(c|a) = \sum_y p(c|y)p(y|a) = \sum_y \vartheta_{y,c} \sum_x \vartheta_{a,x}\vartheta_{x,y}$, which is analogous to the full conditional.

---

a single one (k^ℓ or t^ℓ): $p(x_i|X_{\neg i}, Y) = p(X,Y)/p(X_{\neg i}, Y) = p(X,Y)/[p(X_{\neg i}, Y_{\neg i})p(y_i)] \propto p(X,Y)/p(X_{\neg i}, Y_{\neg i})$. This generalises to an arbitrary number of variables.

[5] For simplicity, this omits sub-sequences as discussed in Chapter 6. Sub-sequences will be discussed in context.

9.3 Sub-structure library

The modularisation of full conditionals and likelihood allows us to view NoMMs on the basis of sub-structures, for instance those identified in Chapter 5: N types characterised according to the probability distributions that their nodes use, E types distinguishing how models branch node information, that is, distribute samples of a given node, and C types distinguishing how models merge information, that is, how incoming data index components of a node.

The actual choice and combination of these sub-structures is decisive for the behaviour of the assembled model, and it is worthwhile to analyse their numerical properties. For the sub-structures from Chapter 5, this is done in Section 9.3.1. Subsequently Section 9.3.2 explains the novel C3 sub-structure, whereas Section 9.3.3 discusses methods to incorporate evidence into models.

9.3.1 Sub-structure numerical properties

To make full use of the sub-structures for model design, their partial full conditional and likelihood terms have been calculated. The method used for this derivation was along the lines of (9.5) and (9.7) (or (6.4) and (6.11), respectively).

The results are summarised in the table in Fig. 9.2. In the quantities in this table, dependencies on A and Θ have been omitted, as well as indices i and $\neg i$ where obvious. To better understand the entries, a few remarks are given for different sub-structure types. For other information on the models and abbreviations of model names, see Chapter 5 where variants are further explained.

N1 Dirichlet–multinomial nodes are the basic form of discrete mixtures, and many models solely consist of N1 and diverse E and C structures. N1 nodes appear to be the simplest adaptive structures, with a standard q-term and multinomials $\vec{\vartheta}_k$ that adapt to the co-occurrences of input and output edge values. While N1A uses a single hyperparameter, N1B introduces a selection function for $\vec{\alpha}_j$ that may be used to add an additional level of grouping among components $\vec{\vartheta}_k$ (see component grouping in Chapter 4). If no special requirements exist, N1 are the first choice to consider when modelling.

E1 and C1 structures create unbranched mixture structures. Sub-type C1A covers sequence indices to select components, $k = i$, and C1B hidden edges, $k = \uparrow x_i^{\ell}$ with a single parent. This corresponds to sequence and topic node behaviour, respectively.

E*S sub-sequences have been discussed in Sections 5.3.1 and 6.3. In the table, this is represented as E1S(z_i) and $n \in i$, meaning that from a single sample of the parent node z_i, e.g., a sentence topic, a whole sub-sequence of tokens $\vec{b}_i$ is produced, as in [Shafiei & Milios 2006]. To sample z_i, the corresponding Gibbs full conditional term becomes $q(z_i, \vec{b}_i)$, which causes (9.1) to deviate from its standard form (9.2) and leads to (6.6) discussed in Chapter 6.[6]

N2–N5 distribution variants: N2 nodes introduced observed distributions over edge values, which may introduce sparsity into the output edge if each component contains only non-zero entries for a subset of those output values. As shown in Section 6.3, they are easy to handle

[6] To sample any token i, a whole set of $n \in i$ needs to be excluded from the sample, with the count vectors having a difference in the ratio of $\Delta(\cdot)$ functions in (9.1).

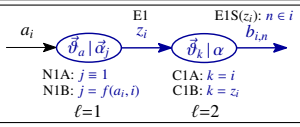
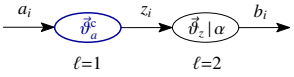
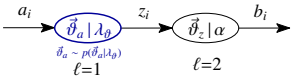
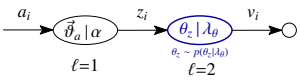
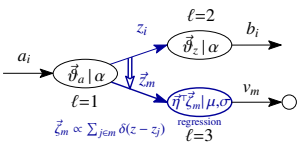
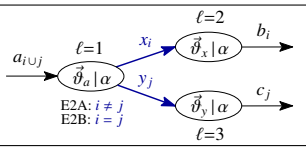
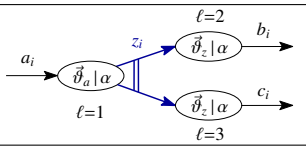
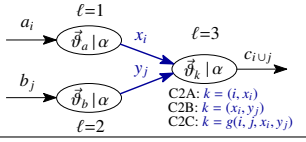
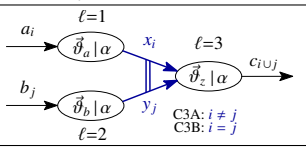
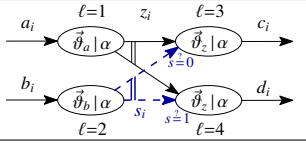
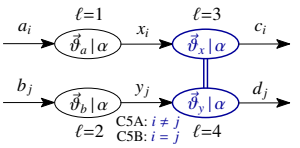
ID. Name	Structure diagram	Gibbs sampler weight w, Likelihood p for single token i Modelled aspect, example models
N1,E1,C1. Dir–Mult nodes, unbranched		$w(z\vert\cdot) = q(a,z)q(z,b)$ E1S(z_i): $q(a,z)q(z,b_1 \oplus b_2 \oplus \ldots b_{N_i})$ $p(b\vert a) = \sum_z \vartheta_{a,z}\vartheta_{z,b}$ *Mixture/admixture*: LDA [Blei et al. 2003b], PAM [Li & McCallum 2006]; LDCC [Shafiei & Milios 2006] (E1S)
N2. Observed parameters		$w(z\vert\vec{\vartheta}^{\mathrm{c}}_a,\cdot) = \vartheta^{\mathrm{c}}_{a,z}\, q(z,b)$ $p(b\vert a) = \sum_z \vartheta^{\mathrm{c}}_{a,z}\vartheta_{z,b}$ *Label distribution*: ATM [Rosen-Zvi et al. 2004]
N3. Non-Dirichlet prior		$w(z\vert\vec{\vartheta}_a,\cdot) = p(z_i\vert a_i,\vec{\vartheta}_a)q(z,b)$; M-step: estimate $\vec{\vartheta}_a$ [Blei & Lafferty 2007] $p(b\vert a) = \sum_z \vartheta_{a,z}\vartheta_{z,b}$ *Alternative distributions on the simplex*: CTM [Blei & Lafferty 2007]: $\vec{\vartheta}_a \propto \exp\vec{\eta}$, $\vec{\eta}\sim\mathcal{N}(\vec{\mu},\underline{\Sigma})$; TLM [Wallach 2008]: hierarchy of Dirichlet priors
N4. Non-discrete output		$w(z\vert\theta,\cdot) = q(a,z)p(v_i\,\vert\,\theta_z)$; M-step: estimate θ_z $p(v\vert a) = \sum_z \vartheta_{a,z}p(v\,\vert\,\theta_z)$ *Non-multinomial observ.*: Corr-LDA [Barnard et al. 2003], GMM [McLachlan & Peel 2000]: $p(v\vert\theta) = \mathcal{N}(\vec{x}\vert\vec{\mu},\underline{\Sigma})$
N5+E4. Aggregation		$w(z\vert\vec{z}_m, v_m,\cdot) = q(a,z)\,q(z,w)\mathcal{N}(v_m\,\vert\,\vec{\eta}_v^{\top}\vec{\zeta}_m,\sigma^2)$; M-step: estimate $\vec{\eta}_v, \sigma^2\,\vert\,\vec{z},\vec{v}$ (for linear regression, N5B) prediction: $v_m = \vec{\eta}_v^{\top}\vec{\zeta}_m$ *Regression/supervised learning*: Supervised LDA [Blei & McAuliffe 2007], Relational topic model [Chang & Blei 2009]
E2. Autonomous edges		$w(x,y\vert\cdot) = q(a,x\oplus y)q(x,b)q(y,c)$ E2A: $q(a_{i\cup j},x_i\oplus y_j) = q(a_i,x_i\oplus\tilde{y}_j)q(a_j,\tilde{x}_i\oplus y_j)$ $p(b,c\vert a) = \sum_x \vartheta_{a,x}\vartheta_{x,b}\sum_y \vartheta_{a,y}\vartheta_{y,c}$ *Common mixture of causes*: Multimodal LDA [Ramage et al. 2009]
E3. Coupled edges		$w(z\vert\cdot) = q(a,z)q(z,b)q(z,c)$ $p(b,c\vert a) = \sum_z \vartheta_{a,z}\vartheta_{z,b}\vartheta_{z,c}$ *Common cause for observations*: Hidden relational model (HRM) [Xu et al. 2006], Link-LDA [Erosheva et al. 2004]
C2. Combined indices		$w(x,y\vert\cdot) = q(a,x)q(b,y)q(k,c)$ $p(c\vert a,b) = \sum_x[\vartheta_{a,x}\sum_y \vartheta_{b,y}\vartheta_{k,c}]$, $k = g(x,y,i,j)$ *Different dependent causes, relation*: hPAM [Li et al. 2007a], HRM [Xu et al. 2006], Multi-LDA [Porteous et al. 2008a]
C3. Interleaved indices		C3A: $w(x_i,y_j\vert\cdot) = q(a_i,x_i)q(b_j,y_j)q(x_i,c_i\oplus\tilde{c}_j)q(y_j,\tilde{c}_i\oplus c_j)$ C3B: $w(x,y\vert\cdot) = q(a,x)q(b,y)[q(x,c\oplus c)]^{\delta(x-y)}\,[q(x,c\oplus\tilde{c})q(y,\tilde{c}\oplus c)]^{1-\delta(x-y)}$ C3A: $p(c\vert a) = \sum_x \vartheta_{a,x}\vartheta_{x,c}$, $p(c\vert b)$ sim., C3B: $p(c\vert a,b) = \sqrt{\sum_x \vartheta_{a,x}\vartheta_{x,c}\sum_y \vartheta_{b,y}\vartheta_{y,c}}$ *Different causes, same effect*: proposed here
C4. Switch		$w(z,s\vert\cdot) = q(a,z)[q(b,1)q(z,c)]^{\delta(s-1)}\cdot[q(b,2)q(z,d)]^{\delta(s-2)}$ $p(c,d\vert a,b) = \sum_z \vartheta_{a,z}[\vartheta_{b,0}\vartheta_{z,c} + \vartheta_{b,1}\vartheta_{z,d}]$ *Select complex submodels*: Multi-grain LDA [Titov & McDonald 2008], Entity-topic models [Newman et al. 2006a]
C5. Node coupling		C5A: $w(x_i,y_j\vert\cdot) = q(a_i,x_i)q(b_j,y_j)q(x_i,c_i\oplus\tilde{d}_j)q(y_j,\tilde{c}_i\oplus d_j)$ C5B: $w(x,y\vert\cdot) = q(a,x)q(b,y)[q(x,c\oplus d)]^{\delta(x-y)}\cdot[q(x,c\oplus\tilde{d})q(y,\tilde{c}\oplus d)]^{1-\delta(x-y)}$ $p(c,d\vert a,b) = \sum_x \vartheta_{a,x}\vartheta_{x,c}\sum_y \vartheta_{b,y}\vartheta_{y,d}$ *Correlation of submodels, relations*: Simple relational component model [Sinkkonen et al. 2008], Relational topic model [Chang & Blei 2009]

Figure 9.2: NoMM sub-structure properties. Notation (also see (9.3)): $a\oplus b$ adds counts $\vec{n}^{(a)}_{\cdot} + \vec{n}^{(b)}_{\cdot}$; $\tilde{a}\oplus b$ prevents $\neg i$ for a in (9.1); $c_{i\cup j}$ combines sequences $\{c_i, c_j, c_{i,j}\}$, as applicable.

numerically. For N3 nodes, $\vec{\vartheta}_a \sim p(\vec{\vartheta}|\lambda)$ refers to any distribution on the simplex and its associated inference methods. The most prominent example is the correlated topic model (CTM), where the logistic normal is used, a Gaussian transformed by the logistic function [Blei & Lafferty 2007] that captures correlation between dimensions (whereas the Dirichlet does not. For N4 nodes, also the Gaussian is the most common distribution, and the parameters of θ of the observation component distributions may be estimated using a separate M-step interlaced with the Gibbs step. Finally, the node/edge pair N5+E4 allows variants that include logistic regression and generalised linear models (variant N5B), as well as sampling-based schemes (variant N5A). Inference methods are not covered here, but some details are given in Appendix C.3.

E2 and E3 branches either generate samples autonomously (E2), or both children are forced to the same latent variable (E3), i.e., edges have identical values. Autonomous samples are drawn jointly. As an interpretation, branching may be seen as a common cause to several observed modalities, with E3 enforcing a token-wise relationship and E2 a correlation of mixed causes. Note that if both branches belong to the same sequence (variant E2B), the $\oplus$ in the E2 Gibbs weighting function in Fig. 9.2 expands according to (9.4). Branches belonging to different sequences (variant E2A) act like separate E1 edges, i.e., $q(a, x \oplus y) = q(a, x \oplus \tilde{y})q(a, \tilde{x} \oplus y)$.

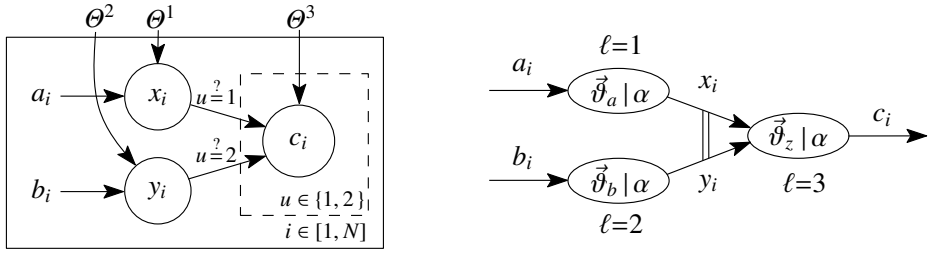
C2 combined indices represent index selection functions that are constructed out of several values. Sub-type C2A is a combination of an observed and one or more hidden edges, $k = (i, x_i)$, simply using them as component index dimensions, as in PAM4 [Li & McCallum 2006]. It allows to condition latent variables on a subset of the corpus and a hidden edge. C2B is the case where several hidden edges are used as index dimensions, $k = (x_i, y_i, \dots)$, thus joining different modalities that co-occur on a token level. Practically, C2B requires a different sequence in each parent edge and joins them in its index.[7] An example is Multi-LDA [Porteous et al. 2008a] where the C2B structure functions as a form of matrix factorisation. Finally, C2C allows arbitrary index selection functions, $k = g(\uparrow x_i, i)$, as in hPAM [Li et al. 2007a], or $k = g(\uparrow x_i, i, y_j, j)$.

C3 interleaved indices are intended as a simple method to merge two influences. Because they are a novel structure, they will be described in more detail in the subsequent Section 9.3.2.

C4 switches join sub-models, learning their importance from the data. Switches filter input edges according to the value of a parent node, as shown in [Titov & McDonald 2008, Newman et al. 2006a]. The full-conditional for this consequently filters count values according to the switch, which is represented by the Kronecker delta. Notably, the count totals in the switched branches then also vary while the sum of counts over the switched branches stays constant. C4 structures may be used as an alternative to C3 that adapts the importance of the parent branches. Note that a C2C structure may be used if only inputs of a single node are switched rather than propagation between a set of sub-models.

C5 node coupling results from sharing parameters among different nodes. This allows coupling of different sub-models without additional need for edges and has been of particular interest in analysing relational data, see, e.g., [Sinkkonen et al. 2008]. Importantly, also the case is covered that the nodes are sampled from jointly (C5B). The Gibbs term has a similar case distinction between $x{=}y$ and $x{\neq}y$ as C3B, which will be described in detail below.

[7] C2A joins C1A and C1B, creating a sequence node, and C2B joins multiple C1B indices, creating a topic node.



(a) Bayesian network (b) NoMM

Figure 9.3: Proposed C3B sub-structure.

Analogous to E2 and C3 types, if the two edge pairs are drawn from disjoint sequences (C5A), learning them with result in a ratio of influence according to the sequence length each of the partial nodes is "exposed" to.

9.3.2 Interleaved indices: Inference in the C3 structure

The novel C3 structure has been proposed in Chapter 5. It allows merging the influence of multiple edges by "interleaving" their values in a single component index of a NoMM node. Of particular interest here is the case C3B where both inputs are on the same sequence, $i = j$.

Its generative process corresponds to sampling for every token both incoming edges jointly "through" the same NoMM node. In the equivalent Bayesian network in Fig. 9.3, this is shown as a dashed plate: The two parent BN nodes x_i and y_i are used as indices to draw two samples c_i^u from the parameters Θ^3. The special constraint here is that the samples have identical values and are assumed to have been created in a single draw: $c_i \equiv c_{i,1} \equiv c_{i,2}$. Correspondingly, the Gibbs term derives from adding two counts and distinguishing the cases x=y and $x \neq y$: $[q(x, c \oplus c)]^{\delta(x-y)}[q(x, c \oplus \tilde{c})q(y, \tilde{c} \oplus c)]^{1-\delta(x-y)}$. If x=y, the total is reduced by 2 at element (x, c), where the $\Delta(\cdot)$ terms expand to two factors. Otherwise, elements (x, c) and (y, c) are decremented by 1 separately, which expands to the two terms $q(x, c \oplus \tilde{c})q(y, \tilde{c} \oplus c)$; cf. the E2B edge type. In both cases, the summed count statistics of both incoming NoMM edges are the main influences on the sampling weights.

The likelihood term for the C3B sub-structure is obtained by computing the likelihood of both partial tokens, $p(c_i^1, c_i^2 \,|\, a, b)$, and taking the square root since in reality there is only a single sample. In effect, this is the geometric mean of the likelihoods due to either branch.[8] The geometric mean is, however, almost always smaller than the arithmetic mean, which may make this structure relatively weak in terms of model likelihood.

In principle, also a simpler case C3A can be thought of where both branches are sampled independently in two distinct sequences. This leads to $w(x \,|\, a, c) = q(a, x)q(x, c \oplus \tilde{c})$ and is analogous for $w(y \,|\, b, c)$. The major numerical difference to C3B is the weight for x=y; both are asymptotically equal for $n_{x,c} + n_{y,c} \gg 1$. In either case, the w-term of node $\ell = 3$ is roughly equal to $q(x, c \oplus \tilde{c})q(y, c \oplus \tilde{c})$. The likelihood just has a term for the particular sequence i or j of the

[8] Note that for the likelihood, the two samples c_i^1 and c_i^2 become conditionally independent because parameters are given: $p(c_i^1, c_i^2 \,|\, \Theta^3, \cdot) = p(c_i^1 \,|\, \Theta^3, \cdot)p(c_i^2 \,|\, \Theta^3, \cdot)$. This is opposed to the partial term $w(x, y|\cdot)$ of the *collapsed* Gibbs sampler where parameters are marginalised.

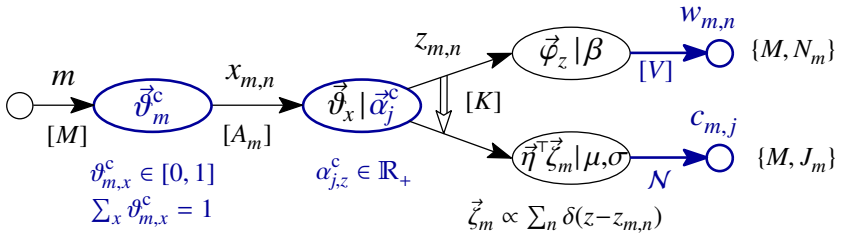


Figure 9.4: Summary of evidence structures.

current token $c_{i \cup j}$ because there is no "virtual" duplication of output tokens as in C3B. For the C3A case, the actual amount of data from each parent edge controls the influence on the model.

Effectively, in the C3 sub-structure the influences of both parent edges are superimposed, and compared to C2, the behavior is rather different: While the effect of a C2 structure is comparable to an *intersection* of the co-occurrences between pairs (x, c) and (y, c), the C3 structure is likely to behave closer to a *union* operator, with the specific behaviour that weights stabilise for configurations (x, y) where both branches are equal.

9.3.3 Incorporating evidence

Beside the structures that connect different quantities in a model, integration of evidence is a central aspect. In the typology introduced in Chapter 5, evidence is introduced by four methods:

- *Output terminals*: This is the primary method to model a sequence of observed data. Its presence does not change the forms of the Gibbs or likelihood equations, as explained above.
- *Nodes with observed parameters, N2*: From these nodes the responsible dimension is sampled for each data item. The respective N2 structure allows either a subset of items to be associated with the token, like a label for a document, or the parameters are given as an explicit set of multinomial parameters.
- *Hyperparameters, N1B, and structured priors, N3*: Some influence can be exerted through Dirichlet hyperparameters, especially structures N1B. Compared to observed parameters, hyperparameters have less influence on the overall model. A variant to introducing hyperparameters is to use a type N3 structured prior.
- *Aggregation output nodes, N5+E4*: A special case of output is the aggregation/regression branch consisting of N5 and E4 structures.

Using these instruments, which are summarised in Fig. 9.4, all data structures common with topic models are covered. With the terminology proposed by [Mimno & McCallum 2008], one may distinguish to incorporate evidence "up-stream", i.e., as N2 inputs, or "down-stream", i.e., introduce them as outputs.

9.4 Towards a model design method

The collection of structures in Section 9.3 along with their numerical properties may be considered a "library" that can be used as a basis for model design. In the following, a design method is sketched that follows this idea.

The development of such a method reconsiders the results of the previous sections and Part II of the thesis from a different viewpoint: asking what is the structure of a model if a specific mining task and the data are given. Our approach is to first look at the "semantics" of mixture levels in Section 9.4.1, then at methods to connect them in Section 9.4.2. Eventually, we pull all these insights together and propose a step-by-step design process in Section 9.4.3.

9.4.1 Mixture levels: The semantics of NoMM nodes

A mixture level in a NoMM consists of a node and its immediate child edges. If that node is a Dirichlet–multinomial (N1 type), it basically re-enacts the core of mixture models with MAP or Bayesian parameter estimation [McLachlan & Peel 2000], but the difference is that NoMMs provide the flexibility needed for advanced applications by flexible combination of levels. Other node types vary the N1 behaviour but (except regression nodes N5) follow the same principle.

Throughout machine learning, pattern recognition and knowledge discovery, there are different high-level assumptions on data that may be modelled using mixtures (or mixture levels):

- *Effect of different causes:* Mixtures merge the effect of different sub-models for an observation, where each sub-model is represented by a component in the node. Ideally, the node will learn its component parameters "automatically" (the sub-models). For this, however, an appropriate component selection function needs to be defined, which will be discussed below. A variant is approximation of complex probability distributions by simpler component distributions, like an exponential distribution by Gaussian components.

- *Co-occurrence of features; clusters:* Features at the output of a node can be considered to co-occur if caused by the same component. For instance, a mixture node with a component for each document m, $(\vec{\vartheta}_m \,|\, \alpha)$, uses the document as co-occurrence context. A document-specific language model is simply $(\vec{\vartheta}_m \,|\, \alpha) \xrightarrow[[V]]{w_{m,n}}$, and its parameters are trained directly from the (co-)occurrence) frequencies of the terms. Non-Dirichlet priors (N3 nodes) vary the clustering behaviour. Note that co-occurrence may be at different levels: Variables/edges may be drawn jointly from a single multinomial ("sub-sequence case" discussed for E*S and E2 structures) or conditionally independent given the node parameters.

  Co-occurrence may also be observed, for instance for labels that are jointly set for a given item. The corresponding structure is an N2 node.

  For continuous observations (N4 node), co-occurrence becomes correlation, referring to the values observed for a component (rather than the correlation between features in multivariate values drawn from a component distributions, e.g., multivariate Gaussians with non-zero off-diagonal covariance).

  Alternatively, a mixture level may be considered a set of clusters, one per component, and these clusters may be clustered themselves, using component grouping with hyperparameters, N3 nodes with non-Dirichlet prior or parent mixture levels.

- *Conditional independence and exchangeability; generative process:* Observations of a mixture level are conditionally independent samples given the component parameters. Furthermore, they are exchangeable, that is, their sequence of observation does not change the likelihood of the model. The exchangeability property directly links NoMM nodes to generative processes and the plate notation in Bayesian networks, so the more "traditional" design of topic models directly can be combined with a NoMM-specific design method.
- *Probabilistic matrices:* Mixture levels depict probabilistic matrices, which allows to create models that perform some sort of matrix decomposition. An N1 node $(\vec{\vartheta}_k \mid \alpha)$ in such a case is represented by a $K \times T$ matrix $\underline{\Theta}$ with elements $\vartheta_{k,t} = p(t|k)$ (cf. Section 3.5.4). Matrix multiplication is done using $\underline{\Theta}^1\underline{\Theta}^2$. Appropriately defining and connecting matrices using N1 nodes can thus be used to perform matrix decomposition, with the actual decomposition operation performed by inference algorithms like Gibbs sampling.

Although many of these assumptions are interchangeable, depending on the particular situation it is more intuitive to use only one when designing a model, or a suitable subset.

The default modelling structure is the Dirichlet–multinomial node type N1. It may be considered an adaptive mixture (with respect to the parameters automatically learned from the data) and thus the "Swiss knife" of topic modelling. The N1 structure can be substituted by other node types as soon as special properties on the data or model come into play: According to Fig. 9.2, non-discrete observations lead to N4 nodes, and specific assumptions on the dependency between the components of a mixture may lead to an N3 node. Moreover, structure types N2 and N5 comprise particular situations of incorporating evidence (see Section 9.3.3).

Looking at the numerical implications of an assumption in the list above, the situation looks straight-forward: An N1 node as the most illustrative correspondence has a straight-forward Gibbs term of $q(k, t)$. The behaviour of this q-term is to increase if k and t are observed together according to the Dirichlet clustering property (cf. Section 3.4.2). This clustering may in addition be controlled using component grouping, i.e., using different vector hyperparameters for subsets of components.

Such building blocks cannot "live" on its own as models, though. The actual mixing properties are encoded in the interaction between different levels.

9.4.2 Mixture interaction: The semantics of NoMM topologies

The other main ingredient of our method is the interaction between the mixture levels. The main side-conditions in this interaction are dictated by the available data, the tasks to be performed by the model (which may be expressed by some desired probability distribution or quantity derived from it) and finally the assumptions made on the data as described in the previous section. These core modelling conditions are complemented by assumptions on interaction between mixture levels, which glue them together between the boundaries of the NoMM terminals (i.e., observed data):

- *Root terminals* represent an observable quantity like a sequence index that is used as a co-occurrence context m, "quantities $\{x, y\} \in \mathcal{V}$ co-occur within m". Many NoMMs have a root terminal that corresponds to the largest logical level in the data, like a document, and many models use just this context to model their data – implying it as the co-occurrence context.

- *Leaf terminals* represent observed tokens along a known sequence. If no aggregation is used, several values in leaf terminals belong to a single value in a parent edge or root terminal, thereby creating grouping. Aggregation transforms a sequence of tokens into a single output using regression or selection of tokens.

Adding building blocks between these terminals is the main task when mapping assumptions to NoMM structures. There are different "instruments" to structure a model, which are used in combination:

- *Grouping data and adding co-occurrence contexts.* To discriminate different mixture components or their features, they need to be spread over different contexts, i.e., different groupings of the data. Data groups may be hierarchically structured, like documents grouping sections that themselves group word tokens, which on the section level leads to E1S edge structures. Furthermore, different completely independent sequences can be defined as root terminals, such as documents and users in a recommender scenario, and both co-occurrence contexts are merged in an edge with a sequence whose index consists of a document–user tuple, e.g., using a C2B structure.

  Furthermore, component grouping may be considered, which allows additional clustering, this time of subsets of components, by associating different vector hyperparameters to subsets of components of a node.

- *Adding dependencies and latent variables.* As discussed in Chapter 4, dependence between model variables spans over different levels that are connected via hidden component indices. Adding a mixture level with a hidden component index therefore connects its input and output edges, and this is done as a latent variable with an intermediate N1 mixture and its associated mixture-level assumptions (see above).

- *Chaining, merging and branching.* Dependent variables may be linked in different ways: By chaining different levels (using "straight" E1 and C1 structures), by branching (E* structures) and by merging (C* structures). The latter are the instruments to handle especially multi-modal data.

  Depending on the branch or merge type, the dependency between sub-models will become more or less strong. An E1 edge or E2 structure with branches on disjoint sequences posits conditional independence given the parent node, leading to a w-term for each of the values t_1 and t_2: $w_{\text{E2A}} = q(k, t_1 \oplus \tilde{t}_2)$. Depending on the configuration, these values may even be drawn from different components. An E2 branch with co-occurring tokens $\{t_1, t_2\}$ leads to a w-term of $w_{\text{E2B}} = q(k, t_1 \oplus t_2)$ that slightly favours joint draws of t_1=t_2 from the same parent component $\vec{\vartheta}_k$ but still associates weight to combinations with $t_1 \neq t_2$. An E3 branch enforces $t_1 \equiv t_2$ and only samples one value from the node. The same is true for component selectors: While a C2 edge that maps to a component for each combination of input indices will require input variables to co-occur on a token level ($w_{\text{C2}} = q((k_1, k_2), t)$), the novel C3 structure requires a much weaker co-occurrence of any of its components with the output value ($w_{\text{C3}} \approx q(k_1, t \oplus \tilde{t})q(k_2, \tilde{t} \oplus t)$).

- *Adding special structures.* In addition to the "standard" mixture levels, special structures like regression branches should be considered.

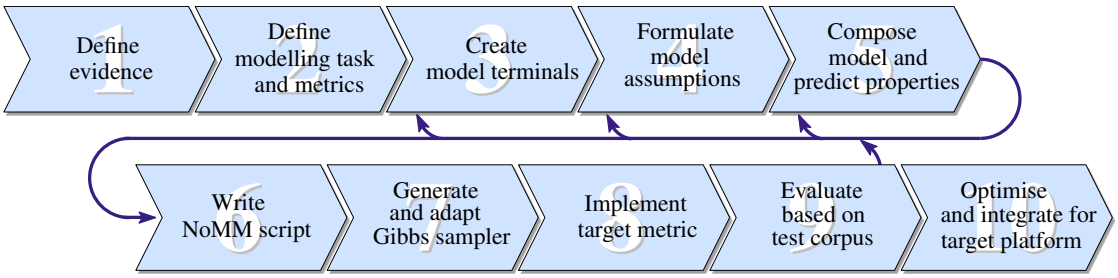


Figure 9.5: NoMM design process.

9.4.3 Design process

Based on the sub-structure library and the considerations on the semantics of NoMM nodes and edges discussed in the previous sections, a design strategy is outlined. Along with any task-specific metrics to capture model quality or computational complexity, this method specifically allows to use the q-terms of the Gibbs full-conditional and the data likelihood as predictors of model behavior. Its steps are as follows:

1. Define *evidence*, i.e., what data are available as model input: This includes a formal definition of the data model modalities (type of documents, metadata, relational structure) and groupings in the data themselves (not assumed groupings or clusterings).

2. Define modelling *task*, i.e., what is expected from the model: This is expressed as a probability distribution or a metric that is based on such a distribution, so it can be related to the quantities in the model itself. Also, side-conditions like maximum computational complexity should be considered here.

3. Create model *terminals*, i.e., map evidence to model structures using the evidence structures described in Section 9.3.3. Predominantly, data will be designed as NoMM outputs and groupings as inputs, but the decision is based on the modelling task. Ideally, the target metric can be directly expressed as a function of the parameters trained by the model, but often it is necessary to transform the parameters via marginalisation, Bayes' rule, the chain rule, etc.

4. Formulate *assumptions*, i.e., how the data modalities behave in mutual relations and what hidden structure they may stem from. This defines assumptions of the types described in Section 9.4.1 (mixtures, co-occurrences, etc.) or other qualitative descriptions of model behaviour.

5. *Compose* the model structure and *predict* its behaviour: Using structures from the "cheat sheet" in Fig. 9.2, map assumptions to structures. This takes into account the structuring criteria in Section 9.4.2 and can start either at the "open" terminals, trying to connect them with appropriate structures, or by considering some successful related model from prior work. The composition/prediction step exploits the benefits of the NoMM-specific modelling method best because whenever the model structure is modified, the implications may be directly seen in the inference and likelihood equations of the complete model, using

Fig. 9.2. A powerful variant is to carry out the composition/prediction step numerically: The model itself is looked at as a product of q-functions, and modifying this product may encode the assumptions directly. In this case, prediction indeed becomes identical to composition.

6. *Iterate* or *implement* the model: Optionally return to Step 3, 4 or 5, depending on how well the model appears to reflect the task at hand. Otherwise implement and evaluate it and optionally iterate subsequently.

Integrated design process. In Fig. 9.5, an overall process for NoMM design is visualised. This process integrates the five initial design steps above as the input to the implementation workflow developed in Chapter 6. Iterations can then be triggered after structuring the model and predicting its behaviour in Step 5 and after implementing it and evaluating it with the likelihood or against the task-specific metric defined in Step 2. The difference is that automatic generation of the likelihood is in most cases supported by the Gibbs meta-sampler, whereas the target metric has to be implemented manually. Better model likelihood values are an indicator for a tendency to better performance in other metrics. This is to be taken with a grain of salt, however, because some metrics, including retrieval precision and subjective topic coherence measures, may be worse for models with better likelihood (see Section 3.7); the latter yields merely a coarse-grained predictor for model quality.

Completing the integration, when defining data and tasks in the domain of virtual communities, Steps 1 and 2 may be supported by formulating the problem with an AMQ schema as introduced in Chapter 2, which will be shown in the next chapter.

9.5 Conclusions

In this chapter, a method was proposed that allows construction of NoMMs from modular structures. As a prerequisite for this, the decomposition of NoMMs has been investigated and a "library" created that collects structures along with their modular Gibbs full conditional and likelihood terms.

Based on the library, which covers the typology in Chapter 5, model construction supports a wide range of models, and its modularised strategy makes it possible to directly view the impacts of NoMM modification on model properties. This way, the model design process is assisted by qualitative criteria.

By integrating the design process with the Gibbs meta-sampler developed in Chapter 6, a complete implementation framework has been realised. Compared to more traditional ways of topic model design, which often involves the formulation of a generative process and mathematical derivation of inference equations "from scratch", the new method can offer a significant increase of design efficiency, in addition to the short-cuts to model implementation provided by the Gibbs meta-sampler.

Beyond usage with NoMMs in Gibbs sampling, it is expected that it is straight-forward to modify the method (1) to work with other inference approaches, such as variational inference and its collapsed counterpart [Teh et al. 2007], and (2) to map structures of Bayesian networks to inference equations. This establishes compatibility with the approaches to topic modelling common in the literature.

Chapter 10

Case study: Topic modelling for virtual communities

To validate the "tools" developed in the thesis, they are applied to a concrete application scenario: expert finding using document content and semantic annotation information. This chapter demonstrates a "round-trip" process to topic modelling, starting from AMQ models as input to the design method, designing the model structure, implementing the algorithms using the Gibbs meta-sampler and finally evaluating the models. Besides serving as exemplary validation for the NoMM design process developed in this thesis, the proposed expert–tag–topic models are contributions in their own right.[1]

10.1 Introduction

This chapter returns to the original motivation of developing topic models in this thesis: their application to data of virtual communities. Such data often exist in the form of text in documents, which have associated with them meta-data like annotations and ratings, comments and tags, as well as relational information like authorship, citation and linkage on the Web, as discussed in the scenarios in Chapter 2.

Because of the diversity of the data structures across the particular modelling tasks, the range of possible model structures is wide. This makes a simplified design method particularly useful, and the goal of this chapter is to demonstrate this for an example case. In particular, a scenario is investigated that uses data of scientific digital libraries as a basis: expert finding using the support of additional document annotations.

The process of model construction follows the ideas developed in the previous chapter, starting from AMQ models as original formalism to define the problem instances. Based on this and modelling assumptions, NoMMs are composed that can be input into the Gibbs meta-sampler implementation workflow of Chapter 6 and augmented by the fast sampling methods of Chapter 8. Finally, evaluation will be based on the methods outlined in Section 3.7.

Chapter outline. This chapter develops several iterations of an expert-finding model in Section 10.2, and Section 10.3 discusses related work. An empirical study of the models is presented in Section 10.4. Finally, Section 10.5 discusses the results and outlines future extensions.

[1] An early version of this chapter has been published in [Heinrich 2011b], including the ETT1 and ETT3 models.

10.2 Expert–tag–topic models: Finding knowledge via tagged documents

The scenario we consider is that of finding knowledgeable people in a business or scientific community, for instance answering the question: "Who knows about models of binaural sound localization?" to start a project on a novel augmented reality system that integrates spatial audio. Imagine an enterprise of a few 100 or 1000 employees that is spread over different geographic locations – a typical form of today's medium-sized businesses. Normally, immense amounts of documentation exist in such virtual communities that give cues to where knowledge is located. Similarly, in the scientific world, much of the knowledge is documented in publications that are openly available along with authorship and citation information.

Recalling the models discussed in the previous chapters, pure expertise finding has been solved as a topic model with the author–topic model (ATM) [Rosen-Zvi et al. 2004], which has been outlined in Fig. 4.5(b). However, in many cases the data available go beyond author–document links. In particular, often meta-data exist that can be used to improve search results. For instance, subject descriptors, such as the Dewey and universal decimal classifications (DDC, UDC), ACM CCS, Medline MeSH or IPTC news subject descriptors may offer important cues for disambiguation and retrieval. Furthermore, user-generated tags with controlled or free vocabulary may be used to enhance the data of the community.

