UNIVERSITY OF CAMBRIDGE

Structured Topic Models for Language

Hanna M. Wallach

B.A., University of Cambridge (2001); M.Sc., University of Edinburgh (2002)

Newnham College
University of Cambridge

THESIS

Submitted for the degree of
Doctor of Philosophy, University of Cambridge

2008

Declaration

I hereby declare that my dissertation, entitled "Structured Topic Models for Language", is not substantially the same as any that I have submitted for a degree or diploma or other qualification at any other university. No part of my dissertation has already been, or is concurrently being, submitted for any degree, diploma, or other qualification. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and acknowledgements. This dissertation does not exceed sixty thousand words in length.

Abstract

This thesis introduces new methods for statistically modelling text using topic models. Topic models have seen many successes in recent years, and are used in a variety of applications, including analysis of news articles, topic-based search interfaces and navigation tools for digital libraries. Despite these recent successes, the field of topic modelling is still relatively new and there remains much to be explored. One noticeable absence from most of the previous work on topic modelling is consideration of language and document structure—from low-level structures, including word order and syntax, to higher-level structures, such as relationships between documents.

The focus of this thesis is therefore structured topic models—models that combine latent topics with information about document structure, ranging from local sentence structure to inter-document relationships. These models draw on techniques from Bayesian statistics, including hierarchical Dirichlet distributions and processes, Pitman-Yor processes, and Markov chain Monte Carlo methods. Several methods for estimating the parameters of Dirichlet-multinomial distributions are also compared.

The main contribution of this thesis is the introduction of three structured topic models. The first is a topic-based language model. This model captures both word order and latent topics by extending a Bayesian topic model to incorporate n-gram statistics. A bigram version of the new model does better at predicting future words than either a topic model or a trigram language model. It also provides interpretable topics.

The second model arises from a Bayesian reinterpretation of a classic generative dependency parsing model. The new model demonstrates that parsing performance can be substantially improved by a careful choice of prior and by sampling hyperparameters. Additionally, the generative nature of the model facilitates the inclusion of latent state variables, which act as specialised part-of-speech tags or "syntactic topics".

The third is a model that captures high-level relationships between documents. This model uses nonparametric Bayesian priors and Markov chain Monte Carlo methods to infer topic-based document clusters. The model assigns a higher probability to unseen test documents than either a clustering model without topics or a Bayesian topic model without document clusters. The model can be extended to incorporate author information, resulting in finer-grained clusters and better predictive performance.

Acknowledgements

Firstly, I would like to thank my supervisor, David MacKay. I have learnt a huge amount from David—his inspiration, insight and feedback have been invaluable. I am also very grateful to him for his tolerance in letting me to pursue my own academic path: During my Ph.D., I've had the fortunate experience of working with three fantastic research groups, providing me with academic opportunities that would not have otherwise been possible, and a group of talented colleagues and good friends.

I would also like to thank Fernando Pereira, with whom I worked at the University of Pennsylvania. Not only does Fernando have an astounding breadth and depth of knowledge, he is also very generous with his time. I am grateful to him for his support and encouragement—as well as many great conversations—during my time at Penn.

Most recently, I have worked with Andrew McCallum at the University of Massachusetts, Amherst. Andrew's enthusiasm and positive outlook have made the final phase of my Ph.D. much more pleasant. Like Fernando, Andrew is also very generous with his time, and always willing to talk about research, for which I am grateful.

I am also grateful to my collaborators: David Mimno, with whom I worked on parts of chapters 2 and 5; Charles Sutton and Andrew McCallum, whose input and high-level advice on chapter 4 were extremely useful; and Mark Dredze, whose discussions, feedback and collaboration on an earlier project influenced the work in chapter 5.

There are, of course, many other people whose advice, discussions, comments and feedback have greatly contributed to my Ph.D. work, both directly and indirectly. At the least, I would like to acknowledge by name Ryan Adams, Moray Allan, John Blitzer, Alex Braunstein, Phil Cowans, Nikhil Dinesh, Rob Hall, Katherine Heller, Julia Hockenmaier, Shane Jensen, Gideon Mann, Ryan McDonald, Iain Murray, Ed Ratzer, Fei Sha, David Stern, Xuerui Wang, Kilian Weinberger, Seb Wills, and John Winn.

I would also like to acknowledge my friends for their continued support, particularly Chris Ball, Madeleine Price Ball, Anne Weinberger Bracy, Margaret Delap, Daf Harries, Mako Hill, Mika Matsuzaki, Alastair Napier, Aline Normoyle, Sarah Osentoski, Alison Sillence, Tom Sillence, Jeff Vaughan, Geoff Washburn, and Jenn Wortman.

I would also like to thank my parents, Rob and Robin, and my sister, Rachael. Their support, encouragement and excellent sense of humour have been truly invaluable.

And, of course, there is Nick. Without Nick's companionship, my Ph.D. would have been a very different experience—a much less enjoyable one. Thank you, Nick.

To my grandmother, Susi.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

This thesis presents new methods for statistically modelling text. The increasing abundance of electronic texts creates both problems and opportunities. Although people are easily overwhelmed by the quantity and variety of available data, these data also provide a fantastic opportunity for researchers, who can build better models of text and language to improve, for example, tools for navigating, organising and managing document collections and systems for predictive text entry and speech recognition.

Two widely-used models of text are probabilistic topic models and n-gram language models. Probabilistic topic models (Steyvers and Griffiths, 2007) capture semantic properties of documents, and have been used in analysis of news articles[1] (Newman et al., 2006) and scientific papers[2] (Blei and Lafferty, 2007), topic-based search interfaces[3], and navigation tools for digital libraries (Mimno and McCallum, 2007). In contrast, n-gram language models (Chen and Goodman, 1998) focus on representing local linguistic structure, as expressed by word order. Language models form an important component of many systems: For example, cell phones use language models for predictive text entry[4], while speech recognition systems use language models to disambiguate acoustically similar phrases (Rabiner and Juang, 1993; Jelinek, 1998).

Probabilistic topic models, such as latent Dirichlet allocation (Blei et al., 2003) and probabilistic latent semantic analysis (Hofmann, 1999, 2001), model documents as finite mixtures of specialised distributions over words, known as topics. An important assumption underlying these topic models is that documents are generated by first choosing a document-specific distribution over topics, and then repeatedly selecting a topic from this distribution and drawing a word from the topic selected. Word order is ignored—each document is modelled as a "bag-of-words". The weakness of this approach, however, is that word order is an important component of document struc-

[1] e.g., News Articles Browser, `http://yarra.ics.uci.edu/topic/nyt/`
[2] e.g., A browsable model of the journal *Science*, `http://www.cs.cmu.edu/~lemur/science/`
[3] e.g., Rexa digital library and search engine, `http://rexa.info/`
[4] e.g., T9 Text Input, `http://www.t9.com/`

ture, and is not irrelevant to topic modelling. For example, two sentences may have the same unigram statistics but be about quite different topics. Information about the order of words used in each sentence may help disambiguate possible topics.

n-gram language models (Good, 1953; Jelinek and Mercer, 1980; Katz, 1987; Witten and Bell, 1991; Kneser and Ney, 1995; MacKay and Peto, 1995; Teh, 2006) decompose the probability of a string of text (such as a sentence, or document) into a product of probabilities of individual words given some number of previous words. Put differently, these models assume that documents are generated by drawing each word from a probability distribution specific to the context consisting of the immediately preceding words. By conditioning word generation on a short sequence of previous words, n-gram language models can be said to use local linguistic structure. One flaw in this method is that word usage can be highly topic-dependent. For instance, "I'll be in the—" is likely to be followed the word "pub" in an email about weekend plans. In an email about a business meeting, however, the word "office" is much more likely.

This thesis addresses the above shortcomings of current probabilistic topic models and n-gram language models, by combining ideas from both modelling approaches.

Word order is just one kind of simple structure present in language. Other linguistic structures, such as syntax, are just as important, perhaps even more so. Dependency grammars (Tesnière, 1959) model syntactic relationships between words in a sentence by treating each word as the dependent of some other word in the sentence. For instance, in the phrase, "the girl hit the ball," the nouns "girl" and "ball" are respectively the subject and object of the verb "hit." As a result, "girl" and "ball" are both considered to be direct dependents of "hit". Another contribution of this thesis is a new dependency model, which reinterprets and extends a classic dependency parser (Eisner, 1996a,b) using a Bayesian perspective and ideas from latent variable topic models.

Document collections also exhibit structure at higher levels, including structure across document boundaries. For instance, academic papers can be thought of as arising from particular groups or communities of individuals working on closely related topics. Models that account for this kind of high-level structure, by capturing latent document groupings, can be useful for organising and navigating document collections. The final contribution of this thesis is therefore concerned with incorporating inter-document structure, as represented by document groupings, into topic models.

Models that combine document structure with latent variables, as described above, are examples of *structured topic models*—in some cases structure refers to local word order or syntactic structure within a sentence, while in other cases it refers to higher-level semantic structure between documents. This thesis addresses the need for such structured topic models for language data, demonstrating that this is an important new research area with much to offer in the way of powerful models and results.

1.1 Bayesian Modelling

In Bayesian statistics, probabilities are used to describe *degrees of belief*. This interpretation is both intuitively appealing and mathematically motivated: Any set of beliefs can be mapped onto probabilities, so long as they satisfy Cox's axioms—a simple set of consistency rules (Cox, 1946). Furthermore, under this definition, probability can be considered to be a direct extension of Boolean algebra (Jaynes, 2003). Using Cox's axioms, probabilities can be used to describe assumptions and to make inferences under those assumptions. This use of probabilities is extremely powerful: Not only does it enable inferences to be drawn in a consistent fashion, even under uncertain information, but it ensures the explicit statement of assumptions. This approach is particularly useful for modelling text data, where it is often the case that the only certainties are word identities—other information, such as underlying topics, syntactic structure and relationships between documents, are unknown. Furthermore, this approach gives rise to a modelling framework in which assumptions about the structure and properties of language must be explicitly stated, resulting in more interpretable models.

1.2 Overview

The next chapter provides a computational foundation for the other chapters in the thesis. I introduce two fixed-point methods for estimating the hyperparameters of a Dirichlet-multinomial distribution, and compare these methods with several previously introduced algorithms, demonstrating that an algorithm originally introduced by MacKay and Peto (1995) and one of the two new methods are more than an order of magnitude faster than other estimation techniques for such distributions. I also explain how to incorporate a gamma hyperprior into the new fixed-point iterations, and describe how the log gamma recurrence relation may be used to efficiently compute the log probability of data under a Dirichlet-multinomial distribution. The derivation of the new algorithms for estimating hyperparameters and computing log probability, along with the inclusion of a gamma hyperprior, are joint work with David Mimno.

The main work in this thesis consists of three chapters: Chapter 3 presents a new n-gram language model that is based on the notion of topics. This model combines n-gram word statistics and latent topic variables by extending a well-known Bayesian topic model—latent Dirichlet allocation (Blei et al., 2003)—to include properties of a hierarchical Dirichlet language model (MacKay and Peto, 1995). I explore several variants of this model, including different priors and model estimation techniques, and derive a new "left-to-right" algorithm that may be used to compute the information rate of unseen test data by sequentially approximating the marginalisation over latent topics. I show that a bigram version of the new topic-based language model exhibits better predictive performance than either a trigram hierarchical Dirichlet language

model or latent Dirichlet allocation. Additionally, the results provide insight into modelling choices that prevent inferred topics from being dominated by stop words. An earlier version of the work in this chapter was presented at ICML[5] (Wallach, 2006).

Chapter 4 introduces a Bayesian dependency parsing model for natural language, based on the hierarchical Pitman-Yor process (Pitman and Yor, 1997; Teh, 2006). I show that a classic dependency parser (Eisner, 1996a,b) can be substantially improved by (a) using a hierarchical Pitman-Yor process as a prior over the distribution over dependents of a word, and (b) sampling the parameters of the prior. These modelling choices give roughly equal improvements in parse accuracy. An advantage of using a Bayesian approach is the ease with which other latent variables can be included in the model. I propose a second Bayesian dependency parsing model in which latent state variables mediate the relationships between words and their dependents. The model clusters parent–child dependencies into states using a similar approach to that employed by Bayesian topic models when clustering words into topics. The latent states may be viewed as specialised part-of-speech tags or "syntactic topics" that arise from the relationships between words and their dependents. This is verified by inspection of the inferred states and by showing that they lead to modestly improved accuracy when substituted for part-of-speech tags in the parsing model. The work in this chapter was done with input from Charles Sutton and Andrew McCallum, who provided useful discussions and high-level advice. This work was presented at the Prior Knowledge for Text and Language Processing workshop[6] (Wallach et al., 2008).

In chapter 5, I present a nonparametric Bayesian model for clustering documents into groups using latent topics. The model alternates between clustering documents into groups and inferring latent topics for each document, resulting in a topic-based grouping of documents. The model is evaluated using a collection of academic papers, and assigns a higher probability to unseen test documents than either a word-based clustering model or latent Dirichlet allocation. Additionally, the cluster-specific distributions over topics exhibit a good correspondence with well-known research areas. Finally, I extended the model to incorporate author information by characterising each cluster by two distributions, one over authors and one over topics. This extension results in finer-grained clusters, and highlights the relationships between particular groups of topics and authors. The work in this chapter arose out of collaborations with David Mimno (Mimno et al., 2007) and Mark Dredze (Dredze and Wallach, 2008).

In the final chapter, I summarise the key contributions of this thesis and discuss the implications of these findings as well as possibilities for future exploration.

[5] 23rd International Conference on Machine Learning, `http://www.icml2006.org/`
[6] `http://prior-knowledge-language-ws.wikidot.com/`

Chapter 2

Efficient Computation in Dirichlet-Multinomial Distributions

The work in this chapter provides a foundation for the models presented in subsequent chapters and for other applications of Dirichlet-multinomial distributions. I introduce two new methods for estimating the hyperparameters of a Dirichlet-multinomial distribution and compare them with several previously-introduced methods, using both real and synthetic data. This comparison demonstrates that an algorithm introduced by MacKay and Peto (1995) is the fastest of the methods compared, followed closely a new method, based on the digamma recurrence relation. These methods are both over an order of magnitude faster than the standard estimation techniques (Minka, 2003). The new method is more accurate than MacKay and Peto's algorithm, and can also be extended to incorporate a gamma prior over the hyperparameters. Finally, I show that it is possible to efficiently compute the log probability of data under a Dirichlet-multinomial distribution using the log gamma recurrence relation. These results have implications not only for situations where data are directly modelled using a Dirichlet-multinomial distribution, but also for situations where a Dirichlet-multinomial distribution forms a component of a larger model.

2.1 Dirichlet-Multinomial Distributions

Many applications involve estimating probabilities from count data—these include text-based applications, such as language modelling (Chen and Goodman, 1998; Rosenfeld, 2000) and topic modelling (Steyvers and Griffiths, 2007), where the probabilities of interest are those of observing particular words in some context or topic, and biological applications (Durbin et al., 1999), where the probabilities of interest of-

ten relate to a particular nucleotide occurring at a some position in a DNA sequence. In Bayesian statistics, such data are typically modelled using a *Dirichlet-multinomial model*. Given a set of data $\mathcal{D} = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(D)}\}$, consisting of D instances or contexts, each of which consists of $N_{\cdot|d}$ K-valued observations, it is assumed that these data were generated from a set of D K-dimensional probability vectors $\Theta = \{\boldsymbol{\theta}_d\}_{d=1}^{D}$—one for each context. The probability of the data under these vectors is given by

$$P(\mathcal{D} \mid \Theta) = \prod_{d=1}^{D} \prod_{k=1}^{K} \boldsymbol{\theta}_{k|d}^{N_{k|d}}, \tag{2.1}$$

where the quantity $N_{k|d}$ is the number of observations in context d that were observed to take on value k. It is clear from this expression that these counts contain all the relevant information conveyed by the data about the probability vectors Θ.

Uncertainty about Θ is represented by a prior distribution over possible values. This prior is typically taken to be a Dirichlet distribution (MacKay, 2003):

$$\text{Dir}\left(\boldsymbol{\theta} \mid \alpha \boldsymbol{m}\right) = \frac{\Gamma(\alpha)}{\prod_{k=1}^{K} \Gamma(\alpha m_k)} \prod_{k=1}^{K} \theta_k^{\alpha m_k - 1} \, \delta\left(\sum_{k=1}^{K} \theta_k - 1\right), \tag{2.2}$$

where $\Gamma(\cdot)$ is the gamma function (Davis, 1972), given by

$$\Gamma(x) = \int_0^\infty u^{x-1} e^{-u} \, \mathrm{d}u \tag{2.3}$$

for $x > 0$. If x is an integer, then $\Gamma(x) = (x-1)!$. The Dirichlet distribution is parameterised by $\alpha\boldsymbol{m}$: Probability vector $\boldsymbol{m}$ is the mean of the distribution, also known as the base measure, while α is a concentration parameter that determines the extent to which typical samples from this distribution will differ from the mean $\boldsymbol{m}$. Given a Dirichlet prior and the data $\mathcal{D}$, the posterior distribution of each $\boldsymbol{\theta}_d$ is another Dirichlet with parameters $\{N_{k|d} + \alpha m_k\}_{k=1}^{K}$. The predictive probability of observing outcome k in context d—the original probability of interest—is therefore given by

$$P(k \mid d, \mathcal{D}, \alpha\boldsymbol{m}) = \int \theta_{k|d} \, \text{Dir}\left(\boldsymbol{\theta}_d \mid \{N_{k|d} + \alpha m_k\}_{k=1}^{K}\right) \mathrm{d}^K \boldsymbol{\theta}_d = \frac{N_{k|d} + \alpha m_k}{N_{\cdot|d} + \alpha}, \tag{2.4}$$

where the quantity $N_{k|d}$ is the number of times that outcome k was observed in context d. The quantity $N_{\cdot|d} = \sum_{k=1}^{K} N_{k|d}$ is the total number of observations in context d. The value αm_k acts as an initial "pseudocount" for outcome k in all contexts.

Given the Dirichlet-multinomial model described above, there are typically three tasks of interest: Inferring the model "hyperparameters" $\alpha\boldsymbol{m}$, computing the probability of some observed data under the model, also known as the "evidence", and making predictions about future observations. Performing these tasks as efficiently and accurately as possible is important, especially for applications where they may be re-

peated many times. For example, when inferring latent topics using a topic model, the inference algorithm may alternate between sampling latent topics (and computing the probability of the data given these topic assignments so convergence can be detected) and inferring model hyperparameters (Wallach, 2006). These steps may be repeated several thousand times, so it is particularly desirable for each individual step to take as little time as possible. For language modelling, hyperparameters need only be inferred once. However, the number of hyperparameters can be vast since there must be an m_k for every word in the vocabulary and it is common for vocabulary sizes for large corpora to be as high as 50,000 words (Chen and Goodman, 1998).

2.2 Hyperparameter Inference

In an ideal Bayesian setting, the hyperparameters should be given a proper prior and marginalised over when making predictions, yielding the true predictive distribution:

$$P(k \mid d, \mathcal{D}) = \int P(k \mid d, \mathcal{D}, \alpha\boldsymbol{m})\, P(\alpha\boldsymbol{m} \mid \mathcal{D})\, \mathrm{d}^K \alpha\boldsymbol{m}. \tag{2.5}$$

However, for many applications the posterior distribution over hyperparameters $P(\alpha\boldsymbol{m} \mid \mathcal{D})$ is sufficiently sharply peaked in $\alpha\boldsymbol{m}$ that it is effectively a delta function in comparison with $P(k \mid d, \mathcal{D}, \alpha\boldsymbol{m})$. Equation 2.5 may therefore be approximated by $P(k \mid d, \mathcal{D}, [\alpha\boldsymbol{m}]^\star)$, where $[\alpha\boldsymbol{m}]^\star$ are the optimal hyperparameters (MacKay, 1992).

Assuming an improper, noninformative prior over the hyperparameters, the optimal hyperparameters $[\alpha\boldsymbol{m}]^\star$ are those that maximise the "evidence" or probability of the data given the hyperparameters $P(\mathcal{D} \mid \alpha\boldsymbol{m})$. The evidence is given by

$$P(\mathcal{D} \mid \alpha\boldsymbol{m}) = \prod_{d=1}^{D} \frac{\Gamma(\alpha)}{\Gamma(N_{\cdot|d} + \alpha)} \prod_{k=1}^{K} \frac{\Gamma(N_{k|d} + \alpha m_k)}{\Gamma(\alpha m_k)} \tag{2.6}$$

and is concave in $\alpha\boldsymbol{m}$, meaning that there are no local maxima.

2.3 Estimation Techniques

The primary resource on finding the optimal hyperparameters $[\alpha\boldsymbol{m}]^\star$ of a Dirichlet-multinomial distribution given data $\mathcal{D}$ is by Minka (2003). Minka describes several methods for jointly estimating $\alpha^\star = \sum_k [\alpha m_k]^\star$ and $\boldsymbol{m}^\star$, including:

- a fixed-point iteration on the log evidence,
- a Newton iteration on the log evidence, and
- a fixed-point iteration on the leave-one-out log evidence.

Unfortunately, Minka does not provide empirical results indicating how these methods compare to each other or to less well-known methods. It is therefore hard to tell which estimation method is most appropriate (i.e., fastest and most accurate) for a particular data set without implementing them all. A comparison of hyperparameter estimation methods for data sets with different dimensionalities, numbers of contexts and numbers of observations per context is consequently a much-needed resource.

In this chapter, I describe the three methods mentioned above, along with a fixed-point iteration on the log evidence introduced by MacKay and Peto (1995). I also present two new methods, one based on the digamma recurrence relation and one based on an approximation for digamma differences originally described by MacKay and Peto. I compare these methods in terms of speed and accuracy using several types of data.

2.3.1 Minka's Fixed-Point Iteration

Minka's fixed-point iteration for estimating optimal hyperparameters $[\alpha\boldsymbol{m}]^\star$ may be derived by starting with the logarithm of the evidence $P(\mathcal{D} \mid \alpha\boldsymbol{m})$:

$$\log P(\mathcal{D} \mid \alpha\boldsymbol{m}) = \sum_{d=1}^{D} \left[\log\Gamma(\alpha) - \log\Gamma(N_{\cdot|d} + \alpha) + \sum_{k=1}^{K} \log\Gamma(N_{k|d} + \alpha m_k) - \log\Gamma(\alpha m_k) \right]. \quad (2.7)$$

This function may be bounded from below using the following bounds:

$$\log\Gamma(z) - \log\Gamma(z+n) \geq \log\Gamma(\hat{z}) - \log\Gamma(\hat{z}+n) + [\Psi(\hat{z}+n) - \Psi(\hat{z})]\,(\hat{z} - z) \quad (2.8)$$

and

$$\log\Gamma(z+n) - \log\Gamma(z) \geq \log\Gamma(\hat{z}+n) - \log\Gamma(\hat{z}) + \hat{z}\,[\Psi(\hat{z}+n) - \Psi(\hat{z})]\,(\log z - \log\hat{z}), \quad (2.9)$$

where n is a constant positive integer, z is a "true" positive real number, $\hat{z}$ is an "approximate" positive real number and $\Psi(\cdot)$ is the first derivative of the log gamma function, known as the digamma function (Davis, 1972). Treating the optimal parameter values $[\alpha\boldsymbol{m}]^\star$ as the "true" z and the current estimate $\alpha\boldsymbol{m}$ as the approximation $\hat{z}$,

equations 2.8 and 2.9 may be substituted into equation 2.7, yielding

$$\begin{aligned}\log P(\mathcal{D} \mid [\alpha \boldsymbol{m}]^\star) \geq B([\alpha \boldsymbol{m}]^\star) = \\ \sum_{d=1}^{D} \Bigg[\log \Gamma(\alpha) - \log \Gamma(N_{\cdot|d} + \alpha) + \left[\Psi(N_{\cdot|d} + \alpha) - \Psi(\alpha)\right](\alpha - \alpha^\star) + \\ \sum_{k=1}^{K} \log \Gamma(N_{k|d} + \alpha m_k) + \log \Gamma(\alpha m_k) + \\ \alpha m_k \left[\Psi(N_{k|d} + \alpha m_k) - \Psi(\alpha m_k)\right](\log [\alpha m_k]^\star - \log [\alpha m_k]) \Bigg]. \end{aligned} \tag{2.10}$$

All terms that do not involve $[\alpha \boldsymbol{m}]^\star$ can be grouped into a constant term C:

$$\begin{aligned}\log P(\mathcal{D} \mid [\alpha \boldsymbol{m}]^\star) \geq B([\alpha \boldsymbol{m}]^\star) = \\ \sum_{d=1}^{D} \Bigg[\left[\Psi(N_{\cdot|d} + \alpha) - \Psi(\alpha)\right](-\alpha^\star) + \\ \sum_{k=1}^{K} \alpha m_k \left[\Psi(N_{k|d} + \alpha m_k) - \Psi(\alpha m_k)\right](\log [\alpha m_k]^\star) \Bigg] + C. \end{aligned} \tag{2.11}$$

It is now possible to take the derivative of bound $B([\alpha \boldsymbol{m}]^\star)$ with respect to $[\alpha m_k]^\star$:

$$\begin{aligned}\frac{\partial B([\alpha \boldsymbol{m}]^\star)}{\partial [\alpha m_k]^\star} = \\ \sum_{d=1}^{D} \left[\frac{\alpha m_k \left[\Psi(N_{k|d} + \alpha m_k) - \Psi(\alpha m_k)\right]}{[\alpha m_k]^\star} - \left[\Psi(N_{\cdot|d} + \alpha) - \Psi(\alpha)\right] \right]. \end{aligned} \tag{2.12}$$

Finally, equation 2.12 can be set to zero and solved for $[\alpha m_k]^\star$:

$$[\alpha m_k]^\star = \alpha m_k \frac{\sum_{d=1}^{D} \Psi(N_{k|d} + \alpha m_k) - \Psi(\alpha m_k)}{\sum_{d=1}^{D} \Psi(N_{\cdot|d} + \alpha) - \Psi(\alpha)}. \tag{2.13}$$

When used repeatedly, this fixed-point iteration will result in the convergence of $[\alpha \boldsymbol{m}]^\star$ to the hyperparameter values that maximise $P(\mathcal{D} \mid \alpha \boldsymbol{m})$ as desired.

2.3.2 Minka's Newton Iteration

Minka's Newton iteration may be obtained using the first and second derivatives of the log evidence. The first derivative of the log evidence is given by

$$g_k = \frac{\partial \log P(\mathcal{D} \,|\, \alpha \boldsymbol{m})}{\partial \left[\alpha m_k\right]} = \sum_{d=1}^{D} \left[\Psi(\alpha) - \Psi(N_{\cdot|d} + \alpha) + \Psi(N_{k|d} + \alpha m_k) - \Psi(\alpha m_k)\right]. \tag{2.14}$$

Similarly, the second derivatives, or Hessian, are given by

$$\frac{\partial \log P(\mathcal{D} \,|\, \alpha \boldsymbol{m})}{\partial \left[\alpha m_k\right]^2} = \sum_{d=1}^{D} \left[\Psi'(\alpha) - \Psi'(N_{\cdot|d} + \alpha) + \Psi'(N_{k|d} + \alpha m_k) - \Psi'(\alpha m_k)\right] \tag{2.15}$$

and

$$\frac{\partial \log P(\mathcal{D} \,|\, \alpha \boldsymbol{m})}{\partial \left[\alpha m_k\right] \partial \left[\alpha m_j\right]} = \sum_{d=1}^{D} \left[\Psi'(\alpha) - \Psi'(N_{\cdot|d} + \alpha)\right] \quad k \neq j, \tag{2.16}$$

where $\Psi'(\cdot)$ is the derivative of the digamma function, known as the trigamma function (Davis, 1972). The Hessian may also be written as a $K \times K$ matrix $\boldsymbol{H}$:

$$\boldsymbol{H} = \boldsymbol{Q} + \mathbf{1}\mathbf{1}^T z \tag{2.17}$$

where

$$q_{jk} = \delta\left(j - k\right) \sum_{d=1}^{D} \left[\Psi'(N_{k|d} + \alpha m_k) - \Psi'(\alpha m_k)\right], \tag{2.18}$$

$$z = \sum_{d=1}^{D} \left[\Psi'(\alpha) - \Psi'(N_{\cdot|d} + \alpha)\right], \tag{2.19}$$

and $\mathbf{1}$ is a K-dimensional vectors, whose elements are all 1.

Given the Hessian matrix $\boldsymbol{H}$ and gradient vector $\boldsymbol{g}$ (with elements given by equation 2.14), a single Newton iteration (Nocedal and Wright, 1999) is

$$\alpha \boldsymbol{m} = [\alpha \boldsymbol{m}]^{\text{old}} - \boldsymbol{H}^{-1} \boldsymbol{g}. \tag{2.20}$$

Minka (2003) showed that it is not necessary to explicitly invert or store $\boldsymbol{H}$ when com-

```
1:  while not converged {
2:    abort := false
3:    λ := 1/2
4:    l_old := log P(D | αm)
5:    while true {
6:      if [H^{-1}g]_k < αm_k for all k {
7:        l := log P(D | αm − H^{-1}g)
8:        if l > l_old {
9:          l_old := l
10:         αm := αm − H^{-1}g
11:         λ := λ/2
12:         break
13:       }
14:     }
15:     λ := λ * 2
16:     if λ > 1^20 {
17:       abort := true
18:       break
19:     }
20:   }
21:   if abort = true {
22:     break
23:   }
24: }
```

Algorithm 2.1: Minka's Newton algorithm for optimising $\alpha\boldsymbol{m}$.

puting the update vector $\boldsymbol{H}^{-1}\boldsymbol{g}$. Since $\boldsymbol{H}^{-1}$ can be written as

$$\boldsymbol{H}^{-1} = \boldsymbol{Q}^{-1} - \frac{\boldsymbol{Q}^{-1}\mathbf{1}\mathbf{1}^T\boldsymbol{Q}^{-1}}{\frac{1}{z} + \mathbf{1}^T\boldsymbol{Q}^{-1}\mathbf{1}}, \tag{2.21}$$

the update term $[\boldsymbol{H}^{-1}\boldsymbol{g}]_k$ for each αm_k may be computed directly as follows:

$$[\boldsymbol{H}^{-1}\boldsymbol{g}]_k = \frac{g_k - b}{q_{kk}}, \tag{2.22}$$

where

$$b = \frac{\mathbf{1}^T\boldsymbol{Q}^{-1}\boldsymbol{g}}{\frac{1}{z}\mathbf{1}^T\boldsymbol{Q}^{-1}\mathbf{1}} = \frac{\sum_j \frac{g_j}{q_{jj}}}{\frac{1}{z} + \sum_j \frac{1}{q_{jj}}}. \tag{2.23}$$

Direct calculation of each update term $[\boldsymbol{H}^{-1}\boldsymbol{g}]_k$ saves both computation time and storage space. Minka's entire Newton algorithm is shown in algorithm 2.1.

2.3.3 Minka's "Leave-One-Out" Fixed-Point Iteration

Instead of finding the hyperparameters that maximise the log evidence, as described in section 2.2, it is also possible to approximate the log evidence by the *leave-one-out* log evidence and compute the hyperparameters that optimise this function. This approximation has the advantage of involving no log gamma, digamma or trigamma functions. It is therefore possible to use this approximation to derive a fixed-point iteration that uses no special functions (even log), thereby reducing computation time.

The leave-one-out log evidence is obtained by treating each observation as the last to arrive and computing the log probability of that observation given all other observations and the hyperparameters using the predictive distribution (given in equation 2.4). The sum of these log probabilities is the leave-one-out evidence:

$$\log P(\mathcal{D} \,|\, \alpha\boldsymbol{m}) \simeq L(\alpha\boldsymbol{m}) = \sum_{d=1}^{D}\sum_{k=1}^{K} N_{k|d} \log\left(\frac{N_{k|d} - 1 + \alpha m_k}{N_{\cdot|d} - 1 + \alpha}\right). \tag{2.24}$$

Minka (2003) uses the following bounds to bound equation 2.24 from below:

$$\log(n + z) \geq q \log z - (1 - q)\log n - q \log q - (1 - q)\log(1 - q) \tag{2.25}$$

where

$$q = \frac{\hat{z}}{N + \hat{z}}, \tag{2.26}$$

and

$$\log z \leq \hat{z}^{-1} z - 1 + \log \hat{z}. \tag{2.27}$$

As was the case with the bounds in section 2.3.1, n is a constant positive integer, z is a "true" positive real number, and $\hat{z}$ is an "approximate" positive real number.

The derivative with respect to $[\alpha m_k]^\star$ of the lower bound on the leave-one-out log evidence can be set to zero and solved for $[\alpha m_k]^\star$, yielding the following expression:

$$[\alpha m_k]^\star = \alpha m_k \frac{\sum_{d=1}^{D} \dfrac{N_{k|d}}{N_{k|d} + \alpha m_k}}{\sum_{d=1}^{D} \dfrac{N_{\cdot|d}}{N_{\cdot|d} + \alpha}}. \tag{2.28}$$

When used repeatedly, this fixed-point iteration will cause the hyperparameters $[\alpha\boldsymbol{m}]^\star$ to converge to the values that maximise the leave-one-out log evidence. A significant advantage of this method is that it does not require any special functions, unlike Minka's fixed-point iteration on the log evidence or Newton iteration.

2.3.4 MacKay and Peto's Fixed-Point Iteration

MacKay and Peto (1995) present a fixed-point iteration on the log evidence for use in situations where $[\alpha m_k]^\star < 1$ for all k and $\alpha^\star > 1$. This iteration may be obtained by starting with the first derivative of the log evidence, as given in equation 2.14.

MacKay and Peto note that for $N_{k|d} \geq 1$, the digamma recurrence relation (Davis, 1972) may be used to express $\Psi(N_{k|d} + \alpha m_k) - \Psi(\alpha m_k)$ as follows:

$$\Psi(N_{k|d} + \alpha m_k) - \Psi(\alpha m_k) = \frac{1}{\alpha m_k} + \frac{1}{1 + \alpha m_k} + \cdots + \frac{1}{N_{k|d} - 1 + \alpha m_k} \tag{2.29}$$

$$= \frac{1}{\alpha m_k} + \sum_{f=2}^{N_{k|d}} \frac{1}{f - 1 + \alpha m_k}. \tag{2.30}$$

If $\alpha m_k < 1$, the sum in equation 2.30 can be approximated using a first-order Taylor series expansion around $\alpha m_k = 0$—i.e., a Maclaurin series (Riley et al., 2006):

$$\sum_{f=2}^{N_{k|d}} \frac{1}{f - 1 + \alpha m_k} \simeq \sum_{f=2}^{N_{k|d}} \frac{1}{f - 1} - \alpha m_k \sum_{f=2}^{N_{k|d}} \frac{1}{(f-1)^2} + O([\alpha m_k]^2). \tag{2.31}$$

Substituting this approximation into equation 2.30 gives

$$\Psi(N_{k|d} + \alpha m_k) - \Psi(\alpha m_k) \simeq \frac{1}{\alpha m_k} + \sum_{f=2}^{N_{k|d}} \frac{1}{f - 1} - \alpha m_k \sum_{f=2}^{N_{k|d}} \frac{1}{(f-1)^2}. \tag{2.32}$$

By taking the derivative of Stirling's approximation for the log gamma function (Riley et al., 2006), $\Psi(\alpha)$ and $\Psi(N_{\cdot|d} + \alpha)$ can be approximated as follows:

$$\Psi(\alpha) \simeq \log \alpha - \frac{1}{2\alpha}, \tag{2.33}$$

$$\Psi(N_{\cdot|d} + \alpha) \simeq \log (N_{\cdot|d} + \alpha) - \frac{1}{2(N_{\cdot|d} + \alpha)}. \tag{2.34}$$

These approximations are appropriate for large α. Substituting equations 2.32, 2.33 and 2.34 into the first derivative of the log evidence (equation 2.14) gives

$$\frac{\partial \log P(\mathcal{D} \,|\, \alpha \boldsymbol{m})}{\partial [\alpha m_k]} \simeq$$

$$\sum_{\{d \,|\, N_{k|d} \geq 1\}} \left[\sum_{f=2}^{N_{k|d}} \frac{1}{f - 1} - \alpha m_k \sum_{f=2}^{N_{k|d}} \frac{1}{(f-1)^2} + \frac{1}{\alpha m_k} \right] +$$

$$\sum_{d=1}^{D} \left[\log \alpha - \frac{1}{2\alpha} - \log (N_{\cdot|d} + \alpha) + \frac{1}{2(N_{\cdot|d} + \alpha)} \right]. \tag{2.35}$$

Letting

$$K(\alpha) = -\sum_{d=1}^{D}\left[\log\alpha - \frac{1}{2\alpha} - \log\left(N_{\cdot|d}+\alpha\right) + \frac{1}{2(N_{\cdot|d}+\alpha)}\right] \tag{2.36}$$

$$= \sum_{d=1}^{D}\log\frac{N_{\cdot|d}+\alpha}{\alpha} + \frac{1}{2}\sum_{d=1}^{D}\frac{N_{\cdot|d}}{\alpha(N_{\cdot|d}+\alpha)}, \tag{2.37}$$

equation 2.35 can be written as

$$\frac{\partial\,\log P(\mathcal{D}\,|\,\alpha\boldsymbol{m})}{\partial\,[\alpha m_k]} \simeq \sum_{\{d\,|\,N_{k|d}\geq 1\}}\left[\sum_{f=2}^{N_{k|d}}\frac{1}{f-1} - \alpha m_k\sum_{f=2}^{N_{k|d}}\frac{1}{(f-1)^2} + \frac{1}{\alpha m_k}\right] - K(\alpha), \tag{2.38}$$

Having done this, equation 2.38 can be set to zero and multiplied by $x = 1/\alpha m_k$:

$$\sum_{\{d\,|\,N_{k|d}\geq 1\}}\left[x^2 + x\sum_{f=2}^{N_{k|d}}\frac{1}{f-1} - \sum_{f=2}^{N_{k|d}}\frac{1}{(f-1)^2}\right] - x\,K(\alpha) = 0. \tag{2.39}$$

The resultant equation can then be rearranged as follows:

$$V_k\,x^2 + \left(G_k - K(\alpha)\right)\,x - H_k = 0, \tag{2.40}$$

where

$$V_k = \sum_{\{d\,|\,N_{k|d}\geq 1\}} 1, \tag{2.41}$$

$$G_k = \sum_{\{d\,|\,N_{k|d}\geq 1\}}\sum_{f=2}^{N_{k|d}}\frac{1}{f-1}, \tag{2.42}$$

$$H_k = \sum_{\{d\,|\,N_{k|d}\geq 1\}}\sum_{f=2}^{N_{k|d}}\frac{1}{(f-1)^2}. \tag{2.43}$$

The quantity V_k is the number of contexts in which outcome k has been seen at least once. The expression for G_k can be simplified by rewriting equation 2.42 as

$$G_k = \sum_{\{d\,|\,N_{k|d}\geq 1\}}\sum_{f=2}^{\max_d N_{k|d}}\frac{1}{f-1}\,\delta\left(f \leq N_{k|d}\right). \tag{2.44}$$

The delta function $\delta\left(f \leq N_{k|d}\right)$ ensures that terms involving $f = N_{k|d}+1 \ \ldots \ \max_d N_{k|d}$ are excluded from the sum over f. Having rewritten G_k

in this manner, the order of the sum over d and f can be reversed to give

$$G_k = \sum_{f=2}^{\max_d N_{k|d}} \sum_{\{d \,|\, N_{k|d} \geq 1\}} \frac{1}{f-1} \, \delta\left(f \leq N_{k|d}\right) \tag{2.45}$$

$$= \sum_{f=2}^{\max_d N_{k|d}} \frac{N_{f_k}}{f-1} \tag{2.46}$$

where $N_{f_k} = \sum_{\{d \,|\, N_{k|d} \geq 1\}} \delta\left(f \leq N_{k|d}\right)$ is the number of contexts in which outcome k has been seen f or more times. Similarly, the expression for H_k may be simplified to

$$H_k = \sum_{f=2}^{\max_d N_{k|d}} \frac{N_{f_k}}{(f-1)^2}. \tag{2.47}$$

MacKay and Peto's fixed-point iteration is finally obtained by solving equation 2.40 for $x = 1/\alpha m_k$ using the quadratic formula (Riley et al., 2006). This gives:

$$\alpha m_k = \frac{2V_k}{K(\alpha) - G_k + \sqrt{(K(\alpha) - G_k)^2 + 4H_k V_k}}. \tag{2.48}$$

The optimal hyperparameters $[\alpha \boldsymbol{m}]^\star$ can be found by alternating between using equation 2.48 to find $\alpha \boldsymbol{m}$ and setting α to $\sum_{k=1}^{K} \alpha m_k$ until convergence is reached. This equation has two nice properties: Firstly, it contains no special functions. Secondly, the only term that depends on the hyperparameters is $K(\alpha)$—the other terms depend only on the data and therefore do not need to be recomputed during estimation.

2.3.5 Two New Fixed-Point Iterations

In this section, I present two new methods for estimating Dirichlet-multinomial hyperparameters. Both methods use Minka's fixed-point iteration on the log evidence as a starting point and neither involve any special functions other than log. Unlike MacKay and Peto's fixed-point iteration (described in the previous section) these new methods are valid for all αm_k and α. The first method arises from two observations: Firstly, that identical terms may be grouped together, and secondly, that the difference between two digamma functions may be computed efficiently using the digamma recurrence relation. The second method also involves the grouping of identical terms, but combines this rearrangement with an approximation for the difference between two digamma functions, originally introduced by MacKay and Peto (1995).

Method 1: Using the Digamma Recurrence Relation

The first method may be derived by starting with Minka's fixed-point iteration:

$$[\alpha m_k]^\star = \alpha m_k \frac{\sum_{d=1}^{D} \Psi(N_{k|d} + \alpha m_k) - \Psi(\alpha m_k)}{\sum_{d=1}^{D} \Psi(N_{\cdot|d} + \alpha) - \Psi(\alpha)}. \tag{2.49}$$

Letting $C_k(n)$ be the number of contexts in which k has been seen exactly n times,

$$C_k(n) = \sum_{d=1}^{D} \delta\left(N_{k|d} - n\right), \tag{2.50}$$

the numerator in equation 2.49 may be rewritten as follows:

$$\sum_{d=1}^{D} \Psi(N_{k|d} + \alpha m_k) - \Psi(\alpha m_k) = \sum_{n=1}^{\max_d N_{k|d}} C_k(n) \left[\Psi(n + \alpha m_k) - \Psi(\alpha m_k)\right]. \tag{2.51}$$

Similarly, the denominator may be rewritten as

$$\sum_{d=1}^{D} \Psi(N_{\cdot|d} + \alpha) - \Psi(\alpha) = \sum_{n=1}^{\max_d N_{\cdot|d}} C_{\cdot}(n) \left[\Psi(n + \alpha) - \Psi(\alpha)\right], \tag{2.52}$$

where $C_{\cdot}(n)$ is the number of contexts that contain a total of n observations:

$$C_{\cdot}(n) = \sum_{d=1}^{D} \delta\left(N_{\cdot|d} - n\right). \tag{2.53}$$

For each outcome k, $\{C_k(n)\}_{n=1}^{\max_d N_{k|d}}$ can be considered to be a histogram with $\max_d N_{k|d}$ bins, each containing the number of contexts in which k has been seen exactly n times. Similarly, $\{C_{\cdot}(n)\}_{n=1}^{\max_d N_{\cdot|d}}$ may be viewed as a histogram with $\max_d N_{\cdot|d}$ bins, each containing the number of contexts that contain exactly n observations.

Substituting equations 2.51 and 2.52 into equation 2.49 gives

$$[\alpha m_k]^\star = \alpha m_k \frac{\sum_{n=1}^{\max_d N_{k|d}} C_k(n) \left[\Psi(n + \alpha m_k) - \Psi(\alpha m_k)\right]}{\sum_{n=1}^{\max_d N_{\cdot|d}} C_{\cdot}(n) \left[\Psi(n + \alpha) - \Psi(\alpha)\right]}. \tag{2.54}$$

The extent to which this rearrangement will speed up computation depends on the number of contexts that are identical to each other along some dimension k. If many contexts d have the same count value $N_{k|d} = n$ for some outcome k, then the time taken to compute equation 2.54 will be reduced. The more outcomes for which this is the case, the greater the reduction. Finally, when using the rearrangement as part

of an algorithm that alternates between updating the counts to reflect latent state and estimating the hyperparameters (e.g., when inferring latent topics in a topic model), the histograms do not need to be computed from scratch prior to each round of hyperparameter estimation—they can be incrementally updated as the counts are changed.

Digamma functions are usually computed using algorithm AS 103 (Bernardo, 1976) which relies on an asymptotic expansion involving Bernoulli numbers. However, if the only calculations involving digamma functions are *differences* of digamma functions, then the digamma recurrence relation (Davis, 1972) can be used instead:

$$\Psi(1+z) = \Psi(z) + \frac{1}{z}. \tag{2.55}$$

This identity can be expanded recursively for any positive integer n to give

$$\Psi(n+z) = \Psi(z) + \sum_{f=1}^{n} \frac{1}{f-1+z}. \tag{2.56}$$

Rewriting gives the following expression:

$$\Psi(n+z) - \Psi(z) = \sum_{f=1}^{n} \frac{1}{f-1+z}. \tag{2.57}$$

Substituting equation 2.57 into equation 2.54 gives:

$$[\alpha m_k]^{\star} = \alpha m_k \frac{\sum_{n=1}^{\max_d N_{k|d}} C_k(n) \sum_{f=1}^{n} \frac{1}{f-1+\alpha m_k}}{\sum_{n=1}^{\max_d N_{\cdot|d}} C_{\cdot}(n) \sum_{f=1}^{n} \frac{1}{f-1+\alpha}}. \tag{2.58}$$

However, for any positive integer n,

$$\sum_{f=1}^{n} \frac{1}{f-1+z} = \sum_{f=1}^{n-1} \frac{1}{f-1+z} + \frac{1}{n-1+z}. \tag{2.59}$$

Consequently, for each n in equation 2.58, where $n = 1 \ldots \max_d N_{k|d}$ in the case of the numerator and $n = 1 \ldots \max_d N_{\cdot|d}$ in the case of the denominator, the (previously-computed) digamma difference for $n-1$ may be used as a starting point, thereby reducing the number of new calculations required for each successive n to one. Pseudocode for the complete fixed-point iteration is given in algorithm 2.2.

Method 2: Approximating Digamma Differences

Instead of decomposing the digamma differences in equation 2.54 using the digamma recurrence relation, it is also possible to approximate them using the following ap-

while not converged {
  $D := 0$
  $S := 0$
  for $n = 1 \ldots \max_d N_{\cdot|d}$ {
    $C_{\cdot}(n) := \sum_{d=1}^{D} \delta\left(N_{\cdot|d} - n\right)$
    $D := D + 1/(n - 1 + \alpha)$
    $S := S + C_{\cdot}(n)\, D$
  }
  for $k = 1 \ldots K$ {
    $D := 0$
    $S_k := 0$
    for $n = 1 \ldots \max_d N_{k|d}$ {
      $C_k(n) := \sum_{d=1}^{D} \delta\left(N_{k|d} - n\right)$
      $D := D + 1/(n - 1 + \alpha m_k)$
      $S_k := S_k + C_k(n)\, D$
    }
    $\alpha m_k := \alpha m_k\, S_k / S$
  }
}

Algorithm 2.2: The first new fixed-point algorithm. This method is based on grouping identical terms and using the digamma recurrence relation.

proximation, described by MacKay and Peto (1995)[1]:

$$\Psi(n + z) - \Psi(z) = \frac{1}{z} + \log \frac{n + z - \frac{1}{2}}{z + \frac{1}{2}}. \tag{2.60}$$

This results in the second of the two new fixed-point iterations:

$$[\alpha m_k]^{\star} = \alpha m_k \frac{\sum_{n=1}^{\max_d N_{k|d}} C_k(n) \left(\frac{1}{\alpha m_k} + \log \frac{n + \alpha m_k - \frac{1}{2}}{\alpha m_k + \frac{1}{2}} \right)}{\sum_{n=1}^{\max_d N_{\cdot|d}} C_{\cdot}(n) \left(\frac{1}{\alpha} + \log \frac{n + \alpha - \frac{1}{2}}{\alpha + \frac{1}{2}} \right)}. \tag{2.61}$$

2.3.6 Efficiently Computing N_{f_k} in MacKay and Peto's Method

The histograms described in the previous section can also be used in MacKay and Peto's fixed-point iteration (section 2.3.4) to efficiently compute each N_{f_k} value (the number of contexts in which outcome k has appeared f or more times) for $f =$

[1] MacKay and Peto originally suggested this approximation for use in a gradient-based algorithm, such as conjugate gradient (Nocedal and Wright, 1999). In practice, even with this approximation, conjugate gradient was found to be much slower than the other methods discussed in this chapter.

$2 \ldots \max_d N_{k|d}$. This can be seen by noting that N_{f_k} may be defined as follows:

$$N_{f_k} = \sum_{d=1}^{D} \delta\left(f \leq N_{k|d}\right) \tag{2.62}$$

$$= \sum_{n=f}^{\max_d N_{k|d}} \sum_{d=1}^{D} \delta\left(N_{k|d} - n\right) \tag{2.63}$$

$$= \sum_{n=f}^{\max_d N_{k|d}} C_k(n) \tag{2.64}$$

$$= N_{(f+1)_k} + C_k(f). \tag{2.65}$$

In other words, the complete set of $\{N_{f_k}\}_{f=2}^{\max_d N_{k|d}}$ values for any k can be computed by starting with $f = \max_d N_{k|d}$ and working down to $f = 2$ using equation 2.65.

2.4 Experiments

The following seven hyperparameter estimation algorithms (all described in the previous section) were compared using synthetic data and natural language data:

- Minka's fixed-point iteration on the log evidence,
- Minka's Newton iteration on the log evidence,
- Minka's fixed-point iteration on the leave-one-out log evidence,
- MacKay and Peto's fixed-point iteration on the log evidence,
- new method based on the digamma recurrence relation,
- new method based on MacKay and Peto's digamma difference approximation,
- MacKay and Peto's method with histogram-based computation of N_{f_k}.

All seven estimation methods were compared in terms of computation time and accuracy. Computation time was measured in milliseconds, while accuracy was computed using two metrics: the Kullback-Leibler divergence between the true base measure and inferred base measure, and the relative error in the concentration parameter estimate. The Kullback-Leibler divergence is a measure (in bits) of the distance between a "true" probability distribution (in this case, the true base measure $\boldsymbol{m}^{\text{true}}$) and some other probability distribution (in this case, the inferred base measure $\boldsymbol{m}^{\star}$):

$$D_{\text{KL}}(\boldsymbol{m}^{\text{true}} \,\|\, \boldsymbol{m}^{\star}) = \sum_{k=1}^{K} m_k^{\text{true}} \log_2 \frac{m_k^{\text{true}}}{m_k^{\star}}. \tag{2.66}$$

The relative error in the concentration parameter estimate $\alpha^\star$ is given by

$$\epsilon = \frac{|\alpha^{\text{true}} - \alpha^\star|}{\alpha^{\text{true}}}. \tag{2.67}$$

The log probability of unseen test data was *not* used as an accuracy metric. A less accurate estimation method may actually assign a higher probability to unseen test data than a more accurate method, by effectively performing smoothing and reducing overfitting.[2] Although this property may superficially seem desirable, using an estimation method to perform smoothing is not a principled way of alleviating overfitting, and there are better ways of addressing this problem (for example, by using an appropriate prior). Consequently, none of the methods discussed in this chapter were evaluated using the log probability of unseen test data. Instead, the estimated hyperparameters were compared to the true hyperparameters, as described above.

2.4.1 Synthetic Data

There are several different quantities that can be varied when generating synthetic data from a Dirichlet-multinomial distribution. These are:

- Dimensionality K,
- the number of instances or contexts D,
- the number of observations per context N,
- the concentration parameter α, and
- the base measure $\boldsymbol{m}$.

To compare the seven hyperparameter estimation methods, 1,296 types of synthetic data were generated, each characterised by a particular set of K, D, N and α values. The values used to generate the data sets are shown in table 2.1. The base measure $\boldsymbol{m}$ was fixed for all data sets of dimensionality K, and was itself drawn from a Dirichlet distribution with a concentration parameter of 1.0 and a uniform base measure. Fifteen data sets of each type (i.e., set of K, D, N and α values) were generated so that results for a given type of data could be averaged over multiple different data sets.

Each method was assumed to have converged when the absolute change between successive iterations in every αm_k value was less than 10^{-6}. The computation times[3] for each method are shown in figure 2.1a. Every point represents the time taken by a

[2] Minka's fixed-point iteration on the leave-one-out log evidence is an example of a less accurate estimation method that often assigns a higher probability to test data than more accurate methods. This is due to its use of the leave-one-out evidence, a function which is based on cross-validation and has previously been used to derive smoothing methods, such as Good-Turing (McAllester and Schapire, 2003).

[3] All code was written in Java™. All experiments were run on a single core of a two processor, dual-core, hyperthreaded Intel® Xeon 3GHz machine with 8GB of RAM, which was otherwise unutilised.

Dimensionality K	# Contexts D	# Observations N	Conc. Parameter α
5	50	50	0.5
10	100	100	1
20	200	200	2
50	500	500	5
100	1,000	1,000	10
200	2,000	2,000	20

Table 2.1: Full set of parameter values used for synthetic data generation.

single method on a particular type of data (characterised by a set of K, D, N and α values), averaged over ten data sets of that type. For each method, the types of data are ordered (left to right) by increasing K, D, N and α, in that order. The fastest estimation method is almost always MacKay and Peto's fixed-point iteration with histogram-based computation of N_{f_k}. The new method based on the digamma recurrence relation and MacKay and Peto's method without histogram-based computation of N_{f_k} are also very fast—often over an order of magnitude faster than any of Minka's methods.

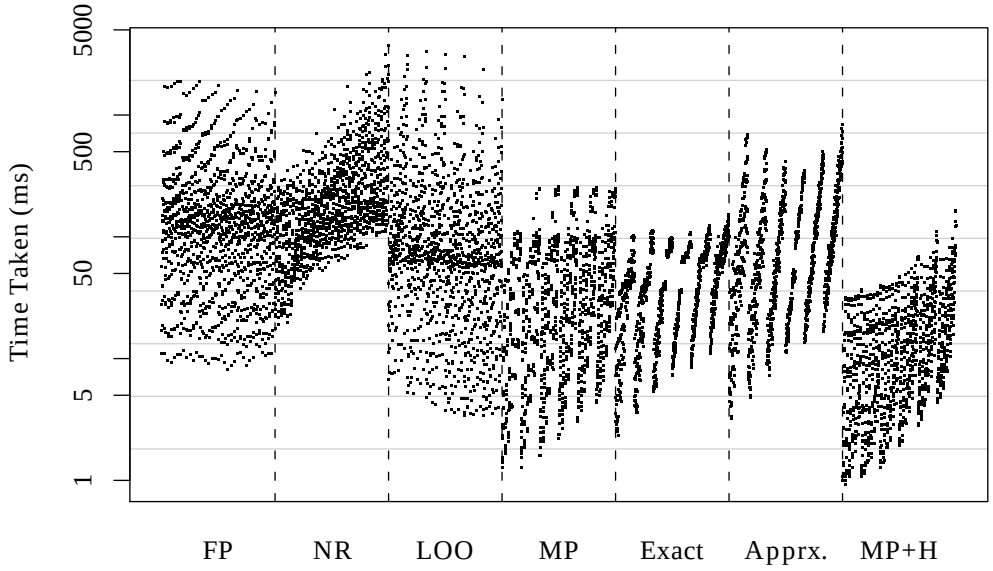
Although figure 2.1a gives an idea of the overall ranking of each method, it is also useful to look at the differences in log time between each method and a single "benchmark" method. This representation eliminates common structure in the results (i.e., estimation always takes longer on some data sets, regardless of the method used) and highlights differences that might otherwise be obscured by the commonalities. For each method and type of data, the difference in log time is computed as follows:

$$\Delta = \log t - \log t_{\text{bench}}, \tag{2.68}$$

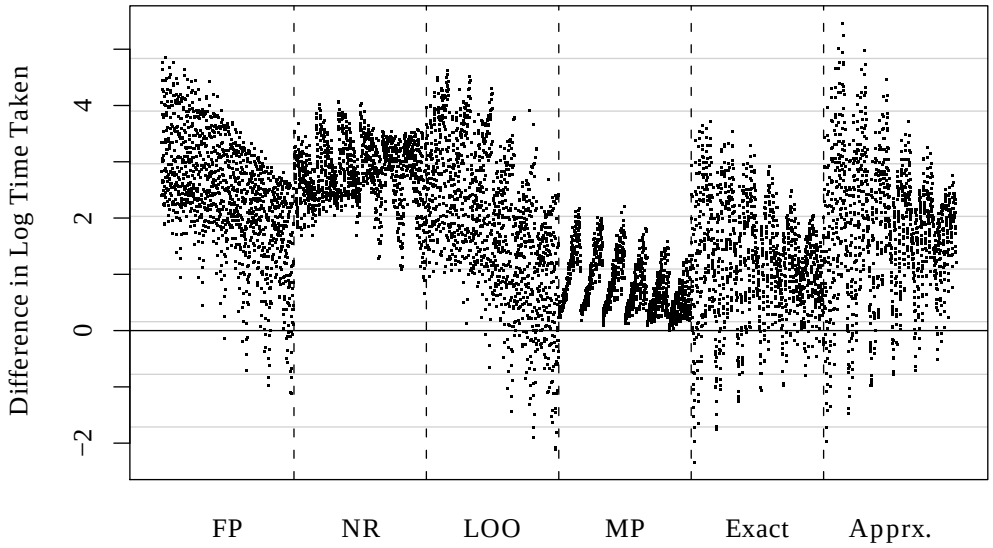
where t is the time taken (measured in milliseconds) by the method in question, and t_{bench} is the time taken by the benchmark method on the same type of data.

The differences in log time are shown in figure 2.1b. MacKay and Peto's method with histogram-based computation of N_{f_k} was chosen as the benchmark method since it appears to be faster than the other methods for almost all types of data. Figure 2.1b confirms that this method is indeed faster than the other methods for almost all types of data. However, Minka's fixed-point iterations on the log evidence and leave-one-out log evidence, as well as the new method based on the digamma recurrence relation and the new method based on MacKay and Peto's digamma difference approximation, are faster than the benchmark method for 30, 95, 134 and 84 (out of 1,296) types of data, respectively. The types of data for which the new methods are faster are characterised by small K and D and large N—exactly the types of data for which the rearrangements that gave rise to the new methods are likely to provide the most benefit.

The Kullback-Leibler divergences between the true base measures and the estimated base measures are shown in figure 2.2a. Again, each point represents the performance

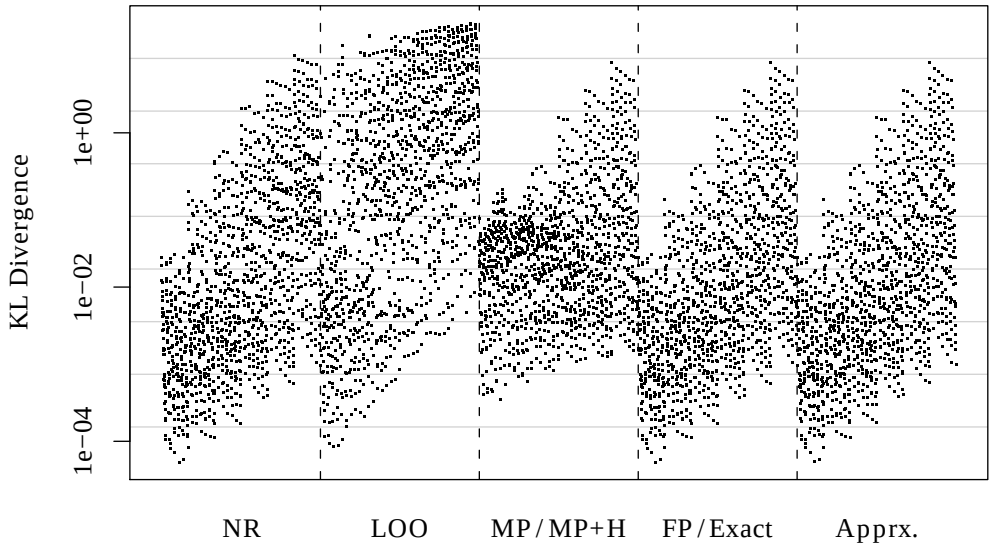


(a) Computation time (ms) for each method.

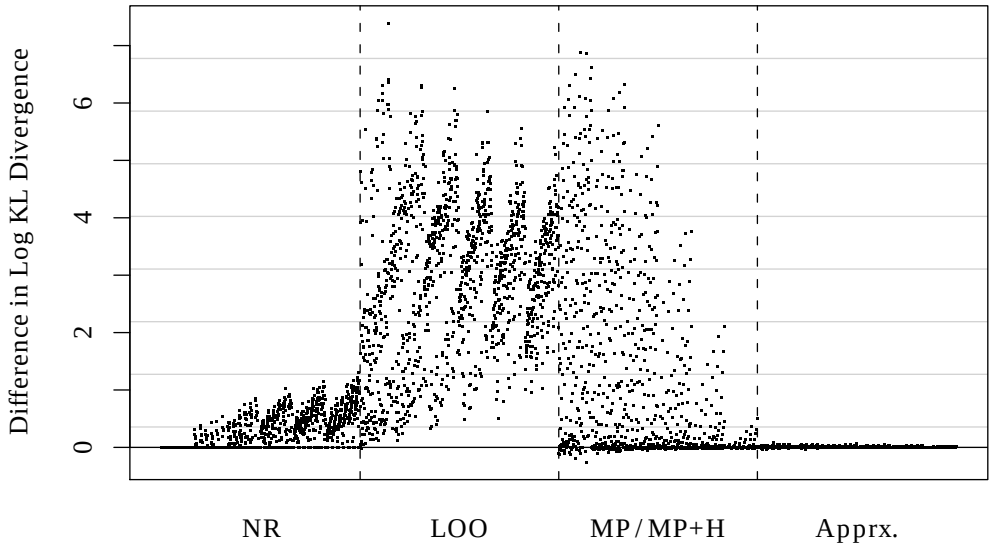


(b) Differences in log computation time (base e) for each method.

Figure 2.1: (a) shows the computation time for each method. (b) shows the differences in log time taken (base e) relative to MacKay and Peto's fixed-point method with histogram-based computation of N_{f_k}. "FP" is Minka's fixed-point iteration on the log evidence, "NR" is Minka's Newton method, "LOO" is Minka's fixed-point iteration on the leave-one-out log evidence, "MP" is MacKay and Peto's method without histogram-based computation of N_{f_k}, "Exact" is the new method based on the digamma recurrence relation, "Apprx." is the new method based on MacKay and Peto's approximation for digamma differences, while "MP+H" is MacKay and Peto's algorithm with histogram-based computation of N_{f_k}.



(a) Kullback-Leibler divergence between the true and inferred base measures.



(b) Differences in log Kullback-Leibler divergence (base e). Higher is worse.

Figure 2.2: (a) shows Kullback-Leibler divergence between the true and inferred base measures for each method. (b) shows the differences in log Kullback-Leibler divergence (base e) relative to the new method based on the digamma recurrence relation. "NR" is Minka's Newton method, "LOO" is Minka's fixed-point iteration on the leave-one-out log evidence, "MP/MP+H" is MacKay and Peto's fixed-point iteration (both with and without histogram-based computation of N_{f_k}, "FP/Exact" is Minka's fixed-point iteration on the log evidence and the new method based on the digamma recurrence relation, and "Apprx." is the new method based on MacKay and Peto's approximation for digamma differences.

of a single method on a particular type of data. For each method, the types of data are ordered (left to right) by increasing K, D, N and α, in that order. Since Minka's fixed-point iteration and the new method based on the digamma difference relation effectively perform the same calculations, albeit using different methods, their Kullback-Leibler divergences are identical. A single set of results is therefore reported for this pair of methods. Similarly, the Kullback-Leibler divergences for MacKay and Peto's fixed-point iterations with and without the histogram-based computation of N_{f_k} are also identical, so a single set of results is reported for this pair of methods too.

The least accurate methods (measured in terms of the Kullback-Leibler divergence between the true and estimated base measures) appear to be Minka's fixed-point iteration on the leave-one-out log evidence and MacKay and Peto's method. Minka's other methods and the two new methods all exhibit relatively similar accuracy.

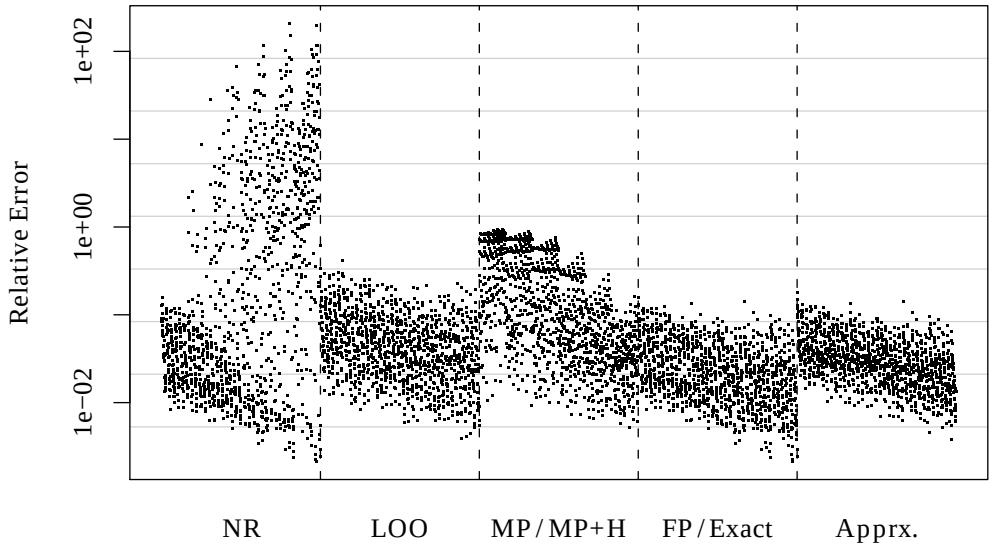
It is also useful to examine the differences in log Kullback-Leibler divergence between each method and a benchmark method. These differences are shown in figure 2.2b, using the new method based on the digamma recurrence relation as the benchmark method. The differences between the benchmark method and new method based on MacKay and Peto's digamma difference approximation are negligible. The other methods—particularly Minka's leave-one-out fixed-point iteration and MacKay and Peto's methods—all achieve worse accuracy than the benchmark method. For MacKay and Peto's methods, these differences are most pronounced for small K.

The relative errors in the estimated concentration parameters are shown in figure 2.3a. The Newton method the worst relative error by far, particularly for data sets with large K—for these data sets the relative error is sufficiently large that the method is rendered effectively useless. MacKay and Peto's fixed point iterations also exhibit a fairly high relative error, particularly for data sets with small K. The new fixed-point iteration based on the digamma recurrence relation exhibits smallest relative error, followed by the new method based on MacKay and Peto's digamma difference approximation, and Minka's fixed-point iteration on the leave-one-out log evidence.

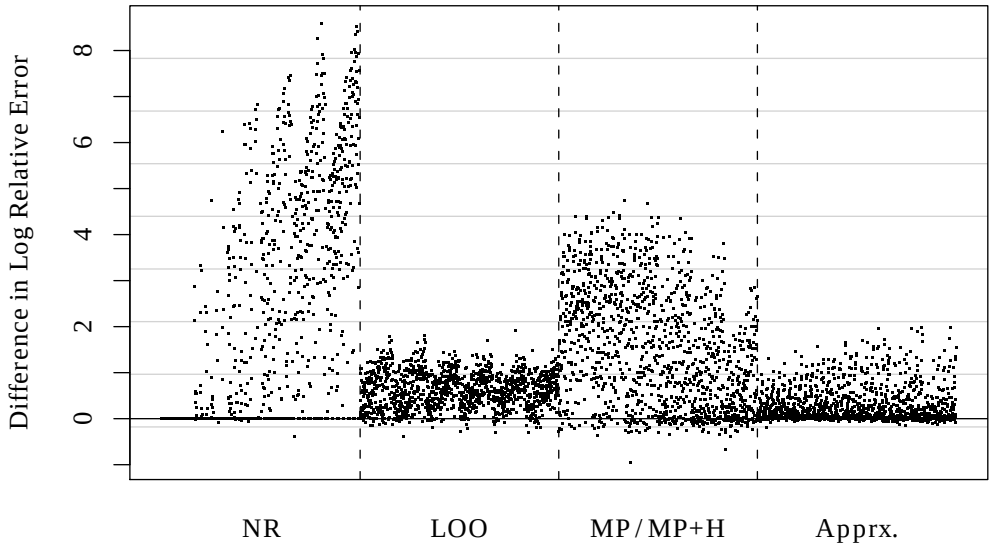
Figure 2.3b shows the differences in log relative error between each estimation method and the new method based on the digamma recurrence relation. It is clear from this figure that the new method is almost always more accurate than the other methods, measured in terms of the relative error in the estimated concentration parameter.

2.4.2 Natural Language Data

While the results in the previous section give a general guide to the relative speed and accuracy of the seven estimation methods, the data used to compare the methods are not especially representative of language data—the focus of this thesis. Each method was therefore also used to estimate the hyperparameters of a Dirichlet-multinomial



(a) Relative error in the estimated concentration parameters.



(b) Differences in log relative error (base e).

Figure 2.3: (a) shows the relative error in the estimated concentration parameters. (b) shows the differences in log relative error (base e), using the the new method based on the digamma recurrence relation as a benchmark. "NR" is Minka's Newton method, "LOO" is Minka's leave-one-out fixed-point iteration, "MP/MP+H" is MacKay and Peto's fixed-point iteration (with and without histogram-based computation of N_{f_k}), "FP/Exact" is Minka's fixed-point iteration and the new method based on the digamma recurrence relation, "Apprx." is the new method using MacKay and Peto's digamma difference approximation.

# Sentences	# Tokens	Vocabulary Size
5,000	119,202	14,048
10,000	238,431	20,269
15,000	357,690	24,828
20,000	476,599	28,562

Table 2.2: Average sizes for the data sets drawn from the Penn Treebank.

bigram language model, using natural language data. In a bigram language model, each word in the vocabulary is treated as a context. The number of observations in each context w is $N_{\cdot|w}$—the number of tokens that immediately follow a token of type w. The data sets used to compare the methods were random subsets of the Penn Treebank (Marcus et al., 1993), with sizes shown in table 2.2. The computation time for each method and data set size was was obtained by averaging over fifteen randomly sampled data sets of that size. Since these data are naturally occurring (the true hyperparameters are unknown), it is not possible to measure the accuracy of each method using the Kullback-Leibler divergence between the true and estimated base measures, or the relative error in the concentration parameter estimate. Although it is possible to compute the probability assigned to unseen test data, this metric is undesirable due to its bias towards less accurate estimation methods (as explained previously).

The computation times for each method are shown in figure 2.4. As data set size increases, so does the time taken. Minka's fixed-point iteration on the log evidence is the slowest method. In contrast, the new fixed-point iteration based on the digamma recurrence relation and MacKay and Peto's method (with and without histogram-based computation of N_{f_k} are over an order of magnitude faster. The fastest method is MacKay and Peto's fixed-point iteration with histogram-based computation of N_{f_k}.

2.5 Incorporating a Gamma Hyperprior

The estimation methods discussed in the previous sections all assumed an improper prior over $\alpha\boldsymbol{m}$. However, if specific properties of the hyperparameters are known, it may be desirable to compute the hyperparameter values that maximise

$$P(\alpha\boldsymbol{m} \mid \mathcal{D}) \propto P(\mathcal{D} \mid \alpha\boldsymbol{m})P(\alpha\boldsymbol{m}), \tag{2.69}$$

where $P(\alpha\boldsymbol{m})$ is a proper "hyperprior" or prior over the hyperparameters.

Minka's fixed-point iteration and the new fixed-point methods presented in section 2.3.5 can all be modified to incorporate a hyperprior. Typically, each hyperparameter αm_k is assumed to have been independently drawn from some univariate prior distribution $P(\alpha m_k)$. The gamma distribution is a common choice for positive

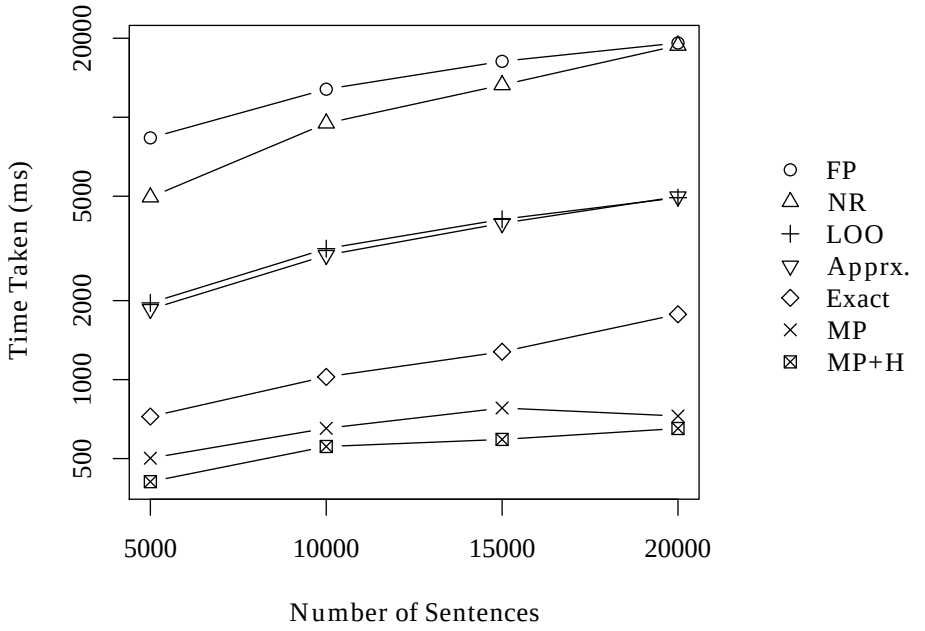


Figure 2.4: Computation time for estimation method on natural language data. "FP" is Minka's fixed-point iteration, "NR" is Minka's Newton method, "LOO" is Minka's leave-one-out fixed-point iteration, "MP" is MacKay and Peto's method (without histogram-based computation of N_{f_k}), "Exact" is the new method based on the digamma recurrence relation, "Apprx." is the new method based on MacKay and Peto's digamma difference approximation, and "MP+H" is MacKay and Peto's fixed-point iteration with histogram-based computation of N_{f_k}.

real-valued variables such as αm_k. It is parameterised by two values s and c:

$$P(\alpha m_k \mid s, c) = \frac{1}{\Gamma(c)s} \left(\frac{\alpha m_k}{s}\right)^{c-1} \exp \frac{-\alpha m_k}{s}. \tag{2.70}$$

In the limit $sc = 1, c \to 0$, this distribution becomes a noninformative improper prior. For certain values of s and c, the gamma distribution exhibits a spike at $\alpha m_k = 0$—an artifact of an inappropriate choice of basis. However, this artifact can be avoid by working in terms of $l_k = \log(\alpha m_k)$. The distribution over l_k is given by

$$P(l_k \mid s, c) = P(\alpha m_k \mid s, c) \left|\frac{\partial \alpha m_k}{\partial l_k}\right| \tag{2.71}$$

$$= \frac{1}{\Gamma(c)} \left(\frac{\alpha m_k}{s}\right)^{c} \exp \frac{-\alpha m_k}{s}. \tag{2.72}$$

Minka's fixed-point iteration can be modified to incorporate a gamma prior over each αm_k by adding the logarithm of equation 2.72 for each k to the lower bound on the log evidence given in equation 2.11. This results in a lower bound on the posterior

distribution over $\alpha\boldsymbol{m}$. Taking the derivative of this bound with respect to αm_k gives

$$\frac{\partial\, B([\alpha\boldsymbol{m}]^\star)}{\partial\, [\alpha m_k]^\star} = \sum_{d=1}^{D}\left[\frac{\alpha m_k\left[\Psi(N_{k|d}+\alpha m_k)-\Psi(\alpha m_k)\right]}{[\alpha m_k]^\star} - \Psi(N_{\cdot|d}+\alpha)+\Psi(\alpha)\right] + \frac{c}{[\alpha m_k]^\star} - \frac{1}{s}. \tag{2.73}$$

Setting this derivative to zero and solving for $[\alpha m_k]^\star$ yields

$$[\alpha m_k]^\star = \alpha m_k \frac{\sum_{d=1}^{D}\left[\Psi(N_{k|d}+\alpha m_k)-\Psi(\alpha m_k)\right]+c}{\sum_{d=1}^{D}\left[\Psi(N_{\cdot|d}+\alpha)-\Psi(\alpha)\right]-\frac{1}{s}}. \tag{2.74}$$

The rearrangements described in section 2.3.5 can be applied to this fixed-point iteration to give variants of the new methods that include a gamma prior over each αm_k.

2.6 Efficiently Computing the Log Evidence

Rearrangements similar to those that gave rise to the new fixed-point methods described in section 2.3.5 can also be used when computing the log evidence.

The log evidence is given by

$$\log P(\mathcal{D}\,|\,\alpha\boldsymbol{m}) = \sum_{d=1}^{D}\left[\log\Gamma(\alpha)-\log\Gamma(N_{\cdot|d}+\alpha)+\sum_{k=1}^{K}\log\Gamma(N_{k|d}+\alpha m_k)-\log\Gamma(\alpha m_k)\right]. \tag{2.75}$$

However, this may be rewritten as

$$\log P(\mathcal{D}\,|\,\alpha\boldsymbol{m}) = \sum_{k=1}^{K}\left[\sum_{n=1}^{\max_d N_{k|d}} C_k(n)\left[\log\Gamma(N_{k|d}+\alpha m_k)-\log\Gamma(\alpha m_k)\right]\right] - \sum_{n=1}^{\max_d N_{\cdot|d}} C_\cdot(n)\left[\log\Gamma(N_{\cdot|d}+\alpha)-\log\Gamma(\alpha)\right], \tag{2.76}$$

where $C_k(n)$ is the number of contexts that contain exactly n observations of value k and $C_\cdot(n)$ is the number of contexts that contain exactly n observations in total. Like

$L := 0$
$S := 0$
for $n = 1 \ldots \max_d N_{\cdot|d}$ {
  $C(n) := \sum_{d=1}^{D} \delta\left(N_{\cdot|d} - n\right)$
  $L := L + \log\left(n - 1 + \alpha\right)$
  $S := S - C_{\cdot}(n)\, L$
}
for $k = 1 \ldots K$ {
  $L := 0$
  for $n = 1 \ldots \max_d N_{k|d}$ {
    $C_k(n) := \sum_{d=1}^{D} \delta\left(N_{k|d} - n\right)$
    $L := L + \log\left(n - 1 + \alpha m_k\right)$
    $S := S + C_k(n)\, L$
  }
}
return S

Algorithm 2.3: Computing the log evidence.

the digamma function, the log gamma function also has a recurrence relation:

$$\log \Gamma(n + z) = \log \Gamma(n - 1 + z) + \log\left(n - 1 + z\right) \tag{2.77}$$

$$= \sum_{f=1}^{n} \log\left(f - 1 + z\right) + \log\left(z\right). \tag{2.78}$$

Rearranging equation 2.78 and substituting it into equation 2.76 gives

$$\log P(\mathcal{D} \,|\, \alpha \boldsymbol{m}) = \sum_{k=1}^{K} \left[\sum_{n=1}^{\max_d N_{k|d}} C_k(n) \sum_{f=1}^{n} \log\left(f - 1 + \alpha m_k\right) \right] - \sum_{n=1}^{\max_d N_{\cdot|d}} C_{\cdot}(n) \sum_{f=1}^{n} \log\left(f - 1 + \alpha\right). \tag{2.79}$$

However, for any positive integer n,

$$\sum_{f=1}^{n} \log\left(f - 1 + z\right) = \sum_{f=1}^{n-1} \log\left(f - 1 + z\right) + \log\left(n - 1 + z\right). \tag{2.80}$$

Consequently, for each n in equation 2.79, the (previously-computed) log gamma difference for $n - 1$ may be used as a starting point when computing the sum over f, thereby reducing the number of new calculations required for each n to one. Pseudocode for computing the log evidence using this method is in algorithm 2.3.

2.7 Conclusions

In this chapter, I introduced two new methods for estimating the hyperparameters of a Dirichlet-multinomial distribution, and compared them with several previously-introduced estimation methods. Using both real and synthetic data, I demonstrated that a method originally introduced by MacKay and Peto (1995), as well as a new method based on the digamma recurrence relation, are faster than standard estimation methods by over an order of magnitude. I also showed that the new method is the most accurate, and can be extended to incorporate a gamma hyperprior. Lastly, I demonstrated that decompositions similar to those used to derive the new estimation methods may be used to derive an algorithm for efficiently computing the log probability of data under a Dirichlet-multinomial model. Due to their speed and accuracy benefits, the estimation method based on the digamma recurrence relation and corresponding log probability algorithm are used throughout subsequent chapters.

Chapter 3

Topic Modelling: Beyond Bag-of-Words

In this chapter, I develop a new hierarchical Bayesian model that incorporates both n-gram statistics and latent topic variables by extending a unigram topic model (Blei et al., 2003) to include properties of a bigram language model (MacKay and Peto, 1995). I compare several variants of this topic-based language model, involving different priors and inference techniques, and introduce a new algorithm for "left-to-right" evaluation of topic models. The new model exhibits better predictive performance than even a trigram language model, and yields topics that are clearly interpretable. Additionally, the model provides insight into modelling choices that prevent latent topics discovered using unigram statistics from being dominated by stop words.

3.1 Introduction

Recently, much attention has been given to generative Bayesian models of textual corpora, designed to reveal inter- or intra-document statistical structure. Such models typically fall into one of two categories—those that generate each word on the basis of some number of preceding words or word classes (MacKay and Peto, 1995; Goldwater et al., 2006; Teh, 2006) and those that generate words based on latent topic variables inferred from word correlations independent of the order in which the words appear (Blei et al., 2003, 2004; Li and McCallum, 2007; Blei and Lafferty, 2007).

Models that make predictions using some number of preceding words are known as n-gram language models. While such models may use conditioning contexts of arbitrary length, this chapter considers only bigram models—i.e., models that generate each word using only the immediately preceding word as available context. To develop a bigram language model, marginal and conditional word counts are determined from a corpus $\boldsymbol{w}$. The marginal count N_w is defined as the number of times that word w

occurs in the corpus, while the conditional count $N_{w|w'}$ is the number of times word w immediately follows word w'. The aim of bigram language modelling is to use these counts to make predictions about the word w_n at position n in any document. In a non-Bayesian setting, this is done by computing estimators of both the marginal probability of word w and the conditional probability of word w following word w', given by $f_w = N_w/N_\cdot$ and $f_{w|w'} = N_{w|w'}/N_{w'}$, where $N_\cdot$ is the number of tokens in the corpus. If there were sufficient data available, the observed conditional frequency $f_{w|w'}$ could be used as an estimator for the predictive probability of w given w'. In practice, the conditional frequency does not provide a good estimate: only a small fraction of possible word pairs will have been observed in the corpus. Consequently, the conditional frequency estimator has too large a variance to be used by itself.