For this chapter, we will refer to such meta-information as "tags", being aware of the danger of over-simplification due to the different provenience, semantic properties and confidence. For our modelling scenario, we assume that these tags are filtered to an extent that ensures a minimum level of (1) confidence with regard to semantic association and (2) coverage in the corpus (document frequency). We do, however, accept that tagging data are neither unique (several tags may have overlapping semantics) nor complete (media may be untagged or have associated to them only a subset of the relevant tags).

The model in question should be able to benefit from the additional tagging information and (1) associate expert authors to meta-information items (because they should have particular meaning in the community) and (2) improve topics in the sense of increased topic-based retrieval performance and improved topic qualities, such as coherence. In the case of the binaural hearing expert above, the model should be able to locate experts that have written documents that contain terms around the subject and/or are tagged by tags like "*hearing*", "*3d acoustics*" or "*human psychophysics*", which may help disambiguate terms of a document term (at training time) and expert finding query (at retrieval time). In addition, experts whose documents are not tagged should be retrievable for semantically related tags.

AMQ representation. Following the ideas of Chapter 2, the AMQ schema of the scenario is shown in Fig. 10.1. In particular, the observed data include the expert authors $a \in \vec{a}$, media m (here: text documents) and the $\mathit{authoring}(a, m)$ relation. Regarding quality classes, in addition to topics z as estimated descriptors of community knowledge we define tagging information c generically as an observable modality of the media, which is connected with media via a $\mathit{hastag}(m, c)$ relation. All of these relations are of arbitrary cardinality, so the community is represented by a text corpus with multi-labelled authorship and tag information.

Regarding AMQ-relations to be inferred, actors are connected to qualities by a $\mathit{knows}(a, q)$ relation, which may be both to topics $z \in \mathcal{Q}$ and labels $c \in \mathcal{Q}$. Furthermore, the topics of word

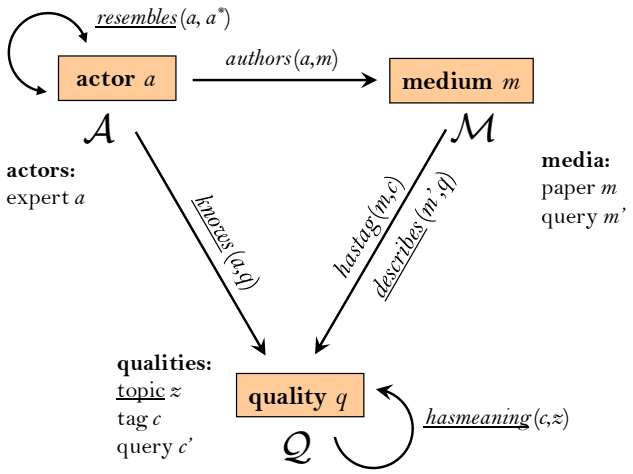


Figure 10.1: Expert–tag–topic scenario, AMQ schema.

queries m' may play a role in modelling, as do the topics of tags c. The former are represented by the *describes*(m', q) relation, and the inferred topics of a tag by *hasmeaning*(c, z).

Based on this, we will run through the first half of the design process outlined in Chapter 9 in the remainder of this section, and in Section 10.4 through the second half.

10.2.1 Designing an expert–tag–topic model

We design a NoMM to illustrate the design workflow in a relatively small example. To demonstrate the process, we will create the model from scratch, deliberately not backed by literature or example models from Chapter 5. We will consider related work subsequently.

Evidence: The data that represent the information in the AMQ representation correspond to the evidence used for the model: Media m contain text $\vec{w} = \{\{w_{m,n}\}_{n=1}^{N_m}\}_{m=1}^{M}$, and the *authors*$(a, m)$ relation corresponds to authorship distributions $\vec{a} = \{\{a_{m,a}\}_{a=1}^{A}\}_{m=1}^{M}$ where $a_{m,a}$ is the portion of document m that expert a is responsible for. Similarly, the *hastag*(m, c) relation is defined between media m and tags c with a tag distributions $\vec{c} = \{\{c_{m,c}\}_{c=1}^{C}\}_{m=1}^{M}$. Both $\vec{a}$ and $\vec{c}$ are sparse, and when iterating over any, by convention we only consider non-zero elements, e.g., the tags for a document can be described as $c_{m,j}\ \forall\ j \in [1, J_m]$ with $J_m > 1$ for multilabelled documents.

For retrieval, word queries $\vec{w}'$ are considered, labels $\vec{c}'$ (allowing combinations), or both. A summary of these quantities is given in Fig. 10.2, already including model quantities described below.

Tasks and metrics. The model should score the expertise of authors given text queries $\vec{w}'$, and tags, $\vec{c}'$. We opt for the query likelihood model of information retrieval discussed in Section 3.7. Word queries will therefore be scored according to the likelihood $p(\vec{w}' \mid a)$, tag queries according to $p(\vec{c}'|a)$. Ranking of experts can be measured using truncated average precision. Furthermore, we are interested in the quality of the topics produced by the model, $p(w \mid z)$, analysing potential improvements by adding tag information to the model.

M, A	number of documents and authors (media, actors) in community corpus
V, C	vocabulary size for terms and tags
K	number of topics
N_m	number of words in document m
$\vec{a}_m, A_m$	author distribution, number of authors for document m (observed)
$\vec{c}_m, C_m, J_m$	tag distribution, number of tags for document m (J_m includes repetitions)
$w_{m,n}$	word sequence (observed)
$c_{m,j}$	tags for document m (observed, $j \in [1, J_m]$)
$x_{m,n}, x_{m,j}, u_{m,n}$	author and tag sequences associated with word $w_{m,n}$ or tag $c_{m,j}$ (inferred)
$y_{m,n}, y_{m,j}, z_{m,n}$	topic sequences associated with word $w_{m,n}$ or tag $c_{m,j}$ (inferred)
u, w, x, y, z	values for tag, word (term), author, topic (twice)
$(\vec{\vartheta}_x \mid \alpha)$	NoMM node for author-specific topic distribution $p(z\|x)$ and hyperparameter α
$(\vec{\varphi}_z \mid \beta)$	NoMM node for topic-specific word distribution $p(w\|z)$ and hyperparameter β
$(\vec{\psi}_z \mid \gamma)$	NoMM node for topic-specific tag distribution $p(c\|z)$ and hyperparameter γ
$(\vec{\zeta}_u \mid \gamma)$	NoMM node for tag-specific topic distribution $p(z\|c)$ and hyperparameter γ

Figure 10.2: Quantities in the ETT models.

Terminals. We choose model terminals according to Section 9.3.3. As we want $p(w \mid a)$ and $p(c \mid a)$, the apparently best choice of NoMM terminals is to put authors a up-stream, i.e., into an N2 node with observed parameters $\vec{a}_m$; cf. Fig. 9.4. This way, the distributions in question can be directly pulled out of the model by marginalising over any hidden edges to be defined in the structuring process. Otherwise, "inversion" of conditional distributions is required using Bayes' rule, which may be costly and may require definition of additional priors. Regarding tags, the most natural terminal seems to be down-stream, due to $p(c|a)$. All of these considerations lead to a black-box model as presented in Fig. 10.3.

Modelling assumptions. We make the following assumptions on the data:

(a) An author a is an expert of all self-authored documents. Association (and expertise) is weighted by the portion of authorship $a_{m,a}$ in the *authors*(a, m) relation.

(b) The thematic description of expertise is based on topics z, and each author has a single field of expertise represented by one topic distribution (this is a simplifying assumption on *knows*(a, z)).

(c) A tag has semantic meaning that may be expressed as a mixture of topics y (relation *hasmeaning*(c, y)) with y used to distinguish tag topics from topics z.

In addition, the standard definition of topics is used: Topics y and z are multinomial mixtures of some vocabulary (here: tags and words), and we assume that topics are global to the corpus.

Composition/prediction. We start from the left, looking at the authorship information $a_{m,a}$ (assumption (a)). As there are multiple authors per document, one way of splitting the document is to identify the author of each paragraph or word token (m, n). We will use words because there is no other data available from the terminals. We thus assume that when every author writes a portion then every word in a document can be associated with one originator. A NoMM structure that allows partitioning a set (document words) into several clusters (authors) is the N2 node,

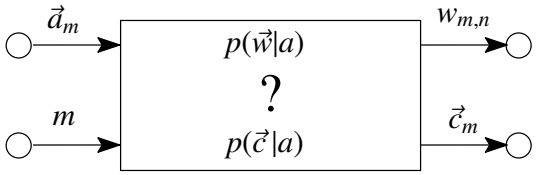


Figure 10.3: ETT model design: Terminals.

$\xrightarrow{m} (\vec{a}_m) \xrightarrow{x_{m,n}}$, cf. Section 9.4.1. Conditioning on $x_{m,n}$, other quantities in the model become author-specific. For prediction of the partial model, the sub-structure library in Fig. 9.2 is used, and the joint Gibbs full-conditional weights (the w-term; see Chapter 9) become:

$$w(x, \ldots | \vec{a}, \ldots) = a_{m,x}\, q(x, \ldots) \ldots , \tag{10.1}$$

assuming that the N2 node feeds some other node with a q-term and leaving out sequence indices for the moment. We focus on prediction using the w-term as a representation of the full conditional because its q-terms show model behaviour as interaction between discrete variables, it allows inclusion of $\oplus$ operators and we can directly rewrite it to one or more full conditionals.

There are several possibilities to connect the authors with the outputs: Because an author has a single field of expertise and this is represented by topics according to assumption (b), we can connect edge $x_{m,n}$ with an N1 node structure that outputs topics $z_{m,n}$, $\xrightarrow{x_{m,n}} (\vec{\vartheta}_x \,|\, \alpha) \xrightarrow{z_{m,n}}$, which has a q-term of $q(x, z)$. There is no grouping by documents because the expertise of an author for all documents is required. Because no other assumptions on the structure of topics exist (such as any hierarchy of expertise topics), the output $w_{m,n}$ can be directly hooked to the topic edge by way of an N1 topic node, adding $\xrightarrow{z_{m,n}} (\vec{\varphi}_z \,|\, \beta) \xrightarrow{w_{m,n}}$ with q-term $q(z, w)$. This leads to:

$$w(x, z, \ldots | \vec{a}, \vec{w}, \ldots) = a_{m,x}\, q(x, z)\, q(z, w) \ldots \tag{10.2}$$

As we have two outputs and assumption (c) defines tags as mixtures of topics, we could add a node structure, $\xrightarrow{x_{m,n}, y_{m,n}} (\vec{\psi}_{x,y} \,|\, \gamma) \xrightarrow{c_{m,j}}$, which is conditioned on the author and tag topic and outputs tags $c_{m,j}$. The component index (x, y) would mean that the association between topics y and labels be author-dependent. We did not, however, make such an assumption. Therefore, we set up the new node as $(\vec{\psi}_y \,|\, \gamma)$, or $q(y, c)$, respectively. Connecting this with the partial model above can be done by adding the tag topic y as output to $q(x, z)$, which leads to the NoMM in Fig. 10.4(a) and the following Gibbs term:[2]

$$w(x, z, y \,|\, \vec{a}, \vec{w}, \vec{c}, \cdot) = a_{m,x}\, q(x, z \oplus y)\, q(z, w)\, q(y, c)\,. \tag{10.3}$$

This equation deliberately avoids to make sequences explicit. The idea is to show how the actual dimensions x, z and y (as elements in the count statistics) influence each other. Inserting concrete

[2] In Appendix E, the Bayesian network and the "standard" derivation of the ETT1 model are given.

sequence indices according to Fig. 10.3(a), this expands into two full conditionals:

$$p(x_{m,n}{=}x, z_{m,n}{=}z \,|\, \vec{a}, \vec{w}, \vec{c}, \vec{x}_{\neg m,n}, \vec{y}, \vec{z}_{\neg m,n}, A) \propto a_{m,x} \cdot \frac{n^{(z)}_{x,z,\neg m,n} + n^{(y)}_{x,z} + \alpha}{n^{(z)}_{x,\neg m,n} + n^{(y)}_{x} + K\alpha} \cdot \frac{n_{z,w,\neg m,n} + \beta}{n_{z,\neg m,n} + V\beta} \tag{10.4}$$

$$p(x_{m,j}{=}x, y_{m,j}{=}y \,|\, \vec{a}, \vec{w}, \vec{c}, \vec{x}_{\neg m,j}, \vec{y}_{\neg m,j}, \vec{z}, A) \propto a_{m,x} \cdot \frac{n^{(z)}_{x,y} + n^{(y)}_{x,y,\neg m,j} + \alpha}{n^{(z)}_{y} + n^{(y)}_{y,\neg m,j} + K\alpha} \cdot \frac{n_{y,c,\neg m,j} + \gamma}{n_{y,\neg m,j} + C\gamma} \tag{10.5}$$

where a superscript in parentheses refers to the branch of a partial count and hyperparameters are collected in the set A.

This first ETT model, ETT1, connects authors, labels and words with an E2 branching structure and assumes that topics $y_{m,j}$ and $z_{m,n}$ are generated disjointly from the author-specific distributions, therefore keeping the structure of $q(x, y \oplus z)$ simple (type E2A; cf. Section 9.3) and allowing to split it into a sweep through (m, n) and one through (m, j).

However, this implies that labels are like a special type of word, and the model depends on the ratio of the number of labels defined for each document, as the counts for $n_{x,z}$ and $n_{x,y}$ in $q(x, y \oplus z)$ effectively add according to (10.3).

Tag boosting. What may be done to adjust the potentially low influence of tag distributions on the topics is to boost them by repeatedly sampling their indices (or equivalently by generalising the Pólya urn scheme to over-replace not one but several items for a sampled one; cf. Section 3.4.2). The variable J_m can then be re-defined as the length of the sequence of repeated topic labels, assuming a round-robin iteration: $c_{m,j} = c_{m,j \bmod C_m}$ where C_m is the number of unique tags for document m.

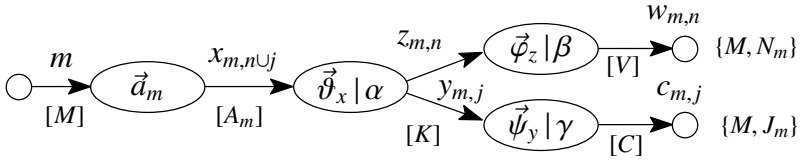
Retrieval functions. To use the trained model for retrieval, we must finally define how to rank authors given a query. We distinguish the case where we want to obtain authors from (1) a term query and (2) a tag or set of tags. Applying the query likelihood method, we can compare authors by how likely their query is according to topic parameters:

$$p(\vec{w}' \,|\, a) = \prod_{w \in \vec{w}'} \sum_z \vartheta_{a,z} \varphi_{z,w} \qquad\qquad p(\vec{c}' \,|\, a) = \prod_{c \in \vec{c}'} \sum_z \vartheta_{a,z} \psi_{z,c} \,. \tag{10.6}$$

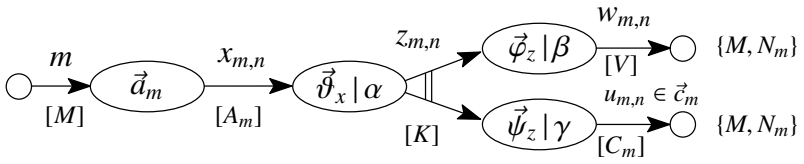
Combined queries use the product of both likelihoods. Note that the simple structure of (10.6) reflects the initial design choices.

However, in (10.6) the amount of contributions an author has published is ignored. While it is well possible that authors are experts even if they published only a single (but potentially seminal) paper in the community, we assume that the frequency of appearance within the collection is an additional relevance cue. An intuitive indicator for author importance is the number of times that an author is associated with word tokens according to the trained model. Based on the authorship edge $x_{m,n}$, we may construct a weight based on the (query-independent) author probability according to the model, $p(a) \propto \sum_m \sum_n \delta(a - x_{m,n})$. The retrieval functions can then be turned into a score for each author given query parameters $\vec{w}'$ and/or $\vec{c}'$:

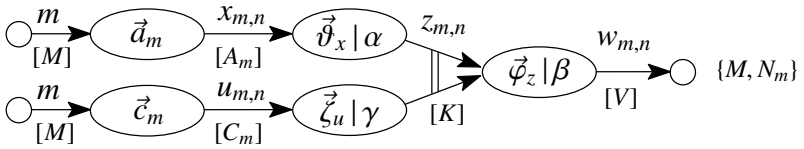
$$s(a \,|\, \vec{c}', \vec{w}') = \log p(\vec{w}', \vec{c}' \,|\, a) + \lambda \log p(a) \tag{10.7}$$



(a) ETT1: E2A-based.



(b) ETT2: E3-based, using filtered parameters.



(c) ETT3: C3B-based, tags as observed parameters.

Figure 10.4: ETT NoMM designs.

where λ weights the influence of author importance on the model, with $\lambda = N'$ (the query length) for an AND condition between query relevance and author importance and 0 for no influence. The logarithms in (10.7) avoid precision underflow for larger queries.

10.2.2 Iterating the model

The E2 structure appears sub-optimal at the core of the ETT1 model, as it might limit the influence of tags on the topic distributions for experts, $\vec{\vartheta}_x$. Thus we iterate the model to explore alternatives.

Iteration 1: Filtered parameters. We can force both topic edges to be equal, $y_{m,n} \equiv z_{m,n}$, using an E3 branch. This creates a stronger dependency of the topics $z_{m,n}$ on the label information. However, in this case the sequence of the tag outputs is (m, n), one for each word, instead of (m, j), one for each distinct or repeated tag. This creates a situation similar to the case of N2 nodes where an author $x_{m,n} \in \vec{a}_m$ is sampled for every word.

As a point of departure, we can force the tags to be in the set of the ones observed for the particular document, defining a tag sequence $u_{m,n} \in \vec{c}_m$, where $u_{m,n}$ is any non-zero element of $\vec{c}$. This may be done by filtering the distributions of topics $\psi_{k,c}$ in ETT1 to be confined to $\vec{c}_m$, the set (or distribution) of labels for a document m:

$$\psi^{\mathrm{f}}_{k,m,u} \propto \delta(u - \vec{c}_m)\, \psi_{k,u} \tag{10.8}$$

where $\delta(u - \vec{c}_m)$ is a Kronecker delta that is 1 if $u \in \vec{c}_m$ as defined above and 0 otherwise.

The consequence of this exotic parametrisation is that the multinomials cease to be Dirichlet-distributed with $\mathrm{Dir}(\gamma)$, but rather obey a "filtered" Dirichlet distribution:

$$\vec{\psi}^{\mathrm{f}}_{k,m} \,|\, \gamma, \vec{c}_m \sim \mathrm{Dir}^{\mathrm{f}}(\vec{\psi}^{\mathrm{f}}_{k,m} \,|\, \gamma^{\mathrm{f}}_m, \vec{c}_m)\,, \tag{10.9}$$

which is a standard Dirichlet $\mathrm{Dir}(\cdot|\,\gamma^{\mathrm{f}}_m)$ over the elements of $\vec{c}_m$ that returns zero for all other dimensions.

This distribution is not an ideal solution because it is only valid for documents with identical $\vec{c}_m$ and thus it will be difficult to estimate $\psi_{k,u}$ globally.[3] We will therefore attempt the "balancing act" of using a standard Dirichlet over the filtered parameters, $\vec{\psi}^{\mathrm{f}}_{k,m} \sim \mathrm{Dir}(\vec{\psi}^{\mathrm{f}}_{k,m} \,|\, \gamma)$, taking into account that estimation methods may not live up to their optimal performance and that the model likelihood may be handicapped. However, we expect this misadaptation to be of limited effect for two reasons: (1) We expect some robustnesss to adverse parametrisations: The scalar hyperparameters commonly used with topic models empirically work well despite the fact that the values of true count statistics are far from equal for all dimensions – they rather follow power-law distributions with few significant weights, not unlike ψ^{f}. And (2) the Dirichlet has an aggregation property: Given some $\{\vartheta_1, \vartheta_2, \vartheta_3, \ldots, \vartheta_K\} \sim \mathrm{Dir}(\{\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_K\})$, any aggregation of dimensions will re-parametrise the distribution on a sub-simplex: $\{\vartheta_1 + \vartheta_2, \vartheta_3, \ldots, \vartheta_K\} \sim \mathrm{Dir}(\{\alpha_1 + \alpha_2, \alpha_3, \ldots, \alpha_K\})$. Transferring this to our case, the filtered Dirichlet in effect aggregates the weight of the components filtered away. Therefore, with a standard Dirichlet we obtain a "view" on the filtered Dirichlets that is less "distorted", the more weights $\{\psi_{k,u} : u \in \vec{c}_m\}$ belong to relevant unfiltered tags. In turn, we may lose some robustness against documents with wrong tags as the distribution will have fewer alternatives to associate topics with.

In Fig. 10.4(b), the NoMM of the complete ETT2 model is shown. The Gibbs weights become:

$$w(x, z, u \,|\, \vec{d}, \vec{w}, \vec{c}, \cdot) = a_{m,x} q(x, z) q(z, w) q(z, u) \tag{10.10}$$

with constraint $u \in \vec{c}_m$. Inserting sequence indices, this expands to:

$$p(x_{m,n}{=}x, z_{m,n}{=}z, u_{m,n}{=}u \,|\, \vec{d}, \vec{w}, u_{m,n} \in \vec{c}_m, \vec{u}_{\neg m,n}, \vec{x}_{\neg m,n}, \vec{z}_{\neg m,n}, A) \propto$$

$$a_{m,x} \cdot \frac{n_{x,z,\neg m,n} + \alpha}{n_{x,\neg m,n} + K\alpha} \cdot \frac{n^{(w)}_{z,w,\neg m,n} + \beta}{n^{(w)}_{z,\neg m,n} + V\beta} \cdot \frac{n^{(u)}_{z,u,\neg m,n} + \gamma}{n^{(u)}_{z,\neg m,n} + C\gamma}\,. \tag{10.11}$$

Due to the filtering constraint, this sampling distribution is restricted to weights for $u \in \vec{c}_m$.

Retrieval functions. One of the core assumptions in ETT2 is that words and tags always appear jointly. What may be done if any are missing in a query is to assume a non-informative distribution $\varphi_{z,t} = 1/V$ or $\psi_{z,c} = 1/C$ for the unknown data, thus in effect ignoring their influence. Under this assumption, ETT2 shares its retrieval functions with ETT1, (10.6). In a combined query, the

[3] We may consider using N3A structures with hierarchies of hyperparameters or structured Dirichlets, cf. [Wallach 2008, Andrzejewski et al. 2009], but we opt for a simpler solution.

retrieval function becomes:

$$p(\vec{w}', \vec{c}' \,|\, a) = \prod_{w \in \vec{w}'} \sum_z \vartheta_{a,z} \varphi_{z,w} \sum_{c \in \vec{c}'} \psi_{z,c} \tag{10.12}$$

where $\sum_{c \in \vec{c}'} \psi_{z,c}$ is a query-specific weighting function over topics that is equal for all authors. As in ETT1, scoring can be done using (10.7).

Discussion. Interpreting (10.10), the stronger influence of tags $u \in \vec{c}_m$ now is clearly seen in the appearance of the topic z in the term $q(z, u)$: The factor rises whenever a topic co-occurs with a certain tag, and at the same time $q(z, w)$ pulls the topics towards appropriate word clusters. On the down-side, this close dependency may add a hard constraint on the topics: The model must fit them to the word and to the label co-occurrences jointly. This may result in lower model likelihood and may even be sensitive to irrelevant tags, which then jeopardise the associations between words and topics.

Moreover, because of the strong dependencies between variables, it may be difficult for the sampler to obtain a good initial state. To alleviate this, a candidate state may be obtained by seeding all topics with the content of a small number of documents from the corpus (similar to [Blei et al. 2003b]): For every k, all seed documents obtain $z_{m,n}{=}k$ and the co-occurrence counts with authors, tags and words are updated accordingly. Empirically, multiple documents assigned to one topic should be semantically coherent (for instance, share a tag).

Iteration 2: Mixture merging. As an alternative to the first model and the filtering approach, we return to the initial decision to set up terminals as in Fig. 10.3. Having both authors and tags as a similar set of inter-document groupings, we may use this similar character as a symmetry in the model. This is possible by "bending around" the tag branch to become an input, i.e., converting the output terminal c to an N2 node at the root of the model. More importantly, not only can we accommodate multiple tags per document, but even explicit tag weights can be introduced because the N2 branch will use $\vec{c}$ explicitly, analogous to the authorship strength $\vec{a}$. Two parent branches in turn lead to using the C3 index interleaving structure, and the resulting model is shown in Fig. 10.4(c). Assembling the structure with the term for C3B from the NoMM structure library will result in the following Gibbs term, where the variables $z^{(x)}$ and $z^{(u)}$ refer to the partial contributions of the respective parent branches:

$$\begin{aligned} w(x, z^{(x)}, z^{(u)}, u \,|\, \vec{a}, \vec{w}, \vec{c}, \cdot) = a_{m,x} q(x, z^{(x)}) c_{m,u} q(u, z^{(u)}) \cdot [q(z^{(x)}, w \oplus w)]^{\delta(z^{(x)} - z^{(u)})} \\ \cdot [q(z^{(x)}, w \oplus \tilde{w}) q(z^{(u)}, \tilde{w} \oplus w)]^{1 - \delta(z^{(x)} - z^{(u)})} \end{aligned} \tag{10.13}$$

$$\approx a_{m,x} q(x, z^{(x)}) c_{m,u} q(u, z^{(u)}) \cdot q(z^{(x)}, w \oplus \tilde{w}) q(z^{(u)}, \tilde{w} \oplus w)\,. \tag{10.14}$$

The approximation is the update for the C3A structure, which is expected to work well for counts $n_{z,w} \gg 1$, as it basically reduces the joint count by 1 if $z^{(x)}{=}z^{(u)}$. Interpreting (10.14), the counts of both the author and the tag branch are added in the node $(\vec{\varphi}_z \,|\, \beta)$. Therefore, the influences of both involved branches are again superimposed.

Expanding the C3A approximation (10.14) with sequence indices, we obtain the following full conditional:

$$p(x_{m,n}{=}x, u_{m,n}{=}u, z^{(x)}_{m,n}{=}k, z^{(u)}_{m,n}{=}l \mid \vec{d}, \vec{c}, \vec{w}, \vec{u}_{\neg m,n}, \vec{x}_{\neg m,n}, \vec{z}^{(x)}_{\neg m,n}, \vec{z}^{(u)}_{\neg m,n}, A) \propto$$
$$a_{m,x} \cdot \frac{n_{x,k,\neg m,n} + \alpha}{n_{x,\neg m,n} + K\alpha} \cdot \frac{n^{(x)}_{k,w,\neg m,n} + n^{(u)}_{k,w} + \beta}{n^{(x)}_{k,\neg m,n} + n^{(u)}_{k} + V\beta}$$
$$\cdot\, c_{m,u} \cdot \frac{n_{u,l,\neg m,n} + \gamma}{n_{u,\neg m,n} + C\gamma} \cdot \frac{n^{(x)}_{l,w} + n^{(u)}_{l,w,\neg m,n} + \beta}{n^{(x)}_{l} + n^{(u)}_{l,\neg m,n} + V\beta}. \tag{10.15}$$

Note that only the sums of partial counts, e.g., $n^{(x)}_{k,w} + n^{(u)}_{l,w}$, need to be stored for the actual algorithm, representing the collapsed state of the central node $(\varphi \mid \beta)$. Also, the C3A approximation allows splitting of the sampler into one for $\{x, z^{(x)}\}$ and the other for $\{u, z^{(u)}\}$.

Retrieval functions. For word queries, the ETT3 model has a structure analogous to ETT1, thus (10.6) can be reused. ETT3 differs, however, in the case of tag queries. Because the tag output was replaced by an N2 node parallel to the N2 author node, tags do only indirectly depend on the authors (via the global parameters φ trained by both authors and tags). One approach to obtain the likelihood of a tag query is by reusing (10.6) with parameters $\psi_{z,c} = p(c|z)$ synthesised from the known parameters $\zeta_{c,z}$ via Bayes' rule:

$$p(\vec{c}\,' \mid a) = \textstyle\prod_{c \in \vec{c}\,'} \sum_z \vartheta_{a,z} p(c|z) = \sum_z \vartheta_{a,z} \dfrac{\zeta_{c,z} p(c)}{\sum_c \zeta_{c,k} p(c)} \tag{10.16}$$

where $p(c)$ is set to the relative strength of tag c after learning the model, which is proportional to n_u for $u = c$. Combined queries use the product of word and tag likelihoods, analogous to ETT1, and again the scoring method of (10.7) may be applied.

10.3 Related work

To illustrate the ideas of the NoMM design approach, we have made an exception from explicitly looking at related models initially. In general, it should be expected from a design method to be able to construct models that are similar to the state of the art in a field that has been active long enough to cover a large portion of the "space" of possible models below a certain structural complexity (or degree of application specialisation). With relatively simple structures as the ones designed here, this is indeed the case.

Aside from general approaches to expertise finding using statistical methods (see, e.g., [Balog et al. 2007] and other models reviewed in Chapter 2), there exist several models that explicitly make use of topics for this task. Most expert-specific topic models are derived from the seminal author–topic model (ATM) [Rosen-Zvi et al. 2004] discussed in Chapter 4. The ETT models are no exception because the N2 structure used to include author distributions as observed parameters into the model has first been proposed in that work. For ATM, in [Rosen-Zvi et al. 2010] a retrieval function is used that is similar to (10.6) (left).

Regarding authorship-based models with additional structure, the work of [Mimno & McCallum 2007] assumed several "personas" or directions of interest for each author by adding a mixture level to ATM as a parent of the author–topic node, and the variant in [Kawamae 2010] goes the opposite way and creates "interests" by adding a level of document classes that each author is associated with.

Focussing on models that explicitly use additional information like tags or class labels to improve systems for expertise search and similar tasks, there are several related approaches. [Tang et al. 2008] have worked on various models that use E3 coupled branches and N5B+E4 regression structures to incorporate label information. Indeed, the model that performed best in retrieval tasks, the E3-based "author–conference–topic" model 1 (ACT1), closely resembles the design of the ETT2 variant. However, it is conceptualised only for a single label per document (the conference an author publishes at) and therefore not directly applicable to the present scenario. Despite having been assembled with a different method, structurally the ETT2 model is an extension of the ACT1 model towards addressing the multi-label annotation problem using filtered parameters. The work in [Tang et al. 2008] is explicitly aimed at expert retrieval, and they use a similar approach to that of [Wei & Croft 2006], combining a language model and their topic models.

Other work adds label or tag information to topic models but does not consider authorship. Naturally, models with the E4+N5 aggregation structure (see Chapters 5 and 9) are relevant: the supervised topic model (sLDA) [Blei & McAuliffe 2007], which uses regression on topic–label association (N5B structure), and the concept–topic model (CT model) [Bundschus et al. 2009], which is based on a selection approach to estimating the labels for a given document (N5A). The regression variant has been investigated in connection with expertise search by [Tang et al. 2008] as the ACT3 model where it yielded inferior retrieval results.[4] Furthermore, the MM-LDA model in [Ramage et al. 2009] uses an E2 structure to create topics from tagged documents, which corresponds to the approach of the ETT1 model. The work shows that the E2-based model yields better F-score results than an LDA model with the tags used as words, which shows that using a separate distinct topic node ($(\psi \,|\, \gamma)$ in ETT1) for the tags may have an advantage over a shared $(\varphi \,|\, \beta)$. Regarding this model, ETT1 is an extension towards tag boosting and author-specific topics.

Similarly, using authorship instead of documents, [Kataria et al. 2011] have worked on an E2-based model to incorporate citations in lieu of tag information to capture author influence. This work is part of a larger body of topic models that explicitly model the semantic relationship between linked and linking documents to determine influence (e.g., [Dietz et al. 2007, Guo et al. 2010, Nallapati et al. 2008; 2011]) and, more generally, relational topic models (e.g., [Chang & Blei 2009, Sinkkonen et al. 2008, Xu et al. 2006]). In the context of ETT models, determining topic-specific influence or predicting links from semantics may be specifically interesting to enhance retrieval functions like (10.6) by influence strengths.

[4] An informal comparison between sLDA [Blei & McAuliffe 2007] and the CT model [Bundschus et al. 2009] resulted in better performance of the former considering likelihood and truncated precision values, but at the expense of algorithm complexity. However, after deeper analysis of the inference in the CT model (cf. [Bundschus 2010]) we identified potential for improvement, which is sketched in Appendix C.3 but left to future exploration.

10.4 Experimental study

The following study investigates the ETT models, based on the data of a representative scientific virtual community that contains the modalities necessary for the model scenario.

Data. The experimental data are derived from the proceedings of the NIPS conference ("Neural Information Processing Systems") between years 1988 and 2000. NIPS is a multi-disciplinary gathering of experts at the crossroads of neurobiology, cognitive science, artifical intelligence, machine learning, computational linguistics and related fields, and its proceedings offer full-text access to the research results (8 pages each).

Used as an evaluation corpus, the NIPS proceedings data contain $M = 1740$ documents, $V = 9570$ unique terms, $W = 1340639$ words, $A = 2037$ authors with $W_A = 3990$ authorship relations. In addition, the current version of the corpus has been manually annotated with 2520 tags with a vocabulary of $C = 165$. The tags do not represent a fully controlled vocabulary or subject hierarchy but have been cleaned from instances with low document frequencies and some semantically similar tags have been merged. However, semantic overlap of tagging data is common, e.g., a document about speech recognition may be tagged as *audio*, *speech* and/or *speech recognition*.

Furthermore, the data have been preprocessed with stop-word lists and terms have been filtered afterwards according to their document frequency: The corpus ensures the constraint on the document frequency df, $\{t : \mathrm{df}(t) \in [10, 500]\}$, which removes a great portion of stopword-like tokens and character recognition errors, which have degraded topic quality.[5] For more details, please refer to Appendix D.2.

Model implementation. Based on the structures in Fig. 10.4, the models have been implemented using NoMM scripts, from which Java source code has been generated by the Gibbs meta-sampler. In effect, this corresponds to the Gibbs full conditionals in (10.5), (10.11) and (10.15) with parallel acceleration using the full-state exact synchronisation strategy (FSE; see Chapter 8). In the generated source code, some adjustments have been manually made to accommodate specific model details like parameter filtering (see Section 10.2.2) and to add retrieval functions.

Connected to this, the first result of the study is the required implementation effort: Compared to manually writing the source code from scratch, only a fraction of the time had to be dedicated to development. Exact comparative figures are difficult to give, but the implementation effort for all three models was below 2 person days for a domain and Java expert, with some additional debugging and verification effort for source code added to the generated models. The final complexity of the models in terms of lines of commented code (LoCC; as defined in Section 6.7) is given in Fig. 10.5.[6]

Model training. All ETT models have been trained for different values of K, using burn-in periods that are measured according to the word likelihood of held-out data, as described in Chapter 6. For this, the corpus is partitioned into subsets of 90% training and 10% test documents. The particular setup of likelihood measurement follows the document completion experiment

[5] The NIPS dataset used is therefore different from the version used in [Heinrich 2011b], which makes results not fully comparable.

[6] For ETT3, the implementation uses the full C3B case; see (10.13). Compared to Fig. 6.6, in Fig. 10.4 different perplexity functions are generated (document completion, see below), leading to different LoCC counts.

Model	Structures	Reference	Generated	Modified	Manual modifications
LDA	—	Fig. 4.5(a)	435	117	querying and thesaurus expansion (10.17)
ATM	N2	Fig. 4.5(b)	572	85	retrieval functions
ETT1	E2	Fig. 10.4(a)	658	150+74	retrieval functions, tag boosting + thesaurus (10.18), (10.19)
ETT2	E3	Fig. 10.4(b)	653	134	retrieval functions, parameter filtering
ETT3	C3B	Fig. 10.4(c)	716	140	retrieval functions

Figure 10.5: Results of the code generator and manual modifications, in lines of commented code (LoCC).

first proposed for ATM [Rosen-Zvi et al. 2004]: The test documents are split into equally sized random subsets, and the first one is added to the training data while the latter is held out. The two baseline models, LDA and ATM, are handled analogously.

The Gibbs sampler Markov chains require between 300 and 400 iterations to reach a stationary state for all models. After this burn-in period, the held-out data can be added back to the training corpus, and the sampler is run with the test documents extended to their full length. This second burn-in period folds in the held-out tokens into the model, adjusting its parameters. After a few iterations, the Gibbs sampler again converges, now judging from the likelihood of the complete data set.

Study outline. Based on these trained models, three major model aspects are analysed: Retrieval performance in Section 10.4.1, model likelihood and clustering behaviour in Section 10.4.2 and finally the quality of the topics themselves in Section 10.4.3.

10.4.1 Retrieval

Retrieval of relevant experts was the main motivation in our scenario, and this first set of experiments validates performance against the author–topic model as a baseline.

Topic-based retrieval. In general, purely topic-based methods on their own cannot be expected to produce competitive results for ad hoc retrieval as implemented in "classical" search engines: Typical text queries are too short to disambiguate the senses of the query terms. In particular, [Wei & Croft 2006] have analysed how LDA may be used for retrieval, finding that the topic-based results lead to inferior retrieval performance. They proposed a linear combination of a language model that computes word probabilities based on literal term matching and the topic-based word probabilities, which performs best for short queries with only a 30% weighting for the topic model: The topic model is used to improve recall where vocabulary differences exist (cf. Chapter 3).

Because we are interested in the effects of the ETT models themselves, we will focus on queries that are expressive enough to allow retrieval solely via topics. One approach to improve precision is to expand queries with words semantically related to the information need [Yi & Allen 2009].