To alleviate this problem, the predictive probability of word w given word w' is obtained by smoothing $f_{w|w'}$ with the marginal frequency estimator f_w:

$$P(w_n = w \mid w_{n-1} = w') = \lambda\, f_w + (1 - \lambda)\, f_{w|w'}. \tag{3.1}$$

The parameter λ may be fixed or determined from data using cross-validation (Jelinek and Mercer, 1980). This procedure works well in practice, despite its ad hoc nature.

The hierarchical Dirichlet language model (MacKay and Peto, 1995) is a bigram model that is entirely driven by principles of Bayesian inference. This model has a similar predictive distribution to models based on equation 3.1, with one key difference: the bigram statistics $f_{w|w'}$ in MacKay and Peto's model are not smoothed with marginal statistics f_w but are smoothed with a quantity related to the number of different contexts in which each word has occurred. Smoothing higher-order counts with lower-order counts that correspond to the number of unique contexts that share some prefix is well-known to yield good predictive performance (Chen and Goodman, 1998).

Latent Dirichlet allocation (Blei et al., 2003) provides an alternative approach to modelling text. Documents are modelled as finite mixtures over an underlying set of latent topics (specialised distributions over words) inferred from correlations between words, independent of word order. The assumption that word order can be ignored—known as the *bag-of-words assumption*—makes sense from a point of view of computational efficiency, but is unrealistic. In many language modelling applications, such as text compression (Bell et al., 1990), speech recognition (Rabiner and Juang, 1993; Jelinek, 1998), and predictive text entry (Ward et al., 2000; Ward, 2001), word order is extremely important. Furthermore, word order can assist in topic inference. The phrases "the department chair couches offers" and "the chair department offers couches" have the same unigram statistics, but are about quite different topics. When inferring which topic generated the word "chair" in the first sentence, knowing that it was immediately preceded by the word "department" makes it more likely to have been generated by a topic that assigns high probability to words about university administration.

Another difference between n-gram language models and topic models is the role of stop words. To ensure that topics inferred using latent Dirichlet allocation do not contain stop words (i.e., non-content words), such as "in", "that", "of" and "for", these words are removed from corpora prior to topic inference. While removing stop words may be appropriate for tasks where word order does not play a significant role—such as information retrieval—this is not appropriate for most language modelling applications, where both stop words and content words must be accurately predicted.

Ideas from n-gram language modelling and Bayesian topic modelling have not previously been combined, yet models of text that capture both word order and topics are clearly appealing. The remainder of this chapter presents a new framework for integrating both approaches in a single Bayesian topic-based language model.

3.2 Hierarchical Dirichlet Language Modelling

As described in the previous section, bigram language models are specified by conditional distributions $P(w_n = w \mid w_{n-1} = w')$, described by $W(W-1)$ free parameters, where W is the size of the vocabulary. These parameters can be denoted by Φ, a matrix whose elements $\phi_{w|w'}$ correspond to $P(w_n = w \mid w_{n-1} = w')$. Φ may be thought of as a transition probability matrix, in which each row $\boldsymbol{\phi}_{w'}$ is the probability vector for transitions from word w'. Given a corpus $\boldsymbol{w}$, the probability of $\boldsymbol{w}$ given Φ is

$$P(\boldsymbol{w} \mid \Phi) = \prod_{w} \prod_{w'} \phi_{w|w'}^{N_{w|w'}}, \tag{3.2}$$

where $N_{w|w'}$ is the number of times that word w' immediately precedes word w. MacKay and Peto (1995) extended this framework with a Dirichlet prior over Φ:

$$P(\Phi \mid \beta \boldsymbol{n}) = \prod_{w'} \operatorname{Dir}(\boldsymbol{\phi}_{w'} \mid \beta \boldsymbol{n}) \tag{3.3}$$

$$= \prod_{w'} \frac{\Gamma(\beta)}{\prod_w \Gamma(\beta n_w)} \prod_{w} \phi_{w|w'}^{\beta n_w - 1} \, \delta\left(\sum_{w} \phi_{w|w'} - 1\right), \tag{3.4}$$

characterised by the hyperparameters β, a nonnegative concentration parameter, and $\boldsymbol{n}$, a base measure whose elements sum to one. Together, equations 3.2 and 3.3 are known as a Dirichlet-multinomial model, as described in the previous chapter.

Combining equations 3.2 and 3.3, and integrating over Φ, yields the probability of the corpus $\boldsymbol{w}$ given the hyperparameters $\beta\boldsymbol{n}$, also known as the "evidence":

$$P(\boldsymbol{w} \mid \beta \boldsymbol{n}) = \prod_{w'} \frac{\prod_w \Gamma(N_{w|w'} + \beta n_w)}{\Gamma(N_{w'} + \beta)} \frac{\Gamma(\beta)}{\prod_w \Gamma(\beta n_w)}. \tag{3.5}$$

As explained in section 2.1, under a Dirichlet-multinomial model, the predictive dis-

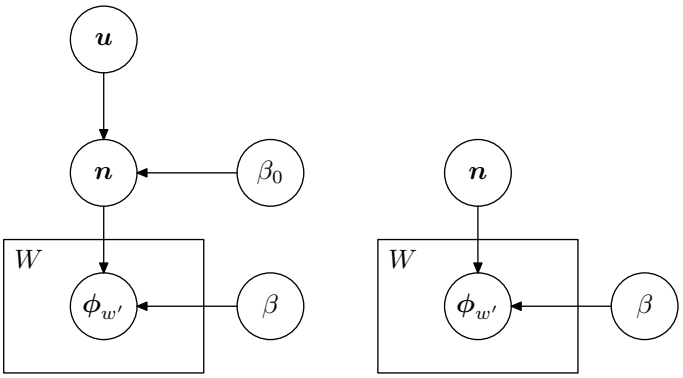


Figure 3.1: (a) depicts a full hierarchical Dirichlet prior over $\boldsymbol{\phi}_{w'}$, while (b) depicts the approximation to this prior used by (MacKay and Peto, 1995).

tribution over words for each context w' given the hyperparameters $\beta\boldsymbol{n}$ is

$$P(w \mid w', \boldsymbol{w}, \beta\boldsymbol{n}) = \int \phi_{w|w'} \operatorname{Dir}\left(\boldsymbol{\phi}_{w'} \mid \{N_{w|w'} + \beta n_w\}_{w=1}^{W}\right) \mathrm{d}^W \boldsymbol{\phi}_{w'} \tag{3.6}$$

$$= \frac{N_{w|w'} + \beta n_w}{N_{w'} + \beta}. \tag{3.7}$$

To explicate the relationship between equation 3.7 and the predictive probability given by a simple non-Bayesian model (equation 3.1), $P(w \mid w', \boldsymbol{w}, \beta\boldsymbol{n})$ may be rewritten as

$$P(w \mid w', \boldsymbol{w}, \beta\boldsymbol{n}) = \lambda_{w'}\, n_w + (1 - \lambda_{w'})\, f_{w|w'}, \tag{3.8}$$

where $f_{w|w'} = N_{w|w'}/N_{w'}$ and

$$\lambda_{w'} = \frac{\beta}{N_{w'} + \beta}. \tag{3.9}$$

Each hyperparameter n_w takes the role of the marginal statistic f_w in equation 3.1, while the concentration parameter β determines the extent of the smoothing.

In an ideal Bayesian setting, $\beta\boldsymbol{n}$ should be given a proper prior, such as a symmetric Dirichlet distribution with uniform base measure $\boldsymbol{u}$ and concentration parameter β_0, as shown in figure 3.1a. The resultant prior induced over Φ is known as a *hierarchical* Dirichlet. Having given $\beta\boldsymbol{n}$ a proper prior, the true predictive distribution can be obtained by computing the expectation of $P(w \mid w', \boldsymbol{w}, \beta\boldsymbol{n})$ under the posterior distribution over $\beta\boldsymbol{n}$. However, as described in the previous chapter, it is often the case that the posterior, $P(\beta\boldsymbol{n} \mid \boldsymbol{w})$, is sufficiently sharply peaked in $\beta\boldsymbol{n}$ that the true predictive distribution may be approximated by $P(w \mid w', \boldsymbol{w}, [\beta\boldsymbol{n}]^\star)$, where $[\beta\boldsymbol{n}]^\star$ is the maximum of $P(\beta\boldsymbol{n} \mid \boldsymbol{w})$. This approximation is shown in figure 3.1b and is exactly the approximation used by MacKay and Peto. Furthermore, MacKay and Peto show that each element of the optimal $\boldsymbol{n}$, when estimated using this "empirical Bayes" procedure, is related to the number of contexts in which the corresponding word has appeared.

3.3 Latent Dirichlet Allocation

Latent Dirichlet allocation, originally introduced by Blei et al. (2003), represents documents as random mixtures over latent topics, where each topic is a specialised distribution over words. Word generation is defined by the conditional distributions $P(w_n = w \mid z_n = t)$, described by $T(W-1)$ free parameters, where T is the number of latent topics and W is the size of the vocabulary. These parameters are denoted by the matrix Φ, with elements $\phi_{w|t} = P(w_n = w \mid z_n = t)$. Φ may be thought of as an emission probability matrix, in which the t^{th} row is the distribution over words for topic t—the probability vector $\boldsymbol{\phi}_t$. Similarly, topic generation is characterised by the conditional distribution $P(z_n = t \mid d_n = d)$, described by $D(T-1)$ free parameters, where D is the number of documents in the corpus and T is the number of latent topics. These parameters form a matrix Θ with elements $\theta_{t|d} = P(z_n = t \mid d_n = d)$. The d^{th} row of this matrix is the distribution over topics for document d—the probability vector $\boldsymbol{\theta}_d$.

The joint probability of a corpus $\boldsymbol{w}$ and corresponding topic assignments $\boldsymbol{z}$ is

$$P(\boldsymbol{w}, \boldsymbol{z} \mid \Phi, \Theta) = \prod_w \prod_t \prod_d \phi_{w|t}^{N_{w|t}} \, \theta_{t|d}^{N_{t|d}}, \tag{3.10}$$

where $N_{t|d}$ is the number of times that topic t has been used in document d and $N_{w|t}$ is the number of times that word w has been generated by topic t. To complete the model, Blei et al. place a nonhierarchical Dirichlet prior over Φ,

$$P(\Phi \mid \beta \boldsymbol{n}) = \prod_t \text{Dir}\,(\boldsymbol{\phi}_t \mid \beta \boldsymbol{n}), \tag{3.11}$$

and another over Θ,

$$P(\Theta \mid \alpha \boldsymbol{m}) = \prod_d \text{Dir}\,(\boldsymbol{\theta}_d \mid \alpha \boldsymbol{m}). \tag{3.12}$$

Combining equations 3.11 and 3.12 with equation 3.10 and marginalising out Φ and Θ and latent variables $\boldsymbol{z}$ gives the evidence for the hyperparameters:

$$\begin{aligned} P(\boldsymbol{w} \mid \alpha \boldsymbol{m}, \beta \boldsymbol{n}) = \\ \sum_{\boldsymbol{z}} \prod_t \frac{\prod_w \Gamma(N_{w|t} + \beta n_w)}{\Gamma(N_{\cdot|t} + \beta)} \frac{\Gamma(\beta)}{\prod_w \Gamma(\beta n_w)} \\ \prod_d \frac{\prod_t \Gamma(N_{t|d} + \alpha m_t)}{\Gamma(N_{\cdot|d} + \alpha)} \frac{\Gamma(\alpha)}{\prod_t \Gamma(\alpha m_t)}. \end{aligned} \tag{3.13}$$

The quantity $N_{\cdot|t} = \sum_w N_{w|t}$ is the total number of times any word has been generated by topic t, while $N_{\cdot|d} = \sum_t N_{t|d}$ is the total number of tokens in document d.

Given a corpus $\boldsymbol{w}$ with corresponding topic assignments $\boldsymbol{z}$, and hyperparameters $\alpha \boldsymbol{m}$

and $\beta\boldsymbol{n}$, the predictive probability of word w being generated by topic t is

$$P(w \mid t, \boldsymbol{w}, \boldsymbol{z}, \beta\boldsymbol{n}) = \frac{N_{w|t} + \beta n_w}{N_{\cdot|t} + \beta}. \tag{3.14}$$

Similarly, the predictive probability of topic t in document d is given by

$$P(t \mid d, \boldsymbol{w}, \boldsymbol{z}, \alpha\boldsymbol{m}) = \frac{N_{t|d} + \alpha m_t}{N_{\cdot|d} + \alpha}. \tag{3.15}$$

These equations may be rewritten as follows:

$$P(w \mid t, \boldsymbol{w}, \boldsymbol{z}, \beta\boldsymbol{n}) = (1 - \lambda_t)\, f_{w|t} + \lambda_t\, n_w, \tag{3.16}$$

$$P(t \mid d, \boldsymbol{w}, \boldsymbol{z}, \alpha\boldsymbol{m}) = (1 - \lambda_d)\, f_{t|d} + \lambda_d\, m_t, \tag{3.17}$$

where $f_{w|t} = N_{w|t}/N_t$, $f_{t|d} = N_{t|d}/N_d$, and

$$\lambda_t = \frac{\beta}{N_{\cdot|t} + \beta}, \tag{3.18}$$

$$\lambda_d = \frac{\alpha}{N_{\cdot|d} + \alpha}. \tag{3.19}$$

The quantity $f_{w|t}$ is effectively smoothed by the hyperparameter n_w while $f_{t|d}$ is smoothed by m_t. Equations 3.16 and 3.17 have the same form as equation 3.8.

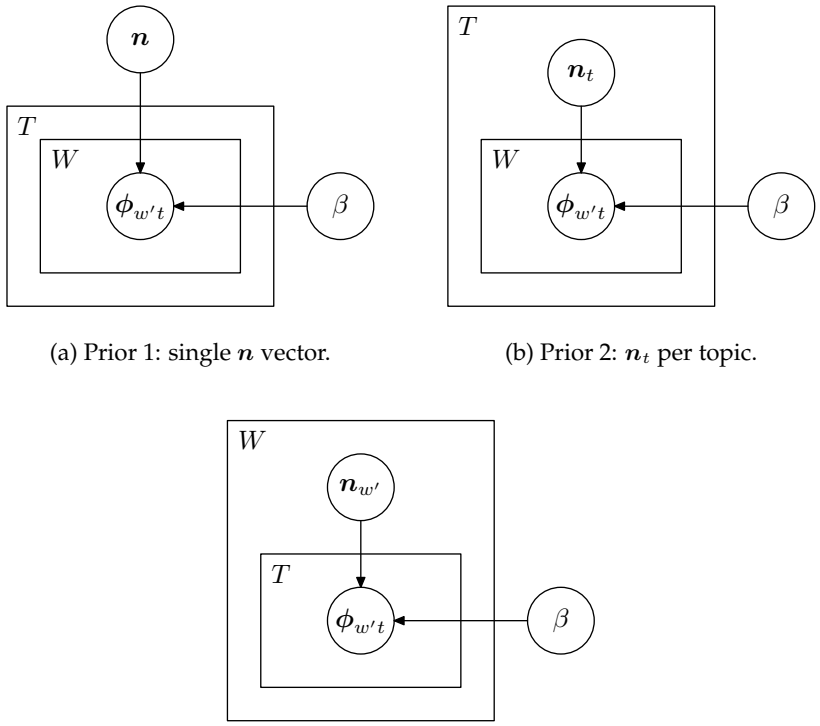
3.4 A Topic-Based Language Model

In this section, I introduce a new model that extends latent Dirichlet allocation by incorporating a notion of word order similar to that employed by MacKay and Peto's hierarchical Dirichlet language model. For simplicity, discussion is restricted to bigrams, however the underlying ideas also are applicable to models of higher order.

The new model characterises each topic by a set of W distributions over words—one for each possible previous word context. Consequently, word generation is defined by conditional distributions $P(w_n = w \mid w_{n-1} = w', z_n = t)$, described by $WT(W-1)$ free parameters. As with latent Dirichlet allocation, these parameters form a matrix Φ—this time with WT rows. Each row $\boldsymbol{\phi}_{w't}$ is the distribution over words for a the context consisting of previous word w' and topic t. Topic generation is identical to latent Dirichlet allocation: Topics are drawn using the conditional probabilities $P(z_n = t \mid d_n = d)$, described by $D(T-1)$ free parameters, which form a matrix Θ.

The joint probability of a corpus $\boldsymbol{w}$ and corresponding set of topic assignments $\boldsymbol{z}$ is

$$P(\boldsymbol{w}, \boldsymbol{z} \mid \Phi, \Theta) = \prod_w \prod_{w'} \prod_t \prod_d \phi_{w|w't}^{N_{w|w't}}\, \theta_{t|d}^{N_{t|d}}, \tag{3.20}$$



(a) Prior 1: single $\boldsymbol{n}$ vector.

(b) Prior 2: $\boldsymbol{n}_t$ per topic.

(c) Prior 3: $\boldsymbol{n}_{w'}$ per previous word.

Figure 3.2: Three nonhierarchical Dirichlet priors over $\boldsymbol{\phi}_{w't}$.

where $N_{w|w't}$ is the number of times word w has occurred in the context of preceding word w' and topic t. $N_{t|d}$ is the number of times topic t has been used in document d.

The prior over Θ is the same as that used in latent Dirichlet allocation:

$$P(\Theta \mid \alpha\boldsymbol{m}) = \prod_d \mathrm{Dir}\,(\boldsymbol{\theta}_d \mid \alpha\boldsymbol{m}). \tag{3.21}$$

However, the additional conditioning context w' in the distributions that define word generation in the new model affords greater flexibility in choosing a prior for Φ than in either latent Dirichlet allocation or the hierarchical Dirichlet language model. The priors over Φ used in both MacKay and Peto's language model and Blei et al.'s latent Dirichlet allocation are "coupled" priors: learning the probability vector for a single context—$\boldsymbol{\phi}_{w'}$ the case of MacKay and Peto's model and $\boldsymbol{\phi}_t$ in Blei et al.'s—gives information about the probability vectors for other contexts w'' and t', respectively. This dependence comes from the hyperparameters $\beta\boldsymbol{n}$, which are shared, in the case of the hierarchical Dirichlet language model, between all possible previous word contexts w' and, in the case of latent Dirichlet allocation, between all possible topics t. In the new model, word generation is conditioned upon both w' and t. Consequently, there is more than one way in which hyperparameters for the prior over Φ might be shared.

Prior 1: A single hyperparameter vector $\beta\boldsymbol{n}$ can be shared between all $w't$ contexts:

$$P(\Phi \mid \beta\boldsymbol{n}) = \prod_{w'} \prod_{t} \text{Dir}\,(\boldsymbol{\phi}_{w't} \mid \beta\boldsymbol{n}). \tag{3.22}$$

Here, learning about one probability vector $\boldsymbol{\phi}_{w't}$ will reveal information about the probability vectors for all other $w''t'$ contexts. This prior is shown in figure 3.2a.

Prior 2: Alternatively, there can be T hyperparameter vectors—one for each topic t:

$$P(\Phi \mid \{\beta\boldsymbol{n}_t\}_{t=1}^{T}) = \prod_{w'} \prod_{t} \text{Dir}\,(\boldsymbol{\phi}_{w't} \mid \beta\boldsymbol{n}_t). \tag{3.23}$$

Information is now shared only between probability vectors that have same topic context as each other: Learning about the distribution over words for context $w't$ yields information about the distributions over words for other contexts $w''t$ that also involve topic t, but not about distributions for other topic contexts t'. This prior (shown in figure 3.2b) captures topic-specific similarities between the distributions over words.

Prior 3: Finally, there can be W sets of hyperparameters—one for each w':

$$P(\Phi \mid \{\beta\boldsymbol{n}_{w'}\}_{w'=1}^{W}) = \prod_{w'} \prod_{t} \text{Dir}\,(\boldsymbol{\phi}_{w't} \mid \beta\boldsymbol{n}_{w'}). \tag{3.24}$$

Here, information is shared between all distributions that share the same previous word context w': Learning about the distribution over words for context $w't$ yields information about only those distributions for other contexts $w't'$ that also correspond to previous word context w'. This prior (shown in figure 3.2c) captures the notion of common bigrams—word pairs that always occur together, regardless of topic.

For each of the three priors described above, it is possible to either (a) integrate out the base measures $\boldsymbol{n}$, $\{\boldsymbol{n}_t\}_{t=1}^{T}$ and $\{\boldsymbol{n}_{w'}\}_{w'=1}^{W}$ after giving them proper priors (known as hyperpriors) or (b) assume noninformative priors over all hyperparameters and estimate $\beta\boldsymbol{n}$, $\{\beta\boldsymbol{n}_t\}_{t=1}^{T}$ and $\{\beta\boldsymbol{n}_{w'}\}_{w'=1}^{W}$ from data, using a similar approach to that described in section 3.2. For completeness, both approaches are described.

3.4.1 Estimating Hyperparameters from Data

Given a corpus $\boldsymbol{w}$ and noninformative hyperpriors, the optimal hyperparameter values may be found by maximising the evidence or probability of $\boldsymbol{w}$ given the hyperparameters. For the hierarchical Dirichlet language model, this procedure is equivalent to estimating the hyperparameters of a Dirichlet-multinomial distribution, as described in chapter 2. For models with topics—either latent Dirichlet allocation or the new topic-based language model—the situation is more complex because the evidence contains latent variables $\boldsymbol{z}$ that must be marginalised out. Previous sampling-based treatments of latent Dirichlet allocation (Griffiths and Steyvers, 2004) have therefore

1: **initialise** $\boldsymbol{z}$ and U
2: **while** not converged {
3: E-step: draw $\{\boldsymbol{z}^{(s)}\}_{s=1}^{S} \sim P(\boldsymbol{z} \,|\, \boldsymbol{w}, U)$
4: M-step: $U := \arg\max_{U} \frac{1}{S} \sum_{s=1}^{S} \log P(\boldsymbol{w}, \boldsymbol{z}^{(s)} \,|\, U)$
5: }

Algorithm 3.1: Gibbs EM for topic models.

not included any form of hyperparameter optimisation—i.e., all base measures are set to the uniform distribution. However, the approach described in this section may be applied to latent Dirichlet allocation as well as the new model.

For the new model, the evidence is given by

$$P(\boldsymbol{w} \,|\, U) = \sum_{\boldsymbol{z}} P(\boldsymbol{w} \,|\, \boldsymbol{z}, U)\, P(\boldsymbol{z} \,|\, U), \tag{3.25}$$

where U denotes the full set of model hyperparameters,

$$P(\boldsymbol{w} \,|\, \boldsymbol{z}, U) = \prod_{w'} \prod_{t} \begin{cases} \dfrac{\prod_w \Gamma(N_{w|w't} + \beta n_w)}{\Gamma(N_{\cdot|w't} + \beta)} \dfrac{\Gamma(\beta)}{\prod_w \Gamma(\beta n_w)} & \text{prior 1} \\ \dfrac{\prod_w \Gamma(N_{w|w't} + \beta n_{w|t})}{\Gamma(N_{\cdot|w't} + \beta)} \dfrac{\Gamma(\beta)}{\prod_w \Gamma(\beta n_{w|t})} & \text{prior 2} \\ \dfrac{\prod_w \Gamma(N_{w|w't} + \beta n_{w|w'})}{\Gamma(N_{\cdot|w't} + \beta)} \dfrac{\Gamma(\beta)}{\prod_w \Gamma(\beta n_{w|w'})} & \text{prior 3} \end{cases} \tag{3.26}$$

and

$$P(\boldsymbol{z} \,|\, U) = \frac{\prod_t \Gamma(N_{t|d} + \alpha m_t)}{\Gamma(N_{\cdot|d} + \alpha)} \frac{\Gamma(\alpha)}{\prod_t \Gamma(\alpha m_t)}. \tag{3.27}$$

The presence of latent variables $\boldsymbol{z}$ means that the evidence may be maximised with respect to the hyperparameters using an expectation-maximisation (EM) algorithm (Dempster et al., 1977). Since each topic assignment z_n can take on one of T values, the expectation step involves a sum over $T^{N_\cdot}$ terms, where $N_\cdot$ is the total number of tokens in the corpus. This sum is intractable. However, it can be approximated using Gibbs sampling (Griffiths and Steyvers, 2004), resulting in a Gibbs EM algorithm (Andrieu et al., 2003), shown in algorithm 3.1. This algorithm can be used to find the hyperparameters that maximise the evidence: $U^\star = \{[\alpha \boldsymbol{m}]^\star, [\beta \boldsymbol{n}]^\star\}$ (prior 1), $U^\star = \{[\alpha \boldsymbol{m}]^\star, \{[\beta \boldsymbol{n}_t]^\star\}_{t=1}^{T}\}$ (prior 2) or $U^\star = \{[\alpha \boldsymbol{m}]^\star, \{[\beta \boldsymbol{n}_{w'}]^\star\}_{w'=1}^{W}\}$ (prior 3).

E-Step

Gibbs sampling involves sequentially resampling each variable of interest, z_n in this case, from its conditional posterior, given the data and current values of all other variables. Letting the subscript "$\backslash n$" denote a quantity that excludes data from the n^{th}

position in the corpus, the conditional posterior for z_n is given by

$$P(z_n{=}t \mid \boldsymbol{z}_{\setminus n}, \boldsymbol{w}, U) \propto P(w_n \mid z_n{=}t, \boldsymbol{z}_{\setminus n}, \boldsymbol{w}, U)\, \frac{(N_{t|d_n})_{\setminus n} + \alpha m_t}{(N_{\cdot|d_n})_{\setminus n} + \alpha} \tag{3.28}$$

where

$$P(w_n \mid z_n{=}t, \boldsymbol{z}_{\setminus n}, \boldsymbol{w}, U) = \begin{cases} \dfrac{(N_{w_n|w_{n-1}t})_{\setminus n} + \beta n_{w_n}}{(N_{\cdot|w_{n-1}t})_{\setminus n} + \beta} & \text{prior 1} \\ \dfrac{(N_{w_n|w_{n-1}t})_{\setminus n} + \beta n_{w_n|t}}{(N_{\cdot|w_{n-1}t})_{\setminus n} + \beta} & \text{prior 2} \\ \dfrac{(N_{w_n|w_{n-1}t})_{\setminus n} + \beta n_{w_n|w_{n-1}}}{(N_{\cdot|w_{n-1}t})_{\setminus n} + \beta} & \text{prior 3} \end{cases} \tag{3.29}$$

Drawing a single set of topic assignments $\boldsymbol{z}^{(s)}$ takes time proportional to the size of the corpus $N_{\cdot}$ and the number of topics T. The E-step therefore takes time proportional to $N_{\cdot}$, T and the number of Gibbs sampling iterations used to obtain the S samples.

M-Step

Given a set of samples $\{\boldsymbol{z}^{(s)}\}_{s=1}^{S}$, $[\alpha\boldsymbol{m}]^{\star}$ can be computed using a variant of the fixed-point iteration described in section 2.3.5, modified to use all S samples:

$$[\alpha m_t]^{\star} = \alpha m_t \frac{\sum_s \sum_d \sum_{n=1}^{\max_d N_{t|d}^{(s)}} C_t(n) \sum_{f=1}^{n} \frac{1}{f-1+\alpha m_t}}{\sum_s \sum_d \sum_{n=1}^{\max_d N_{\cdot|d}^{(s)}} C_{\cdot}(n) \sum_{f=1}^{n} \frac{1}{f-1+\alpha}} \tag{3.30}$$

where $N_{t|d}^{(s)}$ is the number of times topic t has been used in document d in the s^{th} sample. A similar method can be used to optimise the other hyperparameters.

Note that the samples used in this step must come from a single Markov chain. The model is unaffected by permutations of topic indices. Consequently, there is no correspondence between topic indices across samples from different Markov chains: topics with index t in two different chains need not have similar distributions over words.

Predictive Distributions

Given a corpus $\boldsymbol{w}$, corresponding topic assignments $\boldsymbol{z}$, and hyperparameters $U = \{\alpha\boldsymbol{m}, \beta\boldsymbol{n}\}$ (prior 1), $U = \{\alpha\boldsymbol{m}, \{\beta\boldsymbol{n}_t\}_{t=1}^{T}\}$ (prior 2) or $U = \{\alpha\boldsymbol{m}, \{\beta\boldsymbol{n}_{w'}\}_{w'=1}^{W}\}$ (prior 3), the predictive probability of topic t occurring in document d is

$$P(t \mid d, \boldsymbol{w}, \boldsymbol{z}, U) = \frac{N_{t|d} + \alpha m_t}{N_{\cdot|d} + \alpha}. \tag{3.31}$$

Similarly, the predictive probability of word w occurring in context $w't$ is

$$P(w \mid w', t, \boldsymbol{w}, \boldsymbol{z}, U) = \begin{cases} \dfrac{N_{w|w't} + \beta n_w}{N_{\cdot|w't} + \beta} & \text{prior 1} \\ \dfrac{N_{w|w't} + \beta n_{w|t}}{N_{\cdot|w't} + \beta} & \text{prior 2} \\ \dfrac{N_{w|w't} + \beta n_{w|w'}}{N_{\cdot|w't} + \beta} & \text{prior 3} \end{cases} \tag{3.32}$$

In the predictive probability for prior 1 (single $\boldsymbol{n}$ vector), the quantity $N_{w|w't}/N_{\cdot|w't}$ is always smoothed by hyperparameter n_w regardless of the conditioning context $w't$. In contrast, in the predictive probability for prior 2 ($\boldsymbol{n}_t$ for each topic t), $N_{w|w't}/N_{\cdot|w't}$ is smoothed by $n_{w|t}$, which varies depending on the topic t. Finally, in the predictive probability for prior 3 ($\boldsymbol{n}_{w'}$ for each possible previous word context w'), $N_{w|w't}/N_{\cdot|w't}$ is smoothed by $n_{w|w'}$, which varies depending on the previous word w'. These predictive probabilities are very similar to those used in non-Bayesian interpolated language models (equation 3.1). If t were the word two positions before the word being predicted, instead of a topic, the predictive probability for prior 1 would correspond to smoothing trigram counts with some unigram function n_w specific to the current word w. Similarly, the predictive probability for prior 3 would correspond to smoothing trigram counts with some bigram function $n_{w|w'}$ specific to the current word and the word immediately preceding it. Finally, the predictive probability for prior 2 would correspond to smoothing the trigram counts with some function of the *skip-1 bigram* consisting of the current word and the word two positions back. In other words, the three priors can be thought of as different interpolation schemes. Returning to the scenario of interest, where t is a topic, the first prior treats the identity of the word as the most important piece of information after the identity of the word and its entire context $w't$. Meanwhile, the second prior uses the identity of the current word and topic. Finally, the third prior uses the identity of the current and previous words.

3.4.2 Using Hierarchical Priors

Instead of estimating the hyperparameters from data, as described in the previous section, the base measures $\boldsymbol{n}$ (prior 1), $\{\boldsymbol{n}_t\}_{t=1}^T$ (prior 2) or $\{\boldsymbol{n}_{w'}\}_{w'=1}^W$ (prior 3) can themselves be given proper priors—known as hyperpriors—and integrated out.

Prior 1: For prior 1 (single $\boldsymbol{n}$), an appropriate choice of hyperprior is

$$P(\boldsymbol{n} \mid \beta_0 \boldsymbol{u}) = \text{Dir}\,(\boldsymbol{n} \mid \beta_0 \boldsymbol{u}). \tag{3.33}$$

In other words, base measure $\boldsymbol{n}$ is given a Dirichlet hyperprior with uniform base measure $\boldsymbol{u}$ and concentration parameter β_0. (Concentration parameters β and β_0 are both given noninformative hyperpriors.) This hyperprior induces a hierarchical Dirichlet prior over $\boldsymbol{\phi}_{w't}$, henceforth referred to as "hierarchical prior 1" and shown in figure 3.3.

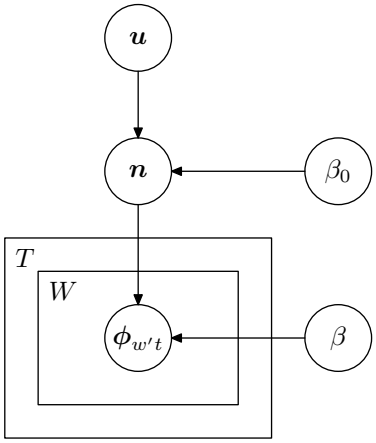


Figure 3.3: Hierarchical version of prior 1 (single $\boldsymbol{n}$ vector).

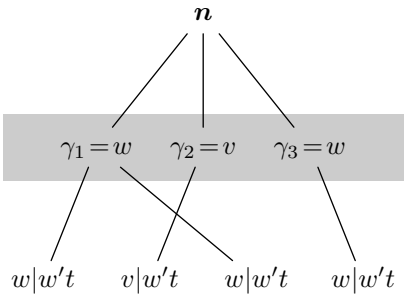


Figure 3.4: Generating four observations (w, v, w, w) from a nonhierarchical Dirichlet-multinomial distribution for context $w't$ with base measure $\boldsymbol{n}$.

The consequences of placing a hierarchical Dirichlet prior over $\phi_{w't}$ are best explained in terms of the effects on the generative process and predictive probabilities, starting without any hyperpriors. For nonhierarchical prior 1 (single $\boldsymbol{n}$), the predictive probability of generating a new observation with value w in context $w't$ is given by

$$P(w \mid w', t, \boldsymbol{w}, \boldsymbol{z}, \beta\boldsymbol{n}) = \frac{N_{w|w't} + \beta n_w}{N_{\cdot|w't} + \beta}. \tag{3.34}$$

If value w has not previously been seen in the context of $w't$, the counts $N_{w|w't}$ and $N_{\cdot|w't}$ will be zero, and the probability of generating w is just n_w. One way of describing the generative process[1], that will become more useful as hyperpriors are added, is to say that generating an observation means instantiating the observation with the value of some context-specific draw from the base measure $\boldsymbol{n}$. Figure 3.4 depicts the process of drawing four observations from the Dirichlet-multinomial for conditioning context $w't$. When drawing the first observation, there are no existing draws from the base measure, so a new one γ_1 must be generated. The first observation is then instantiated with the value of this draw, w in the case of figure 3.4. The second observation is drawn by either selecting γ_1, with probability proportional to the number

[1]The generative process can also be described using the Chinese restaurant process metaphor (Aldous, 1985) and its hierarchical extension, the Chinese restaurant franchise (Teh et al., 2006).

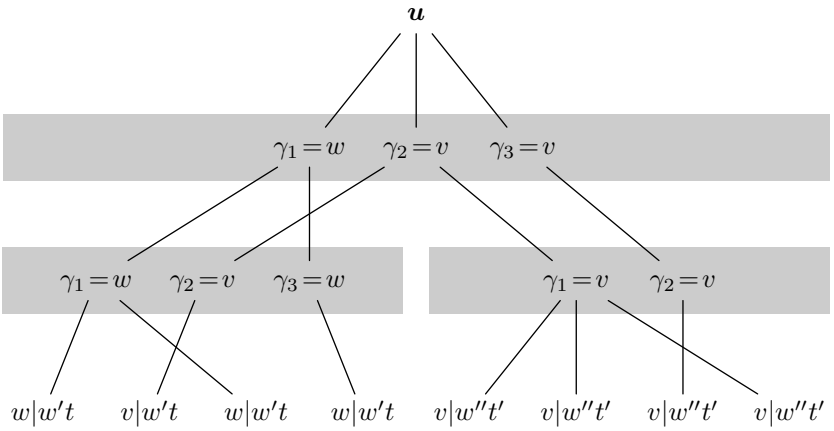


Figure 3.5: Generating observations (w, v, w, w and v, v, v, v) from the hierarchical Dirichlet-multinomial distributions for conditioning contexts $w't$ and $w''t'$.

of observations that have been previously "matched" to γ_1, or a new draw from the base measure, with probability proportional to β. In figure 3.4, a new draw is selected, so γ_2 is drawn from $\boldsymbol{n}$ and the second observation is instantiated with its value, in this case v. The next observation is drawn using the same procedure: existing draws γ_1 and γ_2 are selected with probabilities proportional to the numbers of observations with which they have previously been matched. With probability proportional to β, the observation is matched to a new draw from the base measure. In figure 3.4, γ_1 is selected, meaning there are now two observations matched to γ_1. The third observation is instantiated with the value of γ_1. In general, the probability of a new observation being instantiated with the value of an existing draw from the base measure γ_i is proportional to $N^{(i)}_{\cdot|w't}$—the number of observations previously matched to that draw. Consequently, the probability of generating value w in context $w't$ is given by

$$P(w \mid w', t, \boldsymbol{z}, \boldsymbol{w}, \beta\boldsymbol{n}) = \frac{\hat{N}_{w|w't} + \beta n_w}{\hat{N}_{\cdot|w't} + \beta} \tag{3.35}$$

where $\hat{N}_{\cdot|w't} = \sum_w \hat{N}_{w|w't}$. The quantity $\hat{N}_{w|w't}$ is given by

$$\hat{N}_{w|w't} = \sum_{i=1}^{I} N^{(i)}_{\cdot|w't}\, \delta\left(\gamma_i - w\right), \tag{3.36}$$

where I is the current number of draws from the base measure for context $w't$ and $N^{(i)}_{\cdot|w't}$ is the number of observations matched to γ_i. Since every observation is matched to a draw from the base measure, $\hat{N}_{w|w't}$ is equal to $N_{w|w't}$—the number of times w has been seen in context $w't$. Equations 3.35 and 3.34 are therefore equivalent.

Giving $\boldsymbol{n}$ a Dirichlet prior with parameters $\boldsymbol{u}$ and β_0 (as shown in figure 3.3) and integrating over $\boldsymbol{n}$ has the effect of replacing $\boldsymbol{n}$ in equation 3.35 with a "parent" Dirichlet-multinomial, shared with the Dirichlet-multinomials for all other conditioning contexts $w''t'$. Figure 3.5 depicts the process of drawing eight observations—four in con-

text $w't$ and four in context $w''t'$. When an observation is drawn from the Dirichlet-multinomial for context $w't$, it is (as before) instantiated with the value of an existing "internal draw" γ_i with probability proportional to the number of observations previously matched to that draw. With probability proportional to β, it is instantiated with the value of a new internal draw. However, since the base measure has been integrated out, the new internal draw must be obtained from the parent Dirichlet-multinomial distribution. At the parent level, the new internal draw is treated as if it were an observation, and instantiated with the value of an existing parent-level internal draw γ_j with probability proportional to the number of bottom-level internal draws previously matched to γ_j. With probability proportional to β_0, it is instantiated with the value of a new parent-level draw. In this case, the new parent-level internal draw is drawn from the top-level base measure $\boldsymbol{u}$. In this way, the internal draws at one level are treated as observations by the next level up in the hierarchy, and there is path from every observation to the top-level base measure $\boldsymbol{u}$, via the internal draws. The predictive probability of generating word w in conditioning context $w't$ under the hierarchical version of prior 1 (portrayed in figure 3.3) is therefore given by

$$P(w \mid w', t, \boldsymbol{w}, \boldsymbol{z}, \beta, \beta_0) = \int P(w \mid w', t, \boldsymbol{w}, \boldsymbol{z}, \beta\boldsymbol{n})\, P(\boldsymbol{n} \mid \boldsymbol{w}, \boldsymbol{z}, \beta_0\boldsymbol{u})\, \mathrm{d}^W \boldsymbol{n} \tag{3.37}$$

$$= \int \frac{\hat{N}_{w|w't} + \beta n_w}{\hat{N}_{\cdot|w't} + \beta} \operatorname{Dir}(\boldsymbol{n} \mid \{\hat{N}_w + \beta_0 u_w\}_{w=1}^W)\, \mathrm{d}^W \boldsymbol{n} \tag{3.38}$$

$$= \frac{\hat{N}_{w|w't} + \beta \dfrac{\hat{N}_w + \beta_0 u_w}{\hat{N}_\cdot + \beta_0}}{\hat{N}_{\cdot|w't} + \beta} \tag{3.39}$$

where

$$\hat{N}_{w|w't} = \sum_{i=1}^{I} N^{(i)}_{\cdot|w't}\, \delta(\gamma_i - w), \tag{3.40}$$

$$\hat{N}_w = \sum_{j=1}^{J} N^{(j)}_{\cdot}\, \delta(\gamma_j - w) \tag{3.41}$$

and I and J are the current number of bottom-level and parent-level internal draws, respectively. The quantity $N^{(i)}_{\cdot|w't}$ is the number of observations matched to bottom-level internal draw γ_i, while $N^{(j)}_{\cdot}$ is the number of bottom-level internal draws matched to parent-level internal draw γ_j. As before, the bottom-level count $\hat{N}_{w|w't}$ is equal to $N_{w|w't}$—the number of times word w has been observed in context $w't$. However, $\hat{N}_w$ is not necessarily equal to N_w—the number of tokens of type w.