Query creation. Queries are chosen to widely cover the thematic fields in the corpus. In order to synthesise appropriate queries, supervised query expansion is applied. To find appropriate expansion terms, a "neutral" reference topic model is used that captures some of the semantic similarities inherent in the corpus.

query: "**binaural localization**"	
p = 0.01732: **auditory**	p = 0.00359: **frequencies**
p = 0.01354: **frequency**	p = 0.00354: **phase**
p = 0.01105: **sound**	p = 0.00349: **amplitude**
p = 0.00534: **localization**	p = 0.00344: channels
p = 0.00492: signals	p = 0.00339: cochlea
p = 0.00446: cochlear	p = 0.00314: speech
p = 0.00434: **temporal**	p = 0.00305: **band**
p = 0.00432: **filter**	p = 0.00293: channel
p = 0.00422: **spectral**	p = 0.00291: source
p = 0.00383: sounds	p = 0.00285: owl

(a) Example term distribution $p(t \mid \vec{w}', \cdot)$

query	expansion
associative memory	hopfield network neuron memories synaptic activity address
bayesian variational inference	prior posterior probabilistic likelihood graphical model
stock options	financial trading risk market prediction price future strategy
phoneme speech recognition	speaker acoustic vowel utterances spectral hmm markov word
svm support vector machine	kernel classifier hyperplane regression

(b) Example query expansions

Figure 10.6: Term query expansion: Candidates selection and query examples.

Starting from a raw term query $\vec{w}'$, a plain LDA model with $K = 100$ is computed from the corpus and queried to find the parameters $\vec{\vartheta}'$ for $\vec{w}'$ using sampling, as described in Sections 3.7.2 and 6.5.1. From this, an empirical word distribution (or local language model) is computed:[7]

$$p(t \mid \vec{w}', \cdot) \propto \sum_k \vartheta'_k \varphi_{k,t} n_k \,. \tag{10.17}$$

Ranking the terms t in this distribution leads to candidates of expansion terms, which are manually selected to match the semantic aspects intended for the query. In Fig. 10.6(a), an example empirical distribution used for expansion is shown, with the selected terms in boldface. This selection requires some basic domain knowledge and some decisions on the desired knowledge. For the example raw query "binaural localization", we require that the expert have worked on localisation models based on spectral and phase properties of the ear signals where different spectral bands are an aspect whereas we do not want to focus on models of the cochlea or the localisation system in the owl. In this manner, 30 queries of sizes between 5 and 12 significant vocabulary terms are constructed, some of which are displayed in Fig. 10.6(b). To create query documents $\vec{w}'$, the terms of the raw query are boosted by factor 2.[8]

In addition, a set of 20 queries consisting of 1–3 tags has been constructed, selecting tags and their combination as meaningful information needs, for instance combining tags *blind source separation* (BSS) with *independent component analysis* (ICA), which targets at experts who apply ICA to achieve BSS.

Relevance judgements. In order to measure retrieval quality, we require judgements of the experts returned by the system as ground-truth. To avoid infeasible iterations of the complete corpus for each query, a pooling technique is used: In pooling, sufficiently large results sets from several reference retrieval systems are combined for a query [Spärck Jones & van Rijsbergen 1975]. Relevant items are identified only in this results pool, based on the assumption that the pooled sample represents 100% recall.

[7] This is the likelihood of term t under the query and model parameters, with an additional topic weight n_k; cf. [Yi & Allen 2009] for alternative approaches.

[8] Intuitively, word bigrams such as "associative memory" or "spectral band" seemed to play an important role in term selection and may improve topic modelling. This viewpoint agrees with results in [Heinrich et al. 2005b] and more recent studies [Wallach 2008]. The present study excludes these cues, however, to stay focussed.

In the present case, different variants of the raw queries are combined for pooling using a single reference retrieval system, up to a depth of 30 author results per issued query, and the variations of queries re-enact the improved recall of combined reference systems. The reference system employs a classical tf-idf retrieval function as implemented in the Apache Lucene search library [McCandless et al. 2010]. It is customised to display results for document authors, agglomerating scores across several query expansions.

To be considered a relevant expert in our study, an author needs to cover at least one-third of the authorship in a relevant paper or have contributed to more than one relevant paper on the pooled set of query variants. The query strategy for pooling is similar to that of researching the state of the art in a scientific area described by the query.

Tag query judgements are more difficult to obtain because there is no completeness assumption for tags: Untagged documents may also be relevant to a tag. Because this makes search for tags unreliable in the reference system, we use a "direct pooling" method: For the 20 tag queries, all retrieved experts are pooled and judged by the criterion whether at least one of their articles is labelled or can be labelled with at least one of the query tags and all tags appear at least once (AND condition on author level). Judgements are done on a random author list pooled from all models, making ratings unbiased between models.

Generally, the relevance judgements are not considered of a quality comparable to for instance the TREC test collections.[9] Yet they are considered sufficient for the small comparative study intended. Moreover, other data with author and tag modalities are difficult to obtain elsewhere in combination with relevance judgements.

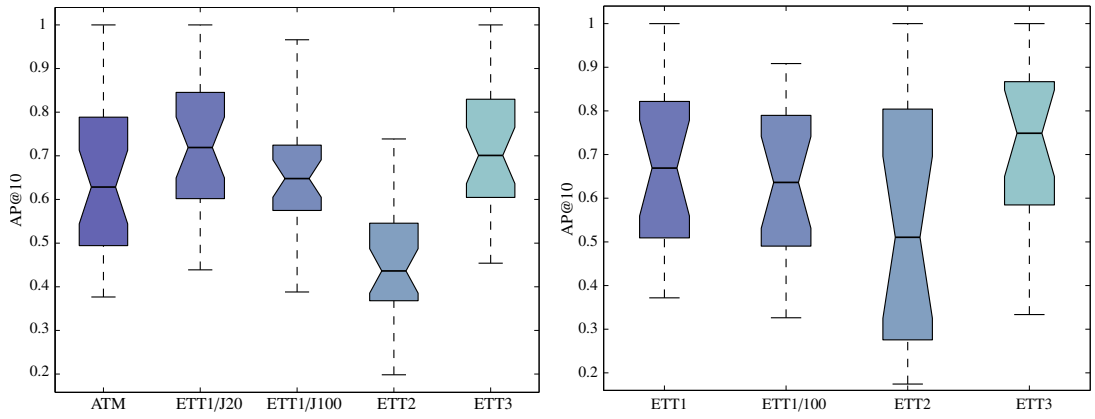
Experiments. Querying has been performed on the three ETT models as well as for word queries with ATM as a baseline, using the full corpus and $K = 100$ for all trained models, which in preliminary trials turned out to be a good value. For ETT1, we evaluate two different settings for tag boosting, using $J_m = N_m/\{20, 100\}$ with larger and smaller influence of tags on the E2 structure. As a metric, we use the average precision at cutoff point $k = 10$ (AP@10), as defined in Section 3.7. Ten items correspond to the first page that a typical search engine returns.

For truncated retrieval measures, it is known that performance may have high variance. The compromise between effort and quality is to use a query count large enough to be able to detect the tendencies of model effectiveness while at the same time accepting that results may be not statistically significant. With 30 term and 20 tag queries, this compromise is approached.

Term query results. Results for AP@10 scores are presented in Fig. 10.7(a), using box plots to facilitate visual comparison via basic statistics at defined significance levels [Frigge et al. 1989]. As expected, there is a large performance variability across queries. Under these conditions, the best model in the study in terms of median AP performance is ETT1/J20, closely followed by ETT3. Both slightly outperform ATM by around 0.08 and 0.06, respectively. With the large confidence intervals (the notches in the box plots), these differences are not statistically significant, though, according to Wilcoxon's rank-sum test ($p = 0.05$).[10] ETT1/J100 performs roughly on par with ATM, however not achieving the top AP score of 1.0, that is, having all retrieved experts judged relevant in the reference pool.

[9] `http://trec.nist.gov`

[10] Because of the small sample sizes, we consider this non-parametric test rather than assuming normal distributions and applying a t-test, as for instance [Park 2010].



(a) Term queries (expanded) (b) Tag queries

Figure 10.7: ETT retrieval results using author weight $\lambda = N'_m/2$ and $K = 100$. Boxes indicate lower and upper quartiles around the median (data ranks $[N/4, N/2]$ and $[N/2, 3N/4]$), the whisker ends correspond to sample minimum and maximum. Notches indicate confidence intervals for the medians according to Wilcoxon's rank-sum test with significance level $p = 0.05$.

Based on these results, one may conclude that the additional information from tags slightly improves the topic structure of ETT1 and ETT3 with respect to the word retrieval task. ETT2, on the other hand, basically fails in validation, as its results range is around 0.2 below the baseline ATM, which is statistically significant even for the small number of queries analysed. It is interesting to see that a model similar to ETT2 has performed so well on data tagged with conference venues in [Tang et al. 2008], but one reason may be the extension to multi-labelling used here. In particular, it appears as if popular topics that are tagged more frequently are not prone to retrieval errors while less frequent ones without specific tags tend to lower quality.

In Fig. 10.8, two example query results are presented. These are based on the ETT1 model but ATM and ETT3 provide similar results, on average, however, with some rankings inferior according to the judgements. Query (a) identifies the experts on support vector machine (SVM) theory in the NIPS community, and the expansion terms are related to typical common methods used with SVMs as well as applications. The top results 1–5 are highly regarded experts in the field, and Scholkopf_B, Smola_A and Vapnik_V are indeed among the most productive contributors to SVM theory, including its usage in classification and regression tasks. The experts judged irrelevant still have good relevance to the query: Steinhage_V works on kernel functions, which is closely related to the approach SVMs use, and Herbrich_R works on classification, which was used as an expansion term. Both irrelevant experts are ranked relatively high also in the baseline model ATM but are not included in the 13 experts judged relevant for the query in the pool.

For query (b), finding experts that use the EM algorithm for ML parameter estimation, the situation is similar: The first four results, Smyth_P, Ghahramani_Z, Jordan_M and Bishop_C, are well-known contributors of methods based on the EM algorithm, which is also reflected in their NIPS articles. But also the other results are plausible. Notably, the false positives Rigoll_G and Vasconcelos_N do mention the EM algorithm in their work but have not been among the 25 experts in the pool.

query: svm support vector machine | kernel classifier hyperplane regression

1. Scholkopf_B, lik = −76.272, tokens = 2830, docs = 10: judged relevant
✓ From Regularization Operators to *Support Vector Kernels* (9); Improving the Accuracy and Speed of *Support Vector Machines* (9); Shrinking the Tube: A New *Support Vector Regression* Algorithm (11) . . .

2. Smola_A, lik = −77.509, tokens = 2760, docs = 11: judged relevant
✓ *Support Vector Regression* Machines (9); Prior Knowledge in *Support Vector Kernels* (10); *Support Vector* Method for Novelty Detection (12) The Entropy Regularization Information Criterion (12, *support vector machines, regularization*) . . .

3. Vapnik_V, lik = −77.525, tokens = 2332, docs = 10: judged relevant
✓ *Support Vector Regression* Machines (9); Prior Knowledge in *Support Vector Kernels* (10); Prior Knowledge in *Support Vector Kernels* (10); *Support Vector* Method for Multivariate Density Estimation (12); . . .

4. Crisp_D, lik = −81.401, tokens = 699, docs = 2: judged relevant
✓ A Geometric Interpretation of t/-*SVM Classifiers* (12); Uniqueness of the *SVM* Solution (12)

5. Burges_C, lik = −81.630, tokens = 1309, docs = 5: judged relevant
✓ Improving the Accuracy and Speed of *Support Vector Machines* (9); A Geometric Interpretation of t/-*SVM Classifiers* (12); Uniqueness of the *SVM* Solution (12) . . .

6. Laskov_P, lik = −84.275, tokens = 738, docs = 1: judged relevant
✓ An Improved Decomposition Algorithm for *Regression Support Vector Machines* (12)

7. Steinhage_V, lik = −84.600, tokens = 438, docs = 1: judged irrelevant
× Nonlinear Discriminant Analysis Using *Kernel* Functions (12)

8. Bennett_K, lik = −86.754, tokens = 384, docs = 1: judged relevant
✓ Semi-Supervised *Support Vector Machines* (11)

9. Herbrich_R, lik = −86.754, tokens = 462, docs = 2: judged irrelevant
× Classification on Pairwise Proximity Data (11); Bayesian Transduction (12, *classification*)

10. Chapelle_O, lik = −87.431, tokens = 494, docs = 2: judged relevant
✓ Model Selection for *Support Vector Machines* (12); Transductive Inference for Estimating Values of Functions (12, *regression, classification*)

(a) AP@10 = 0.768

query: em algorithm | ml likelihood map inference probability estimation

1. Smyth_P, lik = −50.389, tokens = 3153, docs = 7: judged relevant
✓ Stacked Density *Estimation* (10, *probabilistic*); *Probabil.* Anomaly Detection in Dyn. Systems (6, *likelihood, estimation*, classifier); Clustering Seq.s with Hidden Markov Models (9, *probabilistic*, *estimation*) . . .

2. Ghahramani_Z, lik = −57.593, tokens = 4265, docs = 12: judged rel.
✓ Supervised Learning from Incomplete Data via an *EM* Approach (6); Factorial Learning and the *EM* Algorithm (7); Learning Nonlinear Dynamical Systems Using an *EM* Algorithm (11) . . .

3. Jordan_M, lik = −58.281, tokens = 8013, docs = 29: judged relevant
✓ Superv. Learning from Incomplete Data via an *EM* Approach (6); Recursive Algo.s for Approx. Probabilities in Graphical Models (9); Approx. Posterior Distributions in Belief Networks Using Mixtures (10) . . .

4. Bishop_C, lik = −58.849, tokens = 2658, docs = 10: judged relevant
✓ *Estimating* Conditional Probability Densities for Periodic Variables (7); EM Optimization of Latent-Variable Density Models (8); Approximating Posterior Distributions in Belief Networks Using Mixtures (10) . . .

5. Jebara_T, lik = −59.662, tokens = 985, docs = 3: judged relevant
✓ Maximum Conditional *Likelihood* via Bound Maximization and the C*EM* Algorithm (11); Maximum Entropy Discrimination (12). . .

6. Tresp_V, lik = −59.807, tokens = 5013, docs = 15: judged relevant
✓ Discov. Structure in Contin. Var.s Using Bayes. Networks (8); Efficient Methods for Dealing with Missing Data in Superv. Learning (7) . . .

7. Rigoll_G, lik = −60.409, tokens = 1225, docs = 3: judged irrelevant
× A New Approach to Hybrid HMM/ANN Speech Recognition using Mutual Information Neural Networks (9, *EM* algorithm in HMM) . . .

8. Vasconcelos_N, lik = −61.291, tokens = 1261, docs = 3: judged irrelev.
× Learning Mixture Hierarchies (11, *EM* algorithm, *estimation*) . . .

9. Singer_Y, lik = −61.458, tokens = 4074, docs = 9: judged relevant
✓ Training Algorithms for Hidden Markov Models using Entropy Based Distance Functions (9, *EM* in HMM); Batch and On-Line Parameter Estimation of Gaussian Mixtures Based on the Joint Entropy (11, uses *EM*)

10. Bengio_Y, lik = −61.501, tokens = 5365, docs = 16: judged relevant
✓ Neural Network Gaussian Mixture Hybrid for Speech Recognition or Density Estimation (4, *EM* in HMM) . . .

(b) AP@10 = 0.866

Figure 10.8: Example term retrieval results: ETT1/J20. Authors shown with selected articles. Italicised terms are deemed relevant. In parentheses: NIPS volume numbers and relevant terms.

Tag query results. While ETT models allow term-based retrieval partly superior to the ATM baseline, tag-based retrieval is their actual benefit. The results for tag queries are presented in Fig. 10.8(b), comparing the performance of ETT models. In general, the models retrieve relevant experts for tags but precision values scatter widely, which manifests itself in the large boxes in the plot. Notably, ETT2 that resulted in low term retrieval performance outperformed the other models in some queries; on average it stays inferior, nevertheless.

Furthermore, ETT3 outperforms ETT1 in this test slightly. Considering the additional "inversion" necessary to obtain $p(c \mid z)$ in the ETT3 tag retrieval function (10.16) (compared to the other ETT models), this is surprising. One conjecture to explain this is that the ETT3 model effectively adds one degree of freedom by allowing tag responsibilities to be learnt. If for instance a tag is incorrectly assigned to a document, its responsibility is reduced in this context (variable $u_{m,n}$ in (10.15)). Opposed to this, ETT1 keeps the responsibility constant, potentially assigning less optimal topics to the tag (variable $y_{m,j}$ in (10.5)) instead of leaving these assignments to better matching tags in the document context.

In Fig. 10.9, two example queries are shown that give a representative picture for some other observations. Query (a) consists of a combination of two tags, and the information need is to have experts that are related to both *blind source separation* and *independent component analysis*. Results 1–7 and 9 fulfill the sufficient relevance criterion that their authored documents cover all

query: *blind source separation (BSS) + independent component analysis (ICA)*

1. Yang_H, lik = −9.748, tokens = 1604, docs = 6: judged relevant
✓ A New Learning Algorithm for Blind Signal Separation (8, tags: *BSS*); Search for Information Bearing Components in Speech (12, tags: *audio; information theory; speech*) . . .

2. Cichocki_A, lik = −9.966, tokens = 796, docs = 3: judged relevant
✓ Blind Separation of Filtered Sources Using State-Space Approach (11, tags: *BSS; state space methods*); Semiparametric Approach to Multichannel Blind Deconvolution of Nonminimum Phase Systems (12, tags: *deconvolution, semi-parametric methods, signal processing*) . . .

3. Lee_T, lik = −10.054, tokens = 1275, docs = 4: judged relevant
✓ Unsupervised Classification with Non-Gaussian Mixture Models Using *ICA* (11, tags: *classification; ICA; mixture models*); *Blind Separation* of Delayed and Convolved Sources (9, tags: *BSS; deconvolution*) . . .

4. Bell_A, lik = −10.421, tokens = 2822, docs =6: judged relevant
✓ Edges are the "Independent Components" of Natural Scenes (9, tags: *ICA*); A Non-Linear Information Maximisation Algorithm that Performs Blind Separation (8, tags: *BSS, information theory*) . . .

5. Hyvarinen_A, lik = −10.650, tokens = 1392, docs = 3: judged relevant
✓ One-unit Learning Rules for ICA (9, text: *BSS*, tags: *neural networks; ICA*); New Approximations of Differential Entropy for ICA and Projection Pursuit (10, tags: *ICA; information theory; max entropy*) . . .

6. Parra_L, lik = −10.653, tokens = 1639, docs = 4: judged relevant
✓ Maximum Likelihood Blind Source Separation: A Context-Sensitive Generalization of ICA (9, tags: *BSS, ICA, max. likelihood*) . . .

7. Oja_E, lik = −10.722, tokens = 1025, docs = 4: judged relevant
✓ One-unit Learning Rules for ICA (9, text: *BSS*, tags: *neural networks; ICA*); ICA for Identification of Artifacts in Magnetoencephalographic Recordings (10, tags: *ICA, EEG*) . . .

8. Rokni_U, lik = −11.097, tokens = 351, docs = 1: judged irrelevant
× Algo.s for *ICA* and Higher Order Statistics (12, tags: *higher-order stats.*)

9. Shouval_H, lik = −11.267, tokens = 219, docs = 1: judged irrelevant
× Receptive Field Formation in Nat. Scene Env.s: Comparison of Single Cell Learning Rules (10, text: *ICA*, kurtosis, tags: *neural networks*)

10. Lin_J, lik = −11.315, tokens = 979, docs = 2: judged relevant
✓ Source Sep. and Density Estimation by Faithful Equivariant SOM (9, text: separation, independent components, tags: *BSS, density est.*) . . .

(a) AP@10 = 0.780

query: *face recognition*

1. Movellan_J, lik = −4.680, tokens = 3153, docs = 8: judged relevant
✓ Dyn. Features for Visual Speechreading: A System Comparison (9, no tags); Image Representation for Facial Expression Coding (12, tags: *face recognition, image, ICA*); Visual Speech Recognition with Stochastic Networks (7, tags: *HMM, speech recognition*) . . .

2. Bartlett_M, lik = −4.951, tokens = 812, docs = 3: judged relevant
✓ Viewpoint Invariant *Face Recognition* using ICA and Attractor Networks (9, tags: *face recognition, invariances, pattern recognition*); Image Representation for Facial Expression Coding (12, tags: *face recognition, image, ICA*) . . .

3. Dailey_M, lik = −4.952, tokens = 903, docs = 2: judged relevant
✓ Task and Spatial Frequency Effects on Face Specialization (10, tags: *face recognition*); Facial Memory Is Kernel Density Estimation (Almost) (11, no tags)

4. Padgett_C, lik = −4.974, tokens = 499, docs = 1: judged relevant
✓ Representing Face Images for Emotion Classification (9, tags: *classification, face recognition, image*)

5. Hager_J, lik = −5.023, tokens = 377, docs = 2: judged relevant
✓ Classifying Facial Action (8, tags: *classification*); Image Representation for Facial Expression Coding (12, tags: *face recognition, image, ICA*)

6. Ekman_P, lik = −5.027, tokens = 374, docs = 2: judged relevant
✓ Image Representation for Facial Expression Coding (12, tags: *face recognition, image, ICA*); Classifying Facial Action (8, tags: *classification*)

7. Phillips_P, lik = −5.127, tokens = 795, docs = 1: judged relevant
✓ Support Vector Machines Applied to Face Recognition (11, tags: *face recognition, SVM*)

8. Gray_M, lik = −5.159, tokens = 470, docs = 2: judged irrelevant
× Dynamic Features for Visual Speechreading: A Systematic Comparison (9, text: dynamic visual features; no tags)

9. Lawrence_D, lik = −5.217, tokens = 265, docs = 1: judged relevant
✓ SEXNET: A Neural Network Identifies Sex From Human Faces (3, tags: *neural networks, object recognition, pattern recognition*)

10. Ahuja_N, lik = −5.221, tokens = 366, docs = 2: judged relevant
✓ A SNoW-Based Face Detector (12, tags: *face recognition, image, vision*)

(b) AP@10 = 0.879

Figure 10.9: ETT1 tag retrieval results. Italicised terms correspond to tags.

query tags (see above), and especially Cichocki_A, Hyvarinen_A, Oja_E and Parra_L are experts in the NIPS community with several relevant papers.

Query (b) consists of a single tag, *face recognition*, and all results except rank 8 are relevant. Beyond clear experts with much work in the area like Movellan_J and Bartlett_M, here again the matches ranked lower are interesting. For instance, Padgett_C has only a single paper, which matches well with the query despite the low token count. However, the co-author Cottrell_G that has 9 documents with a token count of 3113 in ETT1 appears only at rank 12, despite 4 of his documents being relevant to *face recognition*. Here the other documents like "Non-Linear Dimensionality Reduction" (tagged *dimensionality reduction*) seem to interfere with the topic profile. Similar behaviour is found for Hager_J: One relevant paper is co-authored with a strong expert Sejnowski_T that does not appear among the first 20 results. However, among the 43 documents with 12560 tokens, only a single one is tagged *face recognition*. Even with a high λ weight in (10.7), the model seems to prefer pure matching, which may be due to the modelling assumption (b) that an expert has only a single field of interest.

An interesting property of the ETT models is their potential to retrieve experts with untagged documents for a given query tag, which corresponds to a semi-supervised setting, i.e., learning the model from incompletely labelled data. An example for this is result 8 in query (a): Rokni_U is

tag: *face recognition* (ETT1/J20)
0.82702 face images faces image facial visual human video database detection
0.09392 image images texture pixel resolution pyramid regions pixels region search
0.02696 wavelet video view images tracking user camera image motion shape
0.00117 eeg brain ica artifacts subjects activity subject erp signals scalp
0.00100 image images visual vision optical pixel surface edge disparity receptive
0.00094 orientation cortical dominance ocular cortex development lateral eye cells visual
0.00089 chip neuron synapse digital pulse analog synaptic chips synapses murray
0.00084 hinton object image energy cost images code visible zemel codes

(a) query tag

author: Movellan_J (ETT1/J20)
0.53816: face images faces image facial visual human video database detection
0.16216: image images texture pixel resolution pyramid regions pixels region search
0.08954: speech speaker acoustic vowel phonetic phoneme utterances spoken formant
0.06216: bayesian prior density posterior entropy evidence likelihood distributions
0.03939: filter frequency signals phase channel amplitude frequencies temporal spectrum
0.03508: activation boltzmann annealing temperature neuron stochastic schedule machine
0.02770: cell firing cells neuron activity excitatory inhibitory synaptic potential membrane
0.02154: convergence stochastic descent optimization batch density global update

(b) Movellan_J

author: Bartlett_M (ETT1/J20)
0.77432: face images faces image facial visual human cottrell database detection
0.11350: ica source separation sources blind mixing signals amari entropy bell
0.06111: tangent transformation image simard images invariant invariance euclidean
0.03048: eeg brain ica artifacts subjects activity subject erp signals scalp
0.01759: image images texture pixel resolution pyramid regions pixels region search
0.01145: image images visual vision optical pixel surface edge disparity receptive
0.00041: rbf classifier classifiers radial decision centers lippmann mlp regions knn
0.00008: impulse convolution similarity dot volume product recurrent quantization capacity

(c) Bartlett_M

author: Cottrell_G (ETT1/J20)
0.41865: recurrent nets correlation cascade activation connection epochs representations
0.27523: face images faces image facial visual human video database detection
0.17531: subjects human stimulus cue subject trials experiment perceptual psychophysical
0.11287: tangent transformation image simard images invariant invariance euclidean
0.07130: modules attractors cortex phase olfactory frequency bulb activity oscillatory eeg
0.06143: word connectionist representations words activation production cognitive musical
0.03695: node activation graph cycle nets message recurrence links connection child
0.02049: visual attention contour search selective orientation iiii region saliency segment

(d) Cottrell_G

Figure 10.10: Topics for ETT1 tag query *face recognition* and expert topic examples.

not a relevant expert for performing blind source separation but rather worked on a method of ICA. The author has been retrieved despite his low token count (that influences the score according to (10.7)) and despite his document has been tagged with neither of the query tags, rather with the seemingly unrelated tag *higher-order statistics*. Two analogous cases exist with results 8 and 9 in query (b): In the *face recognition* result, expert Gray_M is not directly relevant but intuitively, visual speech reading seems to share many concepts and terminology with plain face recognition. (Notably, the document may also have boosted the rank of the top expert Movellan_J who is a co-author.) Furthermore, Lawrence_D appears in the results list for one document that is considered relevant despite not being tagged. Here, the existing tags *object recognition* and *pattern recognition* may play a role, which intuitively generalise *face recognition*.

Another observation is that the performance for tags in ETT models apparently depends on their document frequency (the number of documents tagged with them). For instance, the query *back-propagation + neural networks* results in documents that appear to be relevant to *neural networks* but less so to the other tag. The *neural networks* tag is the most frequent tag. For a query assuming both tags to be relevant, this imbalance leads to lower retrieval precision despite the intersection (product) used in the retrieval functions (10.6). Especially ETT2 seems to be sensitive to this, as it gives good results for frequent tags while failing for weaker ones.

For reference, Fig. 10.10 presents topics for the *face recognition* tag query and selected topic distributions for experts. The match between the main topics of Movellan_J and Bartlett_M in Fig. 10.9(b) is obvious, while the topics of the expert Cottrell_G include the main face recognition topic at only 22%, with little additional overlap. Here the breadth of expertise seems to have degraded the relevance match.

Tag topics. The general ability of the models to retrieve experts for tags is also seen in the topics associated with tags, as shown in Fig. 10.11 (apart from the tag query in Fig. 10.10). In subfigures (a)–(d), the topics for the tag *vision* are presented, with one line per topic along with

tag: *vision* (ETT1/J20)
0.57457 image images visual vision optical pixel surface edge disparity receptive
0.13372 orientation cortical dominance ocular cortex development lateral eye cells visual
0.13022 object objects visual attention image grouping parts frame search saliency
0.09578 retina light intensity cells photoreceptor gain contrast background bipolar silicon
0.02681 eye velocity vor head movements saccadic gain motor movement visual
0.01938 wavelet video view images tracking user camera image motion shape
0.00889 face images faces image facial visual human video database detection
0.00128 hinton object image energy cost images code visible zemel codes

(a)

tag: *vision* (ETT1/J100)
0.17815: motion velocity chip intensity image detection analog edge visual flow
0.15645: receptive cells cell fields filter visual spatial linsker orientation principle
0.11882: dominance ocular map cortical orientation eye development cortex maps lateral
0.09592: optical image frequency ring pixel plane loop events beam pixels
0.08801: visual attention contour search selective iiii contours orientation region saliency
0.07733: image images texture pixel resolution vision pixels filters scene regions
0.06345: object objects views view projection pursuit image images edelman intrator
0.05222: view video visual camera object tracking image gesture shape support

(b)

tag: *vision* (ETT2)
0.53789 visual map eye orientation retinal search development activity direction cells
0.44897 image images phase motion flow object pixel vision invariant intensity
0.00880 wavelet murphy joint arm visual dendritic mel video sigma traffic
0.00011 committee members cost bound margin tree effective cascade dimension scaling
0.00011 unsupervised filter convex likelihood competitive image clustering clusters ...
0.00010 neuron cell synaptic firing cells spike activity excitatory phase potential
0.00008 bit memory matlab simd chip processor operations instruction controller ...
0.00008 kernel familiarity discriminant regression discrimination radar theorem fisher ...

(c)

tag: *vision* (ETT3)
0.33384 optical motion flow visual rotation velocity direction location translation image
0.13643 cells visual orientation receptive cortical cortex stimulus spatial contrast tuning
0.10589 visual eye activity map location disparity cortex sejnowski centered head
0.10532 map maps dominance ocular eye development orientation activity lateral cortical
0.05570 image images pixel texture pixels regions color scene vision visual
0.04322 object objects view image views images matching visual match frame
0.03764 activation nets representations internal layers connectionist tasks architectures
0.03493 retina intensity light cells circuit adaptation bipolar insect inhibition feedback

(d)

tag: *pattern recognition* (ETT1/J20)
0.41215 image images texture pixel resolution pyramid regions pixels region search
0.15713 tangent transformation image simard images invariant invariance euclidean
0.10106 face images faces image facial visual human video database detection
0.08961 protein chain sequences region proteins mouse human secondary amino cell
0.06469 projection views view object objects pursuit intrator cooper extraction edelman
0.05461 rbf classifier classifiers radial decision centers lippmann mlp regions knn
0.02613 hinton object image energy cost images code visible zemel codes
0.02125 phase oscillator activity frequency cortex oscillations modules periodic

(e)

tag: *blind source separation* (ETT1/J20)
0.94080 ica source separation sources blind mixing signals amari entropy bell
0.00131 eeg brain ica artifacts subjects activity subject erp signals scalp
0.00112 chip neuron synapse digital pulse analog synaptic chips synapses murray
0.00105 protein chain sequences region proteins mouse human secondary amino cell
0.00100 processor bit memory array hardware bits operations connection activation
0.00095 detection diagnosis patients false normal risk patient fault software accuracy
0.00091 nodes node markov graph conditional arc arcs bayesian ase edge
0.00089 series prediction spline modeling stationary underlying chaotic splines regression

(f)

tag: *biological models* (ETT1/J20)
0.23150 brain activity effects cowan resonance map fibers wave axon chicago
0.15010 cells cortex activity brain region sensory inhibitory stimulus olfactory cell
0.14234 cell firing cells neuron activity excitatory inhibitory synaptic potential membrane
0.12539 synaptic neuron motor phase interneurons biological coupling cell intrinsic
0.11522 dendritic synaptic voltage membrane conductance channels neuron conductances
0.09410 eye velocity vor head movements saccadic gain motor movement visual
0.05983 orientation cortical dominance ocular cortex development lateral eye cells visual
0.04922 stimulus representations subjects stimuli human trials temporal item similarity

(g)

tag: *learning algorithms* (ETT1/J20)
0.30141 tree trees node decision nodes leaf prediction root pruning leaves
0.21200 kernel support svm kernels vapnik machines regression regularization margin sv
0.13659 map organizing som kohonen mapping neighborhood topology structures
0.08868 learner surface merging family game greedy partition samples sampling
0.04137 convergence stochastic descent optimization batch density global update
0.02509 optimization estimation kalman constraint adaptation multipliers sequential
0.01471 clustering cluster similarity unsupervised mutual style partition annealing content
0.00420 perceptron concept hypothesis dimension multilayer capacity mlp binary

(h)

Figure 10.11: Tag topic examples.

the tag-specific topic probabilities $p(z \mid c)$.[11] For all models, the top topics appear semantically correct but the aspects of "vision" vary.

Another difference is the weight assigned to each topic and how quickly it degrades with the topic rank. Here especially ETT1/J100 and ETT3 seem to degrade slower than ETT1/J20. For ETT1 an increase of J_m (more tag tokens) seems to increase this saliency and has an influence on the topic ranking, as suggested by comparing subfigures (a) and (b). However, saliency becomes extreme in ETT2, as shown in (c). The tag basically consists of three topics, with all other topics appearing randomly and with negligible weights. Intuitively, this decreases the effective topic count for each tag. Because for ETT2 this phenomenon occurs for most tag–topic distributions as well as many author–topic distributions, this may be a reason for the inferior retrieval performance.

On the other hand, the saliency of topics also varies across tags of a single model, as can be seen for ETT1/J20 in Fig. 10.11(e)–(h), presumably depending on how well they fit to the topics semantically. This is comparable to principal components that may be close to an input variable.

[11] For ETT3, these probabilities correspond to $\zeta_{c,z}$, but for ETT1 and ETT2 they need to be computed from the parameters $\psi_{z,c}$ using $p(z \mid c) \propto \psi_{z,c} p(z)$ with $p(z) \propto n_z$ (substituting z by y as appropriate).

query: "binaural localization + *computational auditory scene analysis*"

0.05274: auditory	0.00784: stimuli	0.42152: *audio applications*
0.03968: sound	0.00749: periodicity	0.28816: *comp. audit. scene ana.*
0.01819: localization	0.00733: onset	0.09618: *hearing*
0.01560: frequency	0.00732: acoustic	0.06951: *psychophysical models*
0.01454: source	0.00724: band	0.04462: *system identification*
0.01306: tone	0.00715: cues	0.02686: *music*
0.01080: cochlear	0.00706: spectral	0.01063: *blind source separation*
0.01054: spectral	0.00610: signals	0.01063: *neural networks*
0.00984: owl	0.00568: ear	0.00472: *image analysis*
0.00802: location	0.00566: interaural	0.00197: *indep. component ana.*

(a)

query: "*financial applications*"

0.01307: risk	0.00608: asset	0.24154: *financial applications*
0.01193: return	0.00586: capital	0.15801: *markov networks*
0.01093: stock	0.00487: costs	0.03766: *dynamic programming*
0.01038: trading	0.00458: transaction	0.03683: *neural networks*
0.01005: market	0.00458: exchange	0.02165: *learning algorithm*
0.00941: price	0.00454: benchmark	0.01737: *temporal models*
0.00919: strategy	0.00437: investment	0.00868: *reinforcement learning*
0.00721: prediction	0.00412: future	0.00772: *bayesian inference*
0.00703: decision	0.00400: differential	0.00692: *optimisation*
0.00632: financial	0.00377: profit	0.00645: *nonlinear models*

(b)

query: "*bayesian networks + mixture models*"

0.03090: mixture	0.00755: discrete	0.19958: *em algorithm*
0.02874: likelihood	0.00688: covariance	0.19644: *mixture models*
0.02862: em	0.00684: joint	0.13598: *probabilistic models*
0.02093: node	0.00613: mixtures	0.08802: *bayesian networks*
0.01995: density	0.00598: gaussians	0.08589: *gaussian*
0.01590: estimation	0.00592: hmm	0.05030: *hmm*
0.01287: conditional	0.00571: latent	0.03937: *density estimation*
0.01184: markov	0.00519: expectation	0.03364: *latent variables*
0.01038: mixtures	0.00503: posterior	0.02970: *markov networks*
0.00792: probabilistic	0.00414: graph	0.02411: *neural networks*

(c)

query: "*character recognition + audio*"

0.02020: character	0.00546: cochlear	0.28881: *character recognition*
0.01980: auditory	0.00522: digits	0.25639: *audio*
0.01490: sound	0.00512: letters	0.15841: *[untagged]*
0.01460: characters	0.00499: location	0.10815: *hearing*
0.00775: segmentation	0.00491: sounds	0.03618: *image segmentation*
0.00757: image	0.00445: printed	0.02615: *auditory perception*
0.00684: localization	0.00440: signature	0.02314: *pattern recognition*
0.00652: handwritten	0.00419: handwriting	0.01681: *neural networks*
0.00640: word	0.00413: spectral	0.01406: *vision*
0.00630: frequency	0.00412: stimuli	0.00243: *classification*

(d)

query: "monte carlo simulation"

0.02225: posterior	0.00923: hyperparam.s	0.32266: *gaussian processes*
0.02155: carlo	0.00816: mcmc	0.19043: *sampling*
0.02136: monte	0.00804: regression	0.12594: *mcmc*
0.01853: bayesian	0.00785: processes	0.11147: *regression*
0.01524: covariance	0.00778: evidence	0.08840: *neural networks*
0.01480: prior	0.00703: exp	0.02925: *bayesian networks*
0.01372: sampling	0.00690: chain	0.02116: *latent variables*
0.01171: variance	0.00647: prediction	0.02113: *feature selection*
0.00993: latent	0.00639: generalisation	0.01790: *graphical models*
0.00930: markov	0.00570: variational	0.01469: *nonparam. models*

(e)

query: "phonetic"

0.07053: speech	0.00859: mlp	0.30942: *speech recognition*
0.03063: word	0.00764: probabilities	0.25918: *speech*
0.01976: phonetic	0.00715: frames	0.20074: *neural networks*
0.01553: speaker	0.00692: hybrid	0.07763: *hmm*
0.01380: context	0.00688: speakers	0.05181: *audio*
0.01278: phoneme	0.00683: markov	0.04536: *multi-layer networks*
0.01176: acoustic	0.00652: tdnn	0.01660: *markov models*
0.00996: hmm	0.00600: vocabulary	0.01596: *temporal models*
0.00928: vowel	0.00567: spoken	0.00422: *language*
0.00928: frame	0.00557: phone	0.00154: *classification*

(f)

Figure 10.12: ETT1 generated tag–word thesaurus: examples (tags in italics).

Here, especially the tag *blind source separation* in subfigure (f) seems to match very well with a single topic, leaving only 6% of the weight to all others. This good match does not jeopardise retrieval results, as shown in Fig. 10.9(a). Also note that subfigure (e) shows the relation between the tag *pattern recognition* and face detection and recognition terms that has been conjectured for result 9 in Fig. 10.9(b).

Tag–word thesaurus. To enhance retrieval, an interesting aspect in the ETT models is their ability to create "thesauri" of words and tags combined. This extends the LDA-based query expansion method in (10.17). The respective likelihood term is particularly easy to derive for the ETT1 model, allowing to determine similarities between any combination of words and tags:

$$p(t, c \,|\, \vec{w}\,', \vec{c}\,', \cdot) = p(t \,|\, \vec{w}\,', \vec{c}\,', \cdot)\, p(c \,|\, \vec{w}\,', \vec{c}\,', \cdot) \propto \textstyle\sum_k \vartheta'_k \varphi_{k,t} n_k \sum_l \vartheta'_l \psi_{l,c} n_l \,. \tag{10.18}$$

The query parameters $\vec{\vartheta}'$ need to be derived from the ETT1 model as a function of words and tags, which is easiest using Gibbs sampling. We consider an "ideal" author that has written solely the query. To sample the parameters of this query author, we can re-use (10.3) and substitute $a_{m,x} q(x, z \oplus y)$ with $q(1, z \oplus y)$. Furthermore, as the global parameters φ and ψ are known, we

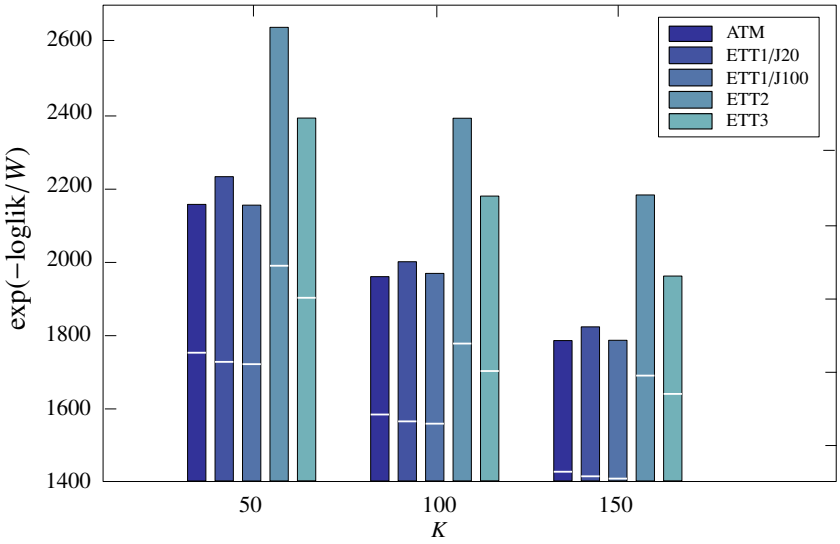


Figure 10.13: ETT perplexity against baseline. Line markers: training-data perplexity.

replace their respective terms $q(z, w)$ and $q(z, c)$, which leads to:

$$p(z_{m,n}{=}z \,|\, \vec{w}', \cdot) \propto (n_{1,z} + \alpha)\, \varphi_{z,w_{m,n}} \qquad p(z_{m,j}{=}z \,|\, \vec{c}\,', \cdot) \propto (n_{1,z} + \alpha)\, \psi_{z,c_{m,j}}\,. \tag{10.19}$$

Some examples of thesaurus correspondences are presented in Fig. 10.12, showing a tag–term query (a), queries with one or more tags (b)–(d), as well as pure term queries (e)–(f). The subjective quality appears high. Notice how subfigure (a) differs from Fig. 10.6(a). In (b), the terms appear to be much better matches than the tags. One explanation is that finance is an area with only cursory coverage in the NIPS community and the tags describe analysis methods for financial applications. Subfigure (d) is notable because it shows a query for unrelated tags, which creates sets of terms and tags where each entry can be clearly associated to one of the query tags.

In summary, the examples in Fig. 10.12 suggest that query expansion indeed is viable using the proposed approach. Moreover, we may compare the parameters $\vec{\vartheta}'$ in (10.19) with those of the authors, creating an alternative retrieval function based on some distance measure, as described in Section 3.7.

10.4.2 Likelihood and clustering behaviour

The following experiments determine numerical metrics of model quality. Held-out word likelihood is a measure of generalisation from the trained model to held-out data, and the VI clustering distance allows evaluation of how similar the trained topics are to the clustering imposed by tags.

Both metrics are seen as "analysis" metrics: We do not necessarily expect them to be better for ETT than for baselines, but rather to gain insights into how models differ.

Log likelihood of held-out data has been a standard metric in topic model evaluation, and in many cases the closely related word perplexity has been used. As described above, we used the document completion approach where the likelihood of 50% of the words in test documents is measured given the model trained on the other 50% and the training data; cf. Section 6.5.2.

Model	ATM			ETT1/J20			ETT1/J100			ETT2			ETT3		
K	50	100	150	50	100	150	50	100	150	50	100	150	50	100	150
H{Z}	3.791	4.436	4.794	3.815	4.464	4.758	3.802	4.452	4.718	3.709	4.359	4.727	3.845	4.481	4.859
H{C}	⟵ 3.506 ⟶														
I{$Z;C$}	0.865	0.980	1.041	1.094	1.300	1.446	1.002	1.208	1.350	1.398	1.531	1.590	1.057	1.238	1.321
VI{$Z\|\|C$}	5.567	5.983	6.218	5.132	5.370	5.372	5.304	5.542	5.524	4.419	4.803	5.053	5.238	5.511	5.723
H{Z,C}	7.721	8.384	8.745	7.535	8.109	8.321	7.693	8.186	8.414	7.145	7.788	8.165	7.605	8.181	8.555
D{$Z\|\|C$}	0.721	0.714	0.711	0.681	0.662	0.646	0.689	0.677	0.656	0.618	0.617	0.619	0.689	0.674	0.669

Figure 10.14: Cluster distances for ETT and ATM models.

Results for varying numbers of topics K are presented in Fig. 10.13. As expected, the baseline model ATM performs best in this metric because it is the only model where objective function and test metric fully comply, whereas for the ETT models the additional constraint to optimise tag–topic associations degrades pure held-out word likelihood. On the other hand, *word* likelihood is the only likelihood that all models have in common, as the joint word–tag likelihood does not apply to ATM and similar strongly varies between ETT models due to their different modelling assumptions (cf. ETT retrieval functions). The measure therefore quantifies the change of optimum state from the baseline ATM by adding the constraint to optimise for both words and tags.

The ETT models clearly create *different* topics, and as has been seen in the previous retrieval experiments, the tags appear to "pull" the model towards topic associations that are meaningful on a semantic level, as the retrieval results suggest.

Compared to ATM, the perplexities of the ETT1 model seem to be closest, and the strength of tags on the model is an influencing factor: If J_m is increased, the model deviates from the topics because constraints are added. However, with $J_m = N_m/100$, the influence of tags on perplexity almost vanishes. The constraints introduced by the other models appear to be much harder, though, introduced by tag filtering in ETT2 and mixture merging in ETT3.

Clustering quality. To measure the general clustering quality, the Variation of Information distance is being applied, as introduced in Section 3.7.3. Considering authors the items to be clustered, the VI metric measures the distance between the (soft) clusterings created with topics and with tags identifying clusters. These clusterings are expressed by the author-specific topics, $\vec{\vartheta}_a$, trained by the models on one hand, and the tag distributions for authors on the other, which are obtained from the evidence by a weighted average:[12] $p(c|a) \propto \sum_m c_{m,c} a_{m,a}$. Based on this, we can define random variables Z and C for a topic and a tag being associated to an author and measure the distance VI{$Z\|\|C$}. It is worthwhile to normalise this distance to the interval [0, 1] using the joint entropy, H{Z, C}:[13]

$$\mathrm{D}\{Z\|\|C\} = \frac{\mathrm{VI}\{Z\|\|C\}}{\mathrm{H}\{Z,C\}} = \frac{\mathrm{H}\{Z\} + \mathrm{H}\{C\} - 2\mathrm{I}\{Z;C\}}{\mathrm{H}\{Z,C\}} = 1 - \frac{\mathrm{I}\{Z;C\}}{\mathrm{H}\{Z,C\}} \tag{10.20}$$

[12] This is obtained via $p(c|a) = \sum_m p(c|m)p(m|a) = \sum_m p(c|m)p(a|m)p(m)/\sum_{m'} p(a|m')p(m')$ with $p(m') = 1/M$.

[13] The joint distribution for Z and C is obtained via $p(Z{=}z, C{=}c) = \sum_a p(z|a)p(c|a)p(a) = \sum_a \vartheta_{a,z} p(c|a)p(a)$ where the observed author strength is used as prior: $p(a) \propto \sum_m a_{m,a}$.

where $\mathrm{I}\{Z;C\} = \mathrm{H}\{Z\} + \mathrm{H}\{C\} - \mathrm{H}\{Z,C\}$ is the mutual information between random variables Z and C. The normalised VI distance may be interpreted as the portion of the total information in both clusterings (the quotient on the right of (10.20)) that they do *not* share (the 1-complement).

A comparison of the normalised VI distance and the related quantities is shown in Fig. 10.14. One would expect that a model that learns topic clusterings with explicit cluster associations will use the extra information to improve the overall clustering quality, as opposed to a model that does not. Note, however, that even plain LDA will implicitly adapt to to ground-truth clusterings as they are related to term co-occurrences, cf. [Heinrich et al. 2005b].

As can be seen in the bottom row in the figure, the ETT models clearly result in topic clusterings that are closer to the tag clusterings than ATM. Remarkably, the ETT models achieve this without measuring any tag-specific structures, just by the differently trained topic parameters. Furthermore, the mutual information between topics and tags is significantly increased, especially for the ETT1 model with high J_m and ETT2.

These results suggest that indeed the topics incorporate information from the tags, which may also partly explain the improved retrieval performance for ETT1 and ETT3. For ETT2, however, the additional constraint imposed by tag filtering may be too strong: Although the VI metric clearly indicates that the ETT2 topics are much closer to the tag clusterings than any other model, the model cannot seem to exploit this advantage in retrieval or word perplexity.

10.4.3 Topic quality

The final experiments analyse topic quality from the perspective of their semantic coherence.

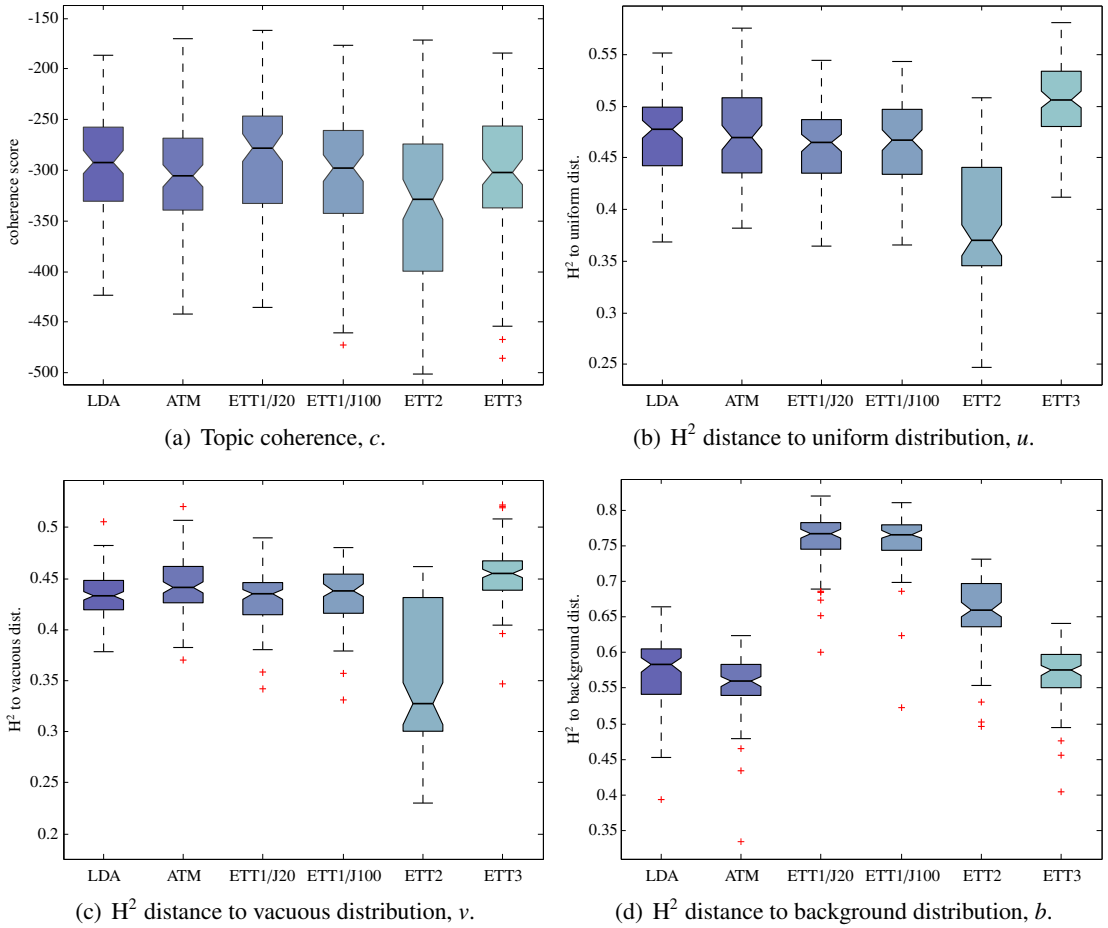
User test. In [Heinrich 2011b], the semantic coherence of ETT1 and ETT3 modelling results was measured by human judgements of five expert voters in the spirit of a topic-coherence experiment [Chang et al. 2009]. In summary, the test evaluates word association to topics as well as topic association to experts and to tags.[14] Voters are presented with groups of (a) 6 topic words and (b) 4 expert/tag topics and asked to detect the least consistent item. In every question, items presented have high probability according to the model, except for an unlikely "intrusion" item that participants may easier identify in semantically coherent groups. Examples are presented in Section 3.7.1.

For ETT models, voting results for word intrusion in ETT topics turned out to be similar as for the baseline LDA. Equally, topic association coherence was found to be of a similar quality for experts and tags as for documents in the baseline LDA. Given the small voter number, no significant improvement (nor adverse effect) of the author or tag influence compared to LDA could be determined.

Numerical methods. Due to the high human effort for subjective tests like the ones above, several researchers have looked into measures that can predict their outcomes with numerical means, in particular [Mimno et al. 2011] and [AlSumait et al. 2009]. As described in Sections 3.7.1 and 3.7.4, such tests try to statistically characterise typical properties of "coherent" and "junk" topics via the respective model parameters.

Applied to the parameters of the ETT models and their baselines, different topic quality metrics are presented in Fig. 10.15.

[14] In [Heinrich 2011b], the conference tracks had been used as a basis for tagging information. Furthermore, different pre-processing for the corpus text was used.



(a) Topic coherence, c.

(b) H^2 distance to uniform distribution, u.

(c) H^2 distance to vacuous distribution, v.

(d) H^2 distance to background distribution, b.

Figure 10.15: ETT topic coherence comparison vs. baselines LDA and ATM. Notches: $p = 0.05$, outliers marked beyond maximum whisker length = 1.5 × box height.

Topic coherence. The topic coherence metric [Mimno et al. 2011], abbreviated c-metric in the following, posits that good topics contain terms t with high $\varphi_{k,t}$ that also co-occur in many documents. In particular, a sum is created where each element is the log ratio between the number of documents that a term co-occurs with a more likely term and those that the more likely term occurs in alone. The total of the sums for the T strongest terms is the coherence score, cf. (3.51).

Fig. 10.15(a) presents the c-metric for all topics and models using the $T = 20$ strongest terms in each topic. Using the baseline LDA, no significant difference could be found for ETT1 and ETT3. This result is in agreement with the subjective tests in [Heinrich 2011b]. For ETT2, performance drops strongly.

However, the figure indicates that also ATM seems to have inferior topic coherence compared to LDA, which may be due to the author-specific word co-occurrence relevant for ATM as opposed to document-specific co-occurrence in LDA. Connected to this, it is interesting to see that indeed the ETT1 and ETT3 models seem to compensate this adverse effect, presumably by the influence of tags on the topics. This suggests that despite the weaker performance on held-out word likelihood observed above, the topics produced are indeed of good quality.

The best model in terms of the c-metric is ETT1 with $J_m = N_m/20$, and the difference is significant according to Wilcoxon's rank-sum test for $p = 0.01$. Reducing the influence of tags in ETT1 and using the ETT3 model creates topics that are slightly but not significantly better than those of ATM, considering typical confidence levels.

It is interesting, however, that the c-metric still measures good values for ETT2. Subjectively, indeed ETT2 produces a portion of good topics and even very good ones but also more incoherent ones compared to the other models. The conjecture is that the strong influence of tags constrains the model causing it to partly miss the optimum word–topic associations. Note, however, that coherence performance does not drop as sharply as for retrieval: The quality of φ in ETT2 may be better than that of ϑ that is crucial for retrieval.

Complementary metrics. To complement topic coherence scores, the metrics from [AlSumait et al. 2009] are presented in Fig. 10.15(b)–(d). Different from this previous work, which uses KL divergence and cosine distance combined, here we use the squared Hellinger distance because of its normalisation properties; cf. Section 3.7. Furthermore, we expect better insight into topic quality by viewing the metrics as separate indicators rather than a combined scalar value.

Fig. 10.15(b) presents the distance of topic distributions to the uniform distribution, $\varphi_t^{\mathrm{u}} = 1/V$, or u-metric. Good topics have few salient terms, re-enacting some power-law distribution between rank and term probability in the topic. It appears that ETT1 topics have a insignificantly lower distance to $\vec{\varphi}^{\,\mathrm{u}}$ than the baseline LDA and are on par with ATM. ETT3 topics are significantly more non-uniform while again ETT2 is worse by a large margin.

A second metric measures the distance of topic distributions to a "vacuous" (empty) distribution, which corresponds to the empirical term frequencies of the entire corpus. These may be computed by averaging topic distributions weighted with their strength, $\varphi_t^{\mathrm{v}} \propto \sum_k \varphi_{k,t} n_k$. The farther topics are different from the vacuous distribution, the better they describe a dedicated theme in the corpus. Under this v-metric, as is seen in Fig. 10.15(c), the topics of ETT1 appear slightly inferior to the baseline ATM but not LDA. ETT3 again is superior while ETT2 drops.

The third metric in [AlSumait et al. 2009] measures how discriminative a topic is for a subset of documents or authors. The maximum non-discriminative topic k will have equal $\vartheta_{m,k}$ or $\vartheta_{x,k}$ for every m or x. This is the background distribution, and the distance to it is computed against the "inverted" topic weights, $\vartheta^{\mathrm{inv}}_{k,x} \propto \vartheta_{x,k} p(x)$ (here for an author-based model). The distance to a background distribution, or b-metric, is presented in Fig. 10.15(d). This metric shows a high advantage of ETT1 and ETT2 models while ETT3 seems on par with the baseline LDA and better than ATM. The good values for ETT2 seem to be due to the model having to decide sharply between topics, which is corroborated for instance in Fig. 10.11(c).

Correlation. We analyse the mutual correlations between quality metrics by applying Spearman's correlation coefficient ϱ to all paired measurements.[15] The main dependencies turn out to be between topic coherence and u-metric ($\varrho = 0.81$) as well as between coherence and topic weight n_k ($\varrho = 0.74$). High correlation also exists between the latter two ($\varrho = 0.84$). For all models, these correlation values are roughly the same (tolerance <0.1), which indicates that it may be possible to create an extended numerical topic quality metric based for instance on the principal components of the different metrics or the combination approach in [AlSumait et al. 2009].

[15] Spearman's ϱ measures how well the ranks of the data correlate rather than the values themselves. The measure is robust to outliers and does not require assumptions on Gaussian distribution or linear relationship for the data.

LDA: "analog hardware"
0.04043 analog
0.03768 circuit
0.03379 chip
0.02492 voltage
0.02158 vlsi
0.01408 synapse
0.01395 cmos
0.01153 neuron
0.00885 transistor
0.00857 mead

c	u	b	v	n_k
−186	.543	.606	.473	31k
1	4	25	7	2

ATM: "analog hardware"
0.05870 circuit
0.03492 voltage
0.03295 analog
0.02423 chip
0.02051 vlsi
0.01691 silicon
0.01227 cmos
0.01227 transistor
0.01223 mead
0.01131 design

c	u	b	v	n_k
−187	.545	.606	.479	22k
3	7	5	6	8

E1/20:"analog hardware"
0.04461 circuit
0.03650 analog
0.03047 chip
0.02677 voltage
0.02042 vlsi
0.01485 silicon
0.01030 transistor
0.00935 gate
0.00921 design
0.00903 cmos

c	u	b	v	n_k
−169	.545	.730	.472	27k
2	1	86	10	4

E1/100: "analog hardware"
0.04870 circuit
0.02958 voltage
0.02812 analog
0.01737 chip
0.01621 vlsi
0.01195 silicon
0.01087 transistor
0.00979 cmos
0.00941 gate
0.00848 silicon

c	u	b	v	n_k
−177	.532	.725	.425	26k
1	4	86	39	5

E2: "analog hardware"
0.02339 chip
0.02310 analog
0.01894 circuit
0.01846 voltage
0.01352 vlsi
0.01076 silicon
0.01027 neuron
0.00892 pulse
0.00853 digital
0.00804 gate

c	u	b	v	n_k
−196	.478	.670	.445	20k
4	4	45	8	14

E3: "analog hardware"
0.03342 chip
0.03107 analog
0.02943 circuit
0.02597 neuron
0.01913 voltage
0.01651 synapse
0.01262 vlsi
0.01250 digital
0.00991 circuits
0.00906 synapses

c	u	b	v	n_k
−200	.566	.623	.455	56k
5	6	9	40	4

(a)

LDA: "phonetics"
0.11002 speech
0.03701 speaker
0.01723 acoustic
0.01467 phoneme
0.01330 vowel
0.01222 phonetic
0.01109 utterances
0.01032 database
0.00912 formant
0.00763 spectral

c	u	b	v	n_k
−226	.511	.594	.471	16k
13	19	41	4	24

ATM: "phonetics"
0.09004 speech
0.02460 speaker
0.02171 context
0.02020 word
0.01799 hmm
0.01731 acoustic
0.01469 frame
0.01377 phonetic
0.01373 utterances
0.01120 recognizer

c	u	b	v	n_k
−260	.544	.623	.480	21k
20	8	2	5	11

E1/20: "phonetics"
0.07128 speech
0.03096 speaker
0.01996 acoustic
0.01569 vowel
0.01394 phonetic
0.01291 phoneme
0.01189 utterances
0.01007 spoken
0.00937 formant
0.00868 consonant

c	u	b	v	n_k
−186	.543	.687	.454	25k
9	2	93	10	6

E1/100: "phonetics"
0.08577 speech
0.02512 speaker
0.01731 acoustic
0.01239 phoneme
0.01172 frame
0.01159 vowel
0.01064 utterances
0.00970 spoken
0.00883 segment
0.00782 word

c	u	b	v	n_k
−236	.506	.767	.473	14k
12	16	49	4	30

E2: "phonetics"
0.04826 speech
0.02290 word
0.00991 phoneme
0.00962 context
0.00871 speaker
0.00853 words
0.00779 letter
0.00762 frame
0.00739 tdnn
0.00705 acoustic

c	u	b	v	n_k
−248	.463	.659	.448	17k
20	8	50	13	20

E3: "phonetics"
0.05505 speech
0.04336 word
0.02302 speaker
0.01507 phoneme
0.01341 frame
0.01314 context
0.01299 phonetic
0.01216 word
0.01206 mlp
0.01087 hmm

c	u	b	v	n_k
−243	.564	.627	.493	40k
17	8	7	4	12

(b)

E1/20: "EM+mixtures"
0.03211 likelihood
0.03101 mixture
0.02899 density
0.02732 em
0.01337 prior
0.01305 estimation
0.01216 bayesian
0.01076 conditional
0.01054 posterior
0.00896 missing

c	u	b	v	n_k
−172	.534	.716	.472	36k
3	4	87	69	3

(c)

E1/20: junk
0.01826 brain
0.01248 spin
0.01176 stochastic
0.01085 chicago
0.01085 resonance
0.01013 correlation
0.00851 voronoi
0.00706 wij
0.00706 markers
0.00652 fibers

c	u	b	v	n_k
−421	.362	.788	.388	4.5k
98	100	15	91	99

(d)

E1/100: missed junk
0.02685 validation
0.02439 prediction
0.01991 regression
0.01914 selection
0.01808 variance
0.01793 cross
0.01451 bias
0.00853 committee
0.00791 bootstrap
0.00728 squared

c	u	b	v	n_k
−262	.512	.722	.423	20k
27	13	88	31	9

(e)

E1/100: "game"
0.02279 game
0.02148 play
0.01651 board
0.01507 move
0.01337 chess
0.01245 expert
0.01219 player
0.01219 backgammon
0.01075 reinforcement
0.01075 carlo

c	u	b	v	n_k
−265	.516	.760	.466	19k
30	7	59	2	10

(f)

E2: "language" + junk?
0.02179 language
0.01139 word
0.01090 stress
0.01008 grammar
0.00861 rules
0.00771 syllable
0.00648 tree
0.00623 vowel
0.00615 connectionist
0.00582 representations

c	u	b	v	n_k
−343	.374	.728	.329	11k
68	48	3	46	50

(g)

E3: "finance" + junk?
0.01885 risk
0.01742 return
0.01551 stock
0.01148 price
0.01148 trading
0.01086 market
0.00936 penalty
0.00882 financial
0.00868 prediction
0.00847 regularizer

c	u	b	v	n_k
−333	.482	.543	.474	13k
69	72	84	7	79

(h)

Figure 10.16: Example topic distributions and metrics. Values for c, u, b and v correspond to the metrics in Fig. 10.15, and n_k is the learnt topic strength with $\langle n_k \rangle = W/K = 12.7$k. Below the numeric values for the metrics, the ranks among all topics of a model are given.

A remarkable finding is the weak correlation between the b-metric and other metrics (values between LDA, $\varrho = -0.12$, and ETT2, $\varrho = 0.10$).

Examples. In Fig. 10.16, examples of different topics are given by their top terms and weights as well as the metrics. An interesting observation is that a portion of topics appears to overlap across all six models considered.[16] This suggests that the models find strong semantic relations inherent in the data, even across the different co-occurrence contexts document and author (LDA vs. ATM and ETT) and different methods to incorporate tags in the ETT models.

[16]This is a stronger finding than stability across different runs of the Gibbs sampler where topics vary slightly.

Examples for this behaviour are given in Fig. 10.16(a) and (b) where terms are clustered that are close to the thematic areas of "analog hardware" and "phonetics", respectively. In both examples, ETT1 outperforms the baselines LDA and ATM in terms of topic coherence while ETT2 is inferior and ETT3 better than ATM for (b). Relatively to the other topics, the two examples have a very high coherence rank (a) and are among the 20% best topics respectively (b). Towards topics with weaker ranks, this cross-model topic stability decreases.

Fig. 10.16(c)–(h) shows some other interesting topics from different ETT models: in subfigure (c), a coherent topic with a low v metric, and in (d) a junk topic where all metrics except b rank low. The topic is subjectively incoherent, which agrees with the low coherence value of -421. However, at times there is a discrepancy between subjective and measured coherence, such as in example (e): With an above-average coherence rank 27 (of 100) and relatively good $c = -262$, the topic is subjectively junk while (f) may be clearly associated with a "game" theme with slightly worse metrics with rank 30 and $c = -265$. Vice versa, examples (g) and (h) show relatively low ranking for topics that subjectively can be assigned to clear thematic areas. However, in the majority of the studied cases, especially the coherence metric seems to predict subjective impressions well. This seems to be even the case when the u and v metrics disagree with it, e.g., in subfigure (c). As already found in the correlation analysis, the b-metric does not appear to predict subjective topic quality well.

10.5 Discussion

This section discusses different aspects of Sections 10.2–10.4, starting with the design itself, the models and finally presenting an outlook to model extensions, including a proof-of-concept of how the ETT models may be used within a visual community browser.

10.5.1 Model design process

In the scenario study undertaken, three ETT models have been designed from the ground up using the novel design method and illustrating its usage. Especially the construction of full conditionals proved viable, for which the q-terms introduced in Chapter 9 appear instrumental. The NoMM models could be structured from assumptions on data and terminals directly to equations (10.3), (10.10) and (10.14). It indeed appears handy to have the Gibbs terms (in their simplified form using q-terms) available for review at each design step, with the sub-structure library as a tool to decide on available modelling possibilities. Compared to this, the more traditional derivation of ETT1 given in Appendix E is significantly more involved in terms of calculation effort, and for more complex models, this difference is expected to increase.

The ETT models output from the design are, in addition, close to state of the art models that target similar assumptions, as shown in Section 10.3. This appears plausible because the sub-structures from Chapter 9 used for design have been synthesised on the basis of a typology of model structures from prior work. Design therefore may be understood to perform a recombination of model structures adapted to new problem requirements.

However, the general idea of constructing models with prediction of properties is restricted to qualitative evaluation: If Gibbs term or likelihood appears to "look right" for a model because it emphasises some co-occurrence assumed in the observed or hidden data (NoMM edges), there's a chance that this model will work well. There is no way around experimentally testing the

model design to gain quantitative insight, as also the ETT models have shown. Fortunately, in the approach developed in the thesis, implementation is facilitated largely by code generation with the Gibbs meta-sampler, still reducing the overall model design effort.

10.5.2 ETT models

The ETT models themselves may be considered innovations in their own right. For the ETT1 model, the innovation over the closest ancestor model MM-LDA [Ramage et al. 2009] consists of (1) using authorship instead of documents as co-occurrence contexts for word and tag features, (2) the tag boosting approach and (3) the retrieval functions.[17] The ETT2 model extends a model by [Tang et al. 2008] by covering multiple labels. Finally, ETT3 introduces a novel topic modelling structure, C3 interleaved indices, which is validated experimentally in the current study.

Model behaviour. Considering the metrics studied, the ETT1 and ETT3 models appear to be best in terms of retrieval performance and topic quality. ETT1 is slightly better for term queries while ETT3 prevails for tag queries in retrieval. For topic coherence and complementary quality metrics, only insignificant improvement of ETT1 over the baseline LDA could be found, corroborating the user study results in [Heinrich 2011b]. In particular tag boosting configurations, ETT1 is able to outperform ATM, however, with statistical significance. ETT3 is on par with ATM in the coherence metric and significantly better than all other models in the topic distances to uniform and vacuous distributions. Regarding term-based retrieval performance, ETT1 and ETT3 are slightly better than the baseline ATM, and in tag retrieval ETT3 performs best, followed by ETT1.

Tag boosting in ETT1 may indeed be the key to making ETT1 competitive: By allowing the sampler to resample tags for documents, their influence on the overall model can be adjusted. As indicated by the decreasing cluster distance values between topics and tags, the more tags are sampled repeatedly in a context (document, indirectly author), the stronger the model is indeed influenced by this additional information, and it appears that the author–topic distributions $\vec{\vartheta}_a$ can exploit this information to improve retrieval results and topic coherence. However, if the influence of tags is too high, e.g., $J_m = N_m$, the model degrades because parameters $\vec{\vartheta}_a$ are "pulled" away from word distributions towards tag distributions (but still generate topics for words independently according to the E2 structure). ETT2 may be considered an extreme case where joint sampling of word–tag pairs seems to limit the available options for the sampler to sub-optimal regions in the latent space. The effect is that the model still works well for some topics and tags (presumably the ones that appear often and have few wrong document associations) while for a large portion it fails, leading to statistically significant performance differences in retrieval and topic quality. Remarkably, the ETT3 model that samples a tag for each word, as well, is robust to this, with even superior tag retrieval performance.

Another finding suggested by the study is the ability of the ETT models to work in a semi-supervised manner. Untagged documents are aligned with the tag semantics via the topic structure and can be retrieved via these tags. For realistic tagging scenarios, this is a requirement. Furthermore, this ability is beneficial for tag recommendation of existing documents, allowing to refine the meta-data on a given corpus. Complementary to this, the thesaurus functionality proposed allows recommendation of relevant tags for unseen documents and authors, and the empirical results on this have been encouraging.

[17] The work in [Kataria et al. 2011] was achieved concurrently to [Heinrich 2011b] and for a different modality.

Limitations. In the current study, the data set has been a compromise to obtain the modalities necessary for the community scenario considered. Especially tests with a larger quantity of judged term and tag queries are necessary to obtain statistically significant performance differences against the baselines. Also, finer aspects of topic behaviour have yet to be elucidated as part of a more general question of how sub-structures influence model behaviour on the semantic level. For example, it is not currently possible to make statements about the difference of the ETT1 and ETT3 models with respect to the topics they produce although their structures differ strongly. Another question is the robustness of the models against incomplete and noisy data, as well as the apparent dependence on the document frequency of observations: Infrequent tags tend to produce inferior topic quality and retrieval results. Such questions are not restricted to ETT models but topic models in general, and answers will be beneficial also as design criteria.

10.5.3 Model extensions

The assumptions underlying the ETT models have been made relatively simple to focus on the design method, and various extensions can be introduced to improve their retrieval performance and enable them for other tasks. In order to improve the expertise finding scenario, there are three directions suggested by the findings in the ETT design and experiments:

- *Multiple author interests:* Assume that an expert can have multiple fields of expertise, which may be achieved by an additional mixture level in the author branch of the models, as demonstrated by a "persona" level in [Mimno & McCallum 2007]. Here, it is of interest also to estimate the number of personas for each author, and non-parametric methods come into play, cf. Section 3.6.2 and for the corresponding NoMM structures Section 5.5. The resulting model should be able to cope with situations as those explained for the tag query that an author's expertise "degrades" if interests are too broad.

- *Topical influence:* A second direction is to include the importance of authors on the community, for which a $cites(m, m')$ relation may be added to the AMQ schema as the typical evidence for document influence. Beyond measures like PageRank [Brin & Page 1998] that find global node importance from directed graph structures, one of the virtues of topic models is to model the influence between entities in a topic-specific manner. Local influence propagation is captured for instance by the citation influence model [Dietz et al. 2007] and Bernoulli process topic model [Guo et al. 2010] that mix original content and cited documents via a dedicated mixture level (whose parameters then determine document influence). A variant approach includes an E2 branch for citation [Kataria et al. 2011] similar to the ETT1 model, which uses the link topology and additionally the citation text for influence analysis. To capture global influence, methods typically exploit graph topology, like TopicFlow [Nallapati et al. 2011] with a topic-specific PageRank algorithm used
  N3 node). Closely related to influence models, link prediction models may be considered, such as the relational topic model (RTM, [Chang & Blei 2009]), which allows prediction of links from content and vice versa using topic-based regression of links with an N5+E4 structure. Alternative approaches like [Gerrish & Blei 2010] perform influence analysis over time (i.e., a $hastime(m, t)$ relation) instead of links.

All of these models use $cites(m, m')$ or $hastime(m, t)$ relations, i.e., they always relate to documents m. A prerequisite to integrate such work with a relational ETT model therefore needs to incorporate a $cites(m, a')$ relation that determines the influence of an author a', apart from accommodating tags in a submodel. Work like [Kataria et al. 2011] used the N2 method from ATM to solve this problem (analogous to the ETT models), but it remains to be studied whether such an approach is effective in combination with more complex citation-based models. As a result of incorporating influence, the author weighting based on token counts in (10.7) may be replaced by an importance term dependent on the strength of the author influence w.r.t. the query topics.

- *Community tagging process:* In many real-world situations where social tagging plays a role, tags have additional aspects that go beyond the assumptions applied in the current study where tags have semantic character (as opposed to rating, e.g., "*excellent*", or functional, e.g., "*to be done*"). Furthermore, the identity of the tagger is ignored. Model extensions may integrate these aspects for larger adoption in "live" virtual communities. This may for instance follow prior work like [Harvey et al. 2010] that model tagging users, documents and tags as a tripartite graph structure, or [Kashoob et al. 2009] who explicitly model the social annotation community and find categories of related tags.

- *Structured tag vocabularies and ontologies:* The set of tags may be structured hierarchically, define similarity and mutual exclusion relationships, etc. More generically, a full ontology of tag concepts may be incorporated that allows logical inference on the tagging information (cf. Section 3.2.1) and connects this with the associated author and document content. Any of such additional information may (1) improve topic coherence and (2) provide additional retrieval precision because tag queries may be better disambiguated. Incorporating hierarchical domain knowledge and imposing similarity and exclusion a priori may be achieved for instance using Dirichlet forest priors in an N3 node as shown by [Andrzejewski et al. 2009] (cf. Chapter 5). Tag hierarchies again can be learnt by topic hierarchies like those of the hierarchical PAM models in the NoMMs or by integrating methods like those of [Navigli et al. 2011] with NoMM inference. Generally, NoMM model structures to exploit tag ontologies will vary strongly with the formal semantics used, and the mapping between both is a research direction in its own right.