In practice, for real-world data $\boldsymbol{w}$ and $\boldsymbol{z}$, the number of internal draws for each Dirichlet-multinomial and the paths from the observations to the top-level base measure $\boldsymbol{u}$ are unknown. This information is only available for synthetic data explicitly

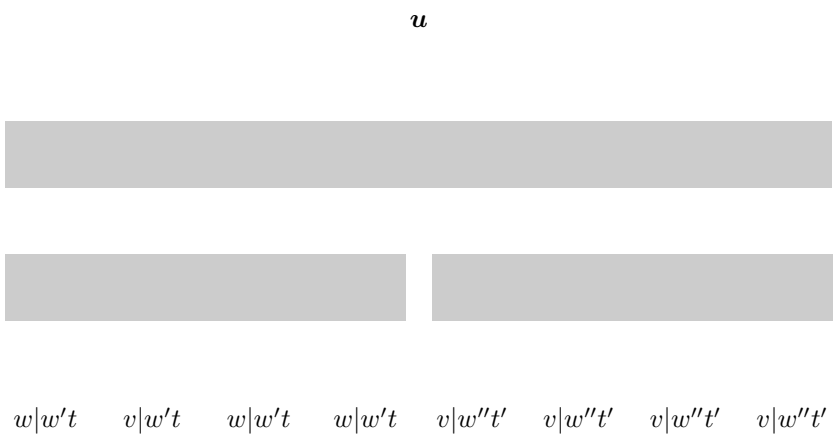
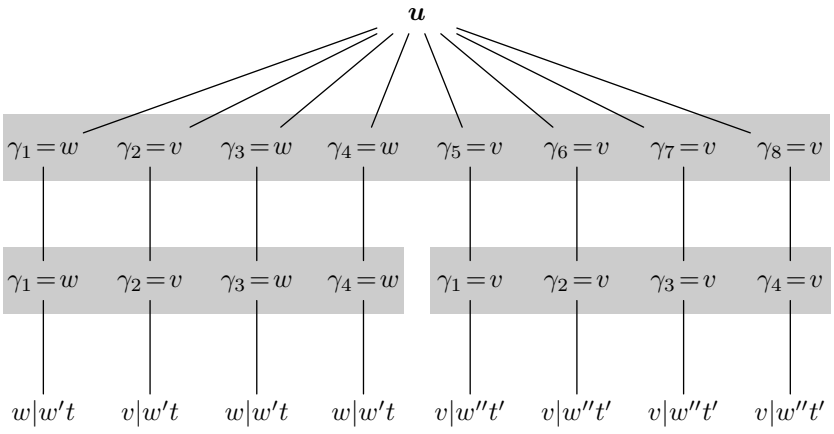


Figure 3.6: The only available information prior to inference: The values of the observed variables (bottom row); that there is Dirichlet-multinomial per conditioning context (bottom grey boxes), each of which can ultimately have a maximum of four internal draws; that these Dirichlet-multinomials share a parent Dirichlet-multinomial (top grey box), which can ultimately have a maximum of eight internal draws; that the internal draws for this parent Dirichlet-multinomial (currently unknown) will ultimately come from the top-level base measure $\boldsymbol{u}$.

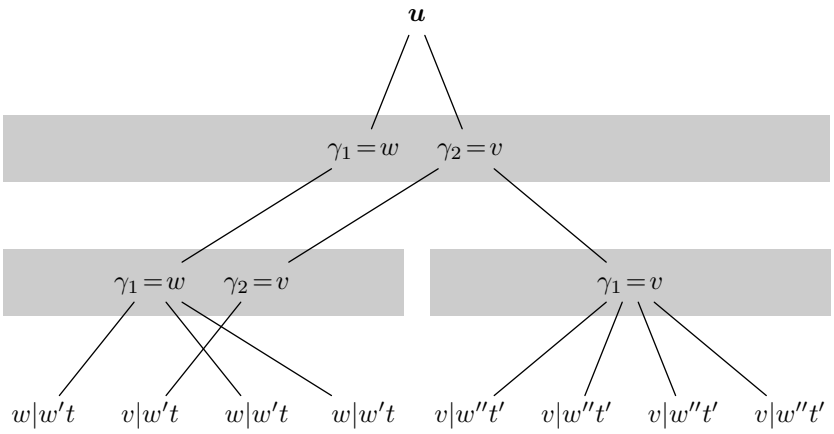
generated using the model. As depicted in figure 3.6, the only information available for real-world data is the value of each observation (and hence the bottom-level counts—the number of times each word w has been observed in each context $w't$), the number of Dirichlet-multinomials at each level, and the number of levels in the hierarchy. It is therefore necessary to infer the internal draws for each Dirichlet-multinomial, along with the path from each observation to the top-level base measure. The most general way of doing this is by using Gibbs sampling (Teh et al., 2006). However, two approximations to the Gibbs sampling procedure (Cowans, 2006) are particularly useful due to their computational efficiency and ease of implementation:

- **Maximal path assumption:** Every observation is assumed to have been generated by using a new internal draw. Furthermore each internal draw is assumed to have been generated by using a new parent-level internal draw. The number of internal draws used in each Dirichlet-multinomial in the hierarchy is therefore the largest possible. Under this assumption, the counts for every level are equal to the raw observation counts: e.g., $\hat{N}_w = \sum_{j=1}^{J} N_{\cdot}^{(j)} \, \delta\left(\gamma_j - w\right)$ is simply equal to N_w—the total number of times that word w has been observed in any context.

- **Minimal path assumption:** An observation is assumed to have been generated by using a new internal draw if and only if there is no existing internal draw with the same value as that observation. Each internal draw is similarly assumed to have been generated by using a new parent-level internal draw if and only if there is no existing parent-level internal draw with the same value as the (child-level) draw in question. The number of internal draws used in each Dirichlet-multinomial is therefore the smallest possible—in any given Dirichlet-multinomial, no two internal draws will have the same value w. Under this as-



(a) Maximal path assumption.



(b) Minimal path assumption.

Figure 3.7: The maximal and minimal path assumptions.

sumption, the counts used in every level in the hierarchy, except for the bottom-most level, are *type* counts: e.g., $\hat{N}_w = \sum_{j=1}^{J} N_{\cdot}^{(j)}\, \delta\,(\gamma_j - w)$ is equal to the number of different conditioning contexts $w't$ in which word w has been observed.

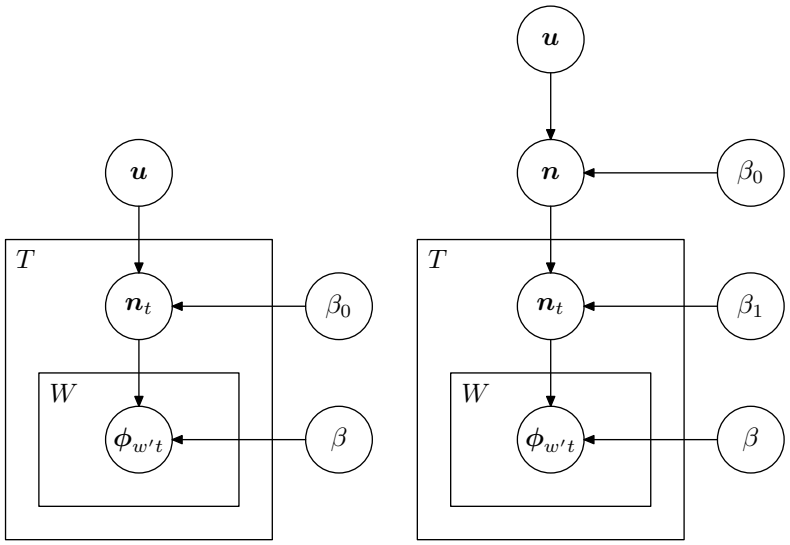
The process of drawing data under each of these assumptions is depicted in figure 3.7.

As in the model variants without hyperpriors, the evidence for the hyperparameters $U = \{\beta, \beta_0, \alpha\boldsymbol{n}\}$ under the hierarchical version of prior 1 is given by

$$P(\boldsymbol{w} \,|\, U) = \sum_{\boldsymbol{z}} P(\boldsymbol{w} \,|\, \boldsymbol{z}, U) P(\boldsymbol{z} \,|\, U). \tag{3.42}$$

However, $P(\boldsymbol{w} \,|\, \boldsymbol{z}, U)$ is now given by

$$P(\boldsymbol{w} \,|\, \boldsymbol{z}, \beta, \beta_0) = \prod_n \frac{(\hat{N}_{w_n|w_{n-1}z_n})_{<n} + \beta\, \dfrac{(\hat{N}_{w_n})_{<n} + \beta_0 u_{w_n}}{(\hat{N}_{\cdot})_{<n} + \beta_0}}{(\hat{N}_{\cdot|w_{n-1}z_n})_{<n} + \beta}, \tag{3.43}$$



(a) (b)

Figure 3.8: Hierarchical versions of prior 2.

where the subscript "$< n$" denotes a quantity that includes only data from positions $1 \ldots n-1$. The probability $P(\boldsymbol{z} \mid U)$ is unchanged from equation 3.27.

Prior 2: There are two possible hyperpriors for prior 2. The first is

$$P(\boldsymbol{n}_t \mid \beta_0 \boldsymbol{u}) = \text{Dir}\,(\boldsymbol{n}_t \mid \beta_0 \boldsymbol{u}), \tag{3.44}$$

while the second is

$$P(\boldsymbol{n}_t \mid \beta_1 \boldsymbol{n}) = \text{Dir}\,(\boldsymbol{n}_t \mid \beta_1 \boldsymbol{n}) \tag{3.45}$$

$$P(\boldsymbol{n} \mid \beta_0 \boldsymbol{u}) = \text{Dir}\,(\boldsymbol{n} \mid \beta_0 \boldsymbol{u}). \tag{3.46}$$

In the first hyperprior (equation 3.44), each topic-specific base measure $\boldsymbol{n}_t$ is given a Dirichlet distribution with uniform base measure $\boldsymbol{u}$ and concentration parameter β_0. In the second hyperprior (equations 3.45 and 3.46), each $\boldsymbol{n}_t$ is given a Dirichlet distribution with base measure $\boldsymbol{n}$ and concentration parameter β_1, where $\boldsymbol{n}$ is itself drawn from a Dirichlet, with uniform base measure $\boldsymbol{u}$ and concentration parameter β_0. (β, β_0 and β_1 are all given noninformative priors.) The hierarchical Dirichlet priors over $\boldsymbol{\phi}_{w't}$ induced by these hyperpriors (referred to henceforth as "hierarchical prior 2a" and "hierarchical prior 2b", respectively) are shown in figures 3.8a and 3.8b. In both of these hierarchical priors, the base measures $\boldsymbol{n}_t$ and $\boldsymbol{n}_{t'}$ for (nonidentical) topics t and t' are related, via $\boldsymbol{u}$ in 2a and via $\boldsymbol{n}$ and $\boldsymbol{u}$ in 2b. In the nonhierarchical version of prior 2, described in section 3.4.1, the base measures for t and t' are independent.

As with prior 1, the effects of these hyperpriors are best discussed in terms of the predictive probability of generating word w in the context of previous word w' and

topic t. Under hierarchical prior 2a ($\boldsymbol{n}_t$ per topic, tied via $\boldsymbol{u}$) this probability is

$$P(w \mid w', t, \boldsymbol{w}, \boldsymbol{z}, \beta, \beta_0) = \frac{\hat{N}_{w|w't} + \beta \dfrac{\hat{N}_{w|t} + \beta_0 u_w}{\hat{N}_{\cdot|t} + \beta_0}}{\hat{N}_{\cdot|w't} + \beta}, \tag{3.47}$$

while under hierarchical prior 2b ($\boldsymbol{n}_t$ per topic, tied via $\boldsymbol{n}$ and $\boldsymbol{u}$) it is

$$P(w \mid w', t, \boldsymbol{w}, \boldsymbol{z}, \beta, \beta_1, \beta_0) = \frac{\hat{N}_{w|w't} + \beta \dfrac{\hat{N}_{w|t} + \beta_1 \dfrac{\hat{N}_{w} + \beta_0 u_w}{\hat{N}_{\cdot} + \beta_0}}{\hat{N}_{\cdot|t} + \beta_1}}{\hat{N}_{\cdot|w't} + \beta}. \tag{3.48}$$

The effects of the two different hyperpriors may be understood by examining equations 3.47 and 3.48: Under hierarchical prior 2a ($\boldsymbol{n}_t$ per topic t, tied via $\boldsymbol{u}$), $\hat{N}_{w|w't}/\hat{N}_{\cdot|w't}$ is effectively smoothed with $\hat{N}_{w|t}/\hat{N}_{\cdot|t}$—a quantity which depends on word w and topic t—and $u_w = 1/W$. Under hierarchical prior 2b ($\boldsymbol{n}_t$ per topic t, tied via $\boldsymbol{n}$ and $\boldsymbol{u}$), $\hat{N}_{w|w't}/\hat{N}_{\cdot|w't}$ is effectively smoothed with $\hat{N}_{w}/\hat{N}_{\cdot}$ (which depends on word w only) as well as $\hat{N}_{w|t}/\hat{N}_{\cdot|t}$ and $u_w = 1/W$. In other words, under 2a, words that have not previously been seen in topic t are given equal probabilities of occurring in topic t in the future. Under 2b, however, words that have never been seen in topic t are given unequal, word-specific probabilities of being seen in that topic in the future.

Prior 3: There are also two possible hyperpriors for prior 3. The first is

$$P(\boldsymbol{n}_{w'} \mid \beta_0 \boldsymbol{u}) = \text{Dir}\,(\boldsymbol{n}_{w'} \mid \beta_0 \boldsymbol{u}), \tag{3.49}$$

while the second is

$$P(\boldsymbol{n}_{w'} \mid \beta_1 \boldsymbol{n}) = \text{Dir}\,(\boldsymbol{n}_{w'} \mid \beta_1 \boldsymbol{n}) \tag{3.50}$$

$$P(\boldsymbol{n} \mid \beta_0 \boldsymbol{u}) = \text{Dir}\,(\boldsymbol{n} \mid \beta_0 \boldsymbol{u}). \tag{3.51}$$

Under the prior over $\boldsymbol{\phi}_{w't}$ induced by the first of these hyperpriors ($\boldsymbol{n}_{w'}$ per previous word context w', tied via $\boldsymbol{u}$), the predictive probability of w given $w't$ is given by

$$P(w \mid w', t, \boldsymbol{w}, \boldsymbol{z}, \beta, \beta_0) = \frac{\hat{N}_{w|w't} + \beta \dfrac{\hat{N}_{w|w'} + \beta_0 u_w}{\hat{N}_{\cdot|w'} + \beta_0}}{\hat{N}_{\cdot|w't} + \beta}. \tag{3.52}$$

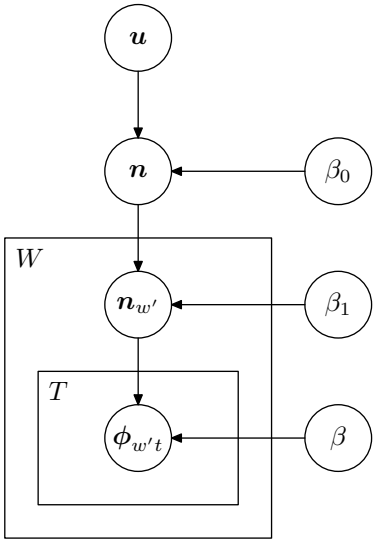


Figure 3.9: Hierarchical version of prior 3.

The second hyperprior ($\boldsymbol{n}_{w'}$ per w', tied via $\boldsymbol{n}$ and $\boldsymbol{u}$) gives rise to

$$P(w \mid w', t, \boldsymbol{w}, \boldsymbol{z}, \beta, \beta_1, \beta_0) = \frac{\hat{N}_{w|w't} + \beta \dfrac{\hat{N}_{w|w'} + \beta_1 \dfrac{\hat{N}_w + \beta_0 u_w}{\hat{N}_\cdot + \beta_0}}{\hat{N}_{\cdot|w'} + \beta_1}}{\hat{N}_{\cdot|w't} + \beta}. \tag{3.53}$$

Again, the effects of the two hyperpriors may be understood from the corresponding predictive probabilities. Using the first hyperprior, $\hat{N}_{w|w't}/\hat{N}_{\cdot|w't}$ is smoothed with $N_{w|w'}/N_{\cdot|w'}$—a quantity specific to words w and w'—and $u_w = 1/W$. Using the second hyperprior, these quantities are also smoothed with $\hat{N}_w/\hat{N}_\cdot$. This means that for a given context w', the first hyperprior will result in all words w that have not previously been seen in this context being given an equal probability of occurring in this context in the future. The second hyperprior, however, will cause each unseen word w to be given a probability that is related to its unigram count (under the maximal path assumption, the number of times w has occurred in $\boldsymbol{w}$; under the minimal path assumption, the number of w' contexts in which w has appeared in $\boldsymbol{w}$). These hyperpriors have been well-studied in the context of language modelling, with the latter giving significantly better results, due to the sparsity of word pair occurrences. For this reason, only the second hyperprior is considered henceforth. The resultant hierarchical Dirichlet prior induced over $\boldsymbol{\phi}_{w't}$ ("hierarchical prior 3") is shown in figure 3.9.

Sampling Concentration Parameters

Given a corpus $\boldsymbol{w}$ and any of the hierarchical priors described above, typical values for the hyperparameters β, β_1 and β_0 can be inferred by alternating between resampling

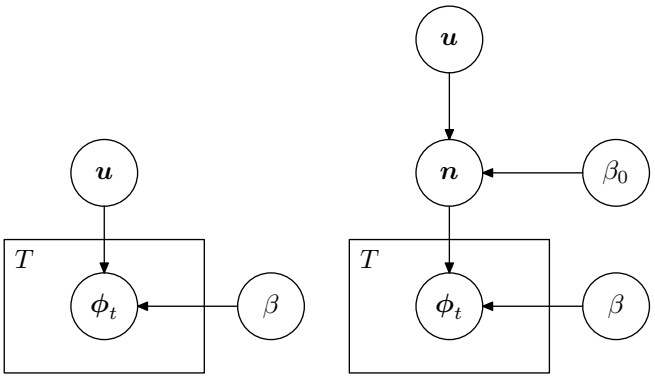


Figure 3.10: Hierarchical priors for latent Dirichlet allocation.

the topic assignments $\boldsymbol{z}$ (using Gibbs sampling) and resampling the hyperparameters given the current topic assignments (using slice sampling (Neal, 2003)). Each topic assignment z_n is resampled from its conditional posterior given all other variables:

$$P(z_n{=}t \,|\, \boldsymbol{z}_{\setminus n}, \boldsymbol{w}, U) \propto P(w_n \,|\, w_{n-1}, z_n{=}t, \boldsymbol{w}_{\setminus n}, \boldsymbol{z}_{\setminus n}, U)\, \frac{(N_{t|d_n})_{\setminus n} + \alpha m_t}{(N_{\cdot|d_n})_{\setminus n} + \alpha}, \tag{3.54}$$

where the subscript "$\setminus n$" denotes a quantity that excludes data from the n^{th} position in the corpus, and $U = \{\beta, \beta_0, \alpha\boldsymbol{m}\}$ (hierarchical priors 1 and 2a) or $\{\beta, \beta_1, \beta_0, \alpha\boldsymbol{m}\}$ (2b and 3). Finally, $P(w_n \,|\, w_{n-1}, z_n = t, \boldsymbol{w}_{\setminus n}, \boldsymbol{z}_{\setminus n}, U)$ may be obtained by treating the n^{th} observation as the last to arrive and using either equation 3.39, 3.47, 3.48 or 3.53.

Slice sampling (Neal, 2003) is a Markov chain Monte Carlo method that adapts to the sampled distribution by sampling uniformly from the area under its density function. In the one-dimensional case, where the goal is to sample from $P(x) \propto P^*(x)$, slice sampling works by making transitions from one two-dimensional point (x, u) under the plot of $P^*(x)$ to another (x', u'), such that the distribution of points tends to a uniform distribution over the area under $P^*(x)$. This approach can be generalised to efficiently produce S multidimensional samples, as shown in algorithm 3.2.

When slice sampling the hyperparameters $\{\beta, \beta_0\}$ (hierarchical priors 1 and 2a) or $\{\beta, \beta_1, \beta_0\}$ (2b and 3), the relevant density is $P(U \,|\, \boldsymbol{w}, \boldsymbol{z})$. Placing an improper noninformative prior over these hyperparameters, this density is proportional to $P(\boldsymbol{w} \,|\, \boldsymbol{z}, U)$, which may be computed using the predictive distribution over words:

$$P(\boldsymbol{w} \,|\, \boldsymbol{z}, U) = \prod_t P(w_n \,|\, w_{n-1}, z_n, \boldsymbol{w}_{<n}, \boldsymbol{z}_{<n}, U). \tag{3.55}$$

for $s := 1 \ldots S$ {
  draw $u' \sim \text{Uniform}\,(0, P^*(\boldsymbol{x}))$
  for each dimension i {
    draw $r \sim \text{Uniform}\,(0, 1)$
    $x_i^l := x_i - r\sigma_i$
    $x_i^r := x_i^l + \sigma_i$
  }
  while true {
    for each dimension i {
      draw $x_i' \sim \text{Uniform}\,(x_i^l, x_i^r)$
    }
    if $P^*(\boldsymbol{x}') > u'$ {
      break
    else
      for each dimension i {
        if $x_i' < x_i$ {
          $x_i^l := x_i'$
        else
          $x_i^r := x_i'$
        }
      }
    }
  }
  $\boldsymbol{x} := \boldsymbol{x}'$
  output $\boldsymbol{x}$
}

Algorithm 3.2: Multidimensional slice sampling. The algorithm requires an initial value $\boldsymbol{x}$, a "step size" vector $\boldsymbol{\sigma}$, and the desired number of samples S.

3.5 Experiments

To evaluate the new topic-based language model, all but one of the model variants described in the previous section[2] were compared with latent Dirichlet allocation and MacKay and Peto's (bigram) hierarchical Dirichlet language model. The model variants without proper hyperpriors were trained identically using 20,000 iterations of the Gibbs EM algorithm described in section 3.4.1. Topic assignments and hyperparameters for the model variants with proper hyperpriors were obtained by alternating between Gibbs sampling topics (once) and slice sampling the hyperparameters (for five iterations). This was repeated 20,000 times. For simplicity, only the maximal path assumption was used. To enable a fair comparison, two versions of the baseline models were used: When evaluating the model variants with improper hyperpriors, latent Dirichlet allocation was trained using the Gibbs EM algorithm, while the hyperparameters of the hierarchical Dirichlet language model were inferred using the fixed-point

[2]Results for the model variant with a $\beta \boldsymbol{n}_{w'}$ hyperparameter vector for each previous word context w' and an improper noninformative hyperprior were not computed as the number of hyperparameters is extremely large—$W(W-1)$—with comparatively little data from which to estimate them.

iteration employed in the M-step. When evaluating the model variants with proper hyperpriors, latent Dirichlet allocation and MacKay and Peto's hierarchical Dirichlet language model were also given proper hyperpriors and their concentration parameters were inferred using slice sampling. Two different hyperpriors were used for latent Dirichlet allocation—the resultant hierarchical Dirichlet priors over $\boldsymbol{\phi}_t$ (depicted in figure 3.10) are analogous to hierarchical priors 2a and 2b for the topic-based language model. The hierarchical Dirichlet prior over $\boldsymbol{\phi}_{w'}$ for MacKay and Peto's model is shown in figure 3.1a. All experiments involving latent Dirichlet allocation and the topic-based language model were run with 2, 5, 10, 20, 50, 100, 200 and 500 topics.

3.5.1 Data

The models were compared using 250 papers from the proceedings of the NIPS conference[3], drawn randomly from the "nips_stream" data set of Steyvers and Griffiths[4]. 200 papers were used as training data, while the remaining fifty were used to evaluate the models. Punctuation characters were replaced with a PUNC type, while all numbers were replaced with a NUMBER type. To enable evaluation on documents containing words not present in the training data, words that occurred once in the training data (and zero times in the test data) or one or more times in the test data, but never in the training data, were replaced with the following UNSEEN types (Eisner, 1996a):

- UNSEEN-SHORT: used for words less than six characters long.
- UNSEEN-XX: used for words of six or more characters in length. XX is replaced with the last two characters of the word.

Preprocessing the data in this manner led to a vocabulary of 4,858 words. To improve computation speed, each paper was truncated after 500 tokens. This truncation resulted in a training data set of 99,836 tokens, and a test data set of 25,000 tokens.

3.5.2 Results

Bayesian generative models are typically evaluated by computing the probability of unseen test data $\boldsymbol{w}$, given training data $\boldsymbol{w}^{\text{train}}$ and hyperparameters U: The better the model, the higher the probability. For models of text, these results are usually reported in terms of the "information rate" of the test data, measured in bits per word. The information rate is computed from $P(\boldsymbol{w} \mid \boldsymbol{w}^{\text{train}}, U)$ as follows:

$$R = -\frac{\log_2 P(\boldsymbol{w} \mid \boldsymbol{w}^{\text{train}}, U)}{N_\cdot}, \tag{3.56}$$

[3]Conference on Neural Information Processing Systems, http://www.nips.cc/

[4]MATLAB Topic Modeling Toolbox 1.3.2, http://psiexp.ss.uci.edu/research/programs_data/toolbox.htm

where $N_{\cdot}$ is the number of tokens in $\boldsymbol{w}$. For the hierarchical Dirichlet language model, $P(\boldsymbol{w} \mid \boldsymbol{w}^{\text{train}}, U)$ may be computed directly. However, for topic-based models, computing $P(\boldsymbol{w} \mid \boldsymbol{w}^{\text{train}}, U)$ involves summing over both training and test topics. As mentioned in section 3.4.1, this is intractable. Within the topic modelling literature (Griffiths and Steyvers, 2004; Griffiths et al., 2005) it is common to approximate these sums using a single set of topics $\boldsymbol{z}^{\text{train}}$ for the training data, obtained using Gibbs sampling. Given these training topics and hyperparameters U, the sum over test topics $\boldsymbol{z}$ can be approximated using importance sampling (Kass and Raftery, 1995), in the following manner: Rather than approximating $\sum_{\boldsymbol{z}} P(\boldsymbol{w}, \boldsymbol{z} \mid \boldsymbol{w}^{\text{train}}, \boldsymbol{z}^{\text{train}}, U)$ using samples drawn from the prior over $\boldsymbol{z}$, the posterior $P(\boldsymbol{z} \mid \boldsymbol{w}, U)$ can be used as the sampling density in an importance sampler, thereby resulting in the following approximation:

$$P(\boldsymbol{w} \mid \boldsymbol{w}^{\text{train}}, \boldsymbol{z}^{\text{train}}, U) \simeq \frac{S}{\sum_{s=1}^{S} \frac{1}{P(\boldsymbol{w} \mid \boldsymbol{z}^{(s)})}} \tag{3.57}$$

$$= \text{HM}\left(\{P(\boldsymbol{w} \mid \boldsymbol{z}^{(s)})\}_{s=1}^{S}\right), \tag{3.58}$$

where $\text{HM}(\cdot)$ denotes the harmonic mean. Unfortunately, this approximation is unstable (Newton and Raftery, 1994). It also results in an unfair bias towards models with topics. This is because the topic assignments used in equation 3.57 are obtained by Gibbs sampling from the posterior distribution. Gibbs sampling repeatedly resamples each topic assignment z_n from the conditional posterior for that variable given all other variables. Consequently, topic assignments from positions $n' > n$ influence the assignment at position n and words from positions $n' > n$ implicitly influence the probability of the word at position n. Sampling topic assignments using Gibbs sampling means that the probability of each word, as used in equation 3.57, is implicitly influenced by future words, via the topic assignments. Models with topics are therefore given an unfair advantage over the hierarchical Dirichlet language model, where evaluation is performed in a strictly "left-to-right" fashion and later words cannot influence the probability of earlier words. A more realistic estimate of predictive performance can be obtained by decomposing $P(\boldsymbol{w} \mid \boldsymbol{w}^{\text{train}}, \boldsymbol{z}^{\text{train}}, U)$[5] as

$$P(\boldsymbol{w} \mid \boldsymbol{w}^{\text{train}}, \boldsymbol{z}^{\text{train}}, U) = \prod_{n} P(w_n \mid \boldsymbol{w}_{<n}, \boldsymbol{w}^{\text{train}}, \boldsymbol{z}^{\text{train}}, U) \tag{3.59}$$

$$= \prod_{n} \sum_{\boldsymbol{z}_{\leq n}} P(w_n, \boldsymbol{z}_{\leq n} \mid \boldsymbol{w}_{<n}, \boldsymbol{w}^{\text{train}}, \boldsymbol{z}^{\text{train}}, U), \tag{3.60}$$

and using an algorithm inspired by sequential Monte Carlo methods (Doucet et al., 2001) to approximate the sums over $\boldsymbol{z}_{\leq n}$, as in algorithm 3.3. This approximation method is appropriate for a wider range of language modelling applications—including predictive text entry systems (Ward et al., 2000; Ward, 2001) and speech recognition systems (Rabiner and Juang, 1993; Jelinek, 1998)—than the importance

[5] As mentioned previously, the training topic assignments $\boldsymbol{z}^{\text{train}}$ should ideally be marginalised out along with $\boldsymbol{z}$. However, for simplicity, they are instead clamped after training.

1: initialise $l := 0$
2: **for** each position n in the test data $\boldsymbol{w}$ {
3: $\quad p_n = 0$
4: $\quad$ **for** each particle $r = 1$ to R {
5: $\quad\quad$ **for** $n' < n$ {
6: $\quad\quad\quad$ resample $z_{n'} \sim P(z_{n'} \mid (\boldsymbol{z}_{<n})_{\setminus n'}, \boldsymbol{w}_{<n}, \boldsymbol{w}^{\text{train}}, \boldsymbol{z}^{\text{train}})$
7: $\quad\quad$ }
8: $\quad\quad p_n := p_n + \sum_t P(w_n \mid z_n = t, \boldsymbol{z}_{<n}, \boldsymbol{w}_{<n}, \boldsymbol{w}^{\text{train}}, \boldsymbol{z}^{\text{train}})\, P(z_n = t \mid \boldsymbol{z}_{<n}, \boldsymbol{z}^{\text{train}})$
9: $\quad$ }
10: $\quad p_n := p_n / R$
11: $\quad l := l + \log p_n$
12: $\quad$ sample $z_n \sim P(z_n \mid \boldsymbol{z}_{<n}, \boldsymbol{w}_{\leq n}, \boldsymbol{w}^{\text{train}}, \boldsymbol{z}^{\text{train}})$
13: }
14: return l

Algorithm 3.3: A "left-to-right" evaluation algorithm for topic models. The algorithm computes $l \simeq \sum_n \log \sum_{\boldsymbol{z}_{\leq n}} P(w_n, z_{\leq n} \mid \boldsymbol{w}_{<n}, \boldsymbol{w}^{\text{train}}, \boldsymbol{z}^{\text{train}}, U)$ using R particles. In practice, it is sufficient to approximate the resampling loop (lines 5–7) by resampling only the topic assignments for tokens in the current document.

Prior 1	Prior 2
$\text{Dir}\,(\boldsymbol{\phi}_{w't} \mid \beta \boldsymbol{n})$	$\text{Dir}\,(\boldsymbol{\phi}_{w't} \mid \beta \boldsymbol{n}_t)$

Table 3.1: The two topic-based language model variants with nonhierarchical priors and optimised hyperparameters that were experimentally compared with the hierarchical Dirichlet language model and latent Dirichlet allocation.

sampling approximation in equation 3.57, because of its "left-to-right" operation.

Nonhierarchical Priors

The information rates of the test data, computed using algorithm 3.3 and twenty particles, are shown in figure 3.11 for the hierarchical Dirichlet language model and two of the topic-based language model variants with improper hyperpriors and optimised hyperparameters (prior 1 and prior 2, summarised in table 3.1). The information rate for latent Dirichlet allocation, even with 500 topics, is much worse than the other models (8.90 bits per word) and is therefore not shown. For the models with topics, performance improves as the number of topics is increased, levelling off at around 100 topics. (With just one topic, both variants of the topic-based language model are identical to the hierarchical Dirichlet language model.) The hierarchical Dirichlet language model exhibits an information rate of 8.12 bits per word, while the best performance (7.98 bits per word) is achieved by the topic-based language model variant with prior 2 ($\beta \boldsymbol{n}_t$ per topic) and 200 topics. These results clearly indicate that for models with improper hyperpriors and optimised hyperparameters, the topic-based language model—particularly with prior 2 ($\beta \boldsymbol{n}_t$ per topic)—is better able to model the test data than either latent Dirichlet allocation or the hierarchical Dirichlet language model.

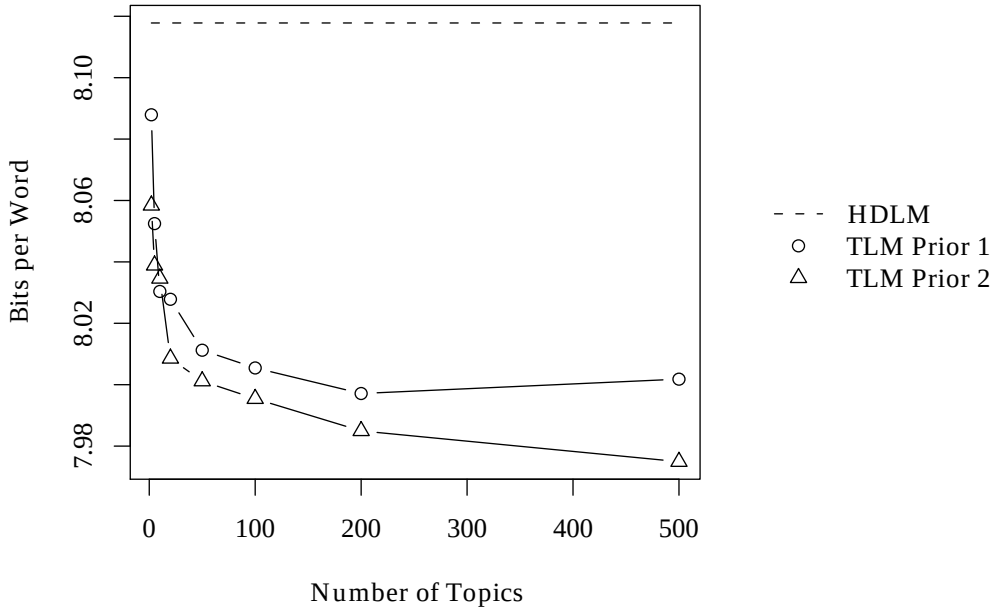


Figure 3.11: Information rates of the test data (computed using the "left-to-right" algorithm) for each of the model variants with improper hyperpriors. The fewer the bits per word, the better the model. The training data consisted of 99,863 tokens, while the test data consisted of 25,000. "HDLM" is MacKay and Peto's hierarchical Dirichlet language model, "TLM Prior 1" is the new topic-based language model with prior 1 (single $\beta\boldsymbol{n}$ vector), while "TLM Prior 2" is the new model with prior 2 ($\beta\boldsymbol{n}_t$ vector per topic). Latent Dirichlet allocation (not shown) achieves a worse information rate than the other models: 8.90 bits per word.

In addition to comparing predictive performance, it is also instructive to look at the inferred topics. Table 3.2 shows the words most frequently assigned to a selection of topics inferred from the training data by latent Dirichlet allocation and the topic-based language model variants with improper hyperpriors. Content words are highlighted in bold. Stop words, such as "to", "and" and "the", were identified by their presence on a standard list of stop words[6]. For each model, the topic shown in the final column consists almost entirely of stop words and is used more often than the other topics.

The topics inferred using latent Dirichlet allocation contain many stop words. This is not normally a problem for latent Dirichlet allocation, because stop words are removed prior to inference. However, when constructing a language model, word order plays a significant role and both content words and stop words must be accurately predicted—removing stop words is therefore inappropriate. The topics for the new topic-based language model with prior 1 (single $\beta\boldsymbol{n}$) are also dominated by stop words, though to a slightly lesser extent. Inspection of the estimated $\alpha\boldsymbol{m}$ hyperparameters indicates that for latent Dirichlet allocation the inferred αm_t value for the topic

[6] http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/stop_words

shown in the final column (mainly stop words) is roughly 0.05, while the mean αm_t value for the other topics is 0.008. For the new model with prior 1 (single $\beta\boldsymbol{n}$), this difference is even more pronounced: the inferred αm_t value for the stop word topic is 4.7, while the mean value for the other topics is 0.04. The larger difference is likely to be the reason why the topics inferred by the new model with prior 1 are slightly less dominated by stop words than are those inferred by latent Dirichlet allocation.

The most interpretable topics are those inferred by the topic-based language model with prior 2 ($\beta\boldsymbol{n}_t$ vector per topic). Except for the topic shown in the final column, very few topics contain stop words and the words in each topic are clearly semantically related. As expected, the inferred αm_t for the stop word topic is much larger than the mean inferred αm_t for the other topics: 50.7 versus 0.80. The separate hyperparameter vector for each topic ($\beta\boldsymbol{n}_t$) means that model is able to use the hyperparameters to capture the characteristics of the different topics. With prior 1 (single $\beta\boldsymbol{n}$), the model is unable to do this and all topic-specific information must be captured by the counts.

Hierarchical Priors

The information rates of the test data, computed using algorithm 3.3, are shown in figure 3.12 for the hierarchical Dirichlet language model and the topic-based language model variants with proper hyperpriors and sampled concentration parameters (hierarchical priors 1, 2a, 2b and 3, summarised in table 3.3). Results for latent Dirichlet allocation are not shown: With hierarchical prior 1a (see figure 3.10a and table 3.4; symmetric Dirichlet), latent Dirichlet allocation achieves a much worse information rate (8.93 bits per word) than the other models. Meanwhile, in every experiment involving latent Dirichlet allocation and hierarchical prior 1b (see figure 3.10b and table 3.4; single $\boldsymbol{n}$ vector), the inferred value of β_0 was sufficiently large that $\boldsymbol{n}$ was effectively ignored, making the model equivalent to latent Dirichlet allocation with hierarchical prior 1a. This was verified by inspection of the sampled values of the other hyperparameters, which were identical to those obtained using latent Dirichlet allocation with hierarchical prior 1a. Information rates for the new topic-based language model with hierarchical prior 3 ($\boldsymbol{n}_{w'}$ per previous word w') are only shown for 50, 100, 200 and 500 topics. With two topics, either the inferred value of β was vast, thereby reducing the model to the hierarchical Dirichlet language model, or the inferred value of β_1 was vast and the model was reduced to the topic-based language model with hierarchical prior 1 (single $\boldsymbol{n}$). This was confirmed by inspection of the other hyperparameters, which were either identical to those of the hierarchical Dirichlet language model or to the new model with hierarchical prior 1. With five topics, the inferred β was also sufficiently large that the model was reduced to the hierarchical Dirichlet language model. With ten topics, the inferred α hyperparameter was so small that all tokens were assigned to a single topic, again causing the model to be reduced to a hierarchical Dirichlet language model. With 50 or more topics, the model outperformed

Latent Dirichlet Allocation				
PUNC	PUNC	PUNC	the	PUNC
the	the	**synapses**	PUNC	the
as	of	the	of	**number**
inputs	**analog**	of	a	*
outputs	and	and	to	is
is	*	**feedback**	**classifiers**	of
to	we	**number**	and	we
extra	**number**	during	is	and
features	a	**learning**	**number**	a
used	to	**associative**	**weak**	in
Topic-Based Language Model (Prior 1)				
PUNC	PUNC	PUNC	PUNC	PUNC
feature	**analog**	**synaptic**	of	the
as	and	**synapses**	**number**	of
and	of	**number**	in	a
inputs	**chip**	*	the	to
features	**number**	of	and	**number**
for	**architecture**	**feedback**	**dimensionality**	*
*	**are**	**learning**	is	is
outputs	is	**feedforward**	**classification**	and
extra	**figure**	**sensory**	*	in
Topic-Based Language Model (Prior 2)				
analog	**synapses**	**capacity**	**associative**	PUNC
signal	**cortical**	**classifier**	**regions**	the
circuit	**excitatory**	**sequential**	**feedforward**	of
high	**programs**	**efficiency**	**region**	a
signals	**shift**	**distinct**	**correct**	to
adaptation	**temporally**	**equal**	**separate**	**number**
frequency	**combining**	**gradients**	**synapses**	*
filtering	**oscillatory**	**minimising**	**storage**	in
background	**attractor**	**vc-dimension**	**select**	and
window	**basins**	**estimates**	**mappings**	for

Table 3.2: Example topics inferred by latent Dirichlet allocation and the new topic-based language model with improper hyperpriors and 100 topics.