Other methods that are worthwhile to be pursued in future ETT models include the query expansion mechanism via topic-based term and tag recommendations, (10.18). This is especially useful for search in domain-specific data where it is difficult to establish thesausi and other auxiliary semantic information to support annotation and retrieval.

Following [Wei & Croft 2006], the topic-based approach may moreover be combined with a language model or other retrieval approach that allows literal queries. It remains to be investigated to what extent the interactive query expansion strategy can make up for the inferior performance of topic models reported in [Wei & Croft 2006] or even improve performance in a combined approach.

Furthermore, the method used for pooling term query results in the study has turned out to be a valuable retrieval strategy: For a set of queries with different expansions (or aspects of the information need), the results scores are agglomerated, which leads to re-ranking of the retrieved top experts and thus optimises the results set.

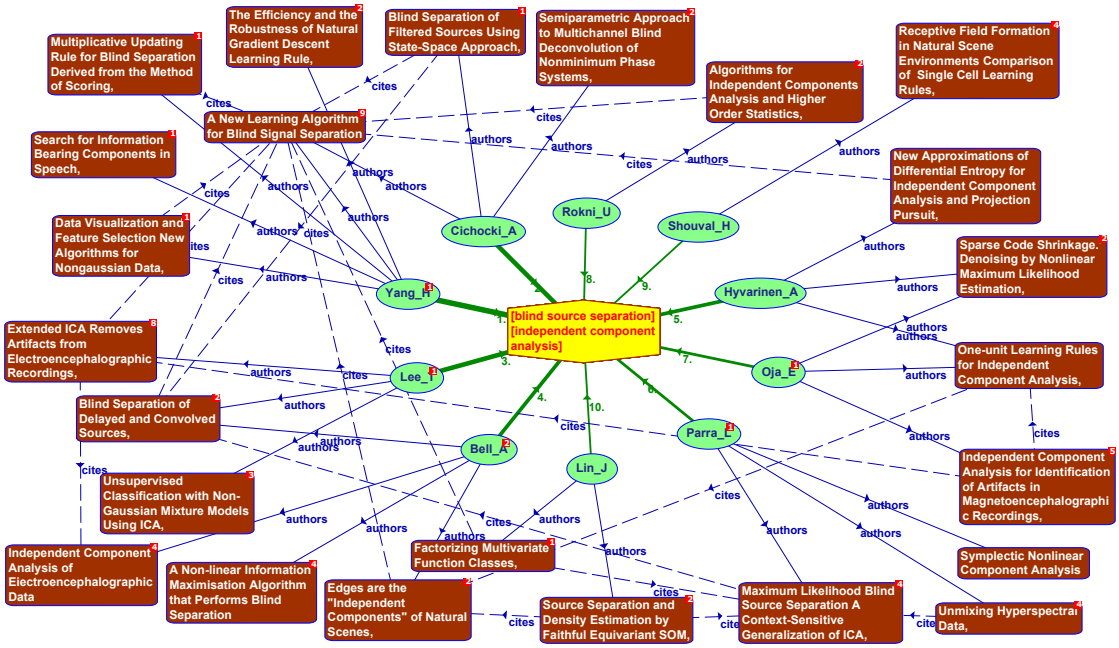


Figure 10.17: ETT example query in community browser.

Towards a community browser. Finally, models like ETT and the discussed extensions may be integrated with a graphical front-end that allows browsing of the community. Identifying expertise then becomes a navigation process in which queries are an inherent part. Motivated by the findings on representing community structure in Chapter 2, a proof-of-concept of such a front-end has been implemented as a graph-based interactive visualisation, extending earlier work in [Heinrich et al. 2005a, Heinrich 2005].

An exemplary screen exported from the browser is shown in Fig. 10.17, visualising the results of the tag query in Fig. 10.9(a). The distinct approach pursued here is to embed the query directly into the community graph and link it to the retrieved experts. Thus, the co-citation and authoring structure of the community relevant for the query (here the $authors(a, m)$ and $cites(m, m')$ relations of the AMQ network) becomes directly visible. In particular, the subgraph of maximum distance ("radius") 2 around the center node is displayed. Interactively, the searcher can view node details and annotate items of interest, collapse irrelevant items and move the center node to explore the community beyond the immediate query context.[18]

In addition to adding dedicated query nodes for term and tag queries as in the example, similarity queries can be issued from any node. This adds edges from the node to the most similar items according to the underlying model. As with term and tag queries, links to irrelevant query results (e.g., Shouval_H in the example) can be removed, which may be used to incorporate relevance feedback mechanisms on model level.

Using the ETT1 model in this browser, the retrieval functions (10.6) or (10.7) can be directly applied to link relevant authors to a query. To allow efficient retrieval of documents in addition,

[18]The numbers on the top-right of the nodes in Fig. 10.17 indicate hidden $authors(a, m)$ and $cites(m, m')$ links. In addition, the context radius may be varied, but graph views with radius >2 typically need additional filtering to curb visual complexity, e.g., showing only co-authors not in the results set (at radius 3).

parameters $\vec{\vartheta}_m$ for documents can be determined and cached. In ETT1, it is straight-forward to obtain these parameters by querying the model with document words and tags using (10.19).

Beyond the ETT models, especially incorporation of citation structure into the visual querying process is of interest. This may result in topic-specific links [Dietz et al. 2007] that may be also used to filter the links in the visualisation, additionally to show query-specific node importance based on models like TopicFlow [Nallapati et al. 2011], or to visually recommend relevant links between experts and articles using an approach based on RTM [Chang & Blei 2009].

In the browser, furthermore novel retrieval strategies may be integrated, such as [El-Arini & Guestrin 2011]. Here documents are identified by stating an initial set of relevant documents and then expanding this set by optimising an objective function based on document influence. In a user study, this method, which may be personalised with a searcher profile, outperformed state-of-the-art retrieval systems in a related-work search task.

10.6 Conclusions

In this chapter, the results of this thesis have been summoned to design topic models for virtual communities, in particular for an expert finding task in a scientific community that incorporates document tags. Central to this was the application of the NoMM development method from Chapter 9, starting with setting up the AMQ schema, making assumptions on the data and performing several iterations of model composition and predition. The designed models have been implemented using the Gibbs meta-sampler and tested against a sample corpus. This way, an end-to-end illustration of the design process has been given, validating the method by an exemplary case.

The expert–tag–topic models (ETT1–3) extend the state of the art towards retrieval of experts from documents with additional semantic tagging information or classification data. Experimentally, two of the models could be shown to outperform baseline models (LDA, ATM) in terms of retrieval performance and topic quality, which is attributed to the influence of additional tagging information. However, as the performance differences to the baselines are mostly not statistically significant, an extended study is indicated, based on larger data sets with more judged retrieval queries.

Based on the ETT models, an outline of model extensions has been given, including aspects of multiple thematic areas for experts, citation structure to capture author influence and thus improve retrieval results, as well as models of social tagging. Finally, a proof-of-concept visual community browser has been presented that combines ETT search with interactive exploration of the community's network structure.

Chapter 11

Conclusions

Discusses the results and contributions of the thesis and proposes future directions.

11.1 Introduction

From their early beginnings as analysis methods for bag-of-words text [Deerwester et al. 1990, Hofmann 1999a, Blei et al. 2002], in the last decade topic models have developed to a widely accepted method of analysis for various types of discrete and partly non-discrete data. In this evolution, the focus of interest has also widened from theoretical foundations to applications, and the fields of application now reach far beyond machine learning or text mining, they include psychology [Navarro et al. 2006], complex clustering tasks in text data bases [Talley et al. 2011], bioinformatics [Liu et al. 2010], computer vision [Cao & Fei-Fei 2007] as well as music analysis [Hu & Saul 2009].

One of the core contributions of this thesis is to reduce the amount of knowledge and experience required to develop and implement models of the topic model family, helping practitioners concentrate on the analysis problems at hand and providing access to more complex model structures than the basic latent Dirichlet allocation model (LDA) that is predominant in fields of application where machine learning and text mining knowledge is not central to the problem domain. For the experts in these fields, topic models may thus become a whole toolbox rather than a tool.

Central to this result is the corroboration of the initial conjecture that topic models may be represented in a generalised form. This could be shown by defining model scope and the generic NoMM representation, and based on this simplifications for different aspects of work with topic models could be derived. This includes that important model properties such as data likelihood and update rules for inference methods have more direct correspondences with NoMM structures than with Bayesian networks. As a result, models can be constructed on a modular basis, from which a design method has been derived that may guide the practitioner to viable model structures.

Chapter outline. In the following, these and other results of this thesis are discussed in more detail, starting with a review of the contributions in Section 11.2. Section 11.3 will then propose future directions.

11.2 Contributions

As outlined above, there are various areas where this thesis could make contributions. The next sections summarise the main results by category.

Topic meta-model and *Networks of Mixed Membership (NoMMs)*: One of the central contributions of this thesis is the formalisation of topic models as a generic class of models, in connection with a derivation of their common properties. To represent topic models generically, a topic "meta-model" has been proposed that is based on an interpretation of the models as higher-order mixtures. The relation of this approach to existing models in the literature was presented and it was shown how the scope of models may be extended by introducing variants of the Dirichlet–multinomial distribution pair that forms the standard case. Furthermore, a graphical notation has been proposed to represent the structure of topic models, which may be seen as a domain-specific, more compact alternative to Bayesian networks: Networks of mixed membership (NoMMs).

This contribution appears to be the first principled approach to "chart the territory" of topic modelling from a high-level viewpoint, taking into account common model structures and their probabilistic properties rather than the properties of each model individually.

Generic inference methods: Inference allows training of models from data and is the key to their practical usage. For two approximate Bayesian inference methods – Gibbs sampling and variational inference – generic derivations of the update formulas have been contributed, and it has been shown that there exist direct mappings between NoMM structures and inference algorithms. In an empirical comparison between both inference methods for a range of topic models, the Gibbs sampler prevailed, measuring test-data perplexity and convergence time.

The novel generic update rules encapsulate the knowledge needed to derive inference for concrete models, avoiding extensive calculations from scratch (the typical approach).

Generic scalable Gibbs sampling: The scalability behaviour of Gibbs samplers has been investigated, especially for more complex models with multiple dependent latent variables, answering the question to what extent scaling methods published for LDA can be transferred to more general model structures. Furthermore, a novel method for scalable sampling has been proposed and the influence of dependencies between latent model variables (a main scalability factor) has been investigated.

An experimental study has shown that generic serial and (to a larger extent) parallel methods significantly increase the performance of the Gibbs sampler, in particular for more complex models. Another important empirical result is that resolving dependencies between latent variables does not degrade model parameters and topic quality significantly while strongly reducing convergence time. High speed-up figures could also be achieved by combining serial and parallel acceleration methods with this approach.

Besides proposal of a novel algorithm, an important contribution is to generalise prior work from LDA to generic NoMMs. The results widen the scope that especially more complex topic models can be used in – and on larger data.

Gibbs meta-sampler: As a direct application of the generic Gibbs sampler and its scalable extensions, a program code generator has been developed. Based on a novel domain-specific NoMM language, this Gibbs "meta-sampler" produces source code for Gibbs samplers in the programming languages Java and C. It thus short-cuts much of the development and testing

work typically necessary for model implementation, especially when it comes to developing complex model structures and/or specific scalable approaches that require more complex code than ordinary Gibbs sampling kernels. The generated samplers are ready to run in experiments and at the same time have legible source code to facilitate manual adjustment and optimisation.

The novelty of the meta-sampler is its focus on topic models, as opposed to more general Bayesian network structures in related work. This focus allows exploitation of the model properties found for NoMMs.

NoMM typology and design method: From an extensive study of the literature on topic models, a typology of NoMM sub-structures has been gathered that considers aspects like node distributions, branching and mixture component selection. Motivated by analysis of the completeness of the typology, a novel NoMM sub-structure type has been proposed.

Connecting the typology with generic inference, a design method has been derived that uses NoMM sub-structures as a "library": Models are assembled from the sub-structures, and their probabilistic properties can be tracked at each construction step (in particular: Gibbs full conditional distributions and data likelihood). This construction may be seen in front of the Gibbs meta-sampler, narrowing down what NoMMs to implement given an analysis task and assumptions on the data.

Prior work has performed analysis of topic models mainly on the basis of related applications, and the novel approach in this thesis is the viewpoint of model sub-structures, also as re-usable building blocks for model design.

Modelling of virtual communities using the *AMQ model*: The thesis contributes an abstract characterisation of knowledge in virtual communities that can be used to describe data structures and tasks for knowledge discovery, information retrieval, visual exploration and other fields. The proposed AMQ model posits that virtual community knowledge can be expressed by three types of entity and their interrelations: actors (i.e., people or agents), media (i.e., documents and other information sources) and qualities (units of knowledge representation). Minimal modelling commitment keeps a high degree of flexibility towards scenarios, yet capturing a large part of the knowledge in virtual communities.

The AMQ model appears to be the first model that formalises the structure of community scenarios and tasks at this generic level. It also serves as a "front-end" for the design method.

Expert-finding models: To demonstrate the other results of this thesis in an application context, several models have been designed in the area of expert finding. The novel expert–tag–topic models (ETT1–3) extend the state of the art for retrieval of experts from documents towards additional semantic tagging information or (multi-)classification data. Experimentally, two of the models could be shown to outperform the baseline author–topic model in terms of retrieval performance (using word and tag queries to find experts) and topic coherence (the semantic relation of high-ranking topic terms). As suggested by an empirical analysis of the topic clustering structure, the good performance can be attributed to the influence of the incorporated tagging information. Finally, it has been shown how ETT models can be integrated into a visual community browser that allows interactive exploration and search of the community's network structure.

In addition to the ETT models, the contribution in this thesis is to show exemplarily how a model may be constructed from assumptions on the data, using the proposed design method.

11.3 Future directions

Based on the achieved results, there are various directions that may be investigated in future work. They are presented in thematic groups in the following.

Applications and NoMM structures: Future work may look at model structures and applications not focussed upon in this thesis. Of special interest here is the combination of discrete data with multi-media features, and for their predominantly Gaussian feature distributions (or mixtures thereof) non-discrete NoMM nodes come into play (type N4 in Chapter 5). NoMM-based models and the associated workflow may for instance complement work on multi-modal classification, such as [Neumayer & Rauber 2007, Mayer & Rauber 2011] where audio signals and textual lyrics data are combined to classify music into genres. Other applications include visual data analysis, extending work like [Wang et al. 2009a, Barnard et al. 2003] where images are mined in conjunction with tagging or caption data, and more generally computer vision [Cao & Fei-Fei 2007, Lacoste-Julien et al. 2008, Philbin et al. 2008].

Furthermore, nodes with non-Dirichlet priors may be looked at (type N3), especially non-parametric distributions, which will be discussed further below, or structures that integrate side-conditions into the topic learning process, such as topic-specific PageRank in the TopicFlow model [Nallapati et al. 2011] to learn topical document importance in citation networks.

Another interesting direction is that of dynamic and sequence-based models, extending work like [Heyer et al. 2009, Wang et al. 2008, Gerrish & Blei 2010, Pruteanu-Malinici et al. 2010]. This would solve a current limitation of the NoMM meta-sampling approach that cannot directly model temporal or sequence information (where model structures are repeated per time interval).

NoMMs and formal semantics. With its inference based on first-order logic, formal semantics may be considered a sibling approach to latent semantics whose inference is based on probabilistic methods in this thesis (cf. Section 3.2). Using NoMM design and inference methods, both methodologies may be brought into a symbiotic relation: Ontology information may be used to improve topic quality by adding ground knowledge into model inference. Vice versa, latent semantics may be used to extract ontology structure and concept–instance associations from raw data, improving current ontology learning approaches, e.g., [Maedche & Staab 2009, Navigli et al. 2011]. For this, one may intuitively default to non-terminal NoMM edges to represent concepts in formal semantics: A NoMM edge may propagate formal concept indicators and its parent node learns their distribution from the data. The result of such inference is then a set of statements based on distributions over ontology concepts that may be transformed into an ontology proper (which typically requires human intervention) or may model the inherent uncertainty explicitly, e.g., in the extended ontology representation PR-OWL [Costa & Laskey 2006] that supports reasoning with uncertainty [Laskey 2008].

However, the way in which axiomatic statements on formal concepts map to particular NoMM structures and how they may be efficiently learnt from unstructured data are open questions. Using NoMMs and the typology established in Chapter 5 may allow to bootstrap answers by re-using prior work that solves inference on particular axiom types. For example, concept hierarchies (i.e., subclass axioms) may be expressed as structured Dirichlet priors (node type N3A in Chapter 5) as in [Andrzejewski et al. 2009], which incorporates a known taxonomy or ontology into a concept learning mechanism but is not an efficient method to learn the taxonomy itself. For this, models along the lines of hierarchical PAM [Li et al. 2007a] and hierarchical LDA [Blei et al. 2004] may

be better suited. Using regression methods (with N5+E4 structures) inspired by the supervised topic model [Blei & McAuliffe 2007] may help learn concept–instance relations and may be extended to relational models based on, e.g., [Chang & Blei 2009, Kemp et al. 2004] to learn class–class relations. Extension and combination of such successful prior work leverages the efficiency of the design process in Chapter 9 because different axioms of a single ontology may map to a multitude of specialised NoMMs that learn or apply particular ontology statements.

Non-parametric priors and NoMM polymorphism: As has been outlined in Chapter 5, NoMM structures may be generalised to parametric (finite) and non-parametric versions of the corresponding models, in the latter case allowing estimation of the data dimensionalities. This is a very active research field, and besides the Dirichlet process especially the more general Pitman–Yor process [Buntine & Hutter 2010] is interesting as a prior, as it has been shown to capture the power-law frequency distributions of language and network data explicitly.

The elegant aspect about a non-parametric extension is that the NoMM representation may become a polymorphism: A single NoMM structure corresponds to a Bayesian network for the parametric and one for the non-parametric case (cf. Chapter 3 for the differences). The behaviour of the underlying probabilistic models should be largely equivalent. Based on the decision to estimate dimensionalities or not, the Gibbs meta-sampler generates the respective algorithm.

Judging from the code structure of the two-level hierarchical Dirichlet process [Teh et al. 2006] (as presented in [Heinrich 2011a] to prepare this future work), the extension of the current Gibbs meta-sampling design is indeed not overly complex in practice; one may follow the methodology outlined in Chapter 6. Moreover, recently [Chen et al. 2011] have reported on a novel sampling technique for the hierarchical Pitman–Yor process with moderate code complexity and often better convergence properties. However, complex model structures with more intricate dependencies between the Dirichlet processes (cf. [Canini & Griffiths 2011]) will require some re-design: As outlined in Chapter 5, the introduction of infinite dimensions cannot be done locally but requires changes all along the NoMM model structures and estimators.

Generic collapsed variational inference: As an alternative to Gibbs sampling and variational inference proposed in Chapters 6 and 7, collapsed variational Bayes (CVB) [Teh et al. 2007, Sung et al. 2008] may be considered for an alternative NoMM inference engine that complements the Gibbs meta-sampler. As has been shown in [Teh et al. 2007; 2008, Asuncion et al. 2009], this technique is comparable to Gibbs sampling in terms of perplexity reduction. It has structurally similar update equations to Gibbs full conditionals, but showed slightly slower convergence in terms of absolute processing time.[1] The clear advantage of CVB over Gibbs sampling is its deterministic character. This allows monitoring of convergence of the algorithm with certainty rather than being dependent on statistical indicators.

Architecture-aware Gibbs meta-sampler: A very promising research direction is to extend the code generation to high-performance computing architectures using the findings of Chapter 8 on scalable sampling, specifically for massively parallel computing (graphics processors) and reconfigurable hardware (FPGAs). This allows model-driven solutions to complex implementations. Usually software for such architectures is hard to optimise for efficiency, and the vision is to use the NoMM language to generate implementations optimised on a generic level.

[1] [Asuncion et al. 2009] report on a parallel version of CVB being faster by comparing this to the serial Gibbs sampler. In the light of the results on fast sampling presented in Chapter 8, this should be re-evaluated.

A variant of such an approach has already proven highly promising: The C code generated by the Gibbs meta-sampler may be directly post-processed into architecture-specific code for FPGAs or GPUs, using architecture-aware code transformation software based on novel code partitioning and hardware mapping algorithms [Rashid et al. 2009, Bertels et al. 2010]. This has been demonstrated for a class of real-time audio algorithms (arrays of convolution operators) that are used for directional sound recording (microphone beamforming) and high-quality spatial sound rendering (wave-field synthesis) [Heinrich et al. 2011].

Improved design method: The design method outlined in Chapter 9 may be extended to develop a more automatic way of topic model construction that does not rely on NoMM structures as an input but constructs the model from high-level criteria covering the available data, the expected co-occurrences and the desired outputs. If the data cover communities, a formalism to express these high-level model attributes may be found in the AMQ model. Another direction is that of directly linking ontology structures to NoMM models, either to incorporate existing formal semantics and/or to create ontology learning models.

Taking this idea further, the design problem may even be considered a recommendation problem, with the goal to suggest suitable model designs given a particular data set and task specification. This may be achieved using the Gibbs meta-sampler (Chapter 6) and automatically optimising NoMM structures, based on a reference data set and the relevant quality metrics as objective functions. This optimisation may also be interactive, for instance rendering the topic structure graphically for the given data using embedding techniques like [Iwata et al. 2004]. In connection to this, the Gibbs meta-sampler may be extended to a graphical editor as an easy-to-use front-end for model construction.

The structure of AMQ networks: The AMQ model opens a new perspective on the information in community-based retrieval and knowledge discovery scenarios, and properties of the combined AMQ networks (and the data of real-world communities instantiating them) could be investigated as generalisations of co-occurrence networks.

A study to characterise AMQ networks may draw from work like [Mehler 2008] where a mathematical model for social ontologies is defined that is readily applicable to other complex networks. This model focuses on (partly novel) graph metrics that may be expressive also for AMQ networks (e.g., Zipfian bipartivity, cohesion, child imbalance and tree-ness). Quantifying this model for AMQ instances may provide insights in the complex co-occurrence and small-world properties within and between their different modalities – also in comparison with the empirical results given in [Mehler 2008] for social tagging scenarios (that themselves are related to the semantic tagging scenario studied in Chapter 10). Properties like Zipfian bipartivity in turn may be used to develop novel retrieval methods, for instance in context with the Pitman–Yor process priors discussed above that model power-law distributions specifically.

Community search engine: Finally, one of the most important goals is to use topic models for practical retrieval and knowledge discovery in virtual community data. The envisioned community search engine may extend the graph-based community browser outlined in Section 10.5.

Among the extensions to the ETT models, especially incorporation of citation structure into the querying process is of interest, for which state-of-the-art models like TopicFlow [Nallapati et al. 2011] or the relational topic model (RTM [Chang & Blei 2009]) are promising points of departure.

However, a more generic and visionary approach is to allow users to define criteria for knowledge discovery or retrieval tasks freely, and the engine creates task-specific topic models on the fly that execute the query. In all steps of this process, substantial research questions are raised, from the way queries are formulated (where AMQ schemas may play a role for community scenarios) to the model generation process (using simplified or even fully automated NoMM design methods) and finally results visualisation (that may be based on the proposed interactive graph browser).

Appendix

Appendix A

Notation and abbreviations

In this appendix, the notations and abbreviations used in this thesis are defined for reference, complementing the descriptions in the text.

A.1 Notation

In this thesis, all symbols are defined in the text preceding or following the respective formulas. In addition, some general conventions are followed to keep the presentation consistent.

Regarding variable symbols, in general Greek letters are used to denote real numbers and Latin letters for integers; exceptions apply to meet conventions in literature. Upper-case roman characters preferably denote constants, but may also denote sets, especially in formulations for generic models. Specific preferences to denote re-appearing notation are contained in the table in Fig. A.1.

Vectors are written in lower-case arrow notation, $\vec{x}, \vec{\alpha}$, which is preferable over bold-face notation as regular and bold Greek symbols are difficult to distinguish. Accordingly, matrices are written upper-case underline notation $\underline{A}$. However, sets that collect parameters (such as Θ or H in generic variational and Gibbs algorithms) are simply written upper-case and without specific markup. The type of these structures becomes clear from the context and surrounding text.

In formulations for generic models, the convention is to use upper-case symbols to refer to a complete sequence, e.g., $X = \{x_i\}_{i \in I}$, while lower-case refers to one of its elements, e.g., x_i, even if it corresponds to multiple dependent variables, $x_i \equiv \{x_i^\ell\}_\ell$. The re-use of the $\{\cdot\}$ operator to concatenate set, vector and matrix elements is clarified in context.

Furthermore, symbols with a superscript $*^\ell$, etc. are assumed specific to a particular mixture level. Sets of mixture levels have a set as superscript, e.g., a dependency group d. To simplify this notation, we define $*_s^\ell = *_{s^\ell}^\ell$ for any subscript s. Moreover, we introduce a bracket notation that indicates level ℓ for all of its contents in which no explicit level is stated: $[*]^{[\ell]}$. Other conventions and symbols are summarised in the table in Fig. A.1.

$z :: x \times y$	z is function of x combined with y (in AMQ representation)
$y \Leftarrow x$	y is obtained from x (in AMQ representation)
$\uparrow\cdot$	parent variable operator
$\neg i$	leave-i-out operator
$\cdot'$	prime: test set, query
$\otimes$	Hadamard (element-wise) matrix product
$\oplus$	sum of indexed elements: $n_{i\oplus j} = n_i + n_j$ (used with)
$\langle X\rangle, \langle X\rangle_{q(X)}$	expectations of X under distributions $p(X)$ (default) and $q(X)$
$\vec{x}, \underline{A}$	vector and matrix; in matrix algebra, vectors $\vec{x} = \{x_i\}_i$ are column, $\vec{x}^\top$ row vectors
$\{x_i\}_{i=1}^N, \{\vec{x}_j^\top\}_{j=1}^M$	set or vector with elements x_i, matrix with rows $\vec{x}_j^\top$
$a \in A \subset \mathcal{A}$	actor (instance, subclass, class; in AMQ representation)
$\alpha, \vec{\alpha}$	Dirichlet parameters, hyperparameters (scalar or undetermined; vector)
A	set of hyperparameters (upper-case α)
$\mathrm{B}(\alpha_1, \alpha_2)$	Beta function (upper-case β)
$\mathrm{BC}\{X\|\|Y\}$	Bhattacharyya coefficient between random variables X and Y
$\mathrm{D}\{x\|\|y\}$	distance between quantities x and y
$\Gamma(x)$	Gamma function, $\Gamma(n+1) = n!$ for natural numbers
$\Delta_K(\alpha), \Delta(\vec{\alpha})$	Dirichlet partition function (scalar and vector, = multidim. $B(\alpha_1, \alpha_2)$), see (3.16) and (3.18)
$\delta(x)$	Kronecker delta (discrete) or Dirac distribution (continuous), $\delta(x) = 1$ if $x = 0$, else 0
$\vec{e}_i$	unit vector with the ith element 1
$\mathrm{H}\{X\}$	entropy of random variable X (upper-case η)
$\mathrm{H}^2\{X\|\|Y\}$	squared Hellinger distance between random variables X and Y
$f(\uparrow x, i)$	mapping of parent variables and indices to hyperparameters
f, g_t, h_{tu}	likelihood function, gradient and Hessian elements
$g(\uparrow x, i)$	mapping of parent variables and indices to components
h_i^ℓ, H^ℓ	hidden variable element and sequence (level-dependent)
$H, *^H$	set of hidden variables, set of $*$ for all hidden levels
θ	generic parameter or set of parameters (opposed to specific ϑ)
$\vec{\vartheta}_k, \Theta$	generic mixture parameters and sets
$\vec{\vartheta}_*, \underline{\Theta}$	mixture parameters and sets (in some concrete models, $* = \{m, k, (m, x), \dots\}$)
i	sequence index (generalises doc. and term indices, m and n, etc.); general iteration index
I	sequence of index i: $i \in I$
$j, j(k), J$	index of a hyperparameter or component group, $f(\uparrow x, i)$, number of groups
$\mathrm{JS}\{X\|\|Y\}$	Jenson–Shannon distance between random variables X and Y
$\mathrm{KL}\{X\|\|Y\}$	Kullback–Leibler divergence between random variables X and Y
k, K	component index, number of components
ℓ, L	mixture level indicator, number of mixture levels
$*^\ell, [*]^{[\ell]}, *_k^\ell$	notation for level index, $* = \{\alpha, A, f(\cdot), g(\cdot), h, H, i, I, j(\cdot), k, K, n_{\cdot}, N, t, T, \vartheta, \Theta, x, X\}$; subscripts like k are assumed level-dependent k^ℓ, distinction from exponents by context
$\mu_t(\vec{\alpha})$	Dirichlet expectation of $\log \alpha_t$, see (7.9)
$m \in M \subset \mathcal{M}$	medium (instance, subclass, class; in AMQ representation)
m, M	document index and count
n, N_m	term index and count (document-specific)
N	total count
$\vec{n}, n_t, n_{k,t}$	vector of counts, count for occurrence of t and joint k, t
$\vec{\xi}_k, \Xi$	variational Dirichlet parameters
$q \in Q \subset \mathcal{Q}$	quality (instance, subclass, class; in AMQ representation)
$q(\cdot)$	variational distribution
$q_i^\ell(k, t\|\alpha), q(k, t)$	NoMM node Gibbs term $\Delta(n_{k,t} + \alpha)/\Delta(n_{k,t,\neg i}^\ell + \alpha^\ell)$
R	number of samples
$R(\cdot, \cdot)$	relation in AMQ model, e.g., *authors*(a, m)
s_k, S; s_l^k	stick-breaking lengths and set; segment length for bound-based sampling
S_K	K-dimensional simplex (embedded in $\mathbb{R}^{K+1}$)
t, T	topic value, value range set (generic models); t term
v_i^ℓ, V^ℓ	visible variable element and sequence (level-dependent)
V	set of visible variables, set of $*$ for all visible levels
v	visible term value (to distinguish from hidden $t^\ell \in \vec{t}$)
$\mathrm{Var}\{X\}$	variance of X
$\vec{w}$	word sequence (concrete models)
$w^d(x, y), w^u(x, y)$	directed and undirected weighted function between node x and y in an AMQ model
W	number of tokens (words) in a corpus
x_i^ℓ, X^ℓ	variable element and sequence (level-dependent)
X	set of all variables (generic models)
$\vec{x}, \vec{y}, \vec{z}$	topic sequence (concrete models)
$\vec{\varphi}_*, \underline{\Phi}$	mixture parameters and sets (in some concrete models, $* = \{k, (x, y), \dots\}$)
$\Psi, \underline{\psi}_u, \psi_{u;\vec{t}}^{\overline{\ell}}$	variational multinomial ("topic field") over complete model and corpus, for observation index u over all hidden states $\vec{t}$, its marginal for level ℓ
$\Psi(x)$	digamma function

Figure A.1: Table of symbols.

A.2 Abbreviations

AMQ	actors, media, qualities (model)
ARS	adaptive rejection sampling
ATM	author–topic model
Beta	beta distribution
BN	Bayesian network
BNF	Backus–Naur form
cdf	cumulative density function
cgf	cumulant-generating function
CGS	collapsed Gibbs sampler
CRP, CRF	Chinese restaurant process, franchise
CTM	correlated topic model
DAG	directed acyclic graph
Dir	Dirichlet distribution
DP	Dirichlet process
EM	expectation-maximisation algorithm
ETT	expert–tag–topic model
FSA	full-state approximate (parallel sampling method)
FSE	full-state exact (parallel sampling method)
Gam	Gamma distribution
GEM	Griffiths–Engen–McCloskey distribution
GMM	Gaussian mixture model
hPAM	hierarchical PAM
HDP	hierarchical DP
i.i.d.	independent and identically distributed (sampling)
IR	information retrieval
LDA	latent Dirichlet allocation
LDCC	latent Dirichlet co-clustering
LoCC	lines of commented code
LR	linear regression
LSA, LSI	latent semantic analysis, indexing
MCMC	Markov-chain Monte Carlo (algorithm)
MIMD	multiple instructions, multiple data (parallelisation scheme)
Mult	multinomial (distribution)
MAP	maximum a posteriori (estimation)
ML	maximum likelihood (estimation)
$\mathcal{N}$	normal distribution
NB	naïve Bayes (classifier)
NoMM	network of mixed membership
OO	object-oriented (methodology)
PAM, PAM4	pachinko allocation model (generic and 4-level PAM)
PCA	principal components analysis
pdf	probability density function
PLSA, PLSI	probabilistic LSA, LSI
pmf	probability mass function
PYP	Pitman–Yor process
r.v.	random variable
SBP	stick-breaking prior
SIMD	same instruction, multiple data (parallelisation scheme)
SSA	split-state approximate (parallel sampling method)
SVD	singular value decomposition
TC	topic coherence
VB	variational Bayes
VI	variation of information (distance) or variational inference
VSM	vector-space model
w.r.t.	with respect to

Figure A.2: Table of abbreviations.

Appendix B

Exponential-family distributions and conjugacy

In Chapter 3, the distributions relevant for this thesis have been introduced without formally giving the mathematical background to conjugacy in exponential families. This was done to make the presentation more focussed. However, to provide some context, the viewpoint of exponential families is given here.

B.1 Exponential families

There is a large class of probability distributions that share a common form of their densities as well as other properties: exponential families [Wainwright & Jordan 2003]. The majority of distributions common in practice belong to it, e.g., Gaussian, gamma, Poisson, geometric, Bernoulli, binomial, multinomial, beta or Dirichlet. The general form of such distributions can be written as:

$$p(x|\theta) = h(x) \cdot \exp[\eta^\top(\theta) \cdot T(x) - A(\theta)] \tag{B.1}$$

where x is the value of a random variable X, θ is the parameter (or parameter set) and $h(x)$ some function of the random value, while $\eta(\theta)$ is called the natural parameter, $T(x)$ the sufficient statistics and $A(\theta)$ the log partition function or cumulant-generating function (cgf). Parameters and values can be vectors.

Pulling the log of the multinomial (discrete) distribution into the exponential, the exponential-family notation of this distribution becomes:[1]

$$\begin{aligned} \text{Mult}(x|\vec{\vartheta}) &= \textstyle\sum_k \vartheta_k \delta(x-k) \\ &= 1 \cdot \exp[\log \vec{\vartheta}^\top \cdot \delta(x-\vec{k}) - 0] \,, \end{aligned} \tag{B.2}$$

where we define $\delta(x-\vec{k})$ to map to a vector with elements $\delta(x-k)\ \forall\ k \in [1, K]$. In a similar

[1] Note the difference between θ, a generic parameter, and $\vec{\vartheta}$, a specific parametrisation of the multinomial.

manner, for the Dirichlet distribution the exponential-family notation becomes:

$$\mathrm{Dir}(\vec{\vartheta}|\vec{\alpha}) = 1 \cdot \exp[(\vec{\alpha} - 1)^\top \cdot \log \vec{\vartheta} - \log \Delta(\vec{\alpha})] \;, \tag{B.3}$$

where $\log \Delta(\vec{\alpha}) = \sum_k \log \Gamma(\alpha_k) - \log \Gamma(\sum_k \alpha_k)$. Analogously, this notation can be given for the symmetric version of the Dirichlet:

$$\mathrm{Dir}(\vec{\vartheta} \,|\, \alpha) = 1 \cdot \exp[(\alpha - 1) \cdot [\textstyle\sum_{k=1}^{K} \log \vartheta_k] - \log \Delta_K(\alpha)] \;, \tag{B.4}$$

where $\log \Delta_K(\alpha) = K \log \Gamma(\alpha) - \log \Gamma(K\alpha)$.

An important property of exponential-family distributions is that the derivatives of the cgf $A(\vartheta)$ w.r.t. the natural parameters $\eta(\vartheta)$ are equal to the cumulants (including mean $\mu = \varkappa_1$ and variance $\sigma^2 = \varkappa_2$) of the sufficient statistics $T(x)$ w.r.t. the distribution (therefore the name cumulant-generating function). For the Dirichlet, the expectation $\langle T(x) \rangle_{p(x)} = \partial / \partial \eta(\vartheta)\; A(\vartheta)$ leads to a result relevant to the variational approaches in Chapter 7:

$$\langle \log \vartheta_k \rangle_{\mathrm{Dir}(\vec{\vartheta}|\vec{\alpha})} = \frac{\partial A(\vec{\alpha})}{\partial(\alpha_k - 1)} = \Psi(\alpha_k) - \Psi(\textstyle\sum_k \alpha_k) = \mu_k(\vec{\alpha}) \tag{B.5}$$

$$\langle \log \vartheta_k \rangle_{\mathrm{Dir}(\vec{\vartheta}|\alpha)} = \frac{\partial A(\alpha)}{\partial(\alpha - 1)} = \Psi(\alpha) - \Psi(K\alpha) \;. \tag{B.6}$$

B.2 Conjugate prior distributions

In Bayesian inference, the difficulty of finding the posterior distribution over the parameters given some data largely depends on the choice of prior distribution, which may or may not allow to find the marginal likelihood in the denominator of (3.8), i.e., the integral (or sum) in (3.24). Choosing the conjugate prior of a given likelihood function solves this problem elegantly [Raiffa & Schlaifer 1961].

Any prior that is conjugate to a given likelihood function is only re-parametrised by the data encoded in the likelihood and maintains its algebraic form as the posterior. This makes conjugate priors algebraically particularly convenient because also the form of the problematic marginal likelihood term (partition function) is shared between the posterior and known prior distribution. In addition, intuitive interpretations of the prior parameters (hyperparameters) are possible. For example, in the results for discrete observations in (3.23) and (3.28), hyperparameters could be interpreted as pseudo-counts.