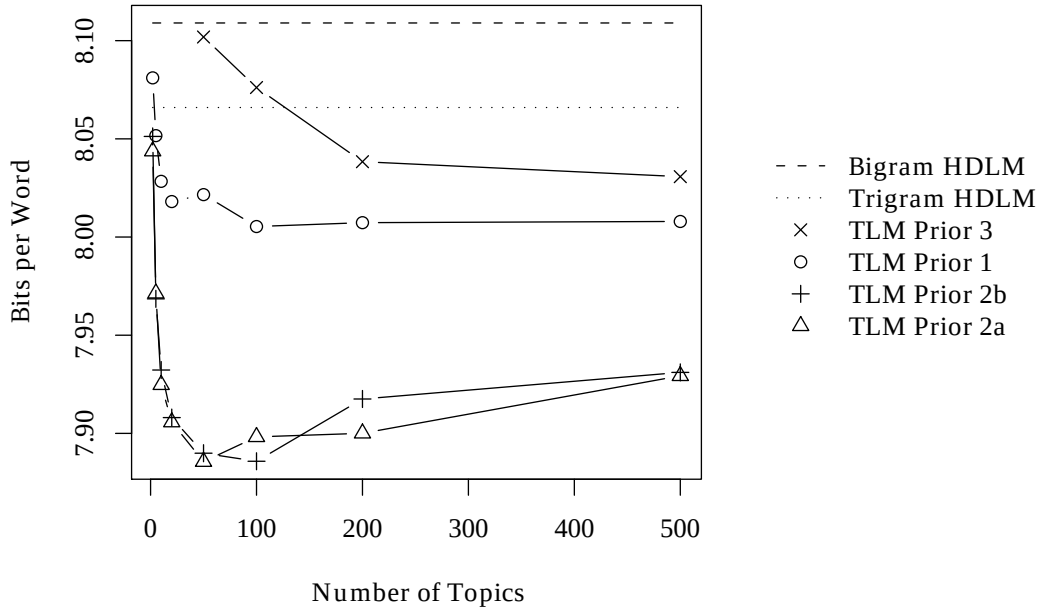


Figure 3.12: Information rates of the test data (computed using the "left-to-right" algorithm) for each of the model variants with proper hyperpriors. The fewer the bits per word, the better the model. The training data consisted of 96,836 tokens, while the test data consisted of 25,000. "Bigram HDLM" is the bigram version of MacKay and Peto's hierarchical Dirichlet language model, while "Trigram HDLM" is the trigram version. "TLM Prior 3" is the new topic-based language model with hierarchical prior 3 ($\boldsymbol{n}_{w'}$ per previous word w', tied via $\boldsymbol{n}$ and $\boldsymbol{u}$), "TLM Prior 1" is the new model with hierarchical prior 1 (single $\boldsymbol{n}$), "TLM Prior 2b" is the new model with hierarchical prior 2b ($\boldsymbol{n}_t$ per topic, tied via $\boldsymbol{n}$ and $\boldsymbol{u}$), and "TLM Prior 2a" is the new model with hierarchical prior 2a ($\boldsymbol{n}_t$ per topic, tied via $\boldsymbol{u}$). Latent Dirichlet allocation with hierarchical prior 1a (symmetric Dirichlet) exhibits the worst performance (8.93 bits per word) and is therefore not shown.

MacKay and Peto's hierarchical Dirichlet language model, but did not perform as well as the other model variants. These results indicate that hierarchical prior 3 ($\boldsymbol{n}_{w'}$ per previous word w') is not a good choice of prior for the topic-based language model.

With hierarchical priors 1 (single $\boldsymbol{n}$), 2a ($\boldsymbol{n}_t$ per topic, tied via $\boldsymbol{u}$) and 2b ($\boldsymbol{n}_t$ per topic, tied via $\boldsymbol{n}$ and $\boldsymbol{u}$), the topic-based language model outperforms MacKay and Peto's hierarchical Dirichlet language model for all numbers of topics. The model variants with hierarchical priors 2a and 2b perform better than the variant with hierarchical prior 1. Interestingly, with 200 or fewer topics, there is almost no difference in the information rates obtained using hierarchical priors 2a and 2b. Inspection of the inferred hyperparameters indicates that this is because the inferred value of β_0 for hierarchical prior 2b ($\boldsymbol{n}_t$ per topic, tied via $\boldsymbol{n}$ and $\boldsymbol{u}$) is sufficiently large that $\boldsymbol{n}$ is effectively ignored, making the prior equivalent to hierarchical prior 2a ($\boldsymbol{n}_t$ per topic, tied via $\boldsymbol{u}$). With 200 or 500 topics, β_0 has a more reasonable value, however the subsequent

Prior 1	Prior 2a	Prior 2b	Prior 3
$\mathrm{Dir}\,(\boldsymbol{\phi}_{w't} \mid \beta\boldsymbol{n})$	$\mathrm{Dir}\,(\boldsymbol{\phi}_{w't} \mid \beta\boldsymbol{n}_t)$	$\mathrm{Dir}\,(\boldsymbol{\phi}_{w't} \mid \beta\boldsymbol{n}_t)$	$\mathrm{Dir}\,(\boldsymbol{\phi}_{w't} \mid \beta\boldsymbol{n}_{w'})$
$\mathrm{Dir}\,(\boldsymbol{n} \mid \beta_0\boldsymbol{u})$	$\mathrm{Dir}\,(\boldsymbol{n}_t \mid \beta_0\boldsymbol{u})$	$\mathrm{Dir}\,(\boldsymbol{n}_t \mid \beta_1\boldsymbol{n})$	$\mathrm{Dir}\,(\boldsymbol{n}_{w'} \mid \beta_1\boldsymbol{n})$
—	—	$\mathrm{Dir}\,(\boldsymbol{n} \mid \beta_0\boldsymbol{u})$	$\mathrm{Dir}\,(\boldsymbol{n} \mid \beta_0\boldsymbol{u})$

Table 3.3: The four topic-based language model variants with hierarchical priors and sampled concentration parameters that were experimentally compared with the hierarchical Dirichlet language model and latent Dirichlet allocation.

Prior 1a	Prior 1b
$\mathrm{Dir}\,(\boldsymbol{\phi}_t \mid \beta\boldsymbol{u})$	$\mathrm{Dir}\,(\boldsymbol{\phi}_t \mid \beta\boldsymbol{n})$
—	$\mathrm{Dir}\,(\boldsymbol{n} \mid \beta_0\boldsymbol{u})$

Table 3.4: The two latent Dirichlet allocation variants that were experimentally compared with the topic-based language model variants with hierarchical priors.

information rates are worse than those obtained using hierarchical prior 2a.

The most striking feature of figure 3.12 is that all variants of the new model—particularly hierarchical prior 2a ($\boldsymbol{n}_t$ per topic, tied via $\boldsymbol{u}$)—exhibit much better information rates than even a *trigram* hierarchical Dirichlet language model. The difference between the information rates achieved by the trigram hierarchical Dirichlet language model and the topic-based language model with hierarchical prior 2a and 50 topics is roughly three times the difference between the trigram and bigram language models.

Example topics inferred by latent Dirichlet allocation with hierarchical prior 1a (symmetric Dirichlet) and the topic-based language model with hierarchical priors 1 (single $\boldsymbol{n}$) and 2a ($\boldsymbol{n}_t$ per topic, tied via $\boldsymbol{u}$) are shown in table 3.5. Example topics are not shown for the model variant with hierarchical prior 3 ($\boldsymbol{n}_{w'}$ per previous word w', tied via $\boldsymbol{n}$ and $\boldsymbol{u}$): Very few of the inferred topics contain anything other than stop words.

The topics inferred by the model variant with hierarchical prior 1 contain several stop words. This is because the quantity $N_{w|w't}/N_{\cdot|w't}$ is smoothed with $N_w/N_{\cdot}$. Since stop words occur more often then other words, $N_w/N_{\cdot}$ is larger for stop words, and they dominate the topics. In contrast, few of the topics inferred by the model variant with hierarchical prior 2a contain any stop words. These are instead captured by a separate (automatically inferred) stop word topic, shown in the final column of table 3.5.

As mentioned previously, the topics inferred by latent Dirichlet allocation with an improper hyperprior and optimised hyperparameters are heavily dominated by stop words. In contrast, only one of the topics (displayed in the final column of table 3.5) inferred by latent Dirichlet allocation with hierarchical prior contains a significant number of stop words. This is most likely due to the fact that under hierarchical prior 1a, the $N_{w|t}/N_{\cdot|t}$ is smoothed by a constant value, $1/W$. The latent Dirichlet allocation variant with improper hyperpriors smooths $N_{w|t}/N_{\cdot|t}$ with n_w—a quantity that is directly related to the number of different topics in which word w occurs. In other

Latent Dirichlet Allocation (Hierarchical Prior 1a)				
analog	**stimuli**	**synapses**	**robot**	the
vlsi	**responses**	**activity**	**navigation**	PUNC
chip	**response**	**feedback**	**environment**	of
architecture	**population**	**synaptic**	**module**	a
circuit	**experiments**	during	have	to
complex	**behavioural**	**connections**	**avoidance**	is
implementation	**activities**	**learning**	**world**	in
circuits	**underlying**	**feedforward**	**modules**	and
design	e.g.	**selective**	**obstacle**	by
hardware	**active**	**associative**	**mobile**	that
Topic-Based Language Model (Hierarchical Prior 1)				
PUNC	PUNC	PUNC	PUNC	PUNC
*	the	**number**	the	the
and	**stimulus**	and	to	of
analog	i.e.	**synapses**	our	a
the	**receptive**	**synaptic**	of	to
number	in	in	*	*
architecture	and	is	**robot**	**number**
with	that	*	in	in
active	several	of	these	and
for	of	during	and	is
Topic-Based Language Model (Hierarchical Prior 2a)				
analog	**stimuli**	**synapses**	**robot**	PUNC
chip	**receptive**	**synaptic**	**modules**	the
hardware	**responses**	during	**hand**	of
digital	**stimulus**	**feedback**	**navigation**	to
implementation	**response**	**activity**	**path**	**number**
circuit	**multiple**	at	**obstacle**	a
circuits	due	**learning**	**environment**	is
technology	**cells**	**models**	**module**	in
real	**behavioural**	**nervous**	**internal**	and
applications	**field**	**associative**	**vlsi**	*

Table 3.5: Example topics inferred by latent Dirichlet allocation and the new topic-based language model with proper hyperpriors and 100 topics.

words, when using an improper hyperprior and optimised hyperparameters, words that occur in many different contexts (e.g., stop words) will have a high probability of occurring in all topics, causing the model to more closely resemble a unigram language model. These results suggest that when using latent Dirichlet allocation with improper hyperpriors, only β should be optimised; $\boldsymbol{n}$ should be fixed to the uniform distribution. It is also important to optimise $\alpha\boldsymbol{m}$—a nonuniform $\boldsymbol{m}$ is precisely what enables the model to infer a separate (more frequently used) stop word topic. This observation has implications for the work of Griffiths et al. (2005) on integrating syntax and topics. Their model, which handles stop words and other function words using a hidden Markov model and content words using latent Dirichlet allocation, was motivated by the fact that they found the topics inferred by latent Dirichlet allocation to be heavily dominated by stop words. In fact, if they had simply optimised $\alpha\boldsymbol{m}$, latent Dirichlet allocation would have automatically handled stop words as desired (by placing them in a separate topic) eliminating the need for a composite model.

3.6 Conclusions

In this chapter, I presented a new Bayesian model that integrates n-gram-based and topic-based approaches to document modelling, and compared several different model variants. The new model, especially with a topic-specific hierarchical Dirichlet prior ($\boldsymbol{n}_t$ per topic; tied via $\boldsymbol{u}$) has several benefits. Firstly, the information rate achieved by the new (bigram) model is much better than that achieved by even trigram hierarchical Dirichlet language model. Secondly, the topics inferred by the new model are clearly interpretable and contain words that are semantically related. Few topics contain stop words; instead, stop words are automatically grouped into a separate stop word topic, which is used more frequently than any of the other topics.

Finally, while investigating different variants of the new model and their effects on the inferred topics, I demonstrated that previous treatments of latent Dirichlet allocation, which either set the base measures of the Dirichlet priors over topics and words to be uniform distributions or optimise both of these base measures along with the concentration parameters, are inappropriate for data containing stop words. When modelling such data using latent Dirichlet allocation, it is important to (a) allow a nonuniform base measure in the Dirichlet prior over topic distributions and (b) use a uniform base measure in the Dirichlet prior over topic-specific word distributions. Together, these modelling choices prevent the topics from being dominated by stop words, and allow the model to automatically discover a separate stop word topic.

Chapter 4

Bayesian Models for Dependency Parsing Using Pitman-Yor Priors

In this chapter, I introduce a Bayesian dependency parsing model for natural language, based on the hierarchical Pitman-Yor process. This model arises from a Bayesian reinterpretation of a classic dependency parser (Eisner, 1996b). I show that parsing performance can be substantially improved by (a) using a hierarchical Pitman-Yor process as a prior over the distribution over dependents of a word, and (b) sampling model hyperparameters. I also present a second Bayesian dependency model in which latent state variables mediate the relationships between words and their dependents. The model clusters parent–child dependencies into states using a similar approach to that employed by Bayesian topic models when clustering words into topics. Each latent state may be viewed as a sort of specialised part-of-speech tag or "syntactic topic" that captures the relationships between words and their dependents. This is verified by inspection of the inferred states and by showing that they lead to improved parse accuracy when substituted for part-of-speech tags in the model.

4.1 Introduction

The previous chapter focused on language structure from the low-level perspective of word order. However, language also exhibits other, more complex, syntactic structures. Dependency graphs—which have seen recent successes in relation extraction (Culotta and Sorensen, 2004), hypernym discovery (Snow et al., 2004) and machine translation (Ding and Palmer, 2005)—are one way of representing this kind of higher-level structure. Dependency graphs encode relationships between words and their sentence-level, syntactic dependents by representing each sentence in a corpus as a directed graph with nodes consisting of the part-of-speech-tagged words in that sen-

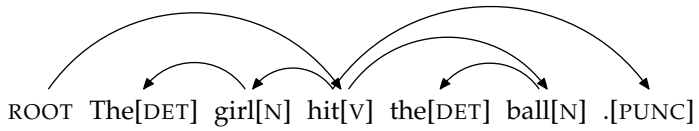


Figure 4.1: An example dependency graph for a tagged, cased sentence.

tence, with their letter case (capitalised or uncapitalised) left intact.[1] Figure 4.1 shows the dependency graph for the sentence, "The[DET] girl[N] hit[V] the[DET] ball[N] .[PUNC]". A directed edge from the word at position n to the word at position n' indicates that the word at position n' is a *dependent* of the word at position n. For example, consider the word "hit[V]" in figure 4.1. "hit[V]" has two immediate dependents—"girl[N]" on the left and "ball[N]" on the right. This is because the noun phrases "The[DET] girl[N]" and "the[DET] ball[N]" are both dependents of a hitting event. More precisely, they are the subject and object of the verb "hit[V]". Since "The[DET]" and "the[DET]" are dependents of "girl[N]" and "ball[N]", respectively, the *immediate* dependents of "hit[V]" are therefore "girl[N]" and "ball[N]" (Manning and Schütze, 2000). Throughout the rest of this chapter, dependent words will be referred to as *children*, while the words upon which they depend will be referred to as *parents*.

These graph structures give rise to an important difference between dependency modelling and n-gram language modelling. In n-gram language modelling, all relevant information is observed: Word identities and word order are known; only model parameters must be inferred in order to compute the probability of a newly observed sentence. In contrast, computing the probability of a new sentence under a generative dependency model also requires inference of the latent structure of the dependency graph for that sentence—i.e., the identity of the parent of each word in the sentence.

Despite this difference, generative models of dependency graphs and n-gram language models share the following property: Both rely on decomposing the probability of words in a sentence into a product of probabilities of individual words given some word-based context. In the case of n-gram language modelling, this context is some number of words immediately preceding the word of interest, while in dependency modelling, this context is the word's parent and sometimes its siblings. Thus, while the actual contexts used by the two types of model are different, the underlying idea—that the contexts that consist of some number of nearby words—is the same.

The work in this chapter exploits this connection between generative dependency models and n-gram language models to expand the reach of Bayesian methods to dependency parsing. I introduce a new dependency parsing model based on the Pitman-Yor process (section 4.4). The resultant model exhibits higher parsing accuracy than previous generative dependency parsing models. Furthermore, the use of a generative framework allows for the incorporation of additional latent variables. In section 4.5, I

[1] Words with intact letter casing will henceforth be referred to as "cased" words.

present a model capable of inferring "syntactic topics" from dependency graphs.

4.2 Generative Dependency Modelling

As described in the previous section, dependency models represent syntactic modification relationships between words in a sentence using graphs. More formally, a dependency graph for a tagged, cased sentence is a directed graph, with nodes corresponding to words and a single unique ROOT node, artificially inserted at the beginning of the sentence (as shown in figure 4.1). In addition to this, most dependency representations, including the one used throughout this chapter, assume that each dependency graph must also satisfy the following properties (McDonald, 2006):

- The graph is weakly connected—that is, every node is reachable from every other node, if the direction of the edges is ignored.
- Each node has a single parent (i.e., incoming edge), except for ROOT.
- The graph is acyclic.
- The graph must contain exactly $N - 1$ edges, where N is the number of tagged, cased words in the sentence (including ROOT).

Any graph that satisfies the above four properties must be a tree. Therefore, the dependency graphs in this chapter will be referred to henceforth as *dependency trees*.

Another common restriction is that of projectivity. For a dependency tree to be *projective*, an edge from w_n to $w_{n'}$ can only exist if there is also a directed path from w_n to every word between w_n and $w_{n'}$ in the sentence. Equivalently, a projective dependency tree is one in which all edges are non-crossing—that is, if the words in the corresponding sentence are written in the order in which they occur and all edges between them are drawn above the words, then it is possible to draw the edges such that no edge crosses any other edge. If this is not possible, the tree is not projective. The following sentence has a non-projective dependency tree: "I bought a computer yesterday which was ThinkPad." Since non-projective dependency trees are not common in written English, and the largest corpus of English dependency trees is automatically constructed from the Penn Treebank so as to be projective (Yamada and Matsumoto, 2003), this chapter deals with projective dependency trees only. This representation of dependencies is isomorphic to a restricted form of phrase-structured grammar.

4.3 Previous Work

In this section I review the generative dependency model of Eisner (1996a,b) and recent research on Bayesian n-gram language modelling. I also briefly discuss other

recent work on applying Bayesian techniques to parsing, and highlight the main differences between the models presented in this chapter and these approaches.

4.3.1 Eisner's Dependency Model

The best-known generative modelling framework for dependency trees is that of Eisner (1996a,b). This model generates a tagged, cased sentence and its corresponding dependency graph using a parent-outward process. Beginning with the ROOT node, each parent generates a sequence of children starting in the centre and moving outward to the left and then similarly to the right. Generation of each child is conditioned upon the identity of the tagged, cased parent, the direction of the child in relation to the parent (left or right) and the most recently generated sibling child. That is, conditioned on the parent, the sequence of children in each direction is a first order Markov chain. The final child on each side of the parent is always a special STOP symbol, indicating that no more children should be generated in that direction. It is these STOP symbols at each level that give rise to the simultaneous generation of words and trees.

For example, to generate the sentence and tree in figure 4.1, the first tagged, cased word to be generated is "hit[V]", as the first child of ROOT (other than ROOT's left STOP). Having done this, "hit[V]" is now considered as a parent, and the subtree rooted at "hit[V]" is generated: First, "girl[N]" is generated as a left child. The process is then recursively repeated at the subtree rooted at "girl[N]", generating "The[DET]" (and its left and right STOPs) to the left, then a left STOP, and then a right STOP. Once the subtree rooted at "girl[N]" has been generated, "hit[V]" generates a left STOP, indicating that generation of left children is now complete, and begins generating right children. The process terminates after the generation of ROOT's right STOP.

The probability of a sentence consisting of words $\boldsymbol{w}$, with corresponding case values $\boldsymbol{c}$, part-of-speech tags $\boldsymbol{s}$ and tree $\boldsymbol{t}$, generated according to this process, is

$$
\begin{aligned}
P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \boldsymbol{t}) = \\
&\prod_n P(s_n, w_n, c_n \mid s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, s_{\sigma(n)}, d_n) \\
&\prod_n P(\text{STOP} \mid s_n, w_n, c_n, s_{y(n)}, d_{\text{STOP}} = \leftarrow)\, P(\text{STOP} \mid s_n, w_n, c_n, s_{y(n)}, d_{\text{STOP}} = \rightarrow)
\end{aligned} \tag{4.1}
$$

where $d_n \in \{\leftarrow, \rightarrow\}$ is the direction of w_n with respect to its parent, $\pi(n)$ is the position of w_n's parent, $\sigma(n)$ is the position of w_n's immediately preceding sibling (moving outward from w_n's parent in direction d_n), $y(n)$ is the position of w_n's final child, and d_{STOP} indicates whether the corresponding STOP is a left ($\leftarrow$) or right ($\rightarrow$) STOP. The case c_n of each word w_n can be one of four values: Lower case, upper case, mixed capitalisation, or first capitalised word in the sentence. Eisner further decomposes the

$P(s_n \mid s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, s_{\sigma(n)}, d_n)$	$P(w_n \mid s_n, s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, d_n)$	$P(c_n \mid s_n, w_n)$
$s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, s_{\sigma(n)}, d_n$	$s_n, s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, d_n$	s_n, w_n
$s_{\pi(n)}, s_{\sigma(n)}, d_n$	$s_n, s_{\pi(n)}, d_n$	$s_n,$
$s_{\pi(n)}, d_n$	$s_n,$	

Table 4.1: Contexts (in order) used by Eisner for estimating probabilities.

probability $P(s_n, w_n, c_n \mid s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, s_{\sigma(n)}, d_n)$ as follows:

$$\begin{aligned} &P(s_n, w_n, c_n \mid s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, s_{\sigma(n)}, d_n) = \\ &\quad P(s_n \mid s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, s_{\sigma(n)}, d_n) \\ &\quad P(w_n \mid s_n, s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, d_n) \\ &\quad P(c_n \mid s_n, w_n). \end{aligned} \tag{4.2}$$

To compute each of the probabilities in equation 4.2 from training data $\mathcal{D}$ (tagged, cased sentences and their corresponding dependency trees), Eisner constructs estimators of the probability of each variable of interest, s_n, w_n and c_n, in contexts of varying length. Each context is a reduction of the full conditioning context for the probability to be estimated. The complete set of context reductions for each variable is shown in table 4.1. For each context, Eisner estimates the probability of the variable of interest in that context using the ratio of conditional and marginal counts. For example, Eisner estimates the probability of w_n in context $s_{\sigma(n)}d_n$ by computing $f_{w_n|s_{\sigma(n)}d_n} = N_{w_n|s_{\sigma(n)}d_n}/N_{\cdot|s_{\sigma(n)}d_n}$ where $N_{w_n|s_{\sigma(n)}d_n}$ is the number of times w_n has occurred in the context of $s_{\sigma(n)}d_n$ in the training data and $N_{\cdot|s_{\sigma(n)}d_n} = \sum_w N_{w|s_{\sigma(n)}d_n}$.

Having constructed estimators for the probability of each variable of interest (s_n, w_n and c_n) in each of the contexts given in table 4.1, Eisner then computes the probability of each variable by interpolating between the relevant estimators. For instance, the probability $P(s_n \mid s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, s_{\sigma(n)}, d_n)$ is computed as follows:

$$\begin{aligned} &P(s_n \mid s_{\pi(n)}, w_{\pi(n)}, c_{\pi(n)}, s_{\sigma(n)}, d_n) = \\ &\quad \lambda_2\, f_{s_n|s_{\pi(n)}w_{\pi(n)}c_{\pi(n)}s_{\sigma(n)}d_n} + \\ &\quad (1-\lambda_2)\,\lambda_1\, f_{s_n|s_{\pi(n)}s_{\sigma(n)}d_n} + \\ &\quad (1-\lambda_2)\,(1-\lambda_1)\,\lambda_0\, f_{s_n|s_{\pi(n)}d_n} + \\ &\quad (1-\lambda_2)\,(1-\lambda_1)\,(1-\lambda_0)\,\frac{1}{S} \end{aligned} \tag{4.3}$$

where

$$\lambda_2 = \frac{N_{\cdot|s_{\pi(n)}w_{\pi(n)}c_{\pi(n)}s_{\sigma(n)}d_n}}{N_{\cdot|s_{\pi(n)}w_{\pi(n)}c_{\pi(n)}s_{\sigma(n)}d_n} + 3} \tag{4.4}$$

$$\lambda_1 = \frac{N_{\cdot|s_{\pi(n)}s_{\sigma(n)}d_n}}{N_{\cdot|s_{\pi(n)}s_{\sigma(n)}d_n} + 3} \tag{4.5}$$

$$\lambda_0 = \frac{N_{\cdot|s_{\pi(n)}d_n}}{N_{\cdot|s_{\pi(n)}d_n} + 0.5} \tag{4.6}$$

and S is the number of possible part-of-speech tags. This approach is similar to that used in non-Bayesian n-gram language modelling (see section 3.1): Estimators for more specific contexts are smoothed by estimators for less specific contexts. The choice of context reductions is different, however. In language modelling, the choice of contexts is obvious: For instance, when estimating the trigram probability of a word w_n the relevant contexts are $w_{n-1}w_{n-2}$ and w_{n-1}—that is, the nearer a word is to w_n the more important it is considered to be when reducing context. In the case of dependency modelling, the choice is not clear and must be decided by the modeller.

4.3.2 Bayesian n-gram Language Models

Bayesian n-gram language modelling was first explored by MacKay and Peto (1995), who drew connections between non-Bayesian interpolated language models and hierarchical Dirichlet priors, as described in section 3.2. More recently, Teh (2006) and Goldwater et al. (2006) demonstrated that Kneser-Ney smoothing (Kneser and Ney, 1995) can be viewed as an approximate inference scheme in a hierarchical Pitman-Yor process, thereby reinterpreting one of the most successful non-Bayesian n-gram language modelling techniques as a hierarchical Bayesian model. In this section, I review the hierarchical Pitman-Yor process and its application to n-gram language modelling.

The Dirichlet distribution, defined in equation 2.2, is the finite version of the Dirichlet process (Ferguson, 1973): The Dirichlet distribution is a prior over finite discrete distributions—i.e., distributions over a set of finite elements—while the Dirichlet process is a prior over infinite continuous probability distributions. (Despite this, draws from a Dirichlet process are discrete with probability one.) Like the Dirichlet distribution, the Dirichlet process may be used hierarchically—the base measure may itself be given a Dirichlet process prior (Teh et al., 2006). In the context of language modelling, the Dirichlet distribution is an appropriate choice of prior for language models with a fixed vocabulary, while the Dirichlet process can be used to create a hierarchical Bayesian language model with a potentially infinite vocabulary (Cowans, 2006).

The Dirichlet distribution and the Dirichlet process are both special cases of the Pitman-Yor process (Pitman and Yor, 1997). Unlike the Dirichlet distribution and process, no distinction is made between the finite and infinite versions of the Pitman-Yor

process in terms of nomenclature. The discussion in the remainder of this section assumes a finite Pitman-Yor process, though the ideas are directly applicable to the infinite case and merely require a change of top-level base measure. Like the Dirichlet distribution, the Pitman-Yor process has a concentration parameter α and a base measure $\boldsymbol{m}$. However, it also has an extra "discount" parameter $0 \leq \epsilon < 1$ which, like α, controls variability around the base measure. When this discount parameter is set to zero, the (finite) Pitman-Yor process reduces to a Dirichlet distribution.

As explained in section 3.2, n-gram language models are specified by conditional distributions $P(w_t = w \mid w_{t-1}w_{t-2}\ldots w_{t-(n-1)} = \boldsymbol{h})$, described by $W^{n-1}(W-1)$ free parameters, where W is the size of the vocabulary. These parameters are typically denoted by the matrix Φ, in which each row $\boldsymbol{\phi_h}$ is the distribution over words given context $\boldsymbol{h}$. The probability of a corpus $\boldsymbol{w}$ given parameters Φ is therefore written as

$$P(\boldsymbol{w} \mid \Phi) = \prod_t P(w_t \mid w_{t-1}, \ldots, w_{t-(n-1)}, \Phi) \tag{4.7}$$

$$= \prod_w \prod_{\boldsymbol{h}} \phi_{w|\boldsymbol{h}}^{N_{w|\boldsymbol{h}}}, \tag{4.8}$$

where the quantity $N_{w|\boldsymbol{h}}$ is the number of times that word w immediately follows $\boldsymbol{h}$ in $\boldsymbol{w}$. Rather than placing a Dirichlet prior over each probability vector $\boldsymbol{\phi_h}$, as in MacKay and Peto's hierarchical Dirichlet language model (1995), Teh (2006) and Goldwater et al. (2006) recommend giving each $\boldsymbol{\phi_h}$ a Pitman-Yor process prior:

$$P(\Phi \mid \alpha_{n-1}, \boldsymbol{m}_{\rho(\boldsymbol{h})}, \epsilon_{n-1}) = \prod_{\boldsymbol{h}} \mathrm{PY}\left(\boldsymbol{\phi_h} \mid \alpha_{n-1}, \boldsymbol{m}_{\rho(\boldsymbol{h})}, \epsilon_{n-1}\right), \tag{4.9}$$

where $\rho(\boldsymbol{h})$ is the reduction of $\boldsymbol{h}$ to a sequence of $n-2$ words (obtained by dropping the left-most word) and $\mathrm{PY}\left(\cdot \mid \alpha_{n-1}, \boldsymbol{m}_{\rho(\boldsymbol{h})}, \epsilon_{n-1}\right)$ is a finite Pitman-Yor prior with parameters α_{n-1}, $\boldsymbol{m}_{\rho(\boldsymbol{h})}$ and ϵ_{n-1}. The base measure $\boldsymbol{m}_{\rho(\boldsymbol{h})}$ is shared between contexts $\boldsymbol{h}'$ with reduction $\rho(\boldsymbol{h})$, while the other parameters α_{n-1} and ϵ_{n-1} are shared between contexts $\boldsymbol{h}'$ with length $n-1$. Although there is no known analytic form for $\mathrm{PY}\left(\cdot \mid \alpha_{n-1}, \boldsymbol{m}_{\rho(\boldsymbol{h})}, \epsilon_{n-1}\right)$, when used as a prior over discrete distributions (Teh, 2006), the resultant predictive distributions (obtained by integrating over Φ) are tractable.

As with the Dirichlet (section 3.4.2), the consequences of using the prior in equation 4.9 are best described in terms of the effects on the generative process and predictive distributions over words for each context $\boldsymbol{h}$ (of length $n-1$) with $\boldsymbol{\phi_h}$ integrated out. The generative process may be described exactly as in section 3.4.2 with one key difference: The new observation is instantiated to the value of "internal" draw γ_i from the base measure $\boldsymbol{m}_{\rho(\boldsymbol{h})}$ with probability proportional to the number of previous observations matched to γ_i *minus some discount* ϵ_{n-1}. This is the only difference between the Pitman-Yor process and the Dirichlet distribution (or process). The predictive probability of

word w in context $\boldsymbol{h}$ under a Pitman-Yor prior over $\boldsymbol{\phi}_{\boldsymbol{h}}$ is therefore

$$P(w \mid \boldsymbol{h}, \alpha_{n-1}, \boldsymbol{m}_{\rho(\boldsymbol{h})}, \epsilon_{n-1}) = \frac{\sum_{i=1}^{I_{\boldsymbol{h}}} (N_{\cdot|\boldsymbol{h}}^{(i)} - \epsilon_{n-1})\, \delta(\gamma_i - w) + (\alpha_{n-1} + \epsilon_{n-1} I_{\boldsymbol{h}})\, m_{w|\rho(\boldsymbol{h})}}{\sum_{i=1}^{I_{\boldsymbol{h}}} N_{\cdot|\boldsymbol{h}}^{(i)} + \alpha_{n-1}}, \tag{4.10}$$

where $I_{\boldsymbol{h}}$ is the current number of internal draws from the base measure $\boldsymbol{m}_{\rho(\boldsymbol{h})}$, and $N_{\cdot|\boldsymbol{h}}^{(i)}$ is the number of observations matched to internal draw γ_i. Since every observation is matched to a draw from the base measure, $\sum_{i=1}^{I_{\boldsymbol{h}}} N_{\cdot|\boldsymbol{h}}^{(i)}\, \delta(\gamma_i - w)$ is equal to $N_{w|\boldsymbol{h}}$—the number of times word w has been seen in context $\boldsymbol{h}$—and $\sum_{i=1}^{I_{\boldsymbol{h}}} N_{\cdot|\boldsymbol{h}}^{(i)} = N_{\cdot|\boldsymbol{h}}$.

The Pitman-Yor process may be used hierarchically. Consequently, $\boldsymbol{m}_{\rho(\boldsymbol{h})}$ can also be given a Pitman-Yor prior, with parameters α_{n-2}, $\boldsymbol{m}_{\rho(\rho(\boldsymbol{h}))}$ and ϵ_{n-2}, and integrated out. Continuing in this fashion, the base measures for all context reductions may be given Pitman-Yor priors and integrated out, leaving only $\boldsymbol{u}$, the base measure for the empty context $\emptyset$ —the uniform distribution over all words in the vocabulary:

$$P(w \mid \boldsymbol{h}, \alpha_{n-1}, \boldsymbol{m}_{\rho(\boldsymbol{h})}, \epsilon_{n-1}) = \frac{\sum_{i=1}^{I_{\boldsymbol{h}}} (N_{\cdot|\boldsymbol{h}}^{(i)} - \epsilon_{n-1})\, \delta(\gamma_i - w) + (\alpha_{n-1} + \epsilon_{n-1} I_{\boldsymbol{h}}) \dfrac{\cdots + (\alpha_0 + \epsilon_0 I_\emptyset)\, u_w}{\cdots}}{\sum_{i=1}^{I_{\boldsymbol{h}}} N_{\cdot|\boldsymbol{h}}^{(i)} + \alpha_{n-1}}. \tag{4.11}$$

There is now a Pitman-Yor process for each context and reductions, arranged in a hierarchical fashion. The resultant generative process is identical to that of the hierarchical Dirichlet (described in section 3.4.2), except for the inclusion of discount parameters.

Given real-world data $\boldsymbol{w}$, the number of internal draws for each Pitman-Yor process and the paths from the observations to the top-level base measure $\boldsymbol{u}$ are unknown and must be inferred. As with the hierarchical Dirichlet, this can be done either using Gibbs sampling or one of two standard approximate inference schemes—the maximal and minimal path assumptions. These schemes are described in detail in section 3.4.2.

Teh (2006) and Goldwater et al. (2006) showed that using a hierarchical Pitman-Yor process prior for n-gram language modelling leads to a model of which Kneser-Ney smoothing (Kneser and Ney, 1995) is a special case: Kneser-Ney smoothing corresponds to setting all α parameters to zero and using the minimal path assumption.

4.3.3 Bayesian Parsing Models

Recently two other Bayesian approaches to parsing have been proposed: Firstly, Johnson et al. (2007b) presented two Markov chain Monte Carlo algorithms for probabilistic context-free grammars. They used these algorithms, in conjunction with nonhierarchical Dirichlet priors, to demonstrate that Bayesian techniques are capable of generat-

ing reasonable morphological analyses. Since probabilistic context-free grammars are unlexicalised and therefore do not suffer from severe sparsity problems, Johnson et al. did not need to use hierarchical Dirichlet priors. They also used fixed concentration parameters. In contrast, the lexicalised nature of dependency models means that hierarchical priors are necessary for achieving good modelling performance. The model presented in the next section therefore uses hierarchical Pitman-Yor priors. Additionally, the model hyperparameters are inferred from training data using slice sampling.

The second application of Bayesian techniques to parsing is that of Johnson et al. (2007a), who presented a framework for combining Pitman-Yor priors and probabilistic context-free grammars. While they did not provide any empirical results applying this framework to syntax, they did show that the framework subsumes their earlier (experimental) work on morphology using Pitman-Yor priors (Goldwater et al., 2006).

4.4 A Hierarchical Pitman-Yor Dependency Model

In this section, I introduce a new Bayesian framework for generative dependency modelling that draws on the similarities between generative dependency models and n-gram language models described in the previous section. The framework uses the same generative process and decomposition of $P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \boldsymbol{t})$ as Eisner's dependency model (Eisner, 1996a,b), combined with a hierarchical Pitman-Yor process prior over each probability, eliminating the estimator-based approach described in section 4.3.1.

4.4.1 Reinterpreting Eisner's Dependency Model

The new model is best explained by starting with a reinterpretation of Eisner's model from a Bayesian perspective. In Eisner's model, the probability of a sentence $\boldsymbol{w}$ with corresponding case values $\boldsymbol{c}$, part-of-speech tags $\boldsymbol{s}$ and tree $\boldsymbol{t}$ may be written as

$$\begin{aligned} P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \boldsymbol{t}) = \\ \prod_n \theta_{s_n | s_{\pi(n)} w_{\pi(n)} c_{\pi(n)} s_{\sigma(n)} d_n} \, \phi_{w_n | s_n, s_{\pi(n)} w_{\pi(n)} c_{\pi(n)} d_n} \, \psi_{c_n | s_n w_n} \\ \prod_n \theta_{\text{STOP} | s_n w_n c_n s_{y(n)} d_{\text{STOP}} = \leftarrow} \, \theta_{\text{STOP} | s_n w_n c_n s_{y(n)} d_{\text{STOP}} = \rightarrow}, \end{aligned} \tag{4.12}$$

where d_n is the direction of token w_n with respect to its parent ($\leftarrow$ or $\rightarrow$), $\pi(n)$ is the position of w_n's parent, $\sigma(n)$ is the position of w_n's immediate sibling (moving outward from w_n's parent in direction d_n), $y(n)$ is the position of w_n's final child, and d_{STOP} indicates whether the corresponding STOP is a left ($\leftarrow$) or right ($\rightarrow$) STOP. The probability vector $\boldsymbol{\theta}_{s' w' c' s'' d}$ is the distribution over part-of-speech tags (and STOP) for the context consisting of parent tag s', parent word w', parent case value c', sibling tag s'', and direction d. Similarly, the vector $\boldsymbol{\phi}_{s s' w' c' d}$ is the distribution over words for

the context defined by part-of-speech tag s, parent tag s', parent word w', parent case value c', and direction d. Finally, the probability vector $\boldsymbol{\psi}_{sw}$ represents the distribution over case values for the context consisting of part-of-speech tag s and word w.

Given a set of training data $\mathcal{D}$ consisting of tagged, cased sentences and their corresponding trees, Eisner computes each of the probabilities in equation 4.12 using estimators, as described in section 4.3.1. In a Bayesian setting, however, each probability vector $\boldsymbol{\theta}_{s'w'c's''d}$, $\boldsymbol{\phi}_{ss'w'c'd}$ and $\boldsymbol{\psi}_{sw}$ should instead be given a prior and integrated out to form the true predictive distribution. One appropriate choice of prior is the hierarchical Dirichlet distribution. Using the same context reductions as Eisner,

$$\begin{gathered} s_{\pi(n)} = s', w_{\pi(n)} = w', c_{\pi(n)} = c', s_{\sigma(n)} = s'', d_n = d \\ \Downarrow \\ s_{\pi(n)} = s', s_{\sigma(n)} = s'', d_n = d \\ \Downarrow \\ s_{\pi(n)} = s', d_n = d, \end{gathered} \tag{4.13}$$

where $\pi(n)$ is the position of w_n's parent, d_n is the direction of w_n with respect to its parent, and $\sigma(n)$ is the position of w_n's immediately preceding sibling moving outward from $w_{\pi(n)}$ in direction d_n, the prior over $\boldsymbol{\theta}_{s'w'c's''d}$ can be defined as

$$P(\boldsymbol{\theta}_{s'w'c's''d} \,|\, \alpha_2, \boldsymbol{m}_{s's''d}) = \text{Dir}\,(\boldsymbol{\theta}_{s'w'c's''d} \,|\, \alpha_2, \boldsymbol{m}_{s's''d}) \tag{4.14}$$

$$P(\boldsymbol{m}_{s's''d} \,|\, \alpha_1, \boldsymbol{m}_{s'd}) = \text{Dir}\,(\boldsymbol{m}_{s's''d} \,|\, \alpha_1, \boldsymbol{m}_{s'd}) \tag{4.15}$$

$$P(\boldsymbol{m}_{s'd} \,|\, \alpha_0, \boldsymbol{u}) = \text{Dir}\,(\boldsymbol{m}_{s'd} \,|\, \alpha_0, \boldsymbol{u}). \tag{4.16}$$

Under this prior, the predictive probability of part-of-speech tag s given α_0, α_1, α_2 and data $\mathcal{D}$ (with $\boldsymbol{\theta}_{s'w'c's''d}$ and the base measures integrated out) is given by

$$\begin{aligned} &P(s \,|\, s', w', c', s'', d, \mathcal{D}, \alpha_0, \alpha_1, \alpha_2) = \\ &\qquad \frac{\sum_i N^{(i)}_{\cdot|s'w'c's''d}\, \delta\,(\gamma_i - s) + \alpha_2 \,\dfrac{\cdots + \alpha_0 u_s}{\cdots}}{\sum_i N^{(i)}_{\cdot|s'w'c's''d} + \alpha_2}. \end{aligned} \tag{4.17}$$

The predictive probabilities for w and and c may be obtained similarly, also using hierarchical Dirichlet priors and the context reductions and orders shown in table 4.1.

As with the language model described in section 4.3.2, number of internal draws for each level in the hierarchy and the paths from the observations to the top-level base measure $\boldsymbol{u}$—equivalently, the counts to be used in all but the bottom-most level of each predictive distribution—are unknown for real data. They must therefore be inferred using either Gibbs sampling or one of the maximal and minimal path assumptions.

In the case where the maximal path assumption is used and $\alpha_2 = \alpha_1 = 3$ and $\alpha_0 = 0.5$, this Bayesian model is equivalent to Eisner's model for dependency trees: Under the

maximal path assumption, the counts in equation 4.17 correspond to the raw observation counts—e.g., $\sum_i N^{(i)}_{\cdot|s'w'c's''d}\,\delta(\gamma_i - s)$ is equal to $N_{s|s'w'c's''d}$, the number of times part-of-speech tag s has been seen in the context of parent tag s', parent word w', parent case value c', sibling tag s'' and direction d. Consequently, using the maximal path assumption and $\alpha_2 = \alpha_1 = 3$, $\alpha_0 = 0.5$, equation 4.17 may be written as follows:

$$P(s \mid s', w', c', s'', d, \mathcal{D}, \alpha_0, \alpha_1, \alpha_2) = \frac{N_{s|s'w'c's''d} + 3\,\dfrac{N_{s|s's''d} + 3\,\dfrac{N_{s|s'd} + 0.5\,\frac{1}{S}}{N_{\cdot|s'd} + 0.5}}{N_{\cdot|s's''d} + 3}}{N_{\cdot|s'w'c's''d} + 3}. \tag{4.18}$$

To make the relationship to Eisner's model explicit, this equation may be rewritten as

$$\begin{aligned} P(s \mid s', w', c', s'', d, \mathcal{D}, \alpha_0, \alpha_1, \alpha_2) = {} & \\ & \lambda_2\,\frac{N_{s|s'w'c's''d}}{N_{\cdot|s'w'c's''d}} + \\ & (1-\lambda_2)\,\lambda_1\,\frac{N_{s|s's''d}}{N_{\cdot|s's''d}} + \\ & (1-\lambda_2)\,(1-\lambda_1)\,\lambda_0\,\frac{N_{s|s'd}}{N_{\cdot|s'd}} + \\ & (1-\lambda_2)\,(1-\lambda_1)\,(1-\lambda_0)\,\frac{1}{S} \end{aligned} \tag{4.19}$$

where the quantities λ_2, λ_1 and λ_0 are given by equations 4.4, 4.5 and 4.6 respectively. Equation 4.19 and equation 4.18 are therefore identical to the predictive distribution over part-of-speech tags used in Eisner's dependency model, given in equation 4.3.

This Bayesian reinterpretation of Eisner's model has three advantages: firstly, the concentration parameters may be sampled, rather than fixed to some particular value, as is the case in Eisner's model. Secondly, the counts need not correspond to the raw observation counts, as is the case when using the maximal path assumption; the minimal path assumption and Gibbs sampling both give rise to other count values. Finally, it is also possible to use priors other than the hierarchical Dirichlet distribution.

4.4.2 Using Pitman-Yor Process Priors

The lexicalised nature of dependency trees means that generative dependency parsing models suffer from the same kinds of data sparsity as n-gram language models. Given the successes of Kneser-Ney smoothing and hierarchical Pitman-Yor process priors for language modelling, it is likely that the hierarchical Pitman-Yor process is a better choice of prior for dependency modelling than the hierarchical Dirichlet distribution. Indeed, the results presented in section 4.4.4 demonstrate that this is in fact the case.

Under a hierarchical Pitman-Yor prior over $\boldsymbol{\theta}_{s'w'c's''d}$, given by

$$P(\boldsymbol{\theta}_{s'w'c's''d} \,|\, \alpha_2, \boldsymbol{m}_{s's''d}, \epsilon_2) = \text{PY}\left(\boldsymbol{\theta}_{s'w'c's''d} \,|\, \alpha_2, \boldsymbol{m}_{s's''d}, \epsilon_2\right) \tag{4.20}$$

$$P(\boldsymbol{m}_{s's''d} \,|\, \alpha_1, \boldsymbol{m}_{s'd}, \epsilon_1) = \text{PY}\left(\boldsymbol{m}_{s's''d} \,|\, \alpha_1, \boldsymbol{m}_{s'd}, \epsilon_1\right) \tag{4.21}$$

$$P(\boldsymbol{m}_{s'd} \,|\, \alpha_0, \boldsymbol{u}, \epsilon_0) = \text{PY}\left(\boldsymbol{m}_{s'd} \,|\, \alpha_0, \boldsymbol{u}, \epsilon_0\right), \tag{4.22}$$

the predictive probability of part-of-speech tag s given data $\mathcal{D}$ is

$$P(s \,|\, s', w', c', s'', d, \mathcal{D}, \alpha_0, \alpha_1, \alpha_2, \epsilon_0, \epsilon_1, \epsilon_2) =$$

$$\frac{\sum_i \left(N^{(i)}_{\cdot|s'w'c's''d} - \epsilon_2\right) \delta\left(\gamma_i - s\right) + \left(\alpha_2 + \epsilon_2 I_{s'w'c's''d}\right) \dfrac{\cdots + \left(\alpha_0 + \epsilon_0 I_{\emptyset}\right) u_s}{\cdots}}{\sum_i N^{(i)}_{\cdot|s'w'c's''d} + \alpha_2}. \tag{4.23}$$

The predictive probabilities for word w and case value c may be obtained similarly. As with $\boldsymbol{\theta}_{s'w'c's''d}$, when defining hierarchical Pitman-Yor priors over $\boldsymbol{\phi}_{ss'w'c'd}$ and $\boldsymbol{\psi}_{sw'}$ the context reductions recommended by Eisner (shown in table 4.1) are used.

The evidence—or probability of data $\mathcal{D}$ given concentration and discount parameters, inferred internal draws and inferred paths from the observations to the top-level base measure via these draws—may be computed using the predictive distributions over tags, words and case values: All counts are zeroed and, starting with the root of each dependency tree, each node (tagged, cased word) is visited in the parent-outward fashion described in section 4.3.1 until all nodes (including STOP nodes) have been processed. As each node is visited, the probability of that tagged, cased word is computed using the predictive distributions given the data seen so far and multiplied into the estimate of the evidence. The node may then be added back into the hierarchy of Pitman-Yor processes according to the current set of inferred internal draws and paths (i.e., the counts are updated to reflect that node) before processing the next node.

4.4.3 Inference

Given the hierarchical Pitman-Yor dependency model introduced in the previous section and a training corpus $\mathcal{D}$, consisting of tagged, cased sentences and their trees, there are two tasks of interest: sampling model hyperparameters given the training data, and inferring trees for unseen test sentences. In this section, I describe how these tasks may be accomplished. For computational efficiency, inference of internal draws and paths is performed using either the maximal or minimal path assumption.

Having inferred a set of internal draws and paths for the training data $\mathcal{D}$, typical concentration and discount parameters can be determined using slice sampling (Neal, 2003). As described in section 3.4.2, slice sampling is a Markov chain Monte Carlo method that adapts to the distribution from which samples are being drawn by uniformly sampling from the area under its density function. When sampling concentra-

tion and discount parameters, the density used is the evidence or probability of the data, which may be computed as described in the previous section. Pseudocode for drawing S multidimensional samples using slice sampling is given in algorithm 3.2.

Although dependency trees have non-trivial structures, the parents for all words in a given sentence can be jointly sampled using an algorithm that combines dynamic programming with the Metropolis-Hastings method. The algorithm is similar to that of Johnson et al. (2007b,a) for unlexicalised probabilistic context-free grammars.

The dynamic program is responsible for proposing a new tree $\boldsymbol{t}'$ for a cased sentence $\boldsymbol{w}$ (with corresponding part-of-speech tags $\boldsymbol{s}$ and previously sampled tree $\boldsymbol{t}$) given all the other trees and sentences in the corpus $\mathcal{D}$. The proposal tree $\boldsymbol{t}'$ is sampled from

$$\begin{aligned} P(\boldsymbol{t}' \mid \boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \mathcal{D}_{\setminus \boldsymbol{s},\boldsymbol{w},\boldsymbol{c},\boldsymbol{t}}, U) \simeq \\ P(\boldsymbol{t}' \mid \boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \mathcal{D}_{\setminus \boldsymbol{s},\boldsymbol{w},\boldsymbol{c},\boldsymbol{t}}, \{\tilde{\boldsymbol{\theta}}_{s'w'c's''d}, \tilde{\boldsymbol{\phi}}_{ss'w'c'd}, \tilde{\boldsymbol{\psi}}_{sw}\}, U) \end{aligned} \tag{4.24}$$

where

$$\begin{aligned} P(\boldsymbol{t}' \mid \boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \mathcal{D}_{\setminus \boldsymbol{s},\boldsymbol{w},\boldsymbol{c},\boldsymbol{t}}, \{\tilde{\boldsymbol{\theta}}_{s'w'c's''d}, \tilde{\boldsymbol{\phi}}_{ss'w'c'd}, \tilde{\boldsymbol{\psi}}_{sw}\}, U) \propto \\ P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \boldsymbol{t}' \mid \mathcal{D}_{\setminus \boldsymbol{s},\boldsymbol{w},\boldsymbol{c},\boldsymbol{t}}, \{\tilde{\boldsymbol{\theta}}_{s'w'c's''d}, \tilde{\boldsymbol{\phi}}_{ss'w'c'd}, \tilde{\boldsymbol{\psi}}_{sw}\}, U), \end{aligned} \tag{4.25}$$

U denotes the concentration and discount parameters and $\mathcal{D}_{\setminus \boldsymbol{s},\boldsymbol{w},\boldsymbol{c},\boldsymbol{t}}$ is the corpus $\mathcal{D}$ excluding the tagged, cased sentence of interest and its previously sampled tree. (When sampling a tree for an unseen test sentence, the corpus $\mathcal{D}$ is considered to be the training data plus all other trees in the test data.) The probability vectors $\tilde{\boldsymbol{\theta}}_{s'w'c's''d}$, $\tilde{\boldsymbol{\phi}}_{ss'w'c'd}$ and $\tilde{\boldsymbol{\psi}}_{sw}$ are the predictive distributions over tags, words and case values given $\mathcal{D}_{\setminus \boldsymbol{s},\boldsymbol{w},\boldsymbol{c},\boldsymbol{t}}$ and the currently inferred internal draws and paths. Conditioned on these probability vectors, each node is independent of the other nodes in the tree. $P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \boldsymbol{t}' \mid \mathcal{D}_{\setminus \boldsymbol{s},\boldsymbol{w},\boldsymbol{c},\boldsymbol{t}}, \{\tilde{\boldsymbol{\theta}}_{s'w'c's''d}, \tilde{\boldsymbol{\phi}}_{ss'w'c'd}, \tilde{\boldsymbol{\psi}}_{sw}\}, U)$ may therefore be computed by taking the product of the probabilities of each tagged, cased word in the sentence under these predictive distributions *without* updating the counts used to construct them. This independence is necessary to derive an efficient dynamic program.

The dynamic program is based on Eisner's $O(n^3)$ algorithm for parsing—that is, for choosing the most probable tree for a given sentence—adapted to perform sampling. The algorithm is analogous to the forward–"sample-backward" algorithm for hidden Markov models: First a bottom-up dynamic programming (forward) pass is performed to marginalise over all possible subtrees for the sentence in question. Sampling is then performed in a top-down (backward) fashion. A four-dimensional dynamic programming chart is used to store the sums over subtrees. Each chart entry $C[a][b][c][d]$ contains the sum of the probabilities of all possible subtrees spanning positions a through $b > a$ with "complete value" $c \in \{0, 1, 2\}$ and direction $d \in \{\leftarrow, \rightarrow, -\}$.

There are five different types of chart entry:

1. $C[a][b][1][\rightarrow]$ contains the sum of the probabilities of all possible complete subtrees rooted at position a, spanning a through $b > a$. The "complete value" of $c = 1$ means that the word at a cannot receive any more right dependents.

2. $C[a][b][1][\leftarrow]$ contains the sum of the probabilities of all possible complete subtrees rooted at b, spanning a through $b > a$. The subtrees are complete (i.e., $c = 1$), in that the word at position b cannot receive any more left dependents.

3. $C[a][b][0][\rightarrow]$ contains the sum of the probabilities of all possible incomplete subtrees rooted at a, spanning positions a through $b > a$. The "complete value" of 0 indicates that these subtrees can still gather more right dependents.

4. $C[a][b][0][\leftarrow]$ contains the sum of the probabilities of all possible incomplete subtrees rooted at b, spanning positions a through $b > a$. The subtrees are incomplete (i.e., $c = 0$), so the word at position b can gather more left dependents.

5. $C[a][b][2][-]$ contains the sum of the probabilities of all possible forests consisting of two trees, one rooted at a spanning a through m, where $a \leq m < b$, and one rooted at b spanning $m + 1$ through b. These trees will ultimately be combined such that the words at both a and b will be dependents of a word at some other position. This type of chart entry is necessary for keeping track of siblings.

The chart is built in a bottom-up fashion by considering subtrees of increasing length. The sum over all possible trees for sentence $\boldsymbol{w}$ with tags $\boldsymbol{s}$ and case values $\boldsymbol{c}$ is contained in the final entry to be completed, $C[0][|\boldsymbol{w}|][1][\rightarrow]$, where $|\boldsymbol{w}|$ is the length of $\boldsymbol{w}$ (excluding ROOT). Algorithm 4.1 shows the dynamic program for building the chart. A proposal tree $\boldsymbol{t}'$ may be sampled by recursively traversing the completed chart.

Having generated a proposal tree $\boldsymbol{t}'$ using the completed chart, $\boldsymbol{t}'$ is accepted as the current tree assignment for $\boldsymbol{w}$ with probability given by the minimum of 1 and

$$\frac{P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \boldsymbol{t}' \mid \mathcal{D}_{\setminus \boldsymbol{s},\boldsymbol{w},\boldsymbol{c},\boldsymbol{t}}, U)}{P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \boldsymbol{t} \mid \mathcal{D}_{\setminus \boldsymbol{s},\boldsymbol{w},\boldsymbol{c},\boldsymbol{t}}, U)} \frac{P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \boldsymbol{t} \mid \mathcal{D}_{\setminus \boldsymbol{s},\boldsymbol{w},\boldsymbol{c},\boldsymbol{t}}, \tilde{\Theta}, \tilde{\Phi}, \tilde{\Psi}, U)}{P(\boldsymbol{w}, \boldsymbol{s}, \boldsymbol{c}, \boldsymbol{t}' \mid \mathcal{D}_{\setminus \boldsymbol{w},\boldsymbol{s},\boldsymbol{c},\boldsymbol{t}}, \tilde{\Theta}, \tilde{\Phi}, \tilde{\Psi}, U)}, \tag{4.26}$$

where $\tilde{\Theta} = \{\tilde{\boldsymbol{\theta}}_{s'w'c's''d}\}$, $\tilde{\Phi} = \{\tilde{\boldsymbol{\phi}}_{ss'w'c'd}\}$ and $\tilde{\Psi} = \{\tilde{\boldsymbol{\psi}}_{sw}\}$. If the proposal tree $\boldsymbol{t}'$ is rejected, then the previously sampled tree $\boldsymbol{t}$ is accepted instead and kept as the current assignment. This Metropolis-Hastings step is necessary to compensate for the fact that $\boldsymbol{t}'$ was not drawn from the true posterior distribution over trees $P(\boldsymbol{t}' \mid \boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \mathcal{D}_{\setminus \boldsymbol{w},\boldsymbol{s},\boldsymbol{c},\boldsymbol{t}}, U)$. (It is not clear how to sample directly from the true posterior by constructing a collapsed, block Gibbs sampler based on the dynamic program described above. Integrating out the model parameters couples the nodes in each tree, invalidating the independence assumptions required by the dynamic program.)

In practice, Metropolis-Hastings rarely rejects a proposed tree: For any given sentence, the conditioning contexts ($s'w'c's''d$, $ss'w'c'd$ or sw) for which the probability vectors

function CREATECHART($\boldsymbol{w}$, $\boldsymbol{s}$)

  % initialise chart
  $C[a][a][1][d] := \text{PROB}(\text{STOP}, a, \text{START}, d) \qquad \forall d \in \{\leftarrow, \rightarrow\}$
  $C[a][a][c][d] := 1.0 \qquad \forall d \in \{\leftarrow, \rightarrow, -\}, \forall c \in \{0, 2\}$
  $C[a][a][1][-] := 1.0 \qquad \forall a \in \{0, \ldots, |\boldsymbol{w}|\}$

  for $k := 1$ to $|\boldsymbol{w}|$ { % width of the subtree
    for $a := 0$ to $|\boldsymbol{w}| - k$ {
      $b := a + k$

      % create "sibling" entry
      $C[a][b][2][-] := \sum_{a \leq m < b} C[a][m][1][\rightarrow]\, C[m+1][b][1][\leftarrow]$

      % parent picks up first child
      $l := C[a][b-1][1][\rightarrow]\, C[b][b][0][\leftarrow]\, \text{PROB}(a, b, \text{START}, \leftarrow)$
      $r := C[a][a][0][\rightarrow]\, C[a+1][b][1][\leftarrow]\, \text{PROB}(b, a, \text{START}, \rightarrow)$

      % parent picks up subsequent child (through sibling)
      $C[a][b][0][\leftarrow] := l + \sum_{a \leq m < b} C[a][m][2][-]\, C[m][e][0][\leftarrow]\, \text{PROB}(a, b, m, \leftarrow)$
      $C[a][b][0][\rightarrow] := r + \sum_{a < m \leq b} C[a][m][0][\rightarrow]\, C[m][b][2][-]\, \text{PROB}(b, a, m, \rightarrow)$

      % create "complete" entries
      $C[a][b][1][\leftarrow] := \sum_{a \leq m < b} C[a][m][1][\leftarrow]\, C[m][b][0][\leftarrow]\, \text{PROB}(\text{STOP}, b, m, \leftarrow)$
      $C[a][b][1][\rightarrow] := \sum_{a < m \leq b} C[a][m][0][\rightarrow]\, C[m][b][1][\rightarrow]\, \text{PROB}(\text{STOP}, a, m, \rightarrow)$
    }
  }

function PROB(a, b, m, d)
  if $a = \text{STOP}$ {
    if $m = \text{START}$ {
      return $\tilde{\theta}_{\text{STOP}|w_b\, s_b\, c_b\, \text{START}\, d}$
    else
      return $\tilde{\theta}_{\text{STOP}|w_b\, s_b\, c_b\, s_m\, d}$
    }
  else
    if $m = \text{START}$ {
      return $\tilde{\theta}_{s_a|w_b\, s_b\, c_b\, \text{START}\, d}\, \tilde{\phi}_{w_a|s_a\, w_b\, s_b\, c_b\, d}\, \tilde{\psi}_{c_a|w_a\, s_a}$
    else
      return $\tilde{\theta}_{s_a|w_b\, s_b\, c_b\, s_m\, d}\, \tilde{\phi}_{w_a|s_a\, w_b\, s_b\, c_b\, d}\, \tilde{\psi}_{c_a|w_a\, s_a}$
    }
  }

Algorithm 4.1: Constructing the dynamic programming chart.

$\tilde{\boldsymbol{\theta}}_{s'w'c's''d}$, $\tilde{\boldsymbol{\phi}}_{ss'w'c'd}$ or $\tilde{\boldsymbol{\psi}}_{sw}$ are exactly equal to the true predictive probabilities (i.e., those that would cause the dynamic program to generate a sample from the true posterior distribution over trees) are those that not only occur exactly once in that sentence, but whose context reductions similarly occur exactly once also. However, for contexts that occur many times in the corpus, $\tilde{\boldsymbol{\theta}}_{s'w'c's''d}$, $\tilde{\boldsymbol{\phi}}_{ss'w'c'd}$ or $\tilde{\boldsymbol{\psi}}_{sw}$ will be very close to the true predictive probabilities. Many contexts fall into one of these two categories. Consequently, for most sentences, the distribution from which the dynamic program generates a sample is close to the true posterior over trees $P(\boldsymbol{t}' \mid \boldsymbol{s}, \boldsymbol{w}, \boldsymbol{c}, \mathcal{D}_{\backslash \boldsymbol{w},\boldsymbol{s},\boldsymbol{c},\boldsymbol{t}}, U)$.

4.4.4 Results

Dependency parsing models are typically evaluated by computing parse accuracy—i.e., the percentage of parents correctly identified. Punctuation is usually excluded. The hierarchical Pitman-Yor dependency model was used to parse the Wall Street Journal sections of the Penn Treebank (Marcus et al., 1993). To facilitate comparison with other dependency parsing algorithms, the standard train/test split was used (sections 2–21 for training, and section 23 for testing), and parse accuracies were computed using the maximum probability trees rather than sampled trees. The Penn Treebank training sections consist of 39,832 sentences, while the test section consists of 2,416 sentences. No preprocessing was performed except for replacing words that occurred once in the training data (and never in the test data) or one or more times in the test data, but never in the training data, with one of several UNSEEN types (Eisner, 1996a):

- UNSEEN-SHORT: used for words less than six characters long.
- UNSEEN-NUM: used for words whose last character is a digit.
- UNSEEN-PUNC: used for words consisting entirely of punctuation characters.
- UNSEEN-XX: used for words of six or more characters in length. XX is replaced with the last two characters of the word.

Gold standard part-of-speech tags were used for the training data, while tags for the test data were inferred using a standard part-of-speech tagger (Ratnaparkhi, 1996).[2]

Parse accuracy was computed for several different model variants:

- Hierarchical Dirichlet (i.e., no discount parameters) with fixed concentration parameters, set to the values used by Eisner. When used with the maximal path assumption, this model variant is identical to Eisner's model.
- Hierarchical Dirichlet with slice-sampled concentration parameters.

[2]The generative nature of the dependency parser means that it is possible to sample part-of-speech tags for test data at the same time as sampling trees. However, it is computationally expensive and results in very similar performance to using part-of-speech tags from Ratnaparkhi's tagger.

- Pitman-Yor with fixed concentration parameters, set to the values used by Eisner, and fixed discount parameters, all set to 0.1.
- Pitman-Yor with slice-sampled concentration and discount parameters.

For each model variant, all experiments were performed using both the maximal and minimal path assumptions. For the variants with fixed concentration and discount parameters, the concentration parameters were set to the values recommended by Eisner (see section 4.3.1), while the discount parameters were set to 0.1. For the model variants with sampled concentration and discount parameters, fifty iterations of slice sampling proved sufficient to reach convergence. Results are shown in figure 4.2.

The parse accuracies for the model variant equivalent to Eisner's dependency model (hierarchical Dirichlet prior, fixed concentration parameters) are lower than those reported in Eisner's work (1996a; 1996b). This is because Eisner's results were obtained using an extensively filtered smaller data set (e.g., sentences with conjunctions are discarded). In the time since Eisner's model was published a different train/test split has become standard, and the results in figure 4.2 are reported on the now-standard split.

The results in figure 4.2 clearly demonstrate that using a hierarchical Pitman-Yor prior and sampling hyperparameters both give considerable improvements over a hierarchical Dirichlet model with fixed concentration parameters and the maximal path assumption (i.e., Eisner's original model). Interestingly, the differences in accuracy between the maximal and minimal path assumptions are not significant. In the hierarchical Dirichlet variant of the model, sampling hyperparameters gives an accuracy improvement of approximately 4%. Using a hierarchical Pitman-Yor prior improves accuracy over the hierarchical Dirichlet variant by approximately 3%. Sampling the hyperparameters of the Pitman-Yor prior gives an accuracy improvement of 5% over the Eisner-equivalent hierarchical Dirichlet model. This corresponds to a 26% reduction in error. Although state-of-the-art dependency models, such as the discriminative maximum-margin method of McDonald (2006), do achieve higher parse accuracy (e.g., 91.5% for McDonald's model; see figure 4.3) the hierarchical Pitman-Yor dependency model uses exactly the same contexts and reductions as Eisner's original model. In contrast, McDonald's model uses a very large number of potentially relevant features. It is therefore possible that further consideration of contexts and reductions, as well as other enhancements to the Pitman-Yor dependency model would yield similar results while retaining the benefits of a generative model. Possible enhancements include aggregation across multiple samples, sampling of internal draws and paths, and a letter-based language model as a top-level base measure (Cowans, 2006).

		Path Assumption	
		Maximal	Minimal
Dirichlet	fixed α values (Eisner, 1996a,b)	80.7	80.2
Dirichlet	sampled α values	84.3	84.1
Pitman-Yor	fixed α and ϵ values	83.6	83.7
Pitman-Yor	sampled α and ϵ values	85.4	**85.7**

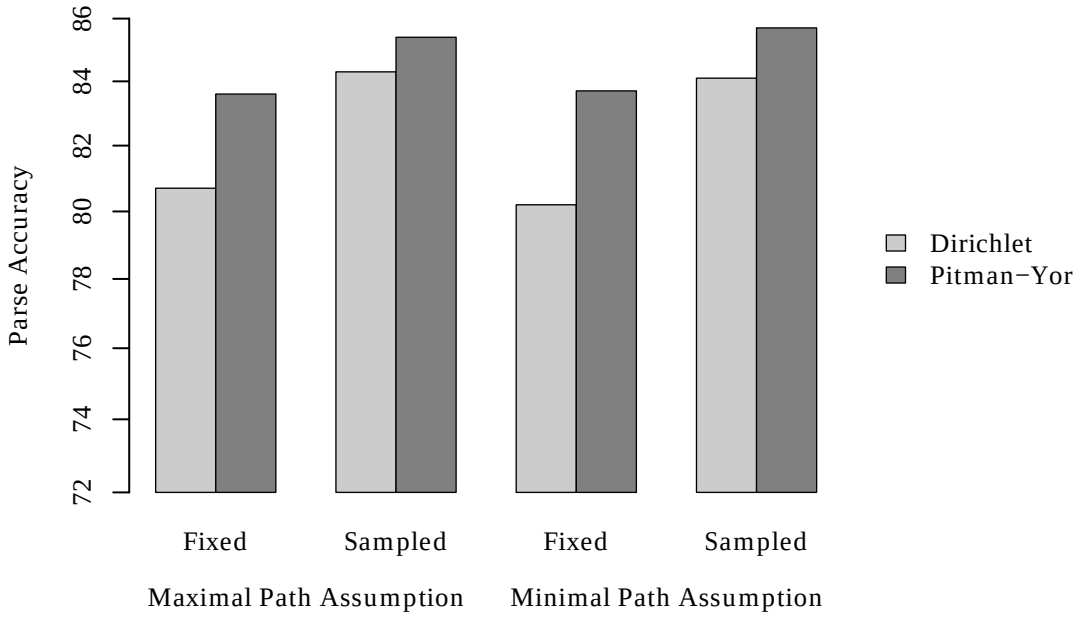


Figure 4.2: Parse accuracy (percentage of words whose parents are correctly identified) for the hierarchical Pitman-Yor dependency model on Penn Treebank data. Results are computed using the maximum probability tree. "Fixed" refers to fixed hyperparameters, while "Sampled" refers to sampled hyperparameters.

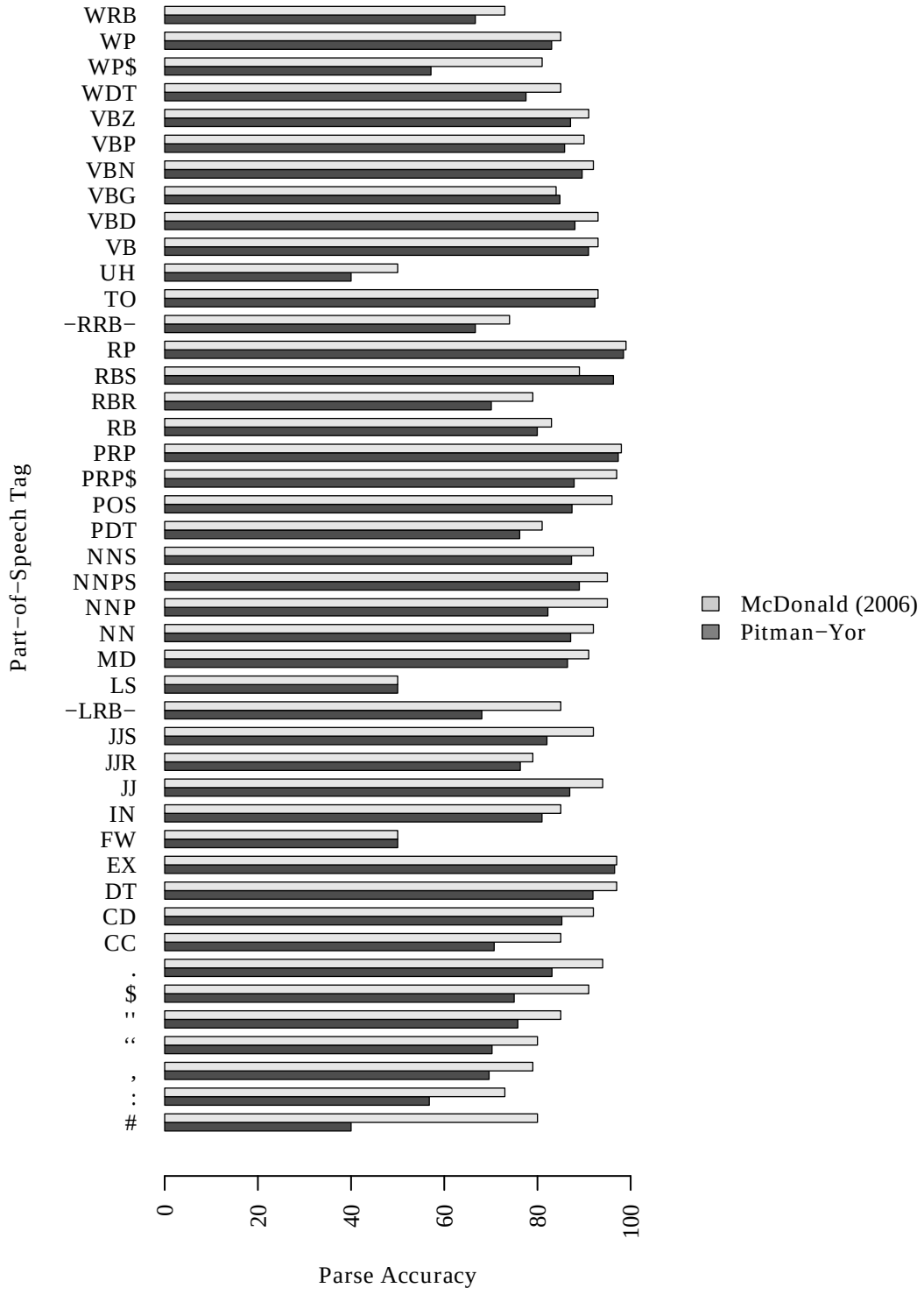


Figure 4.3: Parse accuracy by part-of-speech tag for McDonald's discriminative maximum-margin method (McDonald, 2006) and the best-performing Pitman-Yor model variant (sampled hyperparameters, minimal path assumption).

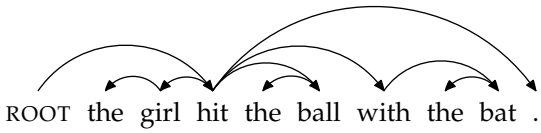


Figure 4.4: An example dependency tree for an untagged, uncased sentence.

4.5 A "Syntactic Topic" Dependency Model

One advantage of a generative approach to dependency modelling is that other latent variables may be incorporated into the model. To demonstrate this, I present a dependency parsing model with latent variables that mediate the relationships between words and their dependents, resulting in a clustering of parent–child dependencies.

This model can be considered to be a dependency-based analogue of the syntactic component from the syntax-based topic model of Griffiths et al. (2005). The models differ in their underlying structure, however: In the model presented in this section, the underlying structure is a tree that combines both words and unobserved syntactic states; in Griffiths et al.'s model, the structure is a simply a chain over latent states. This difference means that there are two kinds of latent information that must be inferred in the dependency-based model: The structure of each dependency tree and the identities of the latent states. In Griffiths et al.'s model, only the latter need be inferred.

4.5.1 Model Structure

The generative process underlying the model in this section is similar to that of the model presented in the previous section, with four key differences:

- Sentences are untagged and uncased,
- STOP symbols are handled differently,
- siblings are not taken into account (i.e., it is a first order model),
- latent state variables mediate the relationships between parents and children.

Generation of STOP symbols is handled by a separate two-valued (STOP/CONTINUE) distribution. Prior to generating a child, a value is sampled from this distribution. If the value is CONTINUE a child is generated; otherwise, no more children are generated in that direction. Although it is possible to handle STOP generation by incorporating the STOP symbol into the distribution over latent states (similar to the way in which it is incorporated into the distribution over part-of-speech tags in the previous model), a separate STOP/CONTINUE distribution results in slightly better performance here.

Models that ignore siblings are more computationally efficient. However, the decision to ignore siblings means that conditioned on their parent, children are independent of

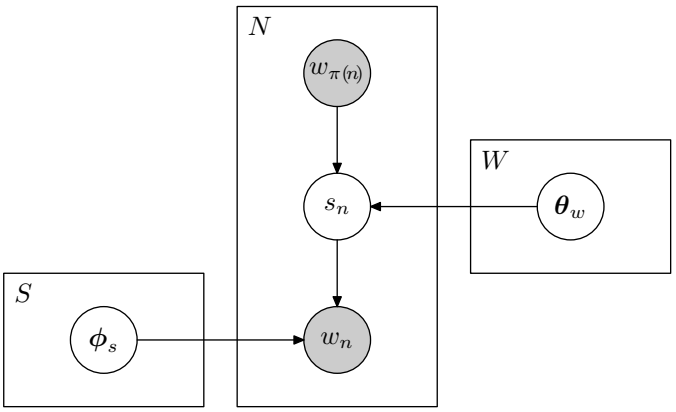


Figure 4.5: Graphical model for the dependency model with latent states.

each other. This affects the model in the following way: Having observed the dependency tree depicted in figure 4.4 (and nothing else), a model that ignores siblings is just as likely to generate "The girl hit . with the bat the ball" as "The girl hit the ball with the bat.". A model that takes siblings into account will have only a very small (or zero, depending on the hyperparameters) probability of generating the first sentence.

Most importantly, the inclusion of latent state variables means that the model does not need to separately learn about the distributions over children for words that are often used similar contexts (e.g., "ate" and "consumed"). Instead, the model can infer that these words should have a high probability of generating some particular state s, which is then responsible for generating children. (The model does not, however, assume that all instances of word w must generate or be generated by the same state.) This means that the model is better able to generalise about future dependencies.

The probability of a sentence $\boldsymbol{w}$ with latent states $\boldsymbol{s}$ and tree $\boldsymbol{t}$ is given by

$$\begin{aligned} P(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{t}) = & \\ & \prod_n \theta_{s_n|w_{\pi(n)}} \, \phi_{w_n|s_n} \, \psi_{\text{CONTINUE}|w_{\pi(n)}d_n} \\ & \prod_n \psi_{\text{STOP}|w_n d_{\text{STOP}}=\leftarrow} \, \psi_{\text{STOP}|w_n d_{\text{STOP}}=\rightarrow}, \end{aligned} \tag{4.27}$$

where the vector $\boldsymbol{\theta}_{w'}$ is the distribution over latent states for parent word w', the vector $\boldsymbol{\phi}_s$ is the distribution over child words for latent state s, and the vector $\boldsymbol{\psi}_{w'd}$ is the distribution that controls tree structure via STOP generation. In other words, parent words are collapsed down to the latent state space and children are generated on the basis of these states. As a result, the clusters induced by the latent states are expected to exhibit syntactic properties and can be thought of as "syntactic topics"—specialised distributions over words with a syntactic flavour. The model is depicted in figure 4.5.

Each of the probability vectors in equation 4.27 is given a Dirichlet prior:

$$P(\boldsymbol{\theta}_{w'} \mid \alpha \boldsymbol{m}) = \text{Dir}\,(\boldsymbol{\theta}_{w'} \mid \alpha \boldsymbol{m}), \tag{4.28}$$

$$P(\boldsymbol{\phi}_s \mid \beta \boldsymbol{u}) = \text{Dir}\,(\boldsymbol{\phi}_s \mid \beta \boldsymbol{u}) \tag{4.29}$$

and

$$P(\boldsymbol{\psi}_{w',d} \mid \zeta \boldsymbol{n}_d) = \text{Dir}\,(\boldsymbol{\psi}_{w',d} \mid \zeta \boldsymbol{n}_d) \tag{4.30}$$

$$P(\boldsymbol{n}_d \mid \zeta_1 \boldsymbol{n}) = \text{Dir}\,(\boldsymbol{n}_d \mid \zeta_1 \boldsymbol{n}) \tag{4.31}$$

$$P(\boldsymbol{n} \mid \zeta_0 \boldsymbol{u}) = \text{Dir}\,(\boldsymbol{n} \mid \zeta_0 \boldsymbol{u}). \tag{4.32}$$

The base measure and concentration parameter for the prior over $\boldsymbol{\theta}_{w'}$ are optimised using first of the two fixed-point methods described in section 2.3.5, while the base measures for the prior over the stop probability vector $\boldsymbol{\psi}_{w'd}$ are integrated out.

4.5.2 Inference

Given a training corpus $\mathcal{D} = \{\boldsymbol{w}, \boldsymbol{t}\}$ consisting of uncased sentences and their corresponding trees, there are two tasks of interest: Sampling latent states for the training data, and sampling states and trees for unseen test sentences. Sampling states for a training sentence is similar to sampling topics in latent Dirichlet allocation (Griffiths and Steyvers, 2004)—the states are initialised randomly and then resampled using Gibbs sampling. Each state s_n is resampled from its conditional distribution given all other state assignments, words and trees in the training data:

$$\begin{aligned} &P(s_n \mid \{\boldsymbol{w}\}, \{\boldsymbol{s}\}_{\setminus n}, \{\boldsymbol{t}\}, U) \propto \\ &\quad P(w_n \mid s_n, \{\boldsymbol{s}\}_{\setminus n}, \{\boldsymbol{w}\}_{\setminus n}, \{\boldsymbol{t}\})\, P(s_n \mid \{\boldsymbol{s}\}_{\setminus n}, \{\boldsymbol{w}\}_{\setminus n}, \{\boldsymbol{t}\}), \end{aligned} \tag{4.33}$$

where the subscript "$\setminus n$" denotes a quantity that excludes data from the n^{th} position in the corpus. The variable U denotes the full set of model hyperparameters.

Given a set of training words and trees and a single sample of training states, the trees and states for unseen test data may be sampled using an augmented version of the inference algorithm described in section 4.4.3 in which states are marginalised over when performing the bottom-up chart-building pass for a test sentence. States and a tree for this sentence can then be sampled simultaneously in a top-down fashion.

4.5.3 Results

Penn Treebank sections 2–21 were used as training data. The true dependency trees and words were used to obtain a single sample of states. These training states, trees and words, were then used to sample states and trees for Penn Treebank section 23.

Some example states or "syntactic topics" are shown in table 4.2. Each column in each row consists of the ten words most likely to be generated by a particular state. The states exhibit a good correspondence with parts-of-speech, but are more finely grained. For example, the states in the first and third columns in the top row both correspond to nouns. However, the first contains job titles, while the third contains place names. Similarly, the states in the fourth and fifth columns in the top row both correspond to verbs. However, the fourth contains transitive past-tense verbs, while the fifth contains present-tense verbs. The state shown in the final column in the bottom row is particularly interesting because the top words are entirely plural nouns. This kind of specificity indicates that these states are likely to be beneficial in other tasks where part-of-speech tags are typically used, such as named entity recognition.

As a measure of the quality of these "syntactic topics", they can be used in place of part-of-speech tags in parsing experiments. The parsing performance (parse accuracy) obtained using the latent state dependency model was compared with the performance of an equivalent model in which the states were fixed to true part-of-speech tags for both training and test data. These results are shown in figure 4.6. Using the sampled states gives an improvement in accuracy of approximately 5% for sampled trees and an improvement of 1.6% for the most probable trees. Although this is a modest improvement in parsing accuracy, it is a clear quantitative indication that the discovered states do indeed capture syntactically meaningful information.

4.6 Conclusions

In this chapter, I introduced a new dependency parsing model based on the hierarchical Pitman-Yor process. Using this model, I showed that the performance of Eisner's generative dependency parsing model can be significantly improved by using a hierarchical Pitman-Yor prior and by sampling model hyperparameters. On the Penn Treebank data, this leads to a 26% reduction in parsing error over Eisner's model. I also presented a second Bayesian dependency model, in which the local dependency distributions are mediated by latent variables that cluster parent–child dependencies. Not only do the inferred latent variables look like finer-grained parts-of-speech, they result in better parse accuracy when substituted for part-of-speech tags in the model.

	Type of Tree	
	Sampled Trees	Most Probable Tree
50 states	59.2	63.8
100 states	60.0	64.1
150 states	60.5	64.7
200 states	60.4	64.5
POS tags	55.3	63.1

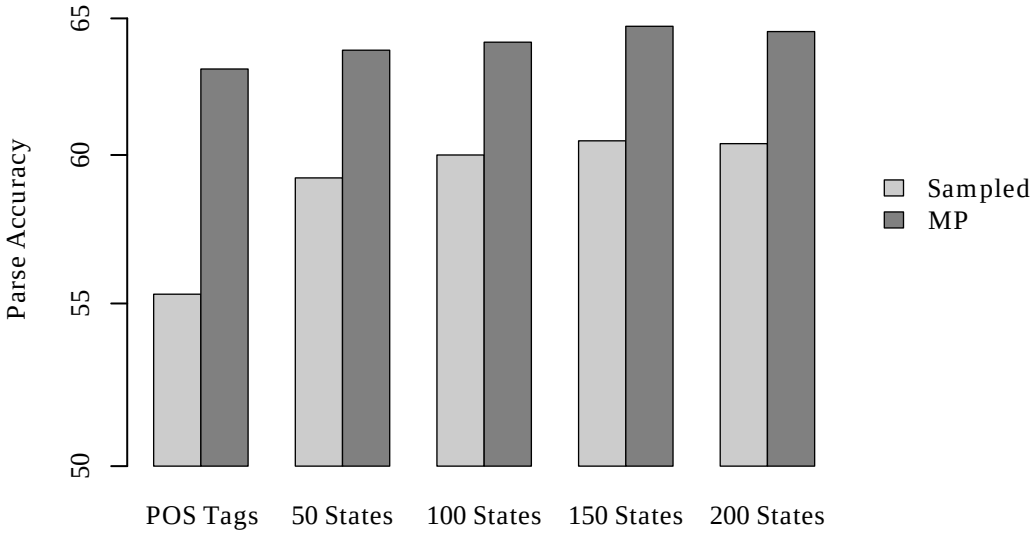


Figure 4.6: Parse accuracy for the "syntactic topic" dependency model (percentage of words whose parents are correctly identified by the model) on the Penn Treebank (standard train/test split). As a baseline, the latent states are fixed to part-of-speech tags. "Sampled" refers to sampled trees, while "MP" refers to the most probable tree. Results for sampled trees are averaged over ten samples.

president	year	u.s.	made	is	in
director	years	california	offered	are	on
officer	months	washington	filed	was	,
chairman	quarter	texas	put	has	for
executive	example	york	asked	have	at
head	days	london	approved	were	with
attorney	time	japan	announced	will	and
manager	weeks	canada	left	had	as
chief	period	france	held	's	by
secretary	week	britain	bought	would	up
10	would	more	his	ms.	sales
8	will	most	their	mrs.	issues
1	could	very	's	who	prices
50	should	so	her	van	earnings
2	can	too	and	mary	results
15	might	than	my	lee	stocks
20	had	less	your	dorrance	rates
30	may	and	own	linda	costs
25	must	enough	'	carol	terms
3	owns	about	old	hart	figures

Table 4.2: The top ten words most likely to be generated as children by twelve of the states inferred from the true dependency trees for the Penn Treebank training sections (sections 2–21). These examples were all from a model with 150 states.