A conjugate prior should be a distribution over the (likelihood) parameters θ with some parameter a. Therefore, the exponential-family form of the prior is:

$$p(\theta \,|\, a) = h(\theta) \exp[\eta^\top(a) \cdot T(\theta) - A(a)] \;, \tag{B.7}$$

while that of the likelihood of a given set of data points $\{x_i\}_{i=1}^N$ is:

$$p(\{x_i\} \,|\, \theta) = \textstyle\prod_{i=1}^{N} h(x_i) \cdot \exp[\eta^\top(\theta)\, (\sum_{i=1}^{N} T(x_i)) - NA(\theta)] \;. \tag{B.8}$$

Using Bayes' rule, the posterior becomes:

$$\begin{aligned} p(\theta \,|\, \{x_i\}, a) &\propto p(\{x_i\}|\theta) \cdot p(\theta|a) \\ &= \exp[\eta^\top(a)\, T(\theta) + \eta^\top(\theta)\, (\textstyle\sum_{i=1}^N T(x_i)) - NA(\theta) - A(a)] \,. \end{aligned} \tag{B.9}$$

As the values of the prior θ are the parameters of the likelihood, an intuitive approach is to find a form of the prior sufficient statistics $T(\theta)$ that allows factoring with the natural parameter $\eta(\theta)$ as well as the cgf $A(\theta)$ of the likelihood. We set $T(\theta) = \{\eta(\theta), A(\theta)\}$, which with the corresponding $a = \{a_\eta, a_A\}$ leads to a valid exponential family of the prior:

$$p(\theta \,|\, \{x_i\}, a) \propto \exp[(\eta(a_\eta) + \textstyle\sum_{i=1}^N T(x_i))^\top \eta(\theta) - (\eta(a_A) + N)A(\theta) - A(a)]$$

$$p(\theta|a) = \exp[\eta^\top(a_\eta)\eta(\theta) - \eta(a_A)A(\theta) - A(a)] \,, \tag{B.10}$$

$$A(a) = \log \textstyle\int \exp[\eta^\top(a_\eta)\eta(\theta) - \eta(a_A)A(\theta)]\, \mathrm{d}\theta \,. \tag{B.11}$$

Dirichlet–multinomial case. For multinomial parameters in the form of (B.2) and concrete prior parameters $\vec{\alpha}$, applying (B.10) and (B.11) results in:

$$\begin{aligned} p(\vec{\vartheta} \,|\, \vec{\alpha}) &= \exp[\eta^\top(\vec{\alpha}_\eta) \cdot \log \vec{\vartheta} - \eta(\alpha_A) \cdot 0 - A(\vec{\alpha})] \\ &= \exp[\eta^\top(\vec{\alpha}) \cdot \log \vec{\vartheta} - A(\vec{\alpha})] \end{aligned} \tag{B.12}$$

$$\begin{aligned} A(\vec{\alpha}) &= \log \textstyle\int \exp[\eta^\top(\vec{\alpha}) \log \vec{\vartheta}]\, \mathrm{d}\vec{\vartheta} \\ &= \log \textstyle\int \prod_{k=1}^K \exp[\eta(\alpha_k) \log \vartheta_k]\, \mathrm{d}\vec{\vartheta} \\ &= \log \textstyle\int \prod_{k=1}^K \vartheta_k^{\eta(\alpha_k)}\, \mathrm{d}\vec{\vartheta} \,. \end{aligned} \tag{B.13}$$

Setting $\eta(\alpha_k) = \alpha_k - 1$, the argument of the log becomes just the form of the Dirichlet integral of the first kind, which results in the Dirichlet partition function $\Delta(\vec{\alpha})$. Consequently, the complete form of the Dirichlet distribution (3.16) or (B.3) is recovered. This result is analogous for the symmetric case.

Mixture of Dirichlets. A mixture of Dirichlet priors can be shown to result in a multinomial likelihood, as well. A mixture of Dirichlet is defined as $p(\vec{\vartheta} \,|\, \vec{\pi}, \{\vec{\alpha}_j\}) = \sum_j \pi_j \mathrm{Dir}(\vec{\vartheta} \,|\, \vec{\alpha}_j)$, which is in fact similar to introducing $\vec{\pi}$ as a prior on the hyperparameters $\{\vec{\alpha}_j\}$. Because the average of several multinomial distributions is identical to a single distribution with the weights averaged (which is special among probability distributions), the mixture of multinomial parameters can be expressed as a single parameter itself: $\vartheta_k = \sum_j \pi_j \vartheta_{k,j}$ with $\{\vartheta_{k,j}\} \sim \mathrm{Dir}(\vec{\vartheta}_j \,|\, \vec{\alpha}_j)$. The remaining unknown, the mixture weight $\vec{\pi}$, can have a conjugate Dirichlet prior again, $\vec{\pi} \sim \mathrm{Dir}(\vec{\pi} \,|\, a_\pi)$, which, in the terminology established in this thesis (see Chapter 4), just acts as an additional mixture level, this time controlling the hyperparameters.

The resulting prior distribution in principle allows to approximate any prior density on the Dirichlet parameters without sacrificing the advantages of conjugacy [Dalal & Hall 1983].

Appendix C

Implementation details

This appendix gives some details of algorithm implementations that were not directly relevant to understand the concepts of the thesis but may give additional insights and reference. It specifically complements the presentation in Part II of the thesis.

C.1 Discrete random variate generation

In Monte Carlo simulation of discrete high-dimensional probability distributions, especially with Gibbs sampling, the generation of discrete random variates from non-uniform distributions plays a central role. Put in plain words, the task is to sample an element from a discrete probability distribution with probability mass function (pmf)

$$f(k \,|\, \vec{\vartheta}) = \{\vartheta_1, \ldots, \vartheta_K\}, \qquad \vartheta_k \in [0, 1]\,. \tag{C.1}$$

Such pmfs may or may not be normalised, and the general case of unnormalised parameter vectors $\vec{\vartheta} : \sum_k \vartheta_k = a$ is considered here because of its practical importance. In the following, we denote unnormalised quantities with f instead of p.

For random number generation from a known distribution, the inversion method is one of the most popular: Its idea is that the cumulative distribution function (cdf) of any probability density or mass function ranges uniformly over the interval $[0, 1]$ for the normalised case and $[0, a]$ for the unnormalised case. If u is a uniform random number on $[0, a]$, then a random number X from a distribution with cdf F is obtained using its inverse: $X = F^{-1}(u)$ [Gentle 2003, p. 102f].

An alternative to the inversion method is the alias method, which has been specially developed for discrete distributions [Walker 1977, Kronmal & Peterson 1979]. Because in Gibbs sampling the weights of the mixtures and the aliases constantly change and would need to be recomputed in the inner loop of the algorithm, the method is deemed too complex for this application.

Independent sampling. Using the inversion method, in the discrete case considered here, X corresponds to the element index k, and a simple generation algorithm is to (1) create an unnormalised cdf from the pmf,

$$F(k \,|\, \vec{\vartheta}) = f(\ell \leq k \,|\, \vec{\vartheta}) = \textstyle\sum_{\ell=1}^{k} \vartheta_\ell\,, \tag{C.2}$$

then (2) generate a uniform random sample, $u \sim \mathcal{U}[0,a]$, and (3) perform a search through the cdf values for the index k satisfying $F(k-1) < u \leq F(k)$. Because the search space is ordered, binary search can be used. Alternatively, the search space might be sorted by criteria like probability mass and some table lookup method may be used as described in [Gentle 2003, p. 104ff].

Block sampling. As an extension to the case of independent samples, a random variate from a joint discrete distribution can be considered. This case is important if several discrete random variables $k_n,\ n \in [1,N]$ are statistically dependent on each other. Sampling then is done as a block of N variates at a time. Using vector notation to collect quantities from index dimensions 1 to N, e.g., $\vec{k} \triangleq \{k_n\}_{n=1}^{N}$, the discrete parameters of the joint distribution are given as

$$f(\vec{k}{=}\vec{x} \,|\, \underline{\Theta}) = \vec{\vartheta} \ . \tag{C.3}$$

The problem in this case is that the corresponding cdf is not directly usable for the inversion method because results may be ambiguous. To identify variables directly, they can be summed along a given complete path through the index space of $\vec{k}$, which yields a cumulative function along one index dimension (which is not equal to the cdf of f).

A straight-forward path through the index space can be obtained from standard mapping of multi-dimensional arrays in linear memory, which formally corresponds to the following index transformation:

$$c = T(\vec{k}) = \textstyle\sum_{n=1}^{N} \left(k_n \prod_{i=1}^{n-1} K_i \right) \ . \tag{C.4}$$

Based on this, the path cdf is analogous to the independent sampler:

$$F(\vec{k}{=}\vec{x} \,|\, \underline{\Theta}) = F(c{=}T(\vec{x}) \,|\, \underline{\Theta}) = f(\ell \leq c \,|\, \underline{\Theta}) = \textstyle\sum_{\ell=1}^{c} \vartheta_{T^{-1}(\ell)} \tag{C.5}$$

where the inverse of the index transformation, $\vec{k} = T^{-1}(c)$, is an iterative algorithm, starting with $c_N = c$ and $M_N = \prod_{i=1}^{N-1} K_i$ and running from index $n = N$ down to $n = 1$:

$$c_{n-1} = c_n \bmod M_n \ ; \quad k_n = \frac{c_n - c_{n-1}}{M_n} = c_n \operatorname{div} M_n \ ; \quad M_{n-1} = \frac{M_n}{K_{n-1}} \ . \tag{C.6}$$

The path cdf can be assembled from the parameters $\underline{\Theta}$ just by incrementing indices without address computations. After obtaining a random sample $u \sim \mathcal{U}[0,a]$ where $a = F(\vec{k}{=}\vec{K})$, a tuple $\vec{k} = T^{-1}(c)$ is searched whose linear correspondence c satisfies $F(c-1) < u \leq F(c)$. This search space is of size $K_1 \times K_2 \times \cdots \times K_N$. After searching the sampled linear index c, the index lookup in (C.6) is used to identify the sample block $\vec{k}$.

In real-life multidimensional Gibbs samplers, this approach can be elegantly solved in the meta-sampler (see Chapter 6 and next appendix section) by generating nested loops that encode dependent variable values directly and break up (linear) search as the sample value u is found. An example for this approach is shown in the generated code for hPAM2 in Fig. 6.7, with the sampling weight computation in lines 32ff and the sampling search in lines 58ff (where the unnormalised $u \sum_{\vec{k}} p(\vec{k}|\cdot)$ is searched).

C.2 Gibbs meta-sampler

Some details on the Gibbs meta-sampler described in Chapter 6 are given. As the meta-sampler is a fairly complex program (with about 10,000 lines of code and comments), for simplicity, some concepts have been "flattened" to increase readability.

C.2.1 Design aspects

The code generator of the Gibbs meta-sampler is written in Java and follows the concept outlined in Section 6.7, most notably to keep the structure of the program and program generators similar, which simplifies updates and extensions considerably. The second design decision was to keep string manipulations central and exploit the possibilities of dedicated code parsers and regular expressions. A third concept was the use of "facets", i.e., functions that the code generator queries on the objects representing NoMM nodes, edges etc. and is returned code that can be emitted by the code renderer, extended by the language-specific syntax.

Implementation approach. The implementation of the kernel generator results from a generalisation process of a straight-forward LDA Gibbs sampler implementation, which had been optimised for clear program structure and readability as well as extensibility towards serial and parallel approaches to fast sampling as discussed in Chapter 8. The implementation steps were as follows:

1. Implementation of LDA as a trivial case of the generic sampler in Fig. 6.1[1]. This code in Java is self-contained and can be run with an example data set. The corresponding C version is structurally very similar, mostly adding memory management using `malloc` and `free` API calls handled by the garbage collector in Java and data I/O functions, as well as explicit size parameters for arrays.

2. Parametrisation of the variables in the raw code in a MixNet structure (the name of a NoMM in the program code) that contains representations of MixNode, MixEdge and MixSequence structures, similar to the structure shown in Fig. 6.4. All of these structures capture the variables and structural properties of the original network. Code is generated by creating the different functions corresponding the raw LDA code, now looping over NoMM objects according to the network structure.

3. Implementation of a parser according to Fig. 6.3 (or rather, the BNF given in Fig. C.2) and synthesis of the original LDA code by generating the corresponding Java objects from the the parsed script and from those the Gibbs sampling code.

4. Generalisation of the network structures: More complex interrelation between MixNet objects are handled in using case switches. This includes a structure classifier that identifies the structures from Chapter 5 and switches between different modes of inference equations (see Chapter 9).

5. Introduction of extensions, such as independent sampling (forcing all hidden variables to be handled as single-element dependency groups), and fast methods (see Chapter 8).

[1] The raw code implementation in Java is found at `http://arbylon.net/resources.html`.

Conceptual parts. The meta-sampler resulting from the above design process consists of five conceptual parts:

- *Data structure:* The data structure, a MixNet object (referring to "mixture network", the synonym for NoMM), represents the actual NoMM and reflects its different parts, including MixNode, MixEdge and MixSequence instances with interrelations and aggregations. The structure is shown in Fig. C.1.
- *Language parser:* The parser can create a MixNet object from the modelling language defined in Section C.2.2.
- *Classifier:* The network classifier detects the different NoMM structures as defined in Chapter 5 and may be used to check structural constraints and detect missing information. In effect, the classifier sets the datatype and linktype fields in the all MixItems (cf. Fig. C.1) and orders sequences into a hierarchy that can then be iterated at runtime.
- *Inference engine:* The inference engine creates the necessary "facets" of the MixNet data structure, which are internal representations of the update equations.
- *Code renderer:* A code renderer transforms the facets of the MixNet objects into actual source code that is output. Currently, there exist two code renderers, one for Java 1.5 and one for ANSI C.

In Fig. C.1, the data structure of the MixNet object is sketched. Facet functions (in red in the figure) are the central functional parts in this structure.

Facet functions work on the graph topology spanned by the sequences, nodes and edges of the MixNet structure and manipulate the strings of the parameters to render code. For instance, a component selector k in a loop over all hidden variables of a sequence (e.g., to calculate the weights of a Gibbs sampler) is represented by `kSel` in the MixNode class. Its content, for instance `[m,x]` for directly addressing components according to the values of the incoming edges `m` and `x`, will be transformed so that local copies of hidden variables of the loop are used. Since in the example, `x` is a hidden variable (while `m` is a sequence index, which is always oberved), and part of the current sequence, a local variable `hx` is generated and assigned the value of `x`. Note that these operations are only necessary at code generation time while the sampler itself is fully implemented at runtime.

Another property of the proposed concept of facets is that they are not only dependent on what they are called with but where they are called from. This idea complements function parametrisation – and thus decisions on the part of the caller – by function *contextualisation*, which changes function behaviour dependent on the call context – thus encapsulating decisions into the facet function itself based on the parameters in context. For instance, the index expansion for `[m,x]` above may return a different result if called in an initialisation context (when used to generate code for function `init()`, cf. Fig. 6.5) than in a sampling context (when function `run()` is generated), or may be different if used to generate code for training or testing (context of function `run()` vs. `runq()`). The advantage of this are leaner internal call interfaces, which simplify maintenance and extension of the generator.

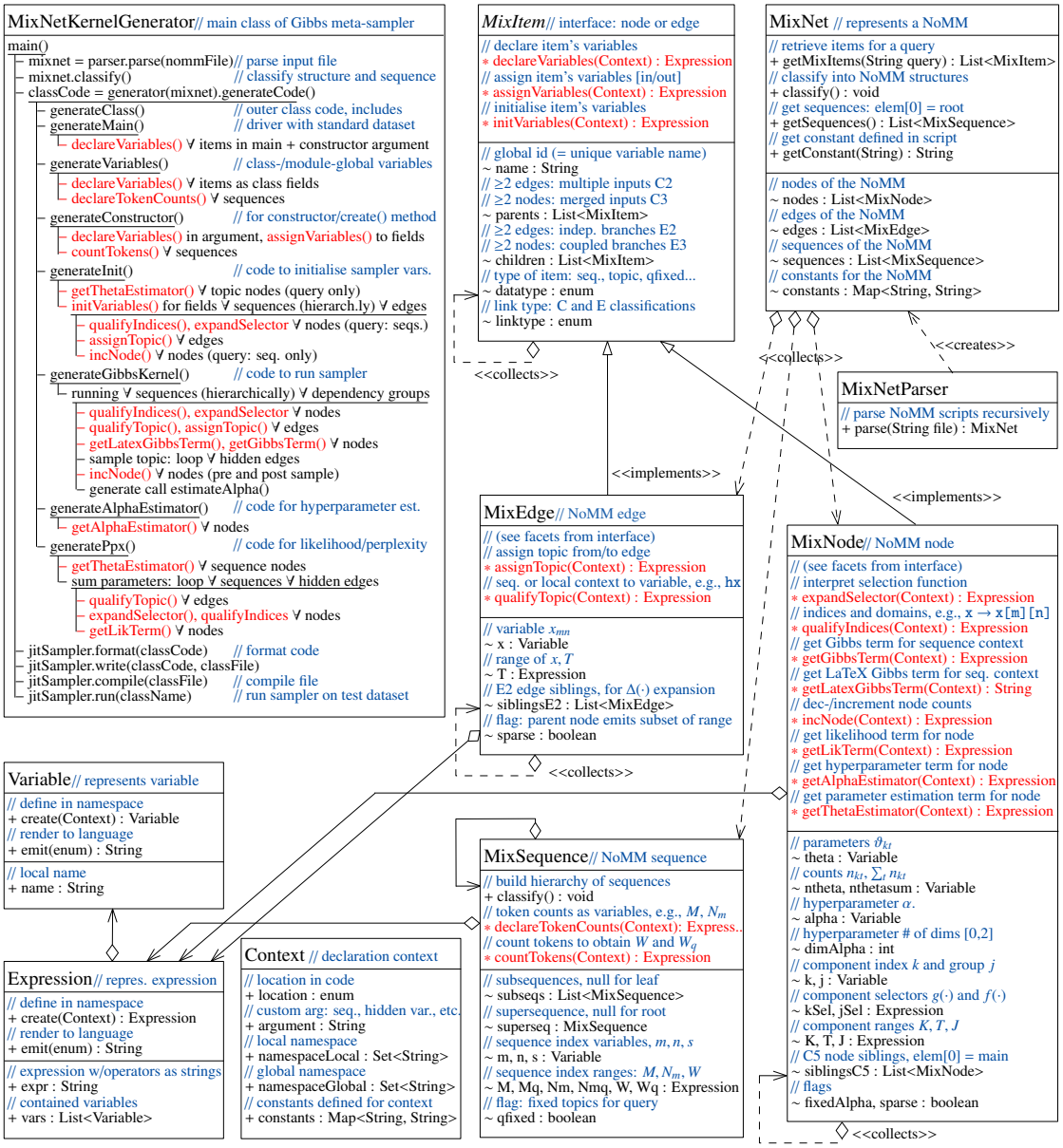


Figure C.1: UML diagram of the different MixNet items and coarse-grained structure of the code generator (simplified to show facet calls = red text).

C.2.2 Modelling language

The NoMM description language can be considered an instance of model description language as is used throughout model-driven software development: By simple domain-specific definition of a given problem, complex software is described concisely.

In Fig. C.2, the Backus–Naur form (BNF) of the grammar is given for the NoMM language used for the code generator in Chapter 6. It is compatible with the JavaCC "compiler compiler".[2]

[2] `https://javacc.dev.java.net/`.

```
/** ignored whitespace and comments */
<DEFAULT> SKIP : { " " | "\t" | <"#" (~["\n"])*> }

<DEFAULT> TOKEN : { <DCOLON: ":" ":"> | <COLON: ":"> | <GENERATES: ">>"> | <EQUIV: "=" "="> 
| <COMMA: ","> | <STOP: "."> | <PRIOR: "|"> | <TIMES: "*"> | <EQUALS: "=">
| <PLUS: "+"> | <MINUS: "-"> | <IN: "i" "n"> | <NI: "n" "i"> }

<DEFAULT> TOKEN : { <TEXT: (<CHARACTER>)* > | <NAME: <TEXT>>
| <VARIABLE: <LETTER> (<CHARACTER>)*> | <NUMBER: (<DIGIT>)+> | <#DIGIT: ["0"-"9"]>
| <#LETTER: ["a"-"z","A"-"Z"]> | <CHARACTER: <LETTER> | <DIGIT>>
| <INDSTART: "["> | <INDEND: "]">  | <SELSTART: "{"> | <SELEND: "}"> | <NEWLINE: "\n">
| <VARLIST: <VARIABLE> ( <COMMA> <VARIABLE> )*>
| <EXPRESSION: (<CHARACTER>+)> }

/* parser BNF productions */
go ::= ( <NEWLINE> )*
        "mixnet" <EQUALS> nameSection
        ( "description:" <NEWLINE> descSection )?
        ( "preferences:" <NEWLINE> prefSection )?
        "nodes:" <NEWLINE> nodeSection
        "sequences:" <NEWLINE> seqSection
        "edges:" <NEWLINE> edgeSection
        "network:" <NEWLINE> netSection

/** model name, example: mixnet = lda */
nameSection ::= <TEXT>

/** parse model description, free text until next section heading */
descSection ::= ( <NEWLINE> | <TEXT> )+

/** parse preferences */
prefSection ::= ( <NEWLINE> | prefLine )+
/* example: fastParallel = 1 */
prefLine ::= <VARIABLE> <EQUALS> <EXPRESSION>

/** parse node section: */
nodeSection ::= ( <NEWLINE> | nodeLine | nodeCoupleLine )+
/* example: doctopics = theta : M, V | alpha : 1 : fixed */
nodeLine ::= ( <NAME> <EQUALS> )? <VARIABLE> <COLON> <DIMENSION>
             ( <PRIOR> <VARIABLE> <COLON> <DIMENSION> )?
             ( <COLON> <VARLIST> )?
/* example: theta == zeta */
nodeCoupleLine ::= ( <NAME> <EQUALS> )? <VARIABLE> ( <EQUIV> <VARIABLE> )+

/** parse sequence section */
seqSection ::= ( <NEWLINE> | seqLine )+
/* example: words = w : m, n in s : M, w[m].length : W */
seqLine ::= ( <NAME> <EQUALS> )? <VARIABLE> <COLON> <DIMENSION>? ( ( <IN> | <NI> ) <DIMENSION> )?
            <COLON> <DIMENSION> <COLON> <DIMENSION>

/** parse edge section, which is structured by sequences */
edgeSection ::= ( <NEWLINE> | edgeSeqSection )+
edgeSeqSection ::= seqLabelLine <NEWLINE> ( <NEWLINE | edgeLine )*
/* example: words = w :: */
seqLabelLine ::= ( <NAME> <EQUALS> )? <VARIABLE> <DCOLON>
/* example: words = w : V */
edgeLine ::= ( <NAME> <EQUALS> )? <VARIABLE> <COLON> <DIMENSION>

/** parse the network section, i.e., the structure of the mixture network model.*/
networkSection ::= ( netLine | selectorBlock )+
/** example: x >> theta[m,x] | alpha[x] >> y */
netLine  ::=  <VARLIST> <GENERATES> parameter ( <PRIOR> parameter )? <GENERATES> <VARLIST> <NEWLINE>
/** examples: theta ; theta[m,x]
parameter ::= <VARIABLE> ( <INDSTART> <VARLIST> <INDEND> )?

/** parse selector block (can be multiple lines, allows Java on model namespace
and two special strings: IDLE to ignore branch, [x,y] as shorthand for [x * Y + y]) */
selectorBlock ::= <VARIABLE> <COLON> <SELSTART> ( <NEWLINE> | <EXPRESSION> )+ <SELEND> <STOP> <NEWLINE>
```

Figure C.2: BNF of NoMM script grammar.

The meaning of the different sections can be summarised as follows (cf. Fig. 6.3 for a mixnet script referred to in examples given):

- *mixnet:* The name of the model. This is used to generate the class and file names of generated code. For instance: `model = hpam2` will generate a class with the name `Hpam2GibbsSampler` or C-language module files `hpam2_gibbs_sampler.c,.h`.
- *description:* A high-level description that is inserted as a comment on top of the generated code, as input to automatic documentation systems like Doxygen or Javadoc.
- *preferences:* For code generation, different preferences may be stated, including:
  - *language* = $\{Java, C\}$: Specify the target language, Java 1.5 or ANSI C. Default = Java.
  - *verbose* = $\{0, 1\}$: Verbose output of comments during code generation. Default = 0.
  - *formatCode* = $\{0, 1\}$: Whether to pretty-print the output code. For Java, an internal code prettifier may be used, for both C and Java shell-based commands may be configured. Default = 0.
  - *fastSerial* = $\{0, 1\}$: Whether to generate serial fast samplers, cf. Chapter 8. Currently, support for automatic generation is done by inserting comments with tasks and fast state variables named to defaults for manual adjustment (approx. 100 lines per NoMM node). Default = 0.[3]
  - *fastParallel* = $\{0, 1\}$: Whether to generate parallel fast samplers, cf. Chapter 8. Default = 0.[3]
  - *syncMode* = $\{0, 1, 2\}$: Synchronisation method of fast parallel samplers Ch. 8: exact full-state sampling (0), the split-state approximate sampler (1) or the full-state approximate sampler (2), as described in Chapter 8. Notably, *parallelMode* may be used with *fastParallel* = 0, so the approximations can be tested with serial samplers, as well. Default = 0.[3]
  - *indepSamplers* = $\{0, 1\}$: Whether to coerce independent samplers, i.e., generate a Gibbs sampler for each variable independently. This speeds up inference but is prone to overfitting. Default = 0.[3]
  - *directAssign* = $\{0, 1\}$: Whether to directly assign indexes from global variables, e.g., `x[ksel][w[m][n]] = hx;`, otherwise temporary variables are created, e.g., `int t = w[m][n]; x[ksel][t] = hx;`. Different compilers may generate faster code for local accesses to the same field, but code size slightly increases. Default = 1.
  - *makeLatex:* For inference equations in the Gibbs samplers, LaTeX source code may be emitted as a comment next to the weights and likelihood calculation. Default = 1.
  - *nameSuffix* = $\{0, 1\}$: Whether to add a suffix to the model name (see *mixnet* section): Currently, this is `i` for independent, `s` for serial accelerated samplers and `p` for parallel samplers, for instance, naming an HPAM2 model with all accelerators `Hpam2ipsGibbsSampler` in Java. Default = 0.[3]

[3] See Chapter 8 for combinations currently supported.

- *syncSuffix* = {0, 1}: Whether to add a suffix for the synchronisation method in parallel sampling approaches. This is `a` for exact, `b` for split-state and `c` for full-state approximation, cf. *syncMode*. Default = 0.[3]
- *package* = String defines the package (for Java) that the class is generated in. Default = `org.knowceans.topics.cgen`
- *checkState* = {0, 1}: Whether to emit code of a `checkState()` method to check the validity of the Markov state. This initialises a set of probe matrices from variables X and compares them with the $n^{\ell}_{k,t}$ and n^{ℓ}_{k} in the model, reporting any inconsistent elements of the count matrices. This is useful when manual modifications are necessary. Default = 0.
- *correctSamples* = {0, 1}: Whether to reset samples that have run out of range to their previous values for precision reasons. If a sample fails and *checkState* = 1, the `checkState()` method is called after re-incrementing the state in the Gibbs sampler. Default = 0.
- *runModel* = {0, 1}: Whether to run the model with the default test corpus with the generated `main()` method that performs initialisation, query sampling, perplexity, training sampler, query sampling, perplexity. Used for coarse-grained sanity checks.
- K or T = integer: Number of topics T used for every hidden edge. As all of the following settings, this may be configured in the `main()` method manually after code generation. Default = 10.
- *alpha* = float: Hyperparameter value at initialisation. Default = 0.1.
- *corpus* = string: Test corpus. Default = `nips/nips` for the NIPS corpus (see Appendix D.2). The corpus is randomly partitioned into a 90/10% training/test split.
- *iter* = integer: Number of main Gibbs iterations. Default = 300.
- *iterq* = integer: Number of query Gibbs iterations. Default = 10.

- *nodes:* Each line contains the definition of a mixture node: Variables for node parameter and hyperparameter (e.g., `thetax` and `alphax`) are complemented by the corresponding dimensions. In front of the variable name with an "=" sign as a separator, a human-readable annotation may be optionally given (e.g., `doc-subtop = theta`). This is possible also for sequences and edges.

  Dimensions are either 1 (scalar) or have one or more dimension variables. The right-most dimension equals that of the outgoing edge and all others refer to the set of component indexes (e.g., `thetax : M, X, Y` is an array of $M \times X$ components of dimension Y).

  Nodes may have additional settings, like `fixed` for omitting hyperparameter estimation, and dimensions may use addition and multiplication of declared variables (e.g., `phi : 1 + X + Y, V`).

  Furthermore, C5 node merging can be defined using a syntax like `theta1 == theta2` on a separate line after both nodes have been declared.

- *sequences:* Each edge belongs to a sequence for which it transfers values. Those sequences (e.g., `words = w`) are defined by their index variables (e.g., `m,n`) and their respective ranges (e.g., `M, w[m].length`[4]). Further, the variable for the total number of tokens of the edge is specified (e.g., `W`).

- *edges:* Edges are defined dependent on the sequences they "run in" (e.g., `words = w ::`, which are specified before. For each edge, on the line a variable and range are given (e.g., `x : X` for $x \in [1, X]$ or, in code, $x \in [0, X-1]$).

- *network:* Based on the previous variable definitions, the network structure itself is specified. Each node with its input and output is given on a separate line (e.g., `m,x >> thetax[m,x] | alpha[x] >> y,z` means $\vec{\vartheta}^{x}_{m,x}$ is fed by m and x edges and feeds y and z edges).

  The hyperparameter is optional if no index is selected (e.g., `m >> theta >> x` is valid). Component indexes are optional if there is a one-to-one mapping to input edges (e.g., `m,x >> thetax[m,x]` and `m,x >> thetax` are equivalent in Fig. 6.3).

  Branching and merging can be easily introduced in the network. Independent branches (E2) are given with two output variables on for a node (e.g., `a >> theta >> b,c`), and coupled branches with (E3) one output that is input in more than one child node (e.g., `theta >> x` input into `m,x >> thetax` and `x,y >> phi[k]` in Fig. 6.3). Correspondingly, independent component indexes (C2) are given as dual input (e.g., `x,y >> phi[k]` or `m,x >> thetax`), and coupled inputs are given as a single component index that is output from several nodes (e.g., `gamma >> x; delta >> x; x >> eta` couples outputs from `gamma` and `delta`).

  Finally, complex component selection functions, can be introduced by referring to a new variable and defining it in a separate block as a function of the input variables (e.g., `x,y >> phi[k]` and `k : { ... } .` in Fig. 6.3).

- *import:* Each line in this section imports a NoMM file that may have all statements except *mixnet*, which defines the root of the import hierarchy. This allows sharing of settings and structures between different model designs, especially overwriting imported settings locally, similar to the concept of class inheritance.

- *code@<location>:* Sections of the *code@* type allow injection of code snippets into the generated sources. The following sections are supported: *import; class; inherit; main::start, end; constructor::start, end; destructor; methods::start, end; init[q]::start, end; run[q]::start, end; run[q]::iter::start, end; alpha::start, end; ppx::start, end*, which refer to the different functions in the sampler, cf. Fig. 6.5.

[4] `w[m].length` is a common Java structure to obtain the length of a vector, here the document length.

C.3 Sampling aggregation branches

In the NoMM library in Chapter 9, aggregation structures N5+E4 have been outlined, and in this appendix we give some details on the inference involved, first on the discrete approach (with node type N5A) and subsequently on regression-based methods (with node type N5B).

Selection-based approach. In a straight-forward selection-based or "discrete" approach as published in [Bundschus et al. 2009], aggregation branches may use the empirical distribution of the co-branch (the edge "parallel" to the E4 edge whose values are aggregated), $\vec{\zeta}_m \propto \sum_n \delta(k - z_{m,n}) = \{n_{m,k}\}_k$, and feed an additional mixture node for the class labels:

$$\cdots \xrightarrow[[K]]{\vec{z}_m} \underset{\text{N5A}}{\boxed{(\vec{\zeta}_m \,|\, \vec{z}_m)}} \xrightarrow[[K]]{\tilde{z}_{m,j}} (\vec{\eta}_{\tilde{z}} \,|\, \vec{\alpha}) \xrightarrow[[V]]{y_{m,j}} \tag{C.7}$$

where $\vec{z}_m$ aggregates a co-branch with variable $z_{m,n}$. In the Gibbs sampler, $\tilde{z}$ is sampled after the co-branch has been sampled and $z_{m,n}$ are known:

$$p(\tilde{z}_{m,j}{=}k | y_{m,j}{=}c, \vec{z}_m, \eta) \propto \zeta_{m,k}\; q(k, c) \tag{C.8}$$

with $\eta = \{\vec{\eta}_k\}_k$ the set of all topic-specific class distributions and the definition of $q(\cdot)$ described in Chapter 9. The parameter η here has the role of a regression coefficient in models like supervised sLDA, see Chapter 5. It follows the simple update rules of Dirichlet–multinomial mixture nodes and therefore does not require a specific optimisation step in the estimator. The full conditional weight of the co-branch is multiplied by $\zeta_{m,\tilde{z}} q(\tilde{z}, y)$ according to the current values of $\vec{\zeta}_m$ and $q(\tilde{z}, y)$. Therefore, the aggregation branch indeed influences the sampling of the co-branch according to the global correspondence between labels and topics. However, in the full derivation in [Bundschus 2010], this reverse dependency appears to have been neglected despite its potential virtues.

Linear regression. As an alternative, we give a Gibbs sampling variant to the model of sLDA [Blei & McAuliffe 2007], which uses normally distributed response variables y and performs variational inference on them. Consider the following structure (cf. (5.10)):

$$\begin{array}{l} m \xrightarrow[[M]]{m} \underset{\text{N1}}{(\vec{\vartheta}_m \,|\, \alpha)} \xrightarrow[[K]]{z_{m,n}=k} \underset{\text{N1}}{(\vec{\varphi}_k \,|\, \beta)} \xrightarrow[[V]]{w_{m,n}} w_{m,n} \qquad \{M, N_m\} \\ \qquad\qquad\quad \Big\downarrow \vec{z}_m \\ \qquad\qquad\quad \underset{[K]}{\boxed{(\vec{\eta}_c^{\top} \vec{\zeta}_m, \sigma^2 \,|\,)}} \xrightarrow[\mathcal{N}]{y_{m,c}} y_{m,c} \qquad \{M, C\} \\ \qquad\qquad\quad \mathcal{N} \\ \qquad\qquad\quad \text{N5} \end{array} \tag{C.9}$$

For this, the values $y_{m,c}$ are to be determined based on the aggregation of the values of the edge $z_{m,n}$, that is, there may exist several class labels c that a document has high association with. The N5B node is governed by a Gaussian distribution $\mathcal{N}(y_m \,|\, \vec{\eta}_c^{\top} \vec{\zeta}_m, \sigma^2)$ with $\vec{\zeta}_m = \{\zeta_{m,k}\}_k$ the empirical topic distribution for document m, $\zeta_{m,k} \propto \sum_n \delta(k - z_{m,n})$, $\vec{\eta}_c$ the regression coefficients and σ^2 the noise variance of the linear model. Here the index c stands for the usage of different response variables with normal distribution that may be also used as class indicators. For the

Gibbs sampler of the model, the value of these Gaussians directly influences topic selection:

$$p(z_{m,n}|\cdot) \propto \sum_c \mathcal{N}(y_{m,c} \,|\, \vec{\eta}_{y_{m,c}}^{\top} \vec{\zeta}_m, \sigma^2)\, q(m, z_{m,n})\, q(z_{m,n}, w_{m,n}) \tag{C.10}$$

where $q(\cdot)$ is defined as in Chapter 9 and $\vec{z}_m$ is the basis to derive $\vec{\zeta}_m$. The inclusion of the normal in the full conditional effectively feeds back label information into topic distributions. Interleaved with this, the free parameters for the Gaussian, $\vec{\eta}_c$ and σ^2 are estimated together with the hyperparameters of each node, and update equations are obtained using the zero-derivatives (cf. Section 3.4.1) of the corpus-wide version of the Gaussian [Blei & McAuliffe 2007] $\mathcal{N}(y|\underline{A}\vec{\eta}, \sigma^2)$ w.r.t. $\vec{\eta}$ and σ^2, with the $M \times K$ matrix $\underline{A} = \{\vec{\zeta}_m^{\top}\}_m$ and M-vector $\vec{y} = \{y_m\}_m$ that concatenate document contributions in rows:

$$\vec{\eta} = (\underline{A}^{\top}\underline{A})^{-1}\underline{A}^{\top}\vec{y} \tag{C.11}$$

$$\sigma^2 = M^{-1}\left(\vec{y}^{\top}\vec{y} - \vec{y}^{\top}\underline{A}\vec{\eta}\right). \tag{C.12}$$

Note that opposed to [Blei & McAuliffe 2007] we don't use the expectation $\sum_m \left\langle \vec{\zeta}_m \vec{\zeta}_m^{\top} \right\rangle$ to estimate $\underline{A}$, but the empirical distributions $\vec{\zeta}_m$ directly, which are often readily available in the collapsed Gibbs sampler as $n_{m,k}$ without prior estimation of the co-branch's node parameters (as needed for $\left\langle \vec{\zeta}_m \vec{\zeta}_m^{\top} \right\rangle$). This approach corresponds to the ordinary least squares (OLS) estimator on the empirical distributions.

Logistic regression. For logistic regression on the topics $\vec{z}_m$ and response variables $y_m \in [-1, 1]$, the auxiliary-variable method proposed by [Mimno et al. 2008] for logistic-normal Gibbs sampling of priors may be used analogously for aggregation branches.

Appendix D

Reference information for experiments

For this thesis, several computers have been used to perform the experiments. As computing architecture has influence especially on the timing and parallelisation of algorithms, the basic specifications are given in this appendix. Furthermore, the data sets used in the experiments are outlined.

D.1 Computing hardware

System 1: Thinkpad notebook. Model: IBM/Lenovo Thinkpad T43, manufactured: late 2006, detailed specification: 2668-75G, processor: Intel Pentium M760/2.0GHz (single core, Intel Sonoma architecture), RAM: 2GB DDR2/533MHz, OS: Microsoft Windows XPPro SP3, Java VM: Sun JDK 1.6.0_11-b03. System has been used in experiments for Chapter 7 in 32 bit mode.

System 2: Quad-core PC. Model: PC based on ASUS P6T7 WS Supercomputer motherboard, manufactured: mid 2009, processor: Intel i7-920/2.67GHz (quad core, 8 virtual cores using HyperThreading technology, 8MB shared L3-cache), RAM: 6GB DDR3/667MHz, OS: Windows 7 SP1, Java VM: 1.6.0 build 20.1-b02 64bit. VM run in experiments for Chapter 8 in 64 bit mode.

System 3: MacBook. Model: Apple MacBook Pro 15", manufactured: late 2010, processor: Intel Core i5/2.4GHz (dual core, 4 virtual cores using HyperThreading technology, 3MB shared L3-cache), RAM: 4GB DDR3/1067MHz, OS: Mac OS X, 10.6.7, Java VM: Sun JDK 1.6.0_24-b07-334-10M3326 for client and build 19.1-b02-334 for server virtual machine. System and Java VM have been used in 64-bit mode, exclusively, in experiments in Chapters 8 and 10. For Chapters 6 and 8, some experiments have been performed on a late-2008 MacBook 13" with an Intel CoreDuo at 2GHz and Mac OS X 10.5. Performance-related experiments have been repeated on System 2.

Virtual processor cores. To allow realistic speed-up measurements, HyperThreading and TurboBoost, two vendor-supplied technologies to increase the number of tasks on a processor by running more than one thread at a time on one core and by overclocking processors by up to 10%, respectively, have been switched off.

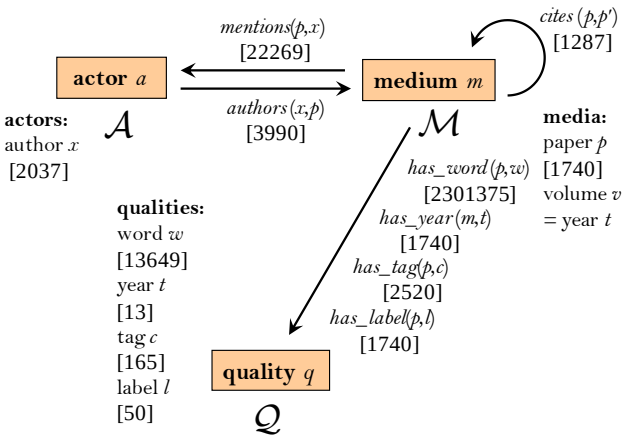


Figure D.1: NIPS corpus, AMQ schema. Data volumes in brackets.

D.2 Data sets

For the data sets used in this thesis, size and structure are given here for reference.

NIPS corpus with labels and citations. The NIPS corpus is a collection of 13 years of the conference "Neural Information Processing Systems" from 1989 to 2002 whose proceedings are openly available.[1] To obtain a corpus with different modalities beyond authorship is usually difficult, because either data are noisy with many scanning errors and duplicates like the CiteSeer data set, or do not allow access to the raw data they have been constructed from like the Cora data set. The NIPS corpus as assembled here has, however, such a wide variety: Full word content, full authorship, vocabulary, citation graph as well as category labels.

The basis for this corpus is the data set constructed by Sam Roweis.[2] What has been added are labelling and citation data. The labels include the name of NIPS conference tracks (e.g., "Speech and Signal Processing"). Furthermore, up to eight annotations per document are added. These tags were assigned freely by two domain experts. Because no controlled vocabulary was used, they have overlapping semantics (e.g., "*visual cortex*" vs. "*vision*"). All of the tags refer to the theme of the paper and are not functional or evaluative (e.g., "*important*" or "*good*") as is often the case in collaborative tagging scenarios. After tagging, tags with similar semantics have been manually merged and tags with low document frequency below 3 removed, leading to a total of 165 labels in 2520 annotations. The resulting tagging data represent a simplified view on a collaborative free-vocabulary annotation process, however, not distinguishing annotators in the data model.

The citation data were extracted from the internal citation graph by via a pattern search for NIPS reference entries and parsing them automatically, followed by a manual postprocessing step to weed out false citation candidates. The resulting data represent a good example of an emerging community: Starting at the early beginnings, researchers did naturally only cite themselves a few

[1] `http://books.nips.cc`.
[2] `http://cs.nyu.edu/~roweis/data.html`.

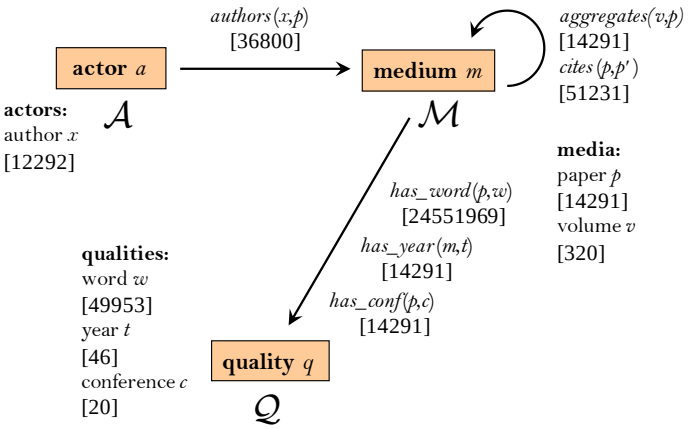


Figure D.2: ACL Anthology corpus, AMQ schema. Data volumes in brackets.

times – interestingly, there are cross-citations even in the first volume. Later, the internal citation community increasingly forms and changes its thematic focus as the conference develops: from neural networks to support vector machines and statistical methods like Bayesian networks.

Although in this process, the co-citation between articles of the conference intensifies, the major portion of academic communication still goes through other conferences and journals, which is not a surprise in an interdisciplinary field like that of NIPS. In order to enhance the internal structure of the citation graph, a second dimension of co-citation is added: Articles that mention members of the community are linked to those ones written by the people mentioned, creating a *mentions* relation between media and authors. Similar to the citations, the data have been extracted by a pattern search, followed by manually pruning results.

The AMQ graph of the NIPS corpus is given in Fig. D.1, with quantities specifying the volumes of available data.

ACL Anthology. The ACL Anthology (ACLA) has been described in Chapter 2, and in Fig. D.2, the quantities of the corpus are given. In particular, this corpus was derived from the ACL Anthology Network[3], which has been carefully extracted from the original online data[4] [Radev et al. 2009]. Compared to the NIPS corpus, ACLA is roughly 10 times larger in terms of documents and authors, with almost 5 times larger vocabulary and 50 times more internal citation links. However, no category information exists, but there are conference and journal associations.

[3] http://clair.si.umich.edu/clair/anthology/.
[4] http://aclweb.org/anthology-new/.

Appendix E

Details on application models

This appendix sketches the traditional derivation of model inference and likelihood equations in Chapter 10. Comparing this to the NoMM-based derivations developed in the thesis illustrates the usefulness of that method to avoid tedious calculations.

E.1 Bayesian networks of application models

For comparison with the NoMM representations in Chaper 10, the Bayesian networks of the three variants of the expert–tag–topic models are presented in Figs. E.1 and E.2. The dashed plate in Fig. E.2(b) refers to the duplicate draw of the C3B structure explained in Sec. 9.3.

E.2 Example derivation: Expert–tag–topic model 1

The Bayesian network of the ETT1 model is shown in Fig. E.1. The details of the derivation strategy have been explained for instance in [Heinrich 2009b]; it is similar to the strategies used in literature.[1] We start with the complete-data likelihood of the corpus:

$$\begin{aligned} p(\vec{w},\vec{c},\vec{a},\vec{x},\vec{z},\underline{\Theta},\underline{\Phi},\underline{\Psi}|\alpha,\beta,\gamma) &= p(\vec{w}|\vec{z},\underline{\Phi})p(\underline{\Phi}|\beta)\cdot p(\vec{c}|\vec{y},\underline{\Psi})p(\underline{\Psi}|\gamma) \\ &\quad\cdot p(\vec{y}\,|\vec{x},\underline{\Theta})p(\vec{z}\,|\vec{x},\underline{\Theta})p(\underline{\Theta}|\alpha)\cdot p(\vec{x}|\vec{a}) \qquad \text{(E.1)} \\ &= \prod_{m=1}^{M}\Bigg(\prod_{n=1}^{N_m} p(w_{m,n}|\vec{\varphi}_{z_{m,n}})p(z_{m,n}|\vec{\vartheta}_{x_{m,n}})\,a_{m,x_{m,n}} \\ &\quad\cdot\prod_{j=1}^{J_m} p(c_{m,j}|\vec{\psi}_{y_{m,j}})p(y_{m,j}|\vec{\vartheta}_{x_{m,j}})\,a_{m,x_{m,j}}\Bigg) \\ &\quad\cdot p(\underline{\Theta}|\alpha)\cdot p(\underline{\Phi}|\beta)\cdot p(\underline{\Psi}|\gamma)\,. \qquad \text{(E.2)} \end{aligned}$$

[1] Alternative derivation strategies for topic model Gibbs samplers have been published in [Griffiths 2002] working via $p(z_i|\vec{z}_{\neg i},\vec{w}) \propto p(w_i|\vec{w}_{\neg i},z)p(z_i|\vec{z}_{\neg i})$ and [McCallum et al. 2007] who use the chain rule via the joint token likelihood, $p(z_i|\vec{z}_{\neg i},\vec{w}_{\neg i}) = p(z_i,w_i|\vec{z}_{\neg i},\vec{w}_{\neg i})/p(w_i|\vec{z}_{\neg i},\vec{w}_{\neg i}) \propto p(\vec{z},\vec{w})/p(\vec{z}_{\neg i},\vec{w}_{\neg i})$, which is similar to the approach taken here.

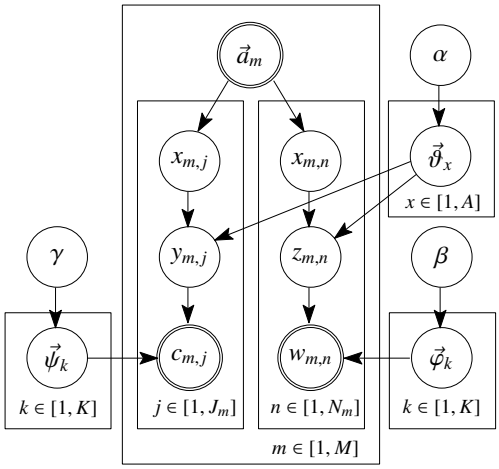
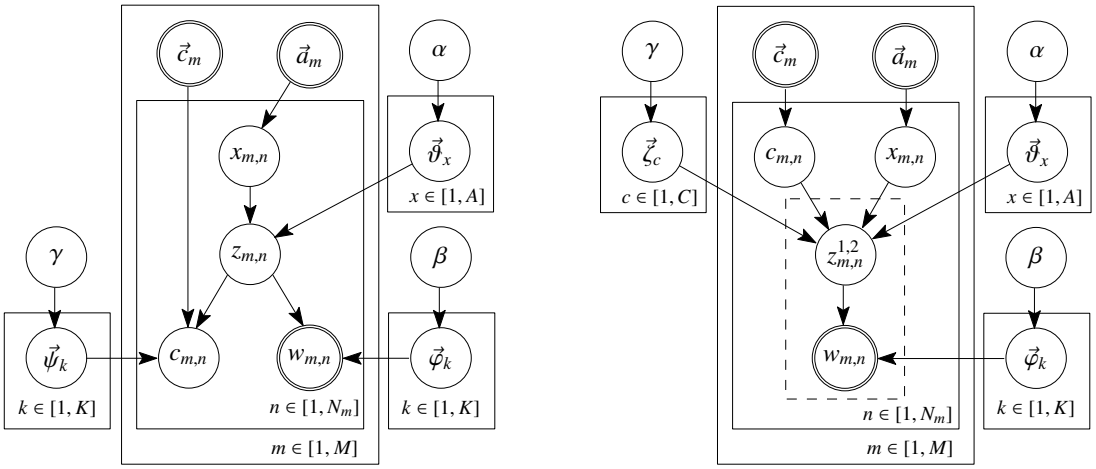


Figure E.1: ETT1 model: Bayesian network.

Next, we integrate out the model parameters, introducing our knowledge on the types of distributions and their conjugacy:

$$
\begin{aligned}
p(\vec{w}, \vec{c}, \vec{a}, \vec{x}, \vec{z} | \alpha, \beta, \gamma) = \iiint \prod_{m=1}^{M} \Bigg(& \prod_{n=1}^{N_m} p(w_{m,n} | \vec{\varphi}_{z_{m,n}}) p(z_{m,n} | \vec{\vartheta}_{x_{m,n}}) \, a_{m,x_{m,n}} \\
& \cdot \prod_{j=1}^{J_m} p(c_{m,j} | \vec{\psi}_{y_{m,j}}) p(y_{m,j} | \vec{\vartheta}_{x_{m,j}}) \, a_{m,x_{m,j}} \Bigg) \\
& \cdot \mathrm{d}p(\underline{\Theta} | \alpha) \cdot \mathrm{d}p(\underline{\Phi} | \beta) \cdot \mathrm{d}p(\underline{\Psi} | \gamma) \qquad \text{(E.3)} \\
= \int \prod_{m=1}^{M} \prod_{n=1}^{N_m} & p(w_{m,n} | \vec{\varphi}_{z_{m,n}}) \prod_{k=1}^{K} p(\vec{\varphi}_k | \beta) \, \mathrm{d}\varphi_k \\
& \cdot \int \prod_{m=1}^{M} \prod_{j=1}^{J_m} p(c_{m,j} | \vec{\psi}_{y_{m,j}}) \prod_{k=1}^{K} p(\vec{\psi}_k | \gamma) \, \mathrm{d}\vec{\psi}_k \\
& \cdot \int \prod_{m=1}^{M} p(\underline{\Theta} | \alpha) \prod_{n=1}^{N_m} p(z_{m,n} | \vec{\vartheta}_{x_{m,n}}) \, a_{m,x_{m,n}} \prod_{j=1}^{J_m} p(y_{m,j} | \vec{\vartheta}_{x_{m,j}}) \, a_{m,x_{m,j}} \, \mathrm{d}\vec{\vartheta}_m \qquad \text{(E.4)} \\
= \int \prod_{k=1}^{K} \frac{1}{\Delta_V(\beta)} \prod_{t=1}^{V} \varphi_{k,t}^{n_{k,t}+\beta-1} \, \mathrm{d}\vec{\varphi}_k & \cdot \int \prod_{k=1}^{K} \frac{1}{\Delta_C(\gamma)} \prod_{c=1}^{C} \psi_{k,c}^{n_{k,c}+\gamma-1} \, \mathrm{d}\vec{\psi}_k \\
& \cdot \int \prod_{a=1}^{A} \frac{1}{\Delta_K(\alpha)} \prod_{k=1}^{K} \vartheta_{a,k}^{n_{a,k}^{(z)}+n_{a,k}^{(y)}+\alpha-1} \, \mathrm{d}\vec{\vartheta}_a \cdot \prod_{m=1}^{M} \prod_{a=1}^{A} a_{m,a}^{n_{m,a}^{(z)}+n_{m,a}^{(y)}} \qquad \text{(E.5)} \\
= \prod_{k=1}^{K} \frac{\Delta(\vec{n}_k^{(z)} + \beta)}{\Delta_V(\beta)} & \cdot \frac{\Delta(\vec{n}_k^{(y)} + \gamma)}{\Delta_C(\gamma)} \prod_{a=1}^{A} \frac{\Delta(\vec{n}_a^{(z)} + \vec{n}_a^{(y)} + \alpha)}{\Delta_K(\alpha)} \prod_{m=1}^{M} a_{m,a}^{n_{m,a}^{(z)}+n_{m,a}^{(y)}} \, . \qquad \text{(E.6)}
\end{aligned}
$$



(a) ETT2 (b) ETT3

Figure E.2: Iterated ETT models: Bayesian networks.

Note the change in indexing from tokens, $w_{m,n}$, $x_{m,n}$, etc., to count statistics, $n_{x,k}$, $n_{k,c}$, etc. The superscripts in counts distinguish the branches of the model: w = words, c = tags. To solve the integrals in (E.5), either the Dirichlet integral of the first type can be used [Abramowitz & Stegun 1964], or one can observe that the Dirichlet distributions just re-parametrise (due to conjugacy with the multinomial, cf. Appendix B). Because the actual distributions integrate to one and vanish, solely their new normalisation must be determined.

The Gibbs full conditional can be determined from (E.6) by applying the chain rule. Because the Gibbs sampler will scan through words and tags in an alternating fashion, the two variables $z_{m,n}$ and $y_{m,j}$ are sampled independently. However, the author association at the root of the model must be sampled jointly for both. Using $i = (m, n)$, $n_{x,k} = n^{(z)}_{x,k} + n^{(y)}_{x,k}$ and the sum notation $n^{(z)}_k = \sum_{t=1}^{V} n_{k,t}$, etc., the full conditional for word tokens becomes:

$$
\begin{aligned}
&p(z_i{=}k, x_i{=}x | w_i{=}t, \vec{z}_{\neg i}, \vec{y}, \vec{x}_{\neg i}, \vec{w}_{\neg i}, \vec{a}, \vec{c}) \\
&\quad = \frac{p(\vec{w}, \vec{z}, \vec{y}, \vec{x})}{p(\vec{w}, \vec{z}_{\neg i}, \vec{y}, \vec{x}_{\neg i})} = \frac{p(\vec{w}|\vec{z}, \vec{y})}{p(\vec{w}_{\neg i}|\vec{z}_{\neg i}, \vec{y}) p(w_i)} \cdot \frac{p(\vec{z}|\vec{x})}{p(\vec{z}_{\neg i}|\vec{x}_{\neg i})} \cdot \frac{p(\vec{x})}{p(\vec{x}_{\neg i})} && \text{(E.7)} \\
&\quad \propto \frac{\Delta(\vec{n}^{(z)}_k + \beta)}{\Delta(\vec{n}^{(z)}_{k,\neg i} + \beta)} \cdot \frac{\Delta(\vec{n}_x + \alpha)}{\Delta(\vec{n}_{x,\neg i} + \alpha)} \cdot a_{m,x} && \text{(E.8)} \\
&\quad = \frac{\Gamma(n_{k,t} + \beta)\, \Gamma(n_{k,\neg i} + V\beta)}{\Gamma(n_{k,t,\neg i} + \beta)\, \Gamma(n_k + V\beta)} \cdot \frac{\Gamma(n^{(z)}_{x,k} + \alpha)\, \Gamma(n^{(z)}_{x,\neg i} + K\alpha)}{\Gamma(n^{(z)}_{x,k,\neg i} + \alpha)\, \Gamma(n^{(z)}_x + K\alpha)} \cdot a_{m,x} && \text{(E.9)} \\
&\quad = \frac{n_{k,t,\neg i} + \beta}{n_{k,\neg i} + V\beta} \cdot \frac{n^{(z)}_{x,k,\neg i} + \alpha}{n^{(z)}_{x,\neg i} + K\alpha} \cdot a_{m,x} && \text{(E.10)} \\
&\quad = q(k, t)\, q(x, k)\, a_{m,x}\,. && \text{(E.11)}
\end{aligned}
$$

For the tag branch, the derivation is analogous, now re-defining $i = (m, j)$:

$$p(y_i{=}k, x_i{=}x | c_i{=}c, \vec{z}_{\neg i}, \vec{y}_{\neg i}, \vec{x}_{\neg i}, \vec{w}, \vec{a}, \vec{c}_{\neg i}) \propto \frac{n_{k,c,\neg i} + \gamma}{n_{k,\neg i} + V\gamma} \cdot \frac{n^{(y)}_{x,k,\neg i} + \alpha}{n^{(y)}_{x,\neg i} + K\alpha} \cdot a_{m,x} \tag{E.12}$$

$$= q(k, c)\, q(x, k)\, a_{m,x}\,. \tag{E.13}$$

The difference of (E.11) and (E.13) to (10.3) is a result of the definition of $n_{x,k}$ as a summed count and the fact that both branches are disjointly sampled.

Bibliography

M. Abramowitz, I. A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, Dover, New York, 1964, URL http://www.math.sfu.ca/~cbm/aands/

J. Aitchison, S. Shen, Logistic-normal distributions: Some properties and uses, *Biometrika*, vol. 67, pp. 261–72, 1980

L. AlSumait, D. Barbará, J. Gentle, C. Domeniconi, Topic significance ranking of LDA generative models, in: *ECML*, 2009, URL http://www.springerlink.com/content/v3jth868647716kg/

J. Anderson, *Language, Memory, and Thought*, Hillsdale, NJ: Erlbaum, 1976

G. E. Andrews, R. Askey, R. Roy, *Special functions*, Cambridge University Press, 1999

D. Andrzejewski, X. Zhu, M. Craven, Incorporating domain knowledge into topic modeling via Dirichlet forest priors, in: *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 25–32, ACM, New York, NY, USA, 2009

N. Antulov-Fantulin, M. Bošnjak, T. Šmuc, M. Jermol, M. Žnidaršič, M. Grčar, P. Keše, N. Lavrač, Discovery challenge: Videolectures.net recommender system challenge, Tech. rep., ECML/PKDD, 2011, URL http://tunedit.org/challenge/VLNetChallenge

A. Asuncion, M. Welling, P. Smyth, Y.-W. Teh, On smoothing and inference for topic models, in: *UAI*, 2009, URL http://www.ics.uci.edu/~asuncion/pubs/UAI_09.pdf

A. Atkin, Peirce's theory of signs, in: E. N. Zalta (ed.), *The Stanford Encyclopedia of Philosophy*, winter 2010 ed., 2010

L. Azzopardi, M. Girolami, K. van Risjbergen, Investigating the relationship between language model perplexity and IR precision-recall measures, in: *Proc. SIGIR*, 2003

F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, P. F. Patel-Schneider (eds.), *The Description Logic Handbook*, Cambridge University Press, 2nd ed., 2007

R. A. Baeza-Yates, B. A. Ribeiro-Neto, *Modern Information Retrieval*, ACM Press & Addison-Wesley, 1999, URL http://citeseer.ist.psu.edu/baeza-yates99modern.html

K. Balog, T. Bogers, L. Azzopardi, M. de Rijke, A. van den Bosch, Broad expertise retrieval in sparse data environments, in: *Proc. SIGIR*, 2007

P. Barna, F. Frasincar, G.-J. Houben, R. Vdovjak, Methodologies for Web information system design, in: *Proc. ITCC*, 2003, URL citeseer.ist.psu.edu/barna03methodologies.html

K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. Blei, M. Jordan, Matching words and pictures, *JMLR – Special Issue on Machine Learning Methods for Text and Images*, vol. 3, no. 6, pp. 1107–1136, 2003, URL http://citeseer.ist.psu.edu/barnard03matching.html

M. J. Beal, *Variational Algorithms for Approximate Bayesian Inference*, Ph.D. thesis, Gatsby Computational Neuroscience Unit, University College London, 2003

M. J. Beal, Z. Ghahramani, Variational Bayesian learning of directed graphical models with hidden variables, *Bayesian Analysis*, vol. 1, pp. 793–832, 2006

J. Bellegarda, Exploiting latent semantic information in statistical language modeling, *Proc. IEEE*, vol. 88, no. 8, pp. 1279–1296, 2000

T. Berners-Lee, J. Hendler, O. Lassila, The Semantic Web, *Scientific American*, 2001

M. W. Berry, M. Browne, *Understanding Search Engines: Mathematical Modeling and Text Retrieval*, SIAM, 2nd ed., 2005

M. W. Berry, S. T. Dumais, G. W. O'Brien, Using linear algebra for intelligent information retrieval, Tech. Rep. UT-CS-94-270, University of Tennessee, 1994, URL http://citeseer.ist.psu.edu/berry95using.html

K. Bertels, V. Sima, Y. Yankova, G. Kuzmanov, W. Luk, J. Coutinho, F. Ferrandi, C. Pilato, M. Lattuada, D. Sciuto, A. Michelotti., hArtes: Hardware-software codesign for heterogeneous multicore platforms, *IEEE Micro*, vol. 30, 2010

J. Bezivin, On the unification power of models, *Software and System Modeling (SoSym)*, vol. 4, no. 2, pp. 171–188, 2006

C. Biemann, Ontology learning from text: A survey of methods, *LDV-Forum*, vol. 20, no. 2, pp. 75–93, 2005

C. Biemann, U. Quasthoff, Networks generated from natural language text, in: *Dynamics on and of complex networks: Modeling and Simulation in Science, Engineering and Technology, Part 2*, pp. 167–185, Springer, 2009

D. Blei, T. Griffiths, M. Jordan, J. Tenenbaum, Hierarchical topic models and the nested Chinese restaurant process, in: *Advances in Neural Information Processing Systems 16*, MIT Press, Cambridge, MA, 2004

D. Blei, M. Jordan, Modeling annotated data, in: *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 127–134, ACM Press, 2003

D. Blei, J. Lafferty, A correlated topic model of Science, *Annals of Applied Statistics*, vol. 1, pp. 17–35, 2007

D. Blei, J. McAuliffe, Supervised topic models, in: *Advances in Neural Information Processing Systems*, 2007

D. Blei, A. Ng, M. Jordan, Latent Dirichlet allocation, in: *Advances in Neural Information Processing Systems 14*, MIT Press, Cambridge, MA, 2002

———, Hierarchical Bayesian models for applications in information retrieval, *Bayesian Statistics*, vol. 7, pp. 25–44, 2003a

———, Latent Dirichlet allocation, *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003b, URL `http://www.cs.berkeley.edu/~blei/papers/blei03a.ps.gz`

M. H. Boisot, *Knowledge assets – securing competitive advantage in the information economy*, Oxford University Press, 1999

K. Börner, J. T. Maru, R. L. Goldstone, The simultaneous evolution of author and paper networks, *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. Suppl 1, pp. 5266–5273, 2004, URL `http://www.pnas.org/content/101/suppl.1/5266.abstract`

J. Boyd-Graber, D. M. Blei, Multilingual topic models for unaligned text, in: *Proc. UAI*, 2009

S. Brin, L. Page, The anatomy of a large-scale hypertextual Web search engine, in: *Proc. Computer Networks and ISDN Systems*, 1998

S. Brooks, A. Gelman, General methods for monitoring convergence of iterative simulations, *J. Comput. Graphi. Stat.*, vol. 7, pp. 434–455, 1998

P. Buitelaar, P. Cimiano, B. Magnini (eds.), *Ontology Learning from Text: Methods, Evaluation and Applications*, Amsterdam: IOS Press, 2005

M. Bundschus, *From Text to Knowledge: Bridging the Gap with Probabilistic Graphical Models*, Ph.D. thesis, University of Munich, 2010

M. Bundschus, S. Yu, V. Tresp, A. Rettinger, M. Dejori, H.-P. Kriegel, Hierarchical Bayesian models for collaborative tagging systems, in: *Proc. IEEE International Conference on Data Mining (ICDM 2009)*, Miami, USA, 2009

W. Buntine, M. Hutter, A Bayesian review of the Poisson-Dirichlet process, arXiv:1007.0296v1 [math.ST], 2010

W. Buntine, A. Jakulin, Discrete principal components analysis, in: *Proc. ECML*, 2005

W. L. Buntine, Variational extensions to EM and multinomial PCA, in: *Proc. ECML*, pp. 23–34, 2002

C. Burgess, K. Livesay, K. Lund, Explorations in context space: Words, sentences, discourse, *Discourse Processes*, vol. 25, pp. 211–257, 1998

K. R. Canini, T. L. Griffiths, A nonparametric Bayesian model of multi-level category learning, in: *Proceedings of the 25th AAAI Conference on Artificial Intelligence*, 2011

L. Cao, L. Fei-Fei, Spatially coherent latent topic model for concurrent object segmentation and classification, in: *Proc. ICCV*, 2007

G. Casella, C. P. Robert, Rao-Blackwellisation of sampling schemes, *Biometrika*, vol. 83, no. 1, pp. 81–94, 1996

J. Chang, D. M. Blei, Relational topic models for document networks, in: *AISTATS*, 2009

J. Chang, J. Boyd-Graber, S. Gerrish, C. Wang, D. Blei, Reading tea leaves: How humans interpret topic models, in: *Proc. Neural Information Processing Systems (NIPS)*, 2009

C. Chen, L. Du, W. Buntine, Sampling table configurations for the hierarchical Poisson-Dirichlet process, in: D. M.-M. V. Dimitrios Gunopulos, Thomas Hofmann (ed.), *Proc. ECML/PKDD*, pp. 296–311, Springer, Athens/Greece, 2011

P. P. Chen, The entity-relationship model – toward a unified view of data, *ACM Transactions on Database Systems*, vol. 1, no. 1, pp. 9–36, 1976, URL `http://csc.lsu.edu/news/erd.pdf`

A. M. Collins, E. F. Loftus, A spreading-activation theory of semantic processing, *Psychological Review*, vol. 82, no. 6, pp. 407–428, 1975

P. C. G. Costa, K. B. Laskey, Pr-owl: A framework for probabilistic ontologies, in: *Proceedings of the 2006 conference on Formal Ontology in Information Systems: Proceedings of the Fourth International Conference (FOIS 2006)*, pp. 237–249, IOS Press, Amsterdam, The Netherlands, The Netherlands, 2006, URL `http://dl.acm.org/citation.cfm?id=1566079.1566107`

A. Cruse, *Lexical Semantics*, Cambridge University Press, Cambridge, UK, 1986

S. Dalal, W. Hall, Approximating priors by mixtures of natural conjugate priors, *Journal of the Royal Statistical Society Series B*, vol. 45, pp. 278–286, 1983

H. Daumé, III, *HBC: Hierarchical Bayes Compiler*, 2007

E. Davenport, M. Graham, J. Kennedy, K. Taylor, Managing social capital as knowledge management – some specification and representation issues, in: *Proc. American Society for Information Science and Technology (ASIS&T)*, pp. 101–108, 2003

E. Davenport, L. McLaughlin, *Trust in knowledge management and systems in organizations*, chap. Interpersonal Trust in Online Partnerships: The Challenge of Representation, pp. 107–123, Idea Group, 2004, URL `http://www.opal-tool.net/Opal-Dateien/05Davenport.pdf`, iSBN:1-59140-220-4

J. Davies, D. Fensel, F. van Harmelen (eds.), *Towards the Semantic Web: Ontology-driven Knowledge Management*, Wiley, 2002

S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, R. A. Harshman, Indexing by latent semantic analysis, *Journal of the American Society of Information Science*, vol. 41, no. 6, pp. 391–407, 1990, URL `http://citeseer.ist.psu.edu/deerwester90indexing.html`

A. P. Dempster, N. M. Laird, D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society, Series B (Methodological)*, vol. 39, pp. 1–38, 1977

K. Dentler, R. Cornet, A. ten Teije, N. de Keizer, Comparison of reasoners for large ontologies in the OWL 2 EL profile, *Semantic Web*, vol. 2, no. 2, pp. 71–87, 2011

R. Diestel, *Graph Theory*, Graduate Texts in Mathematics, Springer, 3rd ed., 2005

L. Dietz, S. Bickel, T. Scheffer, Unsupervised prediction of citation influences, in: *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, Oregon, USA, 2007

G. Doyle, C. Elkan, Accounting for word burstiness in topic models, in: *Proceedings of the Twenty-Sixth International Conference on Machine Learning (ICML '09)*, 2009

P. F. Drucker, The discipline of innovation, *Harvard Business Review*, vol. 63, no. 3, pp. 67–72, 1985

L. Du, W. Buntine, H. Jin, A segmented topic model based on the two-parameter Poisson-Dirichlet process, *Machine Learning*, vol. 81, no. 1, pp. 5–19, 2010

D. Dueck, B. J. Frey, Probabilistic sparse matrix factorization, Technical report PSI TR 2004-023., U. Toronto, September 28, 2004, URL `http://www.psi.toronto.edu/pubs/2004/PSI-TR-2004-23.pdf`

C. Eckart, G. Young, The approximation of one matrix by another of lower rank, *Psychometrika*, vol. 1, pp. 211–218, 1936

K. El-Arini, C. Guestrin, Beyond keyword search: Discovering relevant scientific literature, in: *Proc. KDD*, 2011

C. Elkan, Clustering documents with an exponential-family approximation of the Dirichlet compound multinomial distribution, in: *Proc. ICML*, 2006

E. Erosheva, S. Fienberg, J. Lafferty, Mixed membership models of scientific publications, *PNAS*, vol. 101, no. Suppl. 1, pp. 5220–5227, 2004

M. D. Escobar, M. West, Bayesian density estimation and inference using mixtures, *Journal of the American Statistical Association*, vol. 90, pp. 577–588, 1995

B. Everitt, D. Hand, *Finite Mixture Distributions*, Chapman and Hall, London, 1981

C. Fellbaum (ed.), *WordNet, An Electronic Lexical Database*, Cambridge, MA: MIT Press, 1998

T. S. Ferguson, A Bayesian analysis of some nonparametric problems, *Ann. Statist.*, vol. 1, pp. 209–230, 1973

J. Firth, A synopsis of linguistic theory 1930-55, *Studies in linguistic analysis. The Philological Society, Oxford*, pp. 1–32, 1957

M. Fisz, *Probability Theory and Mathematical Statistics*, New York, London, Sydney: John Wiley & Sons., 1963

M. J. Flynn, Some computer organizations and their effectiveness, *IEEE Transactions on Computers*, vol. C-21, pp. 948–960, 1972

P. Foltz, S. Dumais, Personalized information delivery: An analysis of information filtering methods, *Communications of the ACM*, vol. 35, no. 12, p. 5160, 1992

M. Frigge, D. C. Hoaglin, B. Iglewicz, Some implementations of the boxplot, *The American Statistician*, vol. 43, no. 1, p. 50–54, 1989

G. W. Furnas, T. K. Landauer, L. M. Gomez, S. T. Dumais, The vocabulary problem in human-system communication, *Communications of the ACM*, vol. 30, no. 11, pp. 964–971, 1987, URL citeseer.ist.psu.edu/furnas87vocabulary.html

M. M. Gaber (ed.), *Scientific Data Mining and Knowledge Discovery – Principles and Foundations*, Springer-Verlag, 2010

A. Gelman, D. Rubin, Inference from iterative simulation using multiple sequences, *Statistical Science*, vol. 7, pp. 457–511, 1992

S. Geman, D. Geman, Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images, *IEEE 6*, vol. 6, no. 6, pp. 721–741, 1984

J. E. Gentle, *Random Number Generation and Monte Carlo Methods*, Springer, 2003

S. Gerrish, D. M. Blei, A language-based approach to measuring scholarly impact, in: *ICML*, 2010, URL http://www.cs.princeton.edu/~blei/papers/GerrishBlei2010.pdf

Z. Ghahramani, M. J. Beal, *Advanced Mean Field Methods—Theory and Practice*, chap. Graphical models and variational methods, MIT Press, 2000

J. K. Ghosh, R. V. Ramamoorthi, *Bayesian Nonparametrics*, Springer, 2003

C. L. Giles, K. Bollacker, S. Lawrence, CiteSeer: An automatic citation indexing system, *Proc. 3rd ACM Conf. on Digital Libraries*, pp. 89–98, 1998, URL citeseer.csail.mit.edu/article/giles98citeseer.html

W. Gilks, S. Richardson, D. Spiegelhalter, *Markov chain Monte Carlo Methods in Practice*, Chapman & Hall, 1996

W. R. Gilks, P. Wild, Adaptive rejection sampling for Gibbs sampling, *Applied Statistics*, vol. 41, no. 2, pp. 337–348, 1992

M. Girolami, A. Kaban, On an equivalence between PLSI and LDA, in: *Proc. of ACM SIGIR*, 2003, URL http://citeseer.ist.psu.edu/girolami03equivalence.html

R. Giugno, T. Lukasiewicz, P-SHOQ(D): A probabilistic extension of SHOQ(D) for probabilistic ontologies in the Semantic Web, Research report 1843-02-06, INFSYS, Wien, Austria, 2002

P. Gottschalk, *Strategic knowledge management technology*, Idea Group, 2005

A. G. Gray, B. Fischer, J. Schumann, W. L. Buntine, Automatic derivation of statistical algorithms: The EM family and beyond, in: *NIPS*, pp. 673–680, 2002

T. Griffiths, Gibbs sampling in the generative model of Latent Dirichlet Allocation, Tech. rep., Stanford University, 2002, URL `www-psych.stanford.edu/~gruffydd/cogsci02/lda.ps`

T. L. Griffiths, M. Steyvers, A probabilistic approach to semantic representation, *Proc. of the 24 Annual Conference of the Cognitive Science Society*, 2002

———, Finding scientific topics, *Proceedings of the National Academy of Sciences*, vol. 101, no. Suppl. 1, pp. 5228–5235, 2004

T. L. Griffiths, J. B. Tenenbaum, M. Steyvers, Topics in semantic representation, *Psychological Review*, vol. 114, no. 2, pp. 211–244, 2007