Chapter 5

Cluster-Based Topic Modelling

In this chapter, I present a hierarchical Bayesian model for clustering documents by topic. The model extends a well-known Bayesian topic model, latent Dirichlet allocation (Blei et al., 2003), to incorporate latent document groupings. Given a document collection, these groupings, along with topic assignments for each document, are inferred using an unsupervised approach. The model is evaluated on a collection of academic papers, and exhibits better predictive accuracy than either latent Dirichlet allocation or a clustering model without latent topics. Furthermore, the groups inferred by the new model are clearly interpretable and correspond well to known research areas. Finally, I also show how author information may be incorporated into the model, resulting in a cluster-based author–topic model with even better predictive accuracy and finer-grained groupings than the model variant without author information.

5.1 Introduction

The models presented in the previous two chapters concentrated on sentence-level document structure. However, collections of documents also exhibit higher-level structure, including structure across document boundaries. For example, academic papers from a particular conference or journal may be seen as arising from groups or communities of individuals working on closely related topics. Information about these document groupings is useful for performing coarse-grained analyses, e.g., "How fragmented is this conference? Should it be split in two?", and for making instance-level predictions, e.g., "Which individuals will co-author a paper together next year?", as well as for organising and navigating conference proceedings. In practice, these kinds of relationships between documents are usually unobserved. Consequently, there is a need for models that are able to use available data—such as document content and authorship information—to determine latent document groupings.

This chapter introduces a nonparametric Bayesian model that uses a topic-based ap-

proach to find groups of related documents. Unlike earlier models that cluster documents into groups using raw word counts, such that of Nigam et al. (1998), the model presented in this chapter has the advantage of being robust to variations in vocabulary: Documents that are about similar topics but use slightly different terminology will be grouped together. Furthermore, the use of nonparametric Bayesian techniques means that the new model will automatically discover the most appropriate number of clusters for the data, rather than requiring that this number be specified in advance. Finally, the new model can be extended to account for other relevant information, such as authorship information, thereby resulting in a more informed clustering.

The model presented in this chapter is related to the information bottleneck-based model of Slonim and Tishby (2000), in that document groupings are inferred using a low-dimensional representation of each document. However, the models differ in three ways: Firstly, the topics in Slonim and Tishby's model are categorical—i.e., each word must belong to exactly one topic. In contrast, the topics used by the model in this chapter are componential: Multiple topics can account for the same word. Secondly, in Slonim and Tishby's model, topics are inferred just once, prior to cluster inference, and then fixed. Here, clusters and topics are inferred simultaneously. Thirdly, Slonim and Tishby's model uses a predetermined number of document clusters, whereas the new model can automatically select the number of clusters that best describes the data.

The new model is also similar to Dirichlet enhanced latent semantic analysis (Yu et al., 2005). There is, however, an important difference between the two models: In Dirichlet enhanced latent semantic analysis, the cluster-specific distributions over topics are used, without modification, as document-specific topic distributions. More precisely, when generating a new document using Dirichlet enhanced latent semantic analysis, the distribution over topics for that document is taken to be the distribution over topics for the cluster to which that document belongs. When generating a document using the new model introduced in this chapter, the cluster-specific topic distribution is instead used as the base measure for a Dirichlet distribution, from which the document-specific topic distribution is drawn. This difference means that the new model is more flexible than Dirichlet-enhanced latent semantic analysis: The topic distributions for documents belonging to a single cluster are allowed to vary around the cluster-specific topic distribution. This property is appealing: Although documents in the same cluster should have similar topic distributions, their topic distributions need not be identical. One consequence of this difference is as follows: When making predictions about the occurrence of future topics in some document, Dirichlet enhanced latent semantic analysis treats the topic usage counts for that document as being no more important than the topic usage counts for the entire cluster. In contrast, the model presented in this chapter can automatically determine the extent to which the document-specific topic usage counts should influence the selection of future topics, and can therefore give them greater influence than the cluster-specific counts.

5.2 Topic Modelling

In this section I briefly review latent Dirichlet allocation (Blei et al., 2003), and describe how this framework can be extended to incorporate known document groupings, to give a model that is the finite analogue of Teh et al.'s model (2006) for multiple corpora.

5.2.1 Latent Dirichlet Allocation

As described in section 3.3, latent Dirichlet allocation (Blei et al., 2003) models documents as finite mixtures over latent topics, where each topic is characterised by a distribution over words. Given a corpus $\boldsymbol{w}$, each token w_n is assumed to have been generated by first drawing a topic assignment z_n from a document-specific distribution over topics, and then drawing w_n from the distribution over words that characterises that topic. Letting W, T and D be respectively the size of the vocabulary, the number of topics and the number of documents in the corpus, the model parameters are typically denoted by Φ, a $T \times W$ matrix with elements given by $\phi_{w|t} = P(w_n = w \,|\, z_n = t)$, and Θ, a $D \times T$ matrix with elements given by $\theta_{t|d} = P(z_n = t \,|\, d_n = d)$. The joint probability of corpus $\boldsymbol{w}$ and corresponding topic assignments $\boldsymbol{z}$ is therefore

$$P(\boldsymbol{w}, \boldsymbol{z} \,|\, \Phi, \Theta) = \prod_w \prod_t \prod_d \phi_{w|t}^{N_{w|t}} \, \theta_{t|d}^{N_{t|d}}, \tag{5.1}$$

where $N_{t|d}$ is the number of times that topic t has been used in document d and $N_{w|t}$ is the number of times that word w has been generated by topic t. Finally, Blei et al. place (nonhierarchical) Dirichlet distribution priors over Φ and Θ:

$$P(\Phi \,|\, \beta\boldsymbol{n}) = \prod_t \mathrm{Dir}\,(\boldsymbol{\phi}_t \,|\, \beta\boldsymbol{n}) \tag{5.2}$$

$$P(\Theta \,|\, \alpha\boldsymbol{m}) = \prod_d \mathrm{Dir}\,(\boldsymbol{\theta}_d \,|\, \alpha\boldsymbol{m}). \tag{5.3}$$

The hyperparameters $\beta\boldsymbol{n}$ and $\alpha\boldsymbol{m}$ are given improper noninformative priors.

The use of a nonhierarchical Dirichlet prior over the document-specific topic distributions is certainly appropriate for corpora where all documents are part of the same underlying group. In this case, the base measure $\boldsymbol{m}$ acts as a single "prototype" distribution over topics for the group, while the concentration parameter α controls the extent to which the $\boldsymbol{\theta}_d$ probability vectors will vary from this prototype. For other sorts of corpora, however, this prior may not be the best choice: For example, a collection of news articles may contain some articles about sport and others about business. While the articles about sport are likely to use similar topics to each other, they are less likely to use the topics that occur in articles about business. Consequently, when inferring topics for a new article, the identity of its group—sport or business—will reveal useful information about its topic composition. A topic model that accounts for

these sorts of groupings is therefore a better model of the data than one that doesn't.

5.2.2 Incorporating Document Groupings

If document groupings are known, it is easy to extend latent Dirichlet allocation to incorporate group information: Rather than drawing each document-specific topic distribution $\boldsymbol{\theta}_d$ from a single Dirichlet prior, each distribution can instead be drawn from a *group-specific* Dirichlet, thereby respecting the document groupings. Letting c_d denote the group for document d, the distribution over $\boldsymbol{\theta}_d$ is now given by

$$P(\boldsymbol{\theta}_d \mid \alpha \boldsymbol{m}_{c_d}) = \text{Dir}\,(\boldsymbol{\theta}_d \mid \alpha \boldsymbol{m}_{c_d}). \tag{5.4}$$

In order to capture topic similarities between groups—that is, the overall prevalence of each topic in the corpus—the group-specific base measures $\{\boldsymbol{m}_c\}_{c=1}^{C}$ may also be given Dirichlet priors, with a single, shared, corpus-level base measure $\boldsymbol{m}$:

$$P(\boldsymbol{m}_c \mid \alpha_1 \boldsymbol{m}) = \text{Dir}\,(\boldsymbol{m}_c \mid \alpha_1 \boldsymbol{m}). \tag{5.5}$$

The concentration parameter α_1 determines the extent to which the group-specific base measures (and hence the document-specific distributions over topics) are influenced by the corpus-level base measure. Finally, the corpus-level base measure may itself be given a Dirichlet prior, this time with uniform base measure $\boldsymbol{u}$:

$$P(\boldsymbol{m} \mid \alpha_0 \boldsymbol{u}) = \text{Dir}\,(\boldsymbol{m} \mid \alpha_0 \boldsymbol{u}). \tag{5.6}$$

The prior induced over $\boldsymbol{\theta}_d$ by equations 5.4, 5.5 and 5.6 is a hierarchical Dirichlet. When combined with equation 5.1 and Blei et al.'s prior over topic-specific word distributions (equation 5.2), the resultant model is a finite version of the hierarchical Dirichlet process document model for multiple corpora described by Teh et al. (2006).

Using the terminology and notation introduced in section 3.4.2, the predictive probability of topic t occurring in document d under the prior described above is

$$P(t \mid d, c_d, \boldsymbol{z}, \boldsymbol{c}, \alpha, \alpha_1, \alpha_0) = \frac{\hat{N}_{t|d} + \alpha \dfrac{\hat{N}_{t|c_d} + \alpha_1 \dfrac{\hat{N}_t + \alpha_0\, u_t}{\hat{N}_\cdot + \alpha_0}}{\hat{N}_{\cdot|c_d} + \alpha_1}}{\hat{N}_{\cdot|d} + \alpha} \tag{5.7}$$

where the quantities $\hat{N}_{t|d}$, $\hat{N}_{t|c_d}$ and $\hat{N}_t$ are given by

$$\hat{N}_{t|d} = \sum_{i=1}^{I} N_{\cdot|d}^{(i)} \, \delta\left(\gamma_i - t\right), \tag{5.8}$$

$$\hat{N}_{t|c_d} = \sum_{j=1}^{J} N_{\cdot|c_d}^{(j)} \, \delta\left(\gamma_j - t\right), \tag{5.9}$$

$$\hat{N}_t = \sum_{k=1}^{K} N_{\cdot}^{(k)} \, \delta\left(\gamma_k - t\right), \tag{5.10}$$

and I, J and K are the current numbers of internal draws for the bottom-, middle- and top-level (i.e., document-, group- and corpus-level) Dirichlet-multinomials. The quantity $N_{\cdot|d}^{(i)}$ is the number of observations currently matched to bottom-level internal draw γ_i, while $N_{\cdot|c_d}^{(j)}$ is the number of bottom-level internal draws matched to middle-level internal draw γ_j. Finally, $N_{\cdot}^{(k)}$ is the number of middle-level internal draws matched to top-level internal draw γ_k. Under the maximal path assumption (described, along with the minimal path assumption, in section 3.4.2), $\hat{N}_{t|c_d} = \sum_j N_{\cdot|c_d}^{(j)} \, \delta\left(\gamma_j - t\right)$ is equal to $N_{t|c_d}$, the number of times topic t has been used in group c_d, while under the minimal path assumption it is equal to the number of different documents belonging to c_d that use t. Similarly, under the maximal path assumption $\hat{N}_t = \sum_k N_{\cdot}^{(k)} \, \delta\left(\gamma_k - t\right)$ is equal to the number of times t has been used in the entire corpus, and under the minimal path assumption it is equal to the number of different groups in which t has been used. The bottom-level quantity $\hat{N}_{t|d} = \sum_i N_{\cdot|d}^{(i)} \, \delta\left(\gamma_i - t\right)$ is always equal to the number of times topic t has been used in document d.

It is evident from equation 5.7 that the predictive probability of topic t is influenced not only by document- and corpus-level topic usage (as in latent Dirichlet allocation), but also group-level topic usage, as desired. The extent to which these levels affect the predictive distribution is determined by the concentration parameters α and α_1.

5.3 A Cluster-Based Topic Model

Although the model described in the previous section (as well as the infinite version presented by Teh et al. (2006)) is appropriate for modelling corpora where document groupings are fully observed, it cannot be directly used to model document collections where groups or clusters are known to exist but are unobserved. In this situation, one approach would be to treat clusters and topics separately and use some clustering model to group the documents (on the basis of their word usage alone). The inferred cluster labels could then be used as observed variables in the prior described in the previous section. This is a rather unsatisfactory solution, however, as the latent topics cannot influence cluster inference. A better approach would be to construct a single

combined model, in which latent clusters and topics are simultaneously inferred.

One way of developing a single model for clusters and topics is to extend the model in section 5.2.2 so that each document's group or cluster membership c_d is treated as a latent variable. Assuming the number of clusters C is known, cluster generation is defined by a C-dimensional probability vector $\boldsymbol{\psi}$. The joint probability of a corpus $\boldsymbol{w}$ with topic assignments $\boldsymbol{z}$ and cluster assignments $\boldsymbol{c}$ is now given by

$$P(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{c} \,|\, \Phi, \Theta, \boldsymbol{\psi}) = \prod_w \prod_t \prod_d \prod_c \phi_{w|t}^{N_{w|t}} \, \theta_{t|d}^{N_{t|d}} \, \psi_c^{N_c}, \tag{5.11}$$

where $N_{w|t}$ is the number of times word w has been seen in topic t, $N_{t|d}$ is the number of times topic t has been used in document d and N_c is the number of documents in cluster c. Since probability vector $\boldsymbol{\psi}$ is unknown, it can be given a prior:

$$P(\boldsymbol{\psi} \,|\, \zeta \boldsymbol{u}) = \text{Dir}\,(\boldsymbol{\psi} \,|\, \zeta \boldsymbol{u}), \tag{5.12}$$

where $\boldsymbol{u}$ is a uniform base measure over clusters $1 \ldots C$ and ζ is a concentration parameter. Under this prior, the predictive probability of new document d being generated by cluster c (given the existing cluster membership variables $\boldsymbol{c}_{<d}$) is

$$P(c_d{=}c \,|\, \boldsymbol{c}_{<d}, \zeta \boldsymbol{u}) = \frac{N_c + \zeta u_c}{\sum_c N_c + \zeta}, \tag{5.13}$$

where the quantity N_c is the number of documents generated by cluster c so far.

The prior over $\boldsymbol{\psi}$ (equation 5.12) can be combined with equation 5.11 and the priors over Φ (equation 5.2) and Θ (equations 5.4, 5.5 and 5.6) to give joint distribution of $\boldsymbol{w}$, $\boldsymbol{z}$, $\boldsymbol{c}$, Φ, Θ, Ψ. Marginalising over unknown variables gives the evidence for the model hyperparameters $U = \{\beta \boldsymbol{n}, \alpha, \alpha_1, \alpha_0, \zeta\}$ or the probability of $\boldsymbol{w}$ given U:

$$P(\boldsymbol{w} \,|\, U) = \sum_{\boldsymbol{z}, \boldsymbol{c}} P(\boldsymbol{w} \,|\, \boldsymbol{z}, U) \, P(\boldsymbol{z} \,|\, \boldsymbol{c}, U) \, P(\boldsymbol{c} \,|\, U), \tag{5.14}$$

where

$$P(\boldsymbol{w} \,|\, \boldsymbol{z}, U) = \prod_t \frac{\prod_w \Gamma(N_{w|t} + \beta n_w)}{\Gamma(N_{\cdot|t} + \beta)} \, \frac{\Gamma(\beta)}{\prod_w \Gamma(\beta n_w)}, \tag{5.15}$$

$$P(\boldsymbol{z} \,|\, \boldsymbol{c}, U) = \prod_n P(z_n \,|\, d_n, c_{d_n}, \boldsymbol{z}_{<n}, \boldsymbol{c}_{<d_n}, U) \tag{5.16}$$

and

$$P(\boldsymbol{c} \,|\, U) = \frac{\prod_c \Gamma(N_c + \zeta u_c)}{\Gamma(\sum_c N_c + \zeta)} \, \frac{\Gamma(\zeta)}{\prod_c \Gamma(\zeta u_c)}. \tag{5.17}$$

The document for the n^{th} word in the corpus is denoted by d_n, while c_{d_n} is the group

for that document. $P(z_n \mid d_n, c_{d_n}, \boldsymbol{z}_{<n}, \boldsymbol{c}_{<d_n}, U)$ may be computed using equation 5.7.

The latent cluster and topic assignments ($\boldsymbol{c}$ and $\boldsymbol{z}$, respectively) can be inferred using a Gibbs sampler that alternates between sampling cluster assignments given the current topic assignments and topic assignments given the current cluster assignments.

5.3.1 Using an Unknown Number of Latent Clusters

For most real-world data, the number of groups or clusters, as well as the cluster assignments, are unknown. In this situation, the model described in the previous section can be modified to handle an unknown number of clusters by using a Dirichlet *process* prior (Ferguson, 1973). To facilitate this, it is convenient to work directly in terms of the cluster-specific base measures used in the hierarchical prior over Θ (given in equations 5.4, 5.5 and 5.6), rather than the cluster assignment variables $\boldsymbol{c}$.

The Dirichlet process, briefly mentioned in section 4.3.2, is the infinite generalisation of the Dirichlet distribution. Although the Dirichlet process distributes probability mass over an infinite set of points in a continuous space of probabilities, draws from a Dirichlet process are discrete with probability one. It is therefore an appropriate choice of prior for the cluster-specific base measures $\{\boldsymbol{m}_c\}_{c=1}^{\infty}$, of which there are a finite but unknown number. If a cluster-specific base measure is drawn for each document d (thereby implicitly assigning document d to a cluster), the probability of drawing the same base measure $\boldsymbol{m}_c$ for multiple documents (i.e., assigning multiple documents to the same cluster) should be nonzero. A Dirichlet process prior results in exactly this.

A Dirichlet process prior over the cluster-level base measures can be incorporated into the hierarchical prior over Θ (equations 5.4, 5.5 and 5.6) as follows:

$$P(\Theta \mid \{\alpha \boldsymbol{m}_c\}_{c=1}^{\infty}) = \prod_d \text{Dir}\,(\boldsymbol{\theta}_d \mid \alpha \boldsymbol{m}_d) \tag{5.18}$$

$$P(\boldsymbol{m}_d \mid G) = G(\boldsymbol{m}_d) \tag{5.19}$$

$$P(G \mid \zeta, G_0) = \text{DP}\,(G \mid \zeta, G_0), \tag{5.20}$$

where G is is a random probability measure distributed according to a Dirichlet process with base measure G_0 and concentration parameter ζ. According to the stick-breaking construction (Sethuraman, 1994), if $G \sim \text{DP}\,(G \mid \zeta, G_0)$, then

$$G(\boldsymbol{m}_d) = \sum_{c=1}^{\infty} \pi_c \, \delta_{\boldsymbol{m}_c}(\boldsymbol{m}_d) \tag{5.21}$$

where $\delta_{\boldsymbol{m}_c}(\cdot)$ is a point mass located at $\boldsymbol{m}_c$ and

$$P(\boldsymbol{m}_c \mid G_0) = G_0(\boldsymbol{m}_c) \tag{5.22}$$

$$\pi_c = \pi'_c \prod_{k=1}^{c-1} (1 - \pi'_k) \tag{5.23}$$

$$P(\pi'_c \mid \zeta) = \text{Beta}\,(\pi'_c \mid 1, \zeta) \tag{5.24}$$

Since each $\boldsymbol{m}_c$ is a probability vector, the base measure G_0 must be a distribution over probability vectors. In this case, G_0 is a hierarchical Dirichlet distribution:

$$G_0 = \text{Dir}\,(\boldsymbol{m}_c \mid \alpha_1 \boldsymbol{m}), \tag{5.25}$$

where base measure $\boldsymbol{m}$ is also drawn from a Dirichlet distribution:

$$P(\boldsymbol{m} \mid \alpha_0 \boldsymbol{u}) = \text{Dir}\,(\boldsymbol{m} \mid \alpha_0 \boldsymbol{u}). \tag{5.26}$$

Probability vector $\boldsymbol{u}$ is the uniform distribution over topics. This choice of G_0 ensures that the only effect of the Dirichlet process on the prior over Θ is to allow a variable number of clusters—the predictive probability of topic t in document d (with unknown probability vectors $\boldsymbol{\theta}_d$, $\boldsymbol{m}_d$ and $\boldsymbol{m}$ integrated out) is still given by equation 5.7.

Under this prior, the probability of new document d being assigned to cluster c is

$$P(c_d = c \mid \boldsymbol{c}_{<d}, \zeta) \propto \begin{cases} N_c & c \text{ is an existing cluster} \\ \zeta & c \text{ is a new cluster} \end{cases} \tag{5.27}$$

where $\boldsymbol{c}_{<d}$ is the set of previous cluster assignments, N_c is the number of documents already in existing cluster c. The difference between this equation and equation 5.13 is that this equation reserves probability mass for new clusters while 5.13 does not.

The evidence for $U = \{\beta \boldsymbol{n}, \alpha, \alpha_1, \alpha_0, \zeta\}$ is still given by

$$P(\boldsymbol{w} \mid U) = \sum_{\boldsymbol{z}, \boldsymbol{c}} P(\boldsymbol{w} \mid \boldsymbol{z}, U)\, P(\boldsymbol{z} \mid \boldsymbol{c}, U)\, P(\boldsymbol{c} \mid U), \tag{5.28}$$

where $P(\boldsymbol{w} \mid \boldsymbol{z}, U)$ and $P(\boldsymbol{z} \mid \boldsymbol{c}, U)$ are given by equations 5.15 and 5.16, respectively. However, the probability of cluster assignments $P(\boldsymbol{c} \mid U)$ is now given by

$$P(\boldsymbol{c} \mid U) = \prod_d P(c_d \mid \boldsymbol{c}_{<d}, \zeta) \tag{5.29}$$

$$= \frac{\zeta^C \prod_{c=1}^{C} (N_c - 1)!}{\prod_{d=1}^{D} \zeta + d - 1}, \tag{5.30}$$

where C is the number of currently active clusters, N_c is the number of documents currently in cluster c, and D is the total number of documents in the corpus. Note

that the order in which the documents are processed does not affect the form of equation 5.30—in other words, the cluster assignments are exchangeable.

The cluster and topic assignments ($\boldsymbol{c}$ and $\boldsymbol{z}$, respectively) for a collection of documents $\boldsymbol{w}$ may be inferred using Gibbs sampling. Since the cluster assignments are exchangeable, it is possible to rearrange the documents in any order without changing the probability of their cluster assignments. Given a set of topic assignments $\boldsymbol{z}$, the cluster assignment for document d may therefore be resampled by treating this document as the last to arrive, and drawing its cluster assignment c_d from

$$P(c_d \mid \boldsymbol{c}_{\setminus d}, \boldsymbol{z}, \zeta, \alpha, \alpha_1, \alpha_0) \propto P(c_d \mid \boldsymbol{c}_{\setminus d}, \zeta)\, P(\boldsymbol{z}_d \mid d, c_d, \boldsymbol{c}_{\setminus d}, \boldsymbol{z}_{\setminus d}, \alpha, \alpha_1, \alpha_0). \tag{5.31}$$

The vector $\boldsymbol{z}_d$ is the set of topic assignments for document d and $\boldsymbol{z}_{\setminus d}$ is the set of all other topic assignments. $P(c_d \mid \boldsymbol{c}_{\setminus d}, \zeta)$ can be obtained using equation 5.27, while $P(\boldsymbol{z}_d \mid c_d, \boldsymbol{c}_{\setminus c}, \boldsymbol{z}_{\setminus d}, \alpha, \alpha_1, \alpha_0)$ is the probability of adding $\boldsymbol{z}_d$ to cluster c, given the other documents currently belong to that cluster, and may be computed using

$$\prod_{\{n \mid d_n = d\}} P(z_n \mid d, c_d, (\boldsymbol{z}_d)_{<n}, \boldsymbol{z}_{\setminus d}, \boldsymbol{c}_{\setminus d}, \alpha, \alpha_1, \alpha_0). \tag{5.32}$$

Similarly, each topic assignment z_n can be sampled from

$$\begin{aligned} P(z_n \mid \boldsymbol{w}, \boldsymbol{z}_{\setminus n}, \boldsymbol{c}, \alpha, \alpha_1, \alpha_0, \beta\boldsymbol{n}) \propto \\ P(w_n \mid \boldsymbol{w}_{\setminus n}, \boldsymbol{z}, \beta\boldsymbol{n}) P(z_n \mid d_n, c_{d_n}, \boldsymbol{c}_{\setminus d_n}, \boldsymbol{z}_{\setminus n}, \alpha, \alpha_1, \alpha_0). \end{aligned} \tag{5.33}$$

The topic assignments $\boldsymbol{z}$ can be initialised using latent Dirichlet allocation.

Figure 5.3.1 depicts the full graphical model for the new cluster-based topic model with an unknown number of latent clusters, as introduced in this section.

5.3.2 Experiments

The cluster-based topic model (with an unknown number of clusters) was compared with two baseline models: Latent Dirichlet allocation (Blei et al., 2003) and a word-based Dirichlet process mixture model (depicted in figure 5.2). The latter differs from the cluster-based topic model in that instead of characterising each cluster by a distribution over topics, clusters are characterised by distributions over words $\{\boldsymbol{n}_c\}_{c=1}^{\infty}$. The words that comprise document d are drawn from a document-specific distribution over words $\boldsymbol{\phi}_d$, which is itself drawn from a Dirichlet distribution with base measure $\boldsymbol{n}_d$ and concentration parameter β. Each $\boldsymbol{n}_d$ is distributed as follows:

$$P(\boldsymbol{n}_d|G) = G(\boldsymbol{n}_d) \tag{5.34}$$

$$P(G \mid \zeta, G_0) = \mathrm{DP}\,(G \mid \zeta, G_0), \tag{5.35}$$

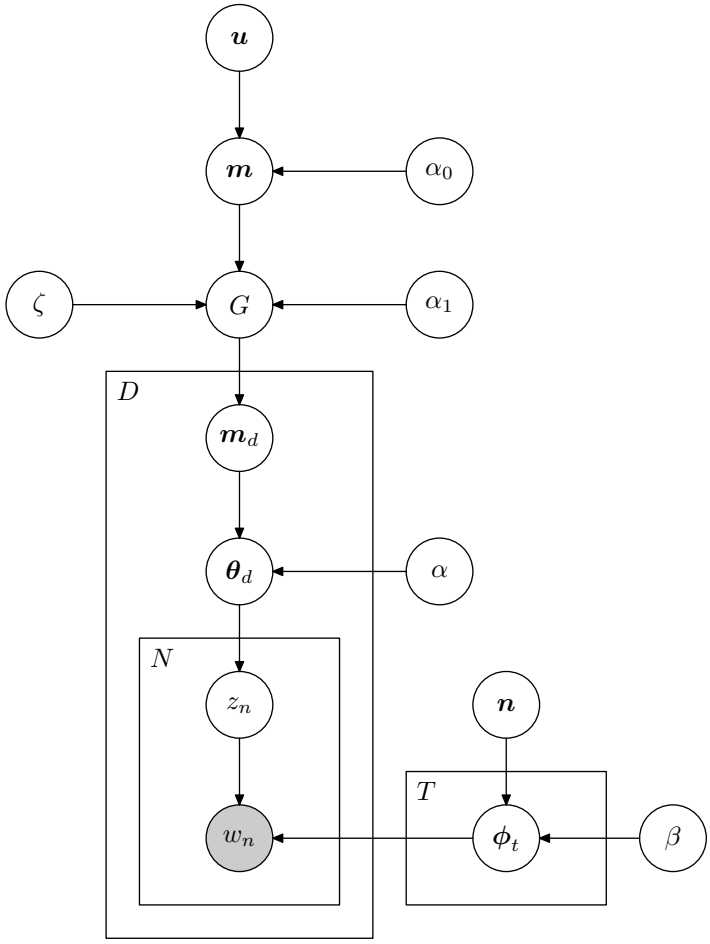


Figure 5.1: Graphical model for the cluster-based topic model with an unknown number of latent clusters. Observed variables (words $\boldsymbol{w}$) are shown in grey. The variables $\boldsymbol{m}$, $\boldsymbol{u}$, α_0 and α_1 comprise the Dirichlet process base measure G_0.

where G is is a random probability measure distributed according to a Dirichlet process with base measure G_0 and concentration parameter ζ. As a result,

$$G(\boldsymbol{n}_d) = \sum_{c=1}^{\infty} \pi_c \, \delta_{\boldsymbol{n}_c}(\boldsymbol{n}_d), \tag{5.36}$$

where $\delta_{\boldsymbol{n}_c}(\cdot)$ is a point mass located at $\boldsymbol{n}_c$ and

$$P(\boldsymbol{n}_c \,|\, G_0) = G_0(\boldsymbol{n}_c) \tag{5.37}$$

$$\pi_c = \pi'_c \prod_{k=1}^{c-1} (1 - \pi'_k) \tag{5.38}$$

$$P(\pi'_c \,|\, \zeta) = \text{Beta}\,(\pi'_c \,|\, 1, \zeta). \tag{5.39}$$

Base measure G_0 is chosen to be a hierarchical Dirichlet distribution:

$$G_0 = \text{Dir}\,(\boldsymbol{n}_c \,|\, \beta_1 \boldsymbol{n}), \tag{5.40}$$

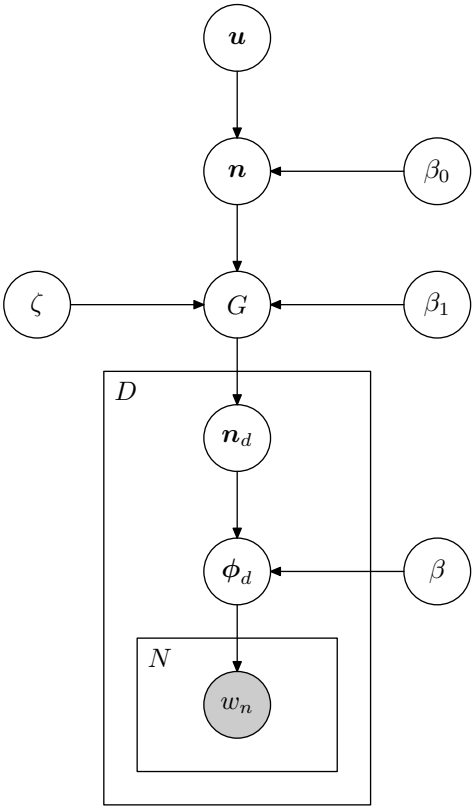


Figure 5.2: Word-based Dirichlet process mixture model. Words $\boldsymbol{w}$ are observed. Variables $\boldsymbol{n}$, $\boldsymbol{u}$, β_1 and β_0 comprise the Dirichlet process base measure G_0.

where $\boldsymbol{n}$ is itself drawn from

$$P(\boldsymbol{n} \mid \beta_0 \boldsymbol{u}) = \text{Dir}\,(\boldsymbol{n} \mid \beta_0 \boldsymbol{u}). \tag{5.41}$$

Given a set of documents $\boldsymbol{w}$, latent cluster assignments $\boldsymbol{c}$ may be inferred using Gibbs sampling. The cluster assignment c_d for document d is resampled from

$$P(c_d \mid \boldsymbol{c}_{\setminus d}, \boldsymbol{w}, \zeta, \beta, \beta_1, \beta_0) \propto P(c_d \mid \boldsymbol{c}_{\setminus d}, \zeta)\, P(\boldsymbol{w}_d \mid c_d, \boldsymbol{c}_{\setminus d}, \boldsymbol{w}_{\setminus d}, \beta, \beta_1, \beta_0), \tag{5.42}$$

where

$$P(c_d \mid \boldsymbol{c}_{\setminus d}, \zeta) \propto \begin{cases} N_{c_d} & c_d \text{ is an existing cluster} \\ \zeta & c_d \text{ is a new cluster} \end{cases} \tag{5.43}$$

and

$$P(\boldsymbol{w}_d \mid c_d, \boldsymbol{c}_{\setminus d}, \boldsymbol{w}_{\setminus d}, \beta, \beta_1, \beta_0) = \prod_{\{n \mid d_n = d\}} P(w_n \mid d, c_d, (\boldsymbol{w}_d)_{<n}, \boldsymbol{w}_{\setminus d}, \boldsymbol{c}_{\setminus d}, \beta, \beta_1, \beta_0). \tag{5.44}$$

The probability $P(w_n \mid d, c_d, (\boldsymbol{w}_d)_{<n}, \boldsymbol{w}_{\backslash d}, \boldsymbol{c}_{\backslash d}, \beta, \beta_1, \beta_0)$ may be computed in a similar fashion to $P(z_n \mid d, c_d, (\boldsymbol{z}_d)_{<n}, \boldsymbol{z}_{\backslash d}, \boldsymbol{c}_{\backslash d}, \beta, \beta_1, \beta_0)$, as shown in equation 5.7. This model captures the fact that documents from different groups or clusters are likely to use different vocabularies. It does not, however, capture the fact that there may be slight variations in vocabulary between documents within a single group or cluster.

Twenty years of proceedings from the NIPS conference[1] were used to compare the models. Papers from 1987–2003 (2,325 papers in total) were treated as training data, while papers from 2004–2006 (614 papers in total) were treated as test data.

All words that occurred exactly once in the training data (and zero times in the test data) or one or more times in the test data, and not at all in the training data, were removed and replaced with one of the following UNSEEN types (Eisner, 1996a):

- UNSEEN-SHORT: used for words less than six characters long.
- UNSEEN-NUM: used for words whose last character is a digit.
- UNSEEN-PUNC: used for words consisting entirely of punctuation characters.
- UNSEEN-XX: used for words of six or more characters in length. XX is replaced with the last two characters of the word.

Words that appeared on a standard list of stop words[2] were also removed. Finally, the first seventy words of each paper were discarded so as to avoid modelling paper titles, author names, affiliations and addresses. Additionally, to improve efficiency each paper was truncated to 180 tokens (roughly the length of a paper abstract). Preprocessing each paper in this fashion resulted in a training data set consisting of 580,983 tokens, test data set consisting of 153,500 tokens, and a vocabulary of 16,376 words,

Training topics for latent Dirichlet allocation were obtained by running a Gibbs sampler for 1,000 iterations. After each iteration, five iterations of slice sampling were used to update the model hyperparameters. For the word-based mixture model, training cluster assignments were obtained using 500 Gibbs sampling iterations. After each iteration, the hyperparameter ζ for the Dirichlet process prior over clusters as well as β, β_1 and β_2 were slice-sampled for five iterations. Cluster assignments and hyperparameters for the new cluster-based topic model were sampled similarly. Topic assignments were initialised using latent Dirichlet allocation and resampled after every iteration. The number of topics was set to fifty for all experiments involving topics.

Experiments were run using both the minimal and maximal path assumptions (described in section 3.4.2), however results are only presented for the minimal path assumption. The maximal path assumption resulted in poor performance for both latent Dirichlet allocation and the new cluster-based topic model, both in terms of the

[1] Data from 1987–2003 were provided by Sam Roweis and Gal Chechik. Data from 2004–1006 were obtained from `http://books.nips.cc/` and converted to plain text using `pdftotext`.

[2] `http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/stop_words`

probability assigned to training and test data, as well as the interpretability of topics and clusters. In latent Dirichlet allocation, the quantity $\hat{N}_{t|d}/\hat{N}_{\cdot|d}$ is "smoothed" with $\hat{N}_t/\hat{N}_\cdot$, where $\hat{N}_t$ is either equal to the number of times that topic t has been used in the entire corpus (under the maximal path assumption) or the number of different documents in which t has previously been seen (under the minimal path assumption). An important difference between these two quantities is that the contributions to $\hat{N}_t$ from each document are unequal under the maximal path assumption—longer documents contribute more than shorter documents. Hence, under the maximal path assumption, a topic that occurs many times in a long document is more likely to be used in a new document than a topic that occurs many times in a short document. This is undesirable—that a topic was used in a longer document rather than a shorter one should not influence the probability of that topic being used in future documents. In addition to this, there are some topics that occur in almost every document, but are used only a few times in each one. For example, in the case of NIPS papers, the topic "methods approach based method problem..." occurs in the first 250 words of most papers, but is used fewer times than other topics relating to the specific models and applications presented in the paper. Under the maximal path assumption, this topic will have a lower probability of occurring in a new document than a topic such as "neurons neuron spike synaptic firing...", which occurs in fewer papers, but many more times in the papers in which it does occur. Again, this is undesirable—a topic that is used in many papers should have a higher probability of occurring in a new paper, regardless of the number of times it is used in each one. These differences between the minimal and maximal path assumptions have not been noted in previous treatments of latent Dirichlet allocation and other topic-based models, most likely because these treatments have used nonhierarchical Dirichlet priors over the document-specific topic distributions (both with and without hyperparameter optimisation).

The models were evaluated by computing the information rate of unseen test data $\boldsymbol{w}$, measured in bits per word: The fewer the bits per word, the better the model. The information rate of $\boldsymbol{w}$ given training data $\boldsymbol{w}^{\text{train}}$ and hyperparameters U is

$$R = -\frac{\log_2 P(\boldsymbol{w} \mid \boldsymbol{w}^{\text{train}}, U)}{N_\cdot}, \tag{5.45}$$

where $N_\cdot$ is the number of tokens in test data $\boldsymbol{w}$. For all three models, computing $P(\boldsymbol{w} \mid \boldsymbol{w}^{\text{train}}, U)$ involves an intractable sum over latent variables—topic assignments for latent Dirichlet allocation, cluster assignments for the word-based mixture model, and both topic and cluster assignments for the new cluster-based topic model. These sums were approximated using the importance sampling approximation (Kass and Raftery, 1995), described in 3.5.2. For latent Dirichlet allocation, 200 sets of topic assignments were used, taken every fifty iterations after a burn-in period of 1,000 iterations. For the word-based Dirichlet process mixture model, 200 sets of cluster assignments were used, taken every twenty-five iterations after 100 burn-in iterations. For

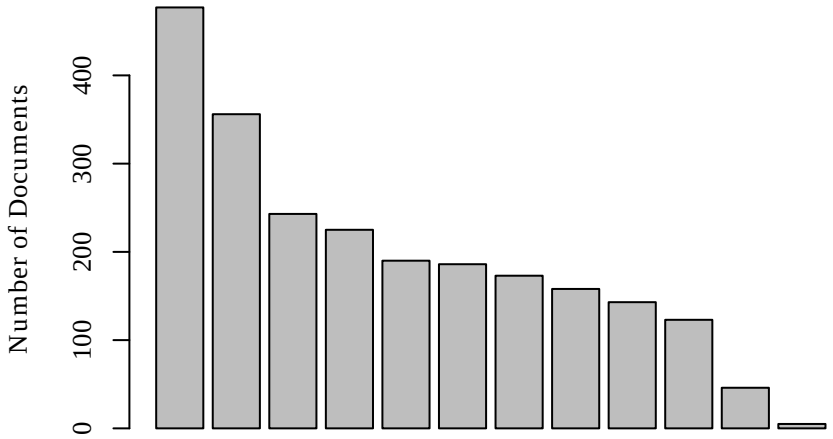


Figure 5.3: Number of training documents assigned to each of the twelve clusters inferred by the word-based Dirichlet process mixture model baseline.

the cluster-based topic model, 200 sets of topic and cluster assignments were used, again taken every twenty-five iterations after a burn-in period of 100 iterations.

The word-based mixture model exhibits much worse performance than the other two models, with an information rate of 10.54 bits per word. Latent Dirichlet allocation exhibits significantly better performance, of 8.38 bits per word. However, the cluster-based topic model achieves the best performance, of 8.33 bits per word. The cluster-based topic model is therefore a much better document model than the word-based mixture model and a slightly better document model than latent Dirichlet allocation.

It is also instructive to examine the clusters inferred by the two cluster-based models. The word-based mixture model baseline inferred twelve clusters, while the cluster-based topic model inferred seven. The cluster sizes for each of the models are shown in figures 5.3 and 5.4. The top ten words generated by each of the twelve clusters inferred by the word-based mixture model are given in table 5.1. Clusters 4, 6, 10 and 12 each contain several words that appear in the top ten words for only one cluster. It is therefore relatively easy to determine what sorts of documents belong to these clusters. Other clusters consist of more general words, such as “function”, “learning” and “introduction”, common to many clusters. Two pairs of clusters have almost identical top words: Clusters 2 and 8 contain words related to general machine learning concepts, while clusters 1 and 9 contain words roughly related to neural networks.