T. Gruber, Towards principles for the design of ontologies used for knowledge sharing, *International Journal of Human and Computer Studies*, vol. 43, no. 5/6, pp. 907–928, 1994

Z. Guo, S. Zhu, Z. Zhang, Y. Chi, Y. Gong, A topic model for linked documents and update rules for its estimation, in: *Proc. 24th AAAI Conference on Artificial Intelligence*, 2010

D. Hall, D. Jurafsky, C. D. Manning, Studying the history of ideas using topic models, in: *Proc. EMNLP*, pp. 363–371, 2008

M. Harvey, M. Baillie, I. Ruthven, M. Carman, Tripartite hidden topic models for personalised tag suggestion, in: *Proc. ECIR*, 2010

G. Heinrich, Teamarbeit nach Maß – Expertisemanagement in Organisationsnetzwerken (in German), in: A. Weisbecker, T. Renner, S. Noll (eds.), *Electronic Business – Innovationen, Anwendungen und Technologien*, pp. 52–59, Fraunhofer IRB-Verlag, Stuttgart, 2004, iSBN 3-8167-6621-8

———, Reinforcement focus+context, in: *Proc. 8th Int. Conf. on Computer Graphics and Artificial Intelligence*, Limoges, France, 2005

———, A generic approach to topic models, in: *Proc. European Conf. on Mach. Learn. / Principles and Pract. of Know. Discov. in Databases (ECML/PKDD), Part 1*, pp. 517–532, 2009a

———, Parameter estimation for text analysis, Technical Report No. 09RP008-FIGD, Fraunhofer Institute for Computer Graphics (IGD), http://www.arbylon.net/publications/text-est2.pdf, 2009b, version 2.9 (version 1.0 May 2005)

———, Actors—media—qualities: a generic model for information retrieval in virtual communities, in: *Proc. 7th International Workshop on Innovative Internet Community Systems (I2CS 2007), part of I2CS Jubilee proceedings, Lecture Notes in Informatics, GI*, 2010

———, "Infinite LDA" – Implementing the HDP with minimum code complexity, Technical note TN2011/1, arbylon.net, 2011a

———, Typology of mixed-membership models: Towards a design method, in: *Proc. European Conf. on Mach. Learn. / Principles and Pract. of Know. Discov. in Databases (ECML/PKDD)*, 2011b

G. Heinrich, M. Goesele, Variational Bayes for generic topic models, in: *Proc. 32nd Annual German Conference on Artificial Intelligence (KI2009)*, 2009

G. Heinrich, T. Keim, C. Jung, U. Krafzig, S. Noll, Smart collaboration networks – a toolkit and a vision for creating and predicting partnership, in: *Proc. Int. Conf. eChallenges*, 2005a

G. Heinrich, J. Kindermann, C. Lauth, G. Paass, J. Sanchez-Monzon, Investigating word correlation at different scopes – a latent concept approach, in: *Workshop Lexical Ontology Learning at Int. Conf. Mach. Learning*, 2005b

G. Heinrich, F. Logemann, V. Hahn, C. Jung, G. Figueiredo, W. Luk, *HW/SW co-design for heterogeneous multi-core platforms: The hArtes toolchain*, chap. Audio array processing for telepresence, pp. 173–207, Springer, 2011

E. Hewitt, L. Savage, Symmetric measures on Cartesian products, *Trans. Amer. Math. Soc.*, vol. 80, p. 470–501, 1955

G. Heyer, Soziale Netzwerke und inhaltsbasierte Suche in Peer-to-Peer Systemen, in: B. Job, A. Mehler (eds.), *Interdependenz und Dynamik sozialer und sprachlicher Netzwerke, VS Verlag: Köln*, 2011

G. Heyer, F. Holz, S. Teresniak, Change of topics over time and tracking topics by their change of meaning, in: A. L. N. Fred (ed.), *KDIR 2009: Proc. of Int. Conf. on Knowledge Discovery and Information Retrieval*, INSTICC Press, 2009

P. Hitzler, M. Krötzsch, B. P. an Peter F. Patel-Schneider, S. Rudolph, OWL 2 Web ontology language primer, W3C recommendation, W3C, 2009

T. Hofmann, Probabilistic latent semantic analysis, in: *Proc. of Uncertainty in Artificial Intelligence, UAI'99*, Stockholm, 1999a, URL `http://citeseer.ist.psu.edu/hofmann99probabilistic.html`

———, Probabilistic Latent Semantic Indexing, in: *Proceedings of the 22nd Annual ACM Conference on Research and Development in Information Retrieval*, pp. 50–57, Berkeley, California, 1999b, URL `http://citeseer.ist.psu.edu/article/hofmann99probabilistic.html`

———, Unsupervised learning by probabilistic latent semantic analysis, *Mach. Learn.*, vol. 42, no. 1-2, pp. 177–196, 2001

A. Hogan, A. Harth, The ExpertFinder Corpus 2007 for the benchmarking and development of expert-finding systems, in: *Proc. First International ExpertFinder Workshop, Berlin, Germany*, 2007

C. W. Holsapple, K. Joshi, A collaborative approach to ontology design, *Comm. ACM*, vol. 45, no. 2, 2002

F. Holz, H. F. Witschel, G. Heinrich, G. Heyer, S. Teresniak, An evaluation framework for semantic search in P2P networks, in: *Proc. I2CS'07*, 2007

R. A. Horn, C. R. Johnson, *Matrix Analysis*, Cambridge University Press, 1985

D. J. Hu, L. K. Saul, A probabilistic topic model for music analysis, in: *Proc. NIPS*, 2009

S. Huang, S. Renals, Unsupervised language model adaptation based on topic and role information in multiparty meetings, in: *Proc. Interspeech*, 2008

M. Huysman, E. Wenger, V. Wulf (eds.), *Communities and Technologies*, Dordrecht: Kluwer, 2003

H. Ishwaran, L. F. James, Gibbs sampling methods for stick breaking priors, *Journal of the American Statistical Association*, vol. 96, no. 453, p. 161ff, 2001, URL `citeseer.ist.psu.edu/322692.html`

T. Iwata, K. Saito, N. Ueda, S. Stromsten, T. L. Griffiths, J. B. Tenenbaum, Parametric embedding for class visualization, in: *Proc. NIPS*, 2004

W. Jank, Stochastic variants of EM: Monte Carlo, quasi-Monte Carlo and more, in: *Proc. American Statistical Association*, Minneapolis, Minnesota., 2005

M. Jansche, Parametric models of linguistic count data, in: *Proc. of the 41st Annual Meeting of the Association for Computational Linguistics (ACL 41)*, pp. 288–295, Sapporo, Japan, 2003, URL `http://acl.ldc.upenn.edu//P/P03/P03-1037.pdf`

K. Jarvelin, J. Kekalainen, Cumulated gain-based evaluation of IR techniques, *ACM Transactions on Information Systems*, vol. 20, no. 4, pp. 422–446, 2002

R. Jäschke, M. Grahl, A. Hotho, B. Krause, C. Schmitz, G. Stumme, Organizing publications and bookmarks in BibSonomy, in: H. Alani, N. Noy, G. Stumme, P. Mika, Y. Sure, D. Vrandecic (eds.), *Proc. Workshop on Social and Collaborative Construction of Structured Knowledge CKC 2007 at WWW 2007, Banff, Canada*, 2007

P. N. Johnson-Laird, *Mental models: Towards a cognitive science of language, inference, and consciousness*, Cambridge University Press, 1983

S. Kashoob, J. Caverlee, E. Khabiri, Probabilistic generative models of the social annotation process, in: *Proc. IEEE International Conference on Social Computing (SocialCom'09), Vancouver*, 2009

S. Kataria, P. Mitra, C. Caragea, C. L. Giles, Context sensitive topic models for author influence in document networks, in: *Proc. IJCAI*, 2011

N. Kawamae, Author interest topic model, in: *SIGIR*, pp. 887–888, 2010

C. Kemp, T. L. Griffiths, J. B. Tenenbaum, Discovering latent classes in relational data, *AI Memo*, vol. 2004, no. 19, 2004

Khronos OpenCL Working Group, *The OpenCL Specification, version 1.0.29*, 2008, URL http://khronos.org/registry/cl/specs/opencl-1.0.29.pdf

W. Kintsch, Predication, *Cognitive Science*, vol. 25, pp. 173–202, 2001

T. G. Kolda, D. P. O'Leary, A semidiscrete matrix decomposition for latent semantic indexing information retrieval, *ACM Trans. Inf. Syst.*, vol. 16, no. 4, pp. 322–346, 1998

A. Kontostathis, W. M. Pottenger, A framework for understanding LSI performance, *Information Processing and Management*, vol. 42, no. 1, pp. 56–73, 2006

R. A. Kronmal, A. V. Peterson, On the alias method for generating random variables from a discrete distribution, *The American Statistician*, vol. 33, pp. 214–218, 1979

S. Kullback, R. A. Leibler, On information and sufficiency, *Ann. Math. Stat.*, vol. 22, pp. 79–86, 1951

S. Lacoste-Julien, F. Sha, M. I. Jordan, Disclda: Discriminative learning for dimensionality reduction and classification, in: *NIPS*, 2008, URL http://books.nips.cc/papers/files/nips21/NIPS2008_0993.pdf

T. Landauer, On the computational basis of learning and cognition: Arguments from LSA, *The psychology of Learning and Motivation*, vol. 41, p. 4384, 2002

T. Landauer, S. Dumais, Latent semantic analysis and the measurement of knowledge, in: R. M. Kaplan, J. C. Burstein (eds.), *Educational testing service conference on natural language processing techniques and technology in assessment and education*, Princeton, NJ: Educational Testing Service, 1994

T. Landauer, D. Laham, B. Rehder, M. Schreiner, How well can passage meaning be derived without using word order? A comparison of latent semantic analysis and humans, in: M. G. Shafto, P. Langley (eds.), *Proceedings of the 19th annual meeting of the Cognitive Science Society*, 1997, URL citeseer.ist.psu.edu/landauer97how.html

T. K. Landauer, S. T. Dumais, Solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge, *Psych. Rev.*, vol. 104, no. 2, pp. 211–240, 1997, cognitive view on LSA

K. B. Laskey, MEBN: A language for first-order Bayesian knowledge bases, *Artificial Intelligence*, vol. 172, no. 2-3, 2008

O. Lassila, D. McGuinness, The role of frame-based representation on the Semantic Web, *Electronic transactions on Artificial Intelligence (ETAI) Journal*, 2001

L. Lee, On the effectiveness of the skew divergence for statistical language analysis, in: *Proc. AISTATS*, 2001

E. Lesser (ed.), *Knowledge and social capital: Foundations and applications*, Oxford: Butterworth-Heinemann, 2000

M. Ley, P. Reuther, Maintaining an online bibliographical database: The problem of data quality, in: *Proc. EGC*, pp. 5–10, 2006

W. Li, D. Blei, A. McCallum, Mixtures of hierarchical topics with pachinko allocation, in: *International Conference on Machine Learning*, 2007a, URL http://www.cs.umass.edu/~mccallum/papers/hpam-icml2007.pdf

———, Nonparametric Bayes pachinko allocation, in: *Proc. 23rd Conference on Uncertainty in Artificial Intelligence*, 2007b

W. Li, A. McCallum, Pachinko allocation: DAG-structured mixture models of topic correlations, in: *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pp. 577–584, ACM, New York, NY, USA, 2006, URL http://portal.acm.org/citation.cfm?id=1143917

J. Lin, Divergence measures based on the Shannon entropy, *IEEE Transactions on Information theory*, vol. 37, pp. 145–151, 1991

B. Liu, L. Liu, A. Tsykin, G. J. Goodall, J. E. Green, M. Zhu, C. H. Kim, , J. Li, Identifying functional miRNA-mRNA regulatory modules with correspondence latent Dirichlet allocation, *Bioinformatics*, vol. 26, no. 24, pp. 3105–11, 2010

J. S. Liu, The collapsed Gibbs sampler in Bayesian computations with applications to a gene regulation problems, *Journal of the American Statistical Association*, vol. 89(427), pp. 958–966, 1994

———, *Monte Carlo Strategies in Scientific Computing*, Springer, 2001

D. Lunn, A. Thomas, N. Best, , D. Spiegelhalter, WinBUGS – a Bayesian modelling framework: Concepts, structure, and extensibility, *Statistics and Computing*, vol. 10, pp. 325–337, 2000

S. MacEachern, Dependent nonparametric processes, *Proceedings of the Section on Bayesian Statistical Science, American Statistical Association*, pp. 50–55, 1999

D. J. MacKay, *Information Theory, Inference, and Learning Algorithms*, Cambridge University Press, 2003, URL http://www.inference.phy.cam.ac.uk/itprnn/book.pdf

R. E. Madsen, D. Kauchak, C. Elkan, Modeling word burstiness using the Dirichlet distribution, in: *ICML '05: Proceedings of the Twenty-Second International Conference on Machine Learning*, pp. 545–552, ACM, New York, NY, USA, 2005, URL http://dx.doi.org/10.1145/1102351.1102420

A. Maedche, S. Staab, Ontology learning, in: *Handbook on Ontologies*, pp. 245–268, Springer, 2009

L. Maicher, J. Park (eds.), *Charting the Topic Maps Research and Applications Landscape. Lecture Notes in Computer Science, Band 3873*, Springer Verlag, 2006

G. S. Mann, D. Mimno, A. McCallum, Bibliometric impact measures leveraging topic analysis, in: *Proceedingsof the 6th ACM/IEEE-CS joint conference on Digital libraries (JCDL)*, pp. 65–74, ACM, New York, USA, 2006

C. D. Manning, P. Raghavan, H. Schütze, *Introduction to Information Retrieval*, Cambridge University Press, 2008

C. D. Manning, H. Schütze, *Foundations of Statistical Natural Language Processing*, The MIT Press, Cambridge, Massachusetts, 1999

T. G. Mattson, B. A. Sanders, B. L. Massingill, *Patterns for Parallel Programming*, Addison Wesley, 2004

R. Mayer, A. Rauber, Music genre classification by ensembles of audio and lyrics features, in: *Proc. ISMIR*, pp. 675–680, 2011

A. McCallum, A. Corrada-Emmanuel, X. Wang, The author-recipient-topic model for topic and role discovery in social networks: Experiments with Enron and academic email, Technical Report UM-CS-2004-096, University of Massachusetts, Amherst, 2004

———, Topic and role discovery in social networks, in: *IJCAI*, 2005

A. McCallum, X. Wang, A. Corrada-Emmanuel, Topic and role discovery in social networks with experiments on Enron and academic email, *Journal of Artificial Intelligence Research*, vol. 30, pp. 249–272, 2007

M. McCandless, E. Hatcher, O. Gospodnetić, *Lucene in action*, Manning, 2010

D. L. McGuinness, F. van Harmelen, OWL Web ontology language overview, W3C recommendation, W3C, 2004

G. McLachlan, D. Peel, *Finite Mixture Models*, Wiley, 2000

A. Mehler, Structural similarities of complex networks: A computational model by example of Wiki graphs, *Applied Artificial Intelligence*, vol. 22, no. 7&8, pp. 619–683, 2008

Q. Mei, X. Ling, M. Wondra, H. Su, C. Zhai, Topic sentiment mixture: Modeling facets and opinions in weblogs, in: *WWW*, 2007a

Q. Mei, X. Shen, C. Zhai, Automatic labeling of multinomial topic models, in: *Proc. KDD*, pp. 490–499, 2007b

M. Meila, Comparing clusterings, in: *Proc. 16th Ann. Conf. on Learn. Theory*, 2003

D. Mimno, W. Li, A. McCallum, Mixtures of hierarchical topics with pachinko allocation, in: *ICML '07: Proceedings of the 24th International Conference on Machine Learning*, pp. 633–640, ACM, New York, NY, USA, 2007

D. Mimno, A. McCallum, Expertise modeling for matching papers with reviewers, in: *KDD*, 2007

———, Topic models conditioned on arbitrary features with Dirichlet-multinomial regression, in: *UAI*, 2008, URL `http://www.cs.umass.edu/~mimno/papers/dmr-uai.pdf`

D. Mimno, H. Wallach, A. McCallum, Gibbs sampling for logistic normal topic models with graph-based priors, in: *Proc. NIPS Workshop on Analyzing Graphs*, 2008

D. Mimno, H. Wallach, J. Naradowsky, D. A. Smith, A. McCallum, Polylingual topic models, in: *EMNLP*, 2009, URL `http://www.cs.umass.edu/~mimno/papers/mimno2009polylingual.pdf`

D. Mimno, H. M. Wallach, E. Talley, M. Leenders, A. McCallum, Optimizing semantic coherence in topic models, in: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, Edinburgh, UK*, p. 262–272, 2011

T. Minka, J. Lafferty, Expectation-propagation for the generative aspect model, in: *Proc. UAI*, 2002

T. P. Minka, The Dirichlet-tree distribution, Tech. rep., Justsystem Pittsburgh Research Center, 1999, revised 2004

———, Estimating a Dirichlet distribution, Web, 2000, URL `http://www.stat.cmu.edu/~minka/papers/dirichlet/minka-dirichlet.pdf`

B. Motik, U. Sattler, A comparison of reasoning techniques for querying large description logic ABoxes, in: *Proc. of the 13th International Conference on Logic for Programming Artificial Intelligence and Reasoning (LPAR 2006)*, 2006

K. Murphy, An introduction to graphical models, Web, 2001, URL `http://www.ai.mit.edu/~murphyk/Papers/intro_gm.pdf`

R. Nallapati, A. Ahmed, E. P. Xing, W. W. Cohen, Joint latent topic models for text and citations, in: *Proc. KDD*, 2008

R. Nallapati, W. Cohen, J. Lafferty, Parallelized variational EM for latent Dirichlet allocation: An experimental evaluation of speed and scalability, in: *Proc. ICDM Workshop on High Performance Data Mining*, 2007

R. Nallapati, D. McFarland, C. Manning, TopicFlow model: Unsupervised learning of topic-specific influences of hyperlinked documents, in: *Proc. 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011

D. Navarro, T. Griffiths, M. Steyvers, D. Lee, Modeling individual differences using Dirichlet processes, *Journal of Mathematical Psychology*, vol. 50, pp. 101–122, 2006

R. Navigli, P. Velardi, S. Faralli, A graph-based algorithm for inducing lexical taxonomies from scratch, in: *Proc. of the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011)*, Barcelona, Spain, 2011

R. M. Neal, Markov chain sampling methods for Dirichlet process mixture models, Tech. rep., Department of Statistics, University of Toronto, 1998

———, Slice sampling, *Ann. Statist.*, vol. 31, no. 3, pp. 705–767, 2003

R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, W. Swartout, Enabling technology for knowledge sharing, *AI Magazine*, vol. 12, no. 3, pp. 36–56, 1991

K. Nemoto, P. A. Gloor, R. Laubacher, Social capital increases efficiency of collaboration among Wikipedia editors, in: *Proc. 22nd ACM Conf. on Hypertext and Hypermedia, Eindhoven*, 2011

R. Neumayer, A. Rauber, Integration of text and audio features for genre classification in music information retrieval, in: *Proc. 29th European Conf. on IR Research (ECIR)*, 2007

J. Neville, *Statistical Models and Analysis Techniques for Learning in Relational Data*, Ph.D. thesis, Stanford University, 2006

D. Newman, A. Asuncion, P. Smyth, M. Welling, Distributed algorithms for topic models, *JMLR*, vol. 10, pp. 1801–1828, 2009

D. Newman, C. Chemudugunta, P. Smyth, Statistical entity-topic models, in: *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 680–686, ACM, New York, NY, USA, 2006a

D. Newman, P. Smyth, M. Steyvers, Scalable parallel topic models, *Journal of Intelligence Community Research and Development*, 2006b

X. Ni, J.-T. Sun, J. Hu, Z. Chen, Mining multilingual topics from Wikipedia, in: *Proc. WWW*, 2009, URL `http://www2009.eprints.org/158/`

K. Nigam, A. McCallum, S. Thrun, T. Mitchell, Text classification from labeled and unlabeled documents using EM., *Machine Learning*, vol. 39, no. 2/3, p. 103–134, 2000

K. P. Nigam, *Using Unlabeled Data to Improve Text Classification*, Ph.D. thesis, Carnegie Mellon University, 2001

I. Nonaka, H. Takeuchi, *The Knowledge Creating Company: How Japanese Companies Create the Dynamics of Innovation*, Oxford Univ. Press New York/Oxford, 1995

M. Papagelis, V. Murdock, R. van Zwol, Individual behavior and social influence in online social systems, in: *Proc. 22nd ACM Conf. on Hypertext and Hypermedia, Eindhoven*, 2011

J. Park, S. Hunting, *XML Topic Maps: Creating and Using Topic Maps for the Web*, Addison-Wesley, 2002

L. A. F. Park, Confidence intervals for information retrieval evaluation, in: *Proc. 15th Australasian Document Computing Symposium, Melbourne, Australia*, 2010

B. Parsia, E. Sirin, Pellet: An OWL DL reasoner, Tech. rep., W3C, 2003

J. Pearl, Bayesian networks: A model of self-activated memory for evidential reasoning, in: *Proc. 7th Conf. of the Cognitive Science Society*, pp. 329–334, 1985, UCLA Technical Report CSD-850017

J. Philbin, J. Sivic, A. Zisserman, Geometric LDA: A generative model for particular object discovery, in: *Proc. British Machine Vision Conference*, 2008

J. Pitman, M. Yor, The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator, *The Annals of Probability*, vol. 25, no. 2, pp. 855–900, 1997

M. Polanyi, *Personal knowledge: Towards a Post-Critical Philosophy*, University of Chicago & Press, Chicago, 1974, originally published in 1958

I. Porteous, E. Bart, M. Welling, Multi-HDP: A non-parametric Bayesian model for tensor factorization, in: *Proc. AAAI*, 2008a

I. Porteous, D. Newman, A. Ihler, A. Asuncion, P. Smyth, M. Welling, Fast collapsed Gibbs sampling for latent Dirichlet allocation, in: *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 569–577, ACM, New York, NY, USA, 2008b

S. E. Preece, *A spreading activation network model for information retrieval*, Ph.D. thesis, Universtiy of Illinois at Urbana-Champaign, 1981

J. K. Pritchard, M. Stephens, P. Donnelly, Inference of population structure using multilocus genotype data, *Genetics*, vol. 155, pp. 945–959, 2000, URL `http://pritch.bsd.uchicago.edu/publications/structure.pdf`

I. Pruteanu-Malinici, L. Ren, J. Paisley, E. Wang, L. Carin, Hierarchical Bayesian modeling of topics in time-stamped documents, *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 32, no. 6, pp. 996–1011, 2010

J. Pujol, R. Sangüesa, J. Delgado, *Web Intelligence*, chap. A Ranking Algorithm Based on Graph Topology to Generate Reputation or Relevance, pp. 382–395, Springer, 2003

M. Purver, K. Körding, T. L. Griffiths, J. Tenenbaum, Unsupervised topic modelling for multi-party spoken discourse, in: *Proc. ACL*, 2006

R. Putnam, *Bowling Alone: The Collapse and Revival of American Community*, New York: Simon & Schuster, 2000

D. R. Radev, P. Muthukrishnan, V. Qazvinian, The ACL anthology network corpus, in: *Proceedings, ACL Workshop on Natural Language Processing and Information Retrieval for Digital Libraries*, Singapore, 2009

H. Raiffa, R. Schlaifer, *Applied Statistical Decision Theory*, Cambridge MA: Harvard University Press., 1961

D. Ramage, S. Dumais, D. Liebling, Characterizing microblogs with topic models, in: *Proc. ICWSM*, 2010, URL `http://www.stanford.edu/~dramage/papers/twitter-icwsm10.pdf`

D. Ramage, P. Heymann, C. D. Manning, H. Garcia-Molina, Clustering the tagged Web, in: *Proc. WSDM*, 2009

R. Rapp, Word sense discovery based on sense descriptor dissimilarity, in: *Proc. Machine Translation Summit IX, New Orleans.*, 2003

M. Rashid, F. Ferrandi, K. Bertels, hArtes design flow for heterogeneous platforms, in: *Proc. 10th International Symposium on Quality of Electronic Design (ISQED)*, pp. 330–338, 2009

C. E. Rasmussen, The infinite Gaussian mixture model, *Advances in Neural Information Processing Systems 12*, pp. 554–560, 2000

T. Reichling, K. Schubert, V. Wulf, Matching human actors based on their texts: Design and evaluation of an instance of the ExpertFinding framework, in: *Proceedings of GROUP 2005*, New York: ACM-Press, 2005

J. Reisinger, M. Pasca, Latent variable models of concept-attribute attachment, in: *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP*, 2009

C. van Rijsbergen, Retrieval effectiveness, *Progress in Communication Sciences*, vol. 1, pp. 91–118., 1979

———, *The geometry of Information Retrieval*, Cambridge University Press, 2004

C. Robert, G. Casella, *Monte-Carlo Statistical Methods*, New York: Springer-Verlag, 2nd ed., 2004

M. Rosen-Zvi, C. Chemudugunta, T. Griffiths, P. Smyth, M. Steyvers, Learning author-topic models from text corpora, *ACM Trans. Inf. Syst.*, vol. 28, no. 1, pp. 4:1–4:38, 2010, URL `http://doi.acm.org/10.1145/1658377.1658381`

M. Rosen-Zvi, T. Griffiths, M. Steyvers, P. Smyth, The author-topic model for authors and documents, in: *Proc. 20th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2004

B. Russell, *Lectures on Logical Atomism, in Logic and Knowledge*, London: Allen and Unwin, 1956

S. Russell, P. Norvig, *Artificial Intelligence A Modern Approach*, Prentice Hall, 2000

G. Salton, M. J. McGill, *Introduction to Modern Information Retrieval*, McGraw-Hill, 1983

H. Scarbrough, J. Swan, Knowledge communities and innovation, *Trends in Communication*, vol. Special Issue on Communities of Practice, pp. 7–20, 2001

H. Schütze, Dimensions of meaning, in: *Proceedings of Supercomputing '92, Minneapolis.*, pp. 787–796, 1992

M. Seeger, H. Nickisch, Fast convergent algorithms for expectation propagation approximate Bayesian inference, in: *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011

J. Sethuraman, A constructive definition of Dirichlet priors, *Statistica Sinica*, vol. 4, pp. 639–650, 1994

R. Shachter, Bayes-Ball: The rational pastime (for determining irrelevance and requisite information in belief networks and influence diagrams), in: G. Cooper, S. Moral (eds.), *Proc. 14th Conf. Uncertainty in Artificial Intelligence*, pp. 480–487, Morgan Kaufmann, San Francisco, CA, 1988

M. M. Shafiei, E. E. Milios, Latent Dirichlet co-clustering, in: *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*, pp. 542–551, IEEE Computer Society, Washington, DC, USA, 2006

C. E. Shannon, *A Mathematical Theory of Communication*, University of Illinois Press, Urbana, IL (reprinted 1998), 1949

J. Sinkkonen, J. Parkkinen, J. Aukia, S. Kaski, A simple infinite topic mixture for rich graphs and relational data, in: *Proc. NIPS Workshop on Analyzing Graphs: Theory and Applications*, 2008

E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, Y. Katz, Pellet: A practical OWL-DL reasoner, *Web Semantics: science, services and agents on the World Wide Web*, vol. 5, no. 2, pp. 51–53, 2007

S. Smolnik, *Wissensmanagement mit Topic Maps in kollaborativen Umgebungen – Identifikation, Explikation und Visualisierung von semantischen Netzwerken in organisationalen Gedächtnissen*, Ph.D. thesis, Universität Paderborn, 2006

J. Sowa, *Knowledge Representation: Logical, Philosophical, and Computational Foundations*, Brooks Cole Publishing Co., Pacific Grove, CA, 2000

K. Spärck Jones, C. van Rijsbergen, Report on the need for and provision of an "ideal" information retrieval test collection, British Library Research and Development Report 5266, Computer Laboratory, University of Cambridge, 1975

V. Spiliopoulos, G. A. Vouros, V. Karkaletsis, Mapping ontologies elements using features in a latent space, in: *Proc. IEEE/WIC/ACM International Conference on Web Intelligence*, 2007

S. Staab, R. Studer (eds.), *Handbook on Ontologies*, International Handbooks on Information Systems, Springer Verlag, 2nd ed., 2009

T. Stahl, M. Voelter, *Model-Driven Software Development: Technology, Engineering, Management*, Wiley, 2006

M. Steyvers, T. Griffiths, Probabilistic topic models, in: T. Landauer, D. Mcnamara, S. Dennis, W. Kintsch (eds.), *Latent Semantic Analysis: A Road to Meaning.*, Laurence Erlbaum, 2006

M. Steyvers, P. Smyth, M. Rosen-Zvi, T. Griffiths, Probabilistic author-topic models for information discovery, in: *Proc. 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004

H. Stockinger, Defining the Grid: A snapshot on the current view, *J Supercomput*, vol. 42, pp. 3–17, 2007

R. E. Story, An explanation of the effectiveness of latent semantic indexing by means of a Bayesian regression model, *Information Processing and Management*, vol. 32, no. 3, pp. 329–344, 1996, URL http://citeseer.ist.psu.edu/story96explanation.html

E. B. Sudderth, *Graphical models for visual object recognition and tracking*, Ph.D. thesis, MIT, 2006

J. Sung, Z. Ghahramani, S.-Y. Bang, Latent-space variational Bayes, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 12, pp. 2236–2242, 2008

R. S. Sutton, A. G. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, Cambridge, MA, 1998

E. M. Talley, D. Newman, D. Mimno, B. W. Herr II, H. M. Wallach, G. A. P. C. Burns, A. G. M. Leenders, A. McCallum, Database of NIH grants using machine-learned categories and graphical clustering, *Nature Methods*, vol. 8, p. 443–444, 2011

V. A. Tamma, *An Ontology Model supporting Multiple Ontologies for Knowledge sharing*, Ph.D. thesis, University of Liverpool, 2001

A. S. Tanenbaum, *Structured Computer Organization*, Pearson/Prentice–Hall, 5th ed., 2006

J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, Z. Su, ArnetMiner: Extraction and mining of academic social networks, in: *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pp. 990–998, ACM, New York, NY, USA, 2008, URL http://doi.acm.org/10.1145/1401890.1402008

J. Tang, J. Zhang, D. Zhang, L. Yao, C. Zhu, J. Li, ArnetMiner: An expertise oriented search system for Web community, in: J. Golbeck, P. Mika (eds.), *Proc. Semantic Web Challenge 2007*, 2007

Y. Teh, A hierarchical Bayesian language model based on Pitman-Yor processes, in: *Proc. 21st ICCL and 44th ACL*, pp. 985–992, 2006

Y. Teh, M. Jordan, M. Beal, D. Blei, Hierarchical Dirichlet processes, Tech. Rep. 653, Department of Statistics, University of California at Berkeley, 2004

———, Hierarchical Dirichlet processes, *Journal of the American Statistical Association*, vol. 101, pp. 1566–1581, 2006

Y. W. Teh, Dirichlet processes, in: C. Sammut, G. I. Webb (eds.), *Encyclopedia of Machine Learning (submitted)*, Springer, 2007

Y. W. Teh, M. I. Jordan, Hierarchical Bayesian nonparametric models with applications, in: N. Hjort, C. Holmes, P. Müller, S. Walker (eds.), *To appear in Bayesian Nonparametrics: Principles and Practice*, Cambridge University Press, 2009

Y. W. Teh, K. Kurihara, M. Welling, Collapsed variational inference for HDP, in: *Advances in Neural Information Processing Systems*, vol. 20, 2008

Y. W. Teh, D. Newman, M. Welling, A collapsed variational Bayesian inference algorithm for latent Dirichlet allocation, in: *Advances in Neural Information Processing Systems*, vol. 19, 2007

I. Titov, R. McDonald, Modeling online reviews with multi-grain topic models, in: *Proc. 17th International World Wide Web Conference (WWW-2008)*, Beijing, China, 2008

D. Titterington, A. Smith, U. Makov, *Statistical Analysis of Finite Mixture Distributions*, Wiley, New York, 1985

M. Wahabzada, Z. Xu, K. Kersting, Topic models conditioned on relations, in: J. Balcazar, F. Bonchi, A. Gionis, M. Sebag (eds.), *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, Springer, Barcelona, Spain, 2010

M. J. Wainwright, M. I. Jordan, Graphical models, exponential families, and variational inference, Tech. rep., EECS Dept., University of California, Berkeley, 2003

A. J. Walker, An efficient method for generating discrete random variables with general distributions, *ACM Transaction on Mathematical Software*, vol. 3, pp. 253–256, 1977

H. Wallach, D. Mimno, A. McCallum, Rethinking LDA: Why priors matter, in: *Proc. NIPS*, 2009a

H. Wallach, I. Murray, R. Salakhutdinov, D. Mimno, Evaluation methods for topic models, in: *Proc. ICML, Montreal, Quebec.*, 2009b

H. M. Wallach, *Structured Topic Models for Language*, Ph.D. thesis, University of Cambridge, 2008

T. Wandmacher, How semantic is latent semantic analysis?, in: *Proc. TALN/RECITAL'05, Dourdan, France*, 2005

T. Wandmacher, E. Ovchinnikova, T. Alexandrov, Does latent semantic analysis reflect human associations?, in: *Proc. Lexical Semantics workshop at ESSLLI'08, Hamburg, Germany*, 2008

C. Wang, D. Blei, F.-F. Li, Simultaneous image classification and annotation, in: *Proc. CVPR*, 2009a

C. Wang, D. M. Blei, D. Heckerman, Continuous time dynamic topic models, in: *Proc. UAI*, 2008, URL `http://uai2008.cs.helsinki.fi/UAI_camera_ready/wang.pdf`

F.-Y. Wang, K. M. Carley, D. Zeng, W. Mao, Social computing: From social informatics to social intelligence, *Intelligent Systems, IEEE*, vol. 22, pp. 79–83, 2007

X. Wang, E. Grimson, Spatial latent Dirichlet allocation, in: *Proc. NIPS*, 2007

X. Wang, X. Ma, W. E. L. Grimson, Unsupervised activity perception in crowded and complicated scenes using hierarchical Bayesian models, *IEEE Trans. Pat. Anal. and Mach. Intell.*, vol. 31, pp. 539–555, 2009b

X. Wang, A. McCallum, Topics over time: A non-Markov continuous-time model of topical trends, in: *KDD*, 2006

X. Wang, N. Mohanty, A. McCallum, Group and topic discovery from relations and their attributes, in: *Neural Informaion Processing Systems (NIPS)*, 2006

S. Wasserman, K. Faust, *Social Network Analysis: Methods and Applications*, Cambridge University Press, 1994

W. Wei, P. Barnaghi, A. Bargiela, Probabilistic topic models for learning terminological ontologies, *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 7, pp. 1028–1040, 2010

X. Wei, *Topic Models in Information Retrieval*, Ph.D. thesis, University of Massachusetts Amherst, 2007

X. Wei, W. B. Croft, LDA-based document models for ad hoc retrieval, in: *Proc. SIGIR*, 2006

T. Weilkiens, *Systems engineering with SysML/UML: Modeling, analysis, design*, Morgan Kaufmann, 2007

E. Wenger, *Communities of Practice: Learning, Meaning, and Identity*, Cambridge University Press, 1998

J. White, ACM opens portal, *Commun. ACM*, vol. 44, no. 7, pp. 14–ff, 2001

J. M. Winn, *Variational Message Passing and its Applications*, Ph.D. thesis, University of Cambridge, 2004

H. Witschel, T. Böhme, Evaluating profiling and query expansion methods for P2P information retrieval, in: *Proc. of the 2005 ACM Workshop on Information Retrieval in Peer-to-Peer Networks (P2PIR)*, 2005

H. Witschel, F. Holz, G. Heinrich, S. Teresniak, An evaluation measure for distributed information retrieval systems, in: *Proc. of the 30th European Conference on Information Retrieval (ECIR)*, 2008

H. F. Witschel, Multi-level association graphs – a new graph-based model for information retrieval, in: *Proc. HLT/NAACL-07 Workshop TextGraphs-2: Graph-Based Algorithms for Natural Language Processing*, pp. 9–16, 2007

L. Wittgenstein, *Tractatus Logico-philosophicus (transl. by C. K. Ogden and F. P. Ramsey)*, Routledge & Kegan Paul, London, 1922

Z. Xu, V. Tresp, K. Yu, H.-P. Kriegel, Infinite hidden relational models, in: *Proc. 22nd Conference in Uncertainty in Artificial Intelligence UAI*, 2006

F. Yan, N. Xu, Y. A. Qi, Parallel inference for latent Dirichlet allocation on graphics processing units, in: *Proc. NIPS*, 2009

L. Yao, D. Mimno, A. McCallum, Efficient methods for topic model inference on streaming document collections, in: *Proc. KDD'09*, 2009

X. Yi, J. Allen, A comparative study of utilizing topic models for information retrieval, in: *Proc. ECIR*, Napa Valley, California, USA, 2009

K. Yu, S. Yu, V. Tresp, Multilabel informed latent semantic indexing, in: *Proc. SIGIR'05*, 2005

S. Yu, K. Yu, V. Tresp, H.-P. Kriegel, Variational Bayesian Dirichlet-multinomial allocation for exponential family mixtures, in: *Proc. ECML 2006*, 2006

B. Zhao, E. P. Xing, HM-BiTAM: Bilingual topic exploration, word alignment, and translation, in: *NIPS*, 2007, URL `http://books.nips.cc/papers/files/nips20/NIPS2007_0188.pdf`

J. Ziman, *Real science*, Cambridge University Press, 2000

G. K. Zipf, *Human behavior and the principle of least effort: An introduction to human ecology*, Addison-Wesley, 1949