In contrast, the clusters inferred by the new cluster-based topic model are easier to interpret. While there are some general topics that appear in every cluster, all but one of the clusters use at least two specialised topics with high probability, thereby making it easy to determine the types of papers associated with each cluster. Figure 5.5 depicts

Cluster 1 (477 Documents)	Cluster 2 (356 Documents)	Cluster 3 (243 Documents)	Cluster 4 (225 Documents)
function	data	network	**neurons**
learning	set	neural	model
set	introduction	networks	introduction
paper	paper	function	neural
network	problem	introduction	**neuron**
model	algorithm	**number**	**activity**
networks	function	**output**	network
introduction	learning	paper	**information**
problem	space	learning	function
neural	training	system	**synaptic**
Cluster 5 (190 Documents)	Cluster 6 (186 Documents)	Cluster 7 (173 Documents)	Cluster 8 (158 Documents)
learning	**visual**	introduction	data
problem	model	model	set
function	introduction	**images**	introduction
paper	system	problem	paper
algorithm	**cells**	based	problem
based	figure	paper	algorithm
introduction	**processing**	set	function
reinforcement	neural	data	learning
problems	**cortex**	approach	space
control	**spatial**	**object**	training
Cluster 9 (143 Documents)	Cluster 10 (123 Documents)	Cluster 11 (46 Documents)	Cluster 12 (5 Documents)
learning	introduction	introduction	learning
network	model	system	set
introduction	**analysis**	model	based
set	**linear**	control	**feedback**
paper	data	**motor**	**individual**
problem	paper	learning	**phase**
networks	**component**	figure	**brain**
model	learning	**shown**	**nucleus**
neural	approach	**movements**	**auditory**
based	**vector**	**movement**	**innate**

Table 5.1: The most frequently used words for the clusters inferred by the word-based mixture model baseline. Clusters are ordered by decreasing size. Words that occur in the top ten words for a single cluster only are highlighted in bold.

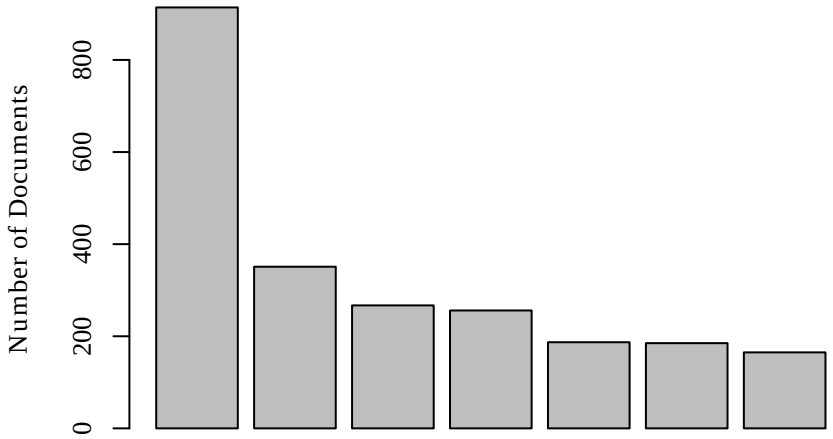
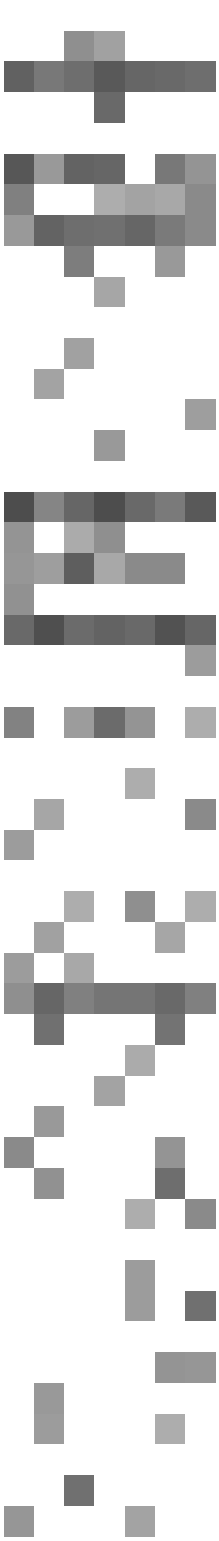


Figure 5.4: Cluster sizes (number of training documents assigned to each cluster) for each of the seven clusters inferred by the cluster-based topic model.

the top fifteen topics used by each cluster, as well as the top five words for each topic. This figure makes it clear that the topics that appear in all clusters are general scientific topics (e.g., "model, models, properties, study, results..."). Topics that appear in only one cluster are much more specific (e.g., "language, words, word, grammar, string...").

Table 5.3 shows the most frequently used topics for each of the seven clusters. Each topic is represented by the ten most probable words for that topic. Topics that occur in the top fifteen topics for all clusters are not shown. Clusters 1 and 2 are the largest clusters, with 914 and 351 documents, respectively. It is evident from looking at tables 5.3a and 5.3b that cluster 1 contains documents about machine learning, while cluster 2 contains documents about neuroscience. This reflects the most prominent dichotomy in the NIPS community: Machine learning versus neuroscience. Cluster 3 (table 5.3c) contains many topics related to neural networks, another research area well-represented at NIPS. Meanwhile, cluster 4 (table 5.3d) contains topics to do with reinforcement learning. Clusters 5 and 6 (shown in shown in tables 5.3e and 5.3f) are slightly less well-defined. Cluster 5 contains several topics that initially seem quite different: Neural networks, speech recognition, object and image recognition, language, face detection. However, these topics are all typical for application papers presented at NIPS. All of the most frequently used topics for cluster 6 are also used by other clusters. Nonetheless, it seems that cluster 6 contains documents about using neural networks and probabilistic models to model concepts from neuroscience. Finally, cluster 7 (shown in table 5.3g), contains several image- and vision-related topics.



bounds bound loss functions error
learning algorithm gradient weight error
number large performance high data
policy action reinforcement learning actions
routing control cost traffic problem
function functions case set paper
model data models parameters likelihood
system systems neural processing computer
memory network dynamics neuron neurons
robot environment modules sensor sensors
eeg brain activity data neural
analog circuit chip circuits vlsi
phase sensory neurons receptor activity
independent source basis component ica
control motor model system trajectory
color light optical reflectance red
methods approach based method problem
problem solution optimization problems constraints
network neural networks architecture output
training data error set selection
model models properties study results
data space clustering points distance
function local functions basis experts
learning task learn problem examples
character recognition characters segmentation game
language words word grammar string
motion visual local scene direction
matrix kernel linear space vector
vector vectors coding codes decoding
representations level representation rules structure
signal auditory signals frequency noise
training learning output generalization error
figure shown shows left single
cells cell visual neurons response
user query data ranking users
sequence prediction sequences series nonlinear
eye visual system movements position
distribution gaussian probability bayesian prior
neurons neuron spike synaptic firing
images face faces detection video
tree graph node nodes trees
speech recognition word hmm system
object objects features recognition feature
music rat place song direction
information entropy statistical optimal mutual
cortex connections cortical orientation activity
stimulus stimuli response attention task
distribution energy field boltzmann approximation
units hidden network layer unit
classification class training classifier classifiers

Figure 5.5: The top fifteen topics used in each cluster and the top five words for each topic. Each cluster is a single column. Clusters are ordered from largest to smallest (left to right). Each square is a single topic/cluster pair: Intensity indicates how common that topic in that cluster (darker means more common).

5.4 Incorporating Author Information

Although the clusters inferred by the new cluster-based topic model correspond well to research areas within the NIPS community, true research areas are characterised by groups of researchers as well as topics. Author information—the identities of the authors responsible for each document—can be incorporated into the model by associating each document d with *two* cluster-specific distributions, one ($\boldsymbol{m}_d$) over topics and one ($\boldsymbol{q}_d$) over authors: $(\boldsymbol{m}_d, \boldsymbol{q}_d) \in \{(\boldsymbol{m}_c, \boldsymbol{q}_c)\}_{c=1}^{\infty}$, distributed according to

$$P(\boldsymbol{m}_d, \boldsymbol{q}_d \,|\, G) = G(\boldsymbol{m}_d, \boldsymbol{q}_d) \tag{5.46}$$

$$P(G \,|\, \zeta, G_0) = \text{DP}\,(G \,|\, \zeta, G_0), \tag{5.47}$$

where G is is a random probability measure distributed according to a Dirichlet process with base measure G_0 and concentration parameter ζ. This means

$$G(\boldsymbol{m}_d, \boldsymbol{q}_d) = \sum_{c=1}^{\infty} \pi_c \, \delta_{\boldsymbol{m}_c, \boldsymbol{q}_c}(\boldsymbol{m}_d, \boldsymbol{q}_d) \tag{5.48}$$

$$P(\boldsymbol{m}_c, \boldsymbol{q}_c \,|\, G_0) = G_0(\boldsymbol{m}_c, \boldsymbol{q}_c) \tag{5.49}$$

$$\pi_c = \pi'_c \prod_{k=1}^{c-1} (1 - \pi'_k) \tag{5.50}$$

$$P(\pi'_c \,|\, \zeta) = \text{Beta}\,(\pi'_c \,|\, 1, \zeta). \tag{5.51}$$

Since each draw from G_0 must be a pair of probability vectors ($\boldsymbol{m}_c$, $\boldsymbol{q}_c$), G_0 is defined as the product of two hierarchical Dirichlet distributions, as follows:

$$G_0 = \text{Dir}\,(\boldsymbol{m}_c \,|\, \alpha_1 \boldsymbol{m}) \, \text{Dir}\,(\boldsymbol{q}_c \,|\, \eta_1 \boldsymbol{q}), \tag{5.52}$$

where

$$P(\boldsymbol{m} \,|\, \alpha_0 \boldsymbol{u}) = \text{Dir}\,(\boldsymbol{m} \,|\, \alpha_0 \boldsymbol{u}), \tag{5.53}$$

$$P(\boldsymbol{q} \,|\, \eta_0 \boldsymbol{u}) = \text{Dir}\,(\boldsymbol{q} \,|\, \eta_0 \boldsymbol{v}), \tag{5.54}$$

and $\boldsymbol{u}$ and $\boldsymbol{v}$ are uniform distributions over topics and authors, respectively. This prior means that authors are conditionally independent of topics given the clusters.

For each document d, the authors $\boldsymbol{a}_d$ responsible for that document are assumed to have been drawn directly from $\boldsymbol{q}_d$, the cluster-specific author distribution. The predictive probability of author a in document d is therefore

$$P(a \,|\, d, c_d, \boldsymbol{a}, \boldsymbol{c}, \eta_1, \eta_0) = \frac{\hat{N}_{a|c_d} + \eta_1 \, \dfrac{\hat{N}_a + \eta_0 v_a}{\hat{N}_{\cdot} + \eta_0}}{\hat{N}_{\cdot|c_d} + \eta_1}, \tag{5.55}$$

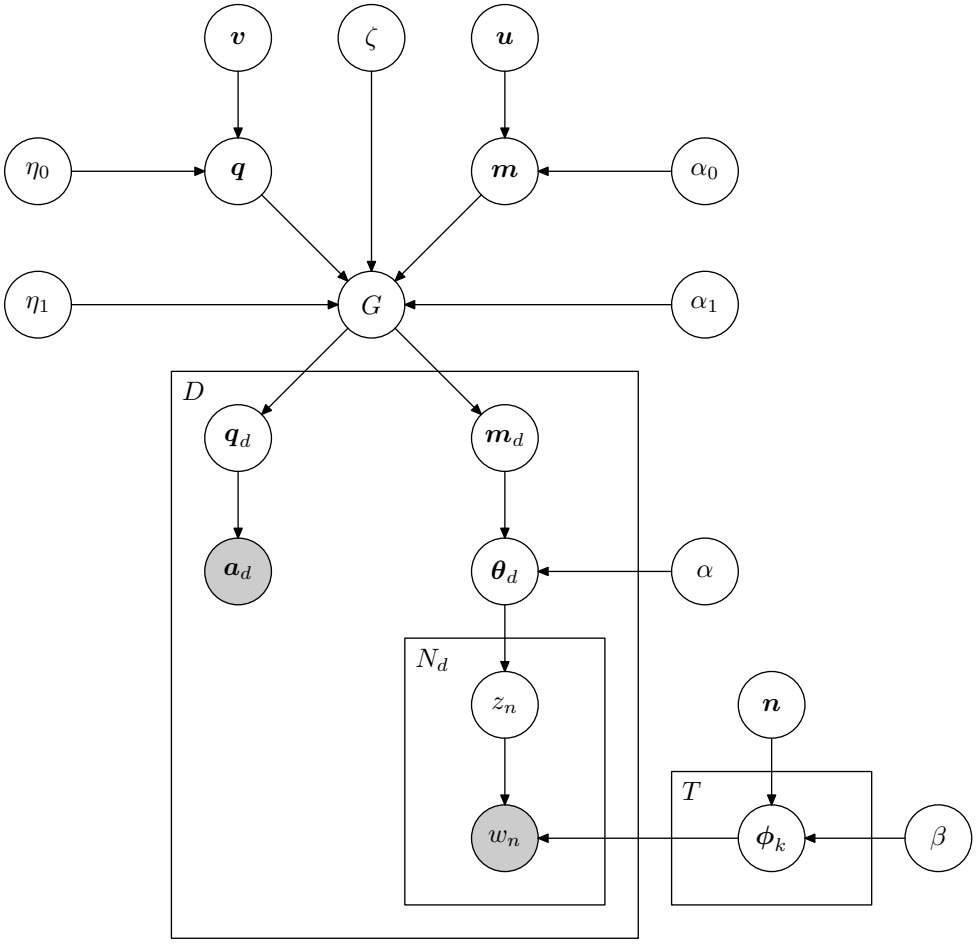


Figure 5.6: The cluster-based author–topic model (an extension of the cluster-based topic model in section 5.3.1). Words $\boldsymbol{w}$ and authors $\boldsymbol{a}$ are observed. Variables $\boldsymbol{m}$, $\boldsymbol{u}$, α_1, α_0, $\boldsymbol{q}$, $\boldsymbol{v}$, η_1, η_0 comprise the Dirichlet process base measure G_0.

where the quantities $\hat{N}_{a|c_d}$ and $\hat{N}_a$ are given by

$$\hat{N}_{a|c_d} = \sum_{l=1}^{L} N_{\cdot|c_d}^{(l)} \, \delta\left(\gamma_l - a\right), \tag{5.56}$$

$$\hat{N}_a = \sum_{m=1}^{M} N_{\cdot}^{(m)} \, \delta\left(\gamma_m - a\right), \tag{5.57}$$

and L and M are the current numbers of internal draws from the bottom- and top-level (i.e., group- and corpus-level) Dirichlet-multinomials. The quantity $N_{\cdot|c_d}^{(l)}$ is the number of author observations currently matched to bottom-level internal draw γ_l, while $N_{\cdot}^{(m)}$ is the number of bottom-level internal draws currently matched to top-level internal draw γ_m. Under the maximal path assumption (section 3.4.2) $\hat{N}_a = \sum_m N_{\cdot}^{(m)} \, \delta\left(\gamma_m - a\right)$ is equal to N_a, the number of times author a has occurred in the corpus. Under the minimal path assumption it is equal to the number of different clusters in which a has appeared. The bottom-level quantity $\hat{N}_{a|c_d} = \sum_l N_{\cdot|c_d}^{(l)} \, \delta\left(\gamma_l - a\right)$ is always equal to the number of times author a has been seen in cluster c_d.

The complete model is referred to henceforth as the *cluster-based author–topic* model, and is shown in figure 5.6. This model is related to the author–topic model of Rosen-Zvi et al. (2004) (see also Steyvers et al. (2004)), but captures the notion of document clusters or groups, unlike Rosen-Zvi et al.'s model. The models also differ in other ways: In the cluster-based author–topic model, each document is generated by first selecting a cluster for that document. Having done this, authors are drawn from a cluster-specific author distribution, while topics are drawn from document- and cluster-specific distributions over words. Authors and topics are independent, given the cluster assignments. Finally, words are drawn from topic-specific word distributions. In Rosen-Zvi et al.'s author–topic model, authors are used as conditioning context—they are not generated by the model. Each word is generated by first selecting an author, uniformly from the set of authors for that document, and then drawing a topic assignment from the corresponding author-specific topic distribution. Having done this, the word is drawn from a topic-specific word distribution. There are no document-specific distributions over topics—all document-specific properties must instead be captured by particular combinations of author-specific topic distributions.

Given observed data $\boldsymbol{w}$ and $\boldsymbol{a}$, and concentration parameters $U = \{\zeta, \alpha, \alpha_1, \alpha_0, \eta_1, \eta_0\}$, the latent topics $\boldsymbol{z}$ and clusters $\boldsymbol{c}$ can be inferred by alternating between sampling topics given the current cluster assignments and clusters given the current topic assignments. The latter is done by assigning document d to cluster c with probability

$$\begin{aligned} &P(c_d = c \,|\, \boldsymbol{c}_{\setminus d}, \boldsymbol{z}, \boldsymbol{a}, U) \propto \\ &\quad P(c_d = c \,|\, \boldsymbol{c}_{\setminus d}, U)\, P(\boldsymbol{z}_d \,|\, d, c_d, \boldsymbol{c}_{\setminus d}, \boldsymbol{z}_{\setminus d}, U)\, P(\boldsymbol{a}_d \,|\, d, c_d, \boldsymbol{c}_{\setminus d}, \boldsymbol{a}_{\setminus d}, U). \end{aligned} \tag{5.58}$$

The first and second terms on the right-hand side are given by equations 5.27 and 5.32 respectively, while $P(\boldsymbol{a}_d \,|\, d, c_d, \boldsymbol{c}_{\setminus d}, \boldsymbol{a}_{\setminus d}, U)$ may be computed as follows:

$$\prod_{\{n \,|\, d_n = d\}} P(a_n \,|\, d, c_d, (\boldsymbol{a}_d)_{<n}, \boldsymbol{a}_{\setminus d}, \boldsymbol{c}_{\setminus d}, U). \tag{5.59}$$

5.4.1 Experiments

The cluster-based author–topic model was evaluated using the data and experimental set-up described in section 5.3.2. As with the cluster-based topic model, fifty topics were used, along with the minimal path assumption. Training cluster and topic assignments were obtained using 500 Gibbs sampling iterations, while the information rate of unseen test data was approximated using 200 sets of test topic and cluster assignments, taken every twenty five iterations after a burn-in period of 100 iterations.

The information rate achieved by the cluster-based author–topic model is 8.28 bits per word. This rate is better than the information rate obtained by either the cluster-based topic model (8.33 bits per word) or latent Dirichlet allocation (8.38 bits per word).

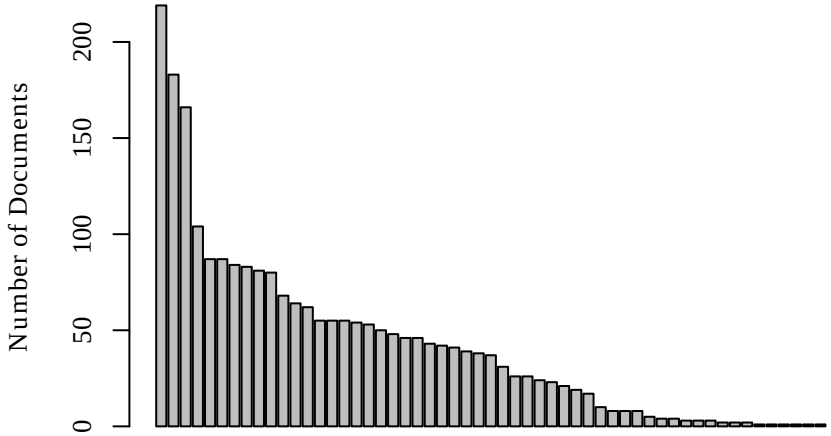


Figure 5.7: Cluster sizes (number of training documents assigned to each cluster) for the fifty-five clusters inferred by the cluster-based author–topic model.

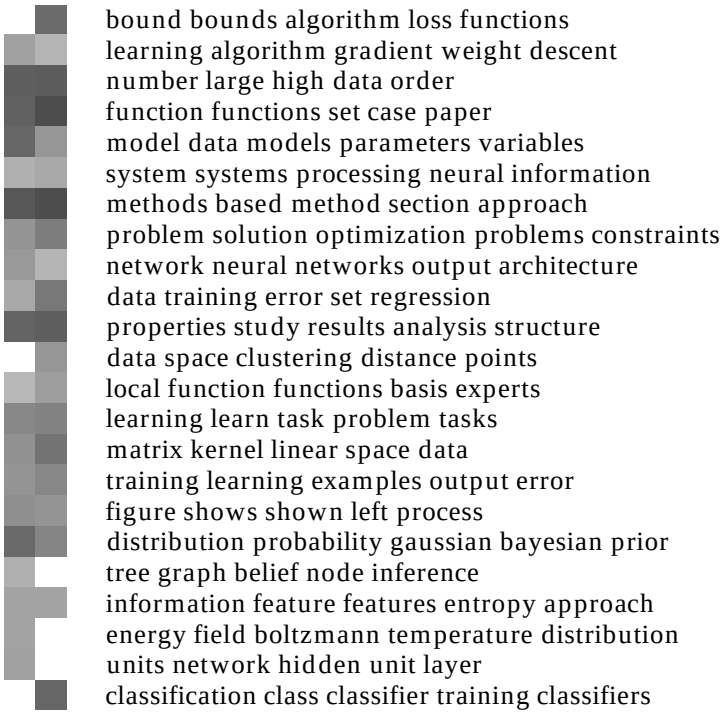


Figure 5.8: The top twenty topics for the two most frequently used clusters, along with the top five words for each topics. Each cluster is a single column (clusters 1 and 2, left to right). Each square represents a single topic/cluster pair: Intensity indicates how common that topic is in that cluster (darker means more common).

These results indicate that the inclusion of author information enables the model to identify cluster and topic assignments that better account for document content.

Cluster sizes for the cluster-based author–topic model are shown in figure 5.7. The

Cluster 1 (219 Documents)	Cluster 2 (183 Documents)
G. Hinton	J. Shawe-Taylor
M. Jordan	A. Smola
C. Williams	B. Scholkopf
Z. Ghahramani	P. Bartlett
M. Opper	R. Williamson
T. Jaakkola	V. Vapnik
D. Saad	Y. Bengio
P. Sollich	R. Herbrich
C. Bishop	T. Graepel
D. Barber	J. Weston

Table 5.2: The ten authors most frequently associated with each of the two largest clusters (see also figure 5.8) inferred by the cluster-based author–topic model.

number of clusters inferred by the model (fifty-five) is significantly higher than the number of clusters inferred by the cluster-based topic model (seven). This is unsurprising: Papers that use similar topics, but are written by different groups of people are unlikely to be placed in the same cluster. This effect is most pronounced for the two largest clusters, inferred by the cluster-based author–topic model. Figure 5.8 shows the top twenty topics used in each of these clusters. Seventeen of the topics are used in both clusters. Both clusters clearly contain papers about machine learning. However the authors most frequently associated with each of these clusters are quite different (figure 5.2). The top authors for cluster 1 are all well-known for research on graphical models, neural networks and Gaussian processes. In contrast, several of the top authors for cluster 2 are known for research on learning theory and support vector machines. The six topics that appear in only one of the clusters reflect this difference. Additionally, many of the top authors for cluster 1 are either currently at or have previously been at universities in the United Kingdom. This reflects the fact that authors at geographically close institutions are more likely to have co-authored papers together.

5.5 Conclusions

In this chapter, I introduced a nonparametric Bayesian model for clustering documents by topic. The model was evaluated using academic papers from the NIPS conference, and was found to assign a higher log probability to unseen test data than either a word-based clustering model or latent Dirichlet allocation. In addition to this, the clusters inferred by the model represent well-known research areas in the NIPS community, and provide a concise representation of the relationships between topics. I also showed how author information can be incorporated into the model, resulting in finer-grained clusters. Finally, I determined that it is necessary to use the minimal path assumption (or Gibbs sampling) when inferring counts for hierarchical Dirichlet

distributions over topics in latent Dirichlet allocation and related models—the maximal path assumption results in poor performance as well as topics that are difficult to interpret. Previous treatments of latent Dirichlet allocation have used nonhierarchical Dirichlet priors over the document-specific topic distributions (with and without hyperparameter optimisation) and have therefore not encountered this issue.

function	model	learning	distribution	**training**
functions	data	task	gaussian	**data**
case	models	learn	probability	**error**
set	parameters	problem	bayesian	**set**
paper	likelihood	examples	prior	**selection**
section	mixture	algorithm	noise	**risk**
defined	variables	set	posterior	**regression**
assume	density	learned	random	**regularisation**
vector	probability	training	density	**generalisation**
general	estimation	tasks	estimate	**parameters**
problem	network	classification	training	**matrix**
solution	neural	class	learning	**kernel**
optimisation	networks	training	output	**linear**
problems	architecture	classifier	generalisation	**space**
constraints	output	classifiers	error	**vector**
function	weights	data	examples	**data**
point	feedforward	classes	inputs	**feature**
solutions	trained	decision	number	**dimensional**
constraint	recurrent	set	set	**pca**
objective	training	pattern	weights	**kernels**

(a) Cluster 1 (914 documents).

Table 5.3: The most frequently used topics for the each of the clusters inferred by the cluster-based topic model. Topics that occur in the top fifteen topics for every cluster are not shown, while those that appear in the top fifteen for a single cluster only are highlighted in bold. Each topic is represented by its top ten words.

cells	neurons	**eye**	**cortex**	function
cell	neuron	**visual**	**connections**	functions
visual	spike	**system**	**cortical**	case
neurons	synaptic	**movements**	**orientation**	set
response	firing	**position**	**activity**	paper
stimulus	spikes	**velocity**	**layer**	section
receptive	membrane	**vor**	**lateral**	defined
field	potential	**model**	**development**	assume
responses	model	**target**	**dominance**	vector
cortex	neuronal	**retina**	**patterns**	general
stimulus	network	signal	**phase**	motion
stimuli	neural	auditory	**sensory**	visual
response	networks	signals	**neurons**	local
attention	architecture	frequency	**receptor**	scene
task	output	noise	**activity**	direction
visual	weights	sound	**olfactory**	field
subjects	feedforward	processing	**oscillatory**	surface
human	trained	source	**oscillators**	contour
information	recurrent	sounds	**binding**	vision
trial	training	system	**inhibitory**	figure

(b) Cluster 2 (351 documents).

Table 5.3: The most frequently used topics for the each of the clusters inferred by the cluster-based topic model. Topics that occur in the top fifteen topics for every cluster are not shown, while those that appear in the top fifteen for a single cluster only are highlighted in bold. Each topic is represented by its top ten words.

network	function	**units**	memory	learning
neural	functions	**hidden**	network	algorithm
networks	case	**network**	dynamics	gradient
architecture	set	**layer**	neuron	weight
output	paper	**unit**	neurons	error
weights	section	**output**	networks	descent
feedforward	defined	**weights**	associative	function
trained	assume	**activation**	model	convergence
recurrent	vector	**networks**	hopfield	algorithms
training	general	**net**	patterns	stochastic
learning	**analog**	training	problem	representations
task	**circuit**	learning	solution	level
learn	**chip**	output	optimisation	representation
problem	**circuits**	generalisation	problems	rules
examples	**vlsi**	error	constraints	structure
algorithm	**digital**	examples	function	knowledge
set	**hardware**	inputs	point	connectionist
learned	**implementation**	number	solutions	structures
training	**output**	set	constraint	rule
tasks	**silicon**	weights	objective	hierarchical

(c) Cluster 3 (267 documents).

Table 5.3: The most frequently used topics for the each of the clusters inferred by the cluster-based topic model. Topics that occur in the top fifteen topics for every cluster are not shown, while those that appear in the top fifteen for a single cluster only are highlighted in bold. Each topic is represented by its top ten words.

function	**policy**	learning	problem	**control**
functions	**action**	task	solution	**motor**
case	**reinforcement**	learn	optimisation	**model**
set	**learning**	problem	problems	**system**
paper	**actions**	examples	constraints	**trajectory**
section	**optimal**	algorithm	function	**controller**
defined	**agent**	set	point	**feedback**
assume	**states**	learned	solution	**movement**
vector	**reward**	training	constraint	**arm**
general	**decision**	tasks	objective	**dynamics**
learning	**sequence**	**robot**	network	model
algorithm	**prediction**	**environment**	neural	data
gradient	**sequences**	**modules**	networks	models
weight	**series**	**sensor**	architecture	parameters
error	**nonlinear**	**sensors**	output	likelihood
descent	**model**	**information**	weights	mixture
function	**models**	**module**	feedforward	variables
convergence	**linear**	**navigation**	trained	density
algorithms	**states**	**task**	recurrent	probability
stochastic	**filter**	**spatial**	training	estimation

(d) Cluster 4 (256 documents).

Table 5.3: The most frequently used topics for the each of the clusters inferred by the cluster-based topic model. Topics that occur in the top fifteen topics for every cluster are not shown, while those that appear in the top fifteen for a single cluster only are highlighted in bold. Each topic is represented by its top ten words.

network	representations	learning	**speech**	object
neural	level	task	**recognition**	objects
networks	representation	learn	**word**	features
architecture	rules	problem	**hmm**	recognition
output	structure	examples	**system**	feature
weights	knowledge	algorithm	**speaker**	images
feedforward	connectionist	set	**acoustic**	transformations
trained	structures	learned	**probabilities**	invariant
recurrent	rule	training	**training**	pattern
training	hierarchical	tasks	**performance**	transformation
classification	model	**user**	**language**	images
class	data	**query**	**words**	face
training	models	**data**	**word**	faces
classifier	parameters	**ranking**	**grammar**	detection
classifiers	likelihood	**users**	**string**	video
data	mixture	**program**	**finite**	human
classes	variables	**queries**	**languages**	facial
decision	density	**processors**	**strings**	resolution
set	probability	**machine**	**symbol**	recognition
pattern	estimation	**parallel**	**context**	low

(e) Cluster 5 (187 documents).

Table 5.3: The most frequently used topics for the each of the clusters inferred by the cluster-based topic model. Topics that occur in the top fifteen topics for every cluster are not shown, while those that appear in the top fifteen for a single cluster only are highlighted in bold. Each topic is represented by its top ten words.

neurons	cells	function	network	information
neuron	cell	functions	neural	entropy
spike	visual	case	networks	statistical
synaptic	neurons	set	architecture	optimal
firing	response	paper	output	mutual
spikes	stimulus	section	weights	output
membrane	receptive	defined	feedforward	measure
potential	field	assume	trained	statistics
model	responses	vector	recurrent	principle
neuronal	cortex	general	training	distribution
distribution	memory	signal	model	stimulus
gaussian	network	auditory	data	stimuli
probability	dynamics	signals	models	response
bayesian	neuron	frequency	parameters	attention
prior	neurons	noise	likelihood	task
noise	networks	sound	mixture	visual
posterior	associative	processing	variables	subjects
random	model	source	density	human
density	hopfield	sounds	probability	information
estimate	patterns	system	estimation	trial

(f) Cluster 6 (185 documents).

Table 5.3: The most frequently used topics for the each of the clusters inferred by the cluster-based topic model. Topics that occur in the top fifteen topics for every cluster are not shown, while those that appear in the top fifteen for a single cluster only are highlighted in bold. Each topic is represented by its top ten words.

object	image	model	motion	function
objects	face	data	visual	functions
features	faces	models	local	case
recognition	detection	parameters	scene	set
feature	video	likelihood	direction	paper
images	human	mixture	field	section
transformations	facial	variables	surface	defined
invariant	resolution	density	contour	assume
pattern	recognition	probability	vision	vector
transformation	low	estimation	figure	general
information	**data**	**independent**	representations	learning
entropy	**space**	**source**	level	task
statistical	**clustering**	**basis**	representation	learn
optimal	**points**	**component**	rules	problem
mutual	**distance**	**ica**	structure	examples
output	**dimensional**	**components**	knowledge	algorithm
measure	**clusters**	**data**	connectionist	set
statistics	**similarity**	**sources**	structures	learned
principle	**cluster**	**analysis**	rule	training
distribution	**algorithm**	**linear**	hierarchical	tasks

(g) Cluster 7 (165 documents).

Table 5.3: The most frequently used topics for the each of the clusters inferred by the cluster-based topic model. Topics that occur in the top fifteen topics for every cluster are not shown, while those that appear in the top fifteen for a single cluster only are highlighted in bold. Each topic is represented by its top ten words.

Chapter 6

Conclusions and Future Work

Topic models have seen many successes in recent years, and are used in a variety of applications, including analysis of news articles, topic-based search interfaces and navigation tools for digital libraries. Despite these recent successes, the field of topic modelling is still relatively new and there remains much to be explored. One of the most noticeable absences from most of the previous work on topic models is a consideration of the structure of language and text—from low-level structures, such as word order and syntax, to higher-level structures, such as relationships between documents.

This thesis presented structured topic models—models that combine document structure with latent topic variables. Three Bayesian models were introduced, each capturing a different type of structure: Word order, sentence-level syntactic structure, and relationships between semantically related documents. The models were applied to real-world document collections, demonstrating that structured topic modelling is an important and useful research area with much to offer in the way of good results.

In chapter 2, I introduced two fixed point-methods for estimating the hyperparameters of Dirichlet-multinomial distributions. Using synthetic and real data, I compared these method with several previously-introduced algorithms for Dirichlet-multinomial hyperparameter estimation, demonstrating that one of the new methods, and an algorithm introduced by MacKay and Peto (1995), are significantly faster than other techniques. I also explained how a gamma hyperprior can be incorporated into both of the new methods, and described how the log gamma recurrence relation may be used to efficiently compute the probability of data under a Dirichlet-multinomial distribution. This work has significance not only for situations where data are directly modelled using Dirichlet-multinomial distributions, but also for those where Dirichlet-multinomial distributions are used as components of a larger model.

Chapter 3 presented a new hierarchical Bayesian model that integrates n-gram-based and topic-based approaches to document modelling. An algorithm for "left-to-right" evaluation of topic models was also introduced. A bigram version of the new model

achieves better language modelling performance than do either latent Dirichlet allocation or a trigram language model. Additionally, the topics inferred by the model are clearly interpretable. I also determined that previous treatments of latent Dirichlet allocation, in which the base measures of the Dirichlet priors over words and topics are either both set to the uniform distribution or both optimised, are inappropriate for data containing stop words. Instead, such data should be modelled using (a) a nonuniform base measure in the Dirichlet prior over topic distributions, combined with (b) a uniform base measure in the Dirichlet prior over topic-specific word distributions. These modelling choices prevent the topics from being dominated by stop words by allowing the model to automatically discover a separate stop word topic. There is much scope for further work in this area. Firstly, it is likely that using the Pitman-Yor language model of Teh et al. (2006) rather than MacKay and Peto's Dirichlet language model would result in improved performance. In addition to this, the use of a letter-based language model as a top-level prior, as described by Cowans (2006), would eliminate the need for a fixed vocabulary. Finally, a complete investigation of the effects of topics for longer n-gram context lengths would be informative.

In chapter 4, I extended the reach of Bayesian methods to dependency parsing by introducing a new generative dependency model based on the hierarchical Pitman-Yor process. I showed that the performance of one of the best-known dependency parsers (Eisner, 1996a,b) can be significantly improved by using a Pitman-Yor prior over the distribution over dependents of a word, and by sampling model hyperparameters. To illustrate the flexibility of using a generative Bayesian approach, I also presented a second dependency model, in which dependencies between words are mediated by latent "syntactic topics". These topics look like finer-grained parts-of-speech and result in better parse accuracy when used instead of part-of-speech tags in the parsing model. Future work includes integration of other other latent variables, such as semantic topics, into the model. This may yield improved performance and reveal additional interactions between syntactic and semantic structure. As with the model presented in chapter 2, a letter-based language model could be used instead of a uniform distribution as a top-level prior, allowing the model to account for an unlimited vocabulary. Finally, the application of Bayesian techniques to the "left-to-right" dependency parsing framework of Chelba and Jelinek (1999) would result in a dependency model that could be combined with the topic-based language model from chapter 2 to give a single model that integrates word order, syntax and topics.

Chapter 5 focused on higher-level document structure—namely relationships between documents. I introduced a nonparametric Bayesian model for clustering documents using latent topics. The model assigned a higher probability to unseen academic papers from the NIPS conference than either a word-based clustering model or latent Dirichlet allocation. Furthermore, the cluster-specific distributions over topics correspond well to research areas within the NIPS community and highlight the topics that are likely to co-occur together. I also determined that when using a hierarchical

Dirichlet as the prior over topics in latent Dirichlet allocation or related topic models, the minimal path assumption results in more interpretable topics and a higher log probability for unseen data than the maximal path assumption. Finally, I extended the model to incorporate author information by characterising each cluster by two distributions: one over authors and one over topics. This results in finer-grained clusters, and explicates the relationships between particular groups of authors and topics. The model presented in chapter 5 was categorical in nature—each document was treated as belonging to a single cluster or category. An alternative to this characterisation, worthy of future investigation, is a *componential* model (MacKay, 1994), in which the distribution over topics for each document would be indexed by hyperparameters that are components in the space of document types. Such a model would easily capture regularities or correlations between document-specific distributions over topics, and could be compared with the cluster-based topic model introduced in chapter 5.

Bibliography

D. Aldous. Exchangeability and related topics. In *École d'Été de Probabilités de Saint-Flour XIII-1983*, pages 1–198. Springer, Berlin, 1985. (page 53)

C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50:5–43, 2003. (page 50)

T. C. Bell, J. G. Cleary, and I. H. Witten. *Text Compression*. Prentice Hall, 1990. (page 43)

J. Bernardo. Algorithm AS 103: Psi (digamma) function. *Applied Statistics*, 25:315–317, 1976. (page 28)

D. Blei and J. Lafferty. A correlated topic model of science. *Annals of Applied Statistics*, 1(1):17–35, 2007. (pages 12 and 42)

D. Blei, T. Griffiths, M. Jordan, and J. Tenenbaum. Hierarchical topic models and the nested Chinese restaurant process. In S. Thrun, L. K. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 17–24. The MIT Press, 2004. (page 42)

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, January 2003. (pages 12, 14, 42, 43, 46, 98, 100, and 106)

C. Chelba and F. Jelinek. Putting language into language modeling. In *Proceedings of Eurospeech*, Budapest, Hungary, 1999. (page 129)

S. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard University, August 1998. (pages 12, 16, 18, and 43)

P. J. Cowans. *Probabilistic Document Modelling*. PhD thesis, University of Cambridge, 2006. (pages 56, 78, and 129)

R. Cox. Probability, frequency, and reasonable expectation. *American Journal of Physics*, 14:1–13, 1946. (page 14)

A. Culotta and J. Sorensen. Dependency tree kernels for relation extraction. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2004. (page 73)

P. J. Davis. Gamma function and related functions. In M. Abramowitz and I. A. Stegun, editors, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, chapter 6. Dover, 1972. (pages 17, 19, 21, 24, and 28)

A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977. (page 50)

Y. Ding and M. Palmer. Machine translation using probabilistic synchronous dependency insertion grammars. In *Proceedings of the 43rd Annual Meeting of the Association of Computational Linguistics*, 2005. (page 73)

A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo in Practice*. Springer-Verlag, 2001. (page 64)

M. Dredze and H. M. Wallach. User models for email activity management. In *Proceedings of the 5th International Workshop on Ubiquitous User Modeling*, Gran Canaria, Spain, 2008. (page 15)

R. Durbin, S. R. Eddy, and G. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1999. (page 16)

J. M. Eisner. An empirical comparison of probability models for dependency grammar. Technical Report IRCS-96-11, Institute for Research in Cognitive Science, University of Pennsylvania, 1996a. (pages 13, 15, 63, 75, 76, 81, 88, 89, 90, 109, and 129)

J. M. Eisner. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th Interational Conference on Computational Linguistics*, pages 340–345, Copenhagen, August 1996b. (pages 13, 15, 73, 75, 76, 81, 89, 90, and 129)

T. Ferguson. Bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1:209–230, 1973. (pages 78 and 104)

S. Goldwater, T. Griffiths, and M. Johnson. Interpolating between types and tokens by estimating power-law generators. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 459–466. The MIT Press, 2006. (pages 42, 78, 79, 80, and 81)

I. J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3 and 4):237–264, 1953. (page 13)

T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101(suppl. 1):5228–5235, 2004. (pages 49, 50, 64, and 94)

T. L. Griffiths, M. Steyvers, D. M. Blei, and J. B. Tenenbaum. Integrating topics and syntax. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 536–544. The MIT Press, 2005. (pages 64, 72, and 92)

T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning Journal*, 42(1):177–196, 2001. (page 12)

T. Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, 1999. (page 12)

E. T. Jaynes. *Probability Theory: The Logic of Science*. Cambridge University Press, 2003. (page 14)

F. Jelinek. *Statistical Methods for Speech Recognition*. The MIT Press, 1998. (pages 12, 43, and 64)

F. Jelinek and R. L. Mercer. Interpolated estimation of Markov source parameters from sparse data. In E. S. Gelsema and L. N. Kanal, editors, *Proceedings of the Workshop on Pattern Recognition in Practice*, pages 381–402, Amsterdam, The Netherlands, 1980. North-Holland. (pages 13 and 43)

M. Johnson, T. L. Griffiths, and S. Goldwater. Adaptor grammars: A framework for specifying compositional nonparametric Bayesian models. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 641–648. The MIT Press, 2007a. (pages 81 and 85)

M. Johnson, T. L. Griffiths, and S. Goldwater. Bayesian inference for PCGFs via Markov chain Monte Carlo. In *Proceedings of the North American Conference on Computational Linguistics*, 2007b. (pages 80 and 85)

R. E. Kass and A. E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90:773–795, 1995. (pages 64 and 110)

S. M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-35(3):400–401, March 1987. (page 13)

R. Kneser and H. Ney. Improved backing-off for M-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 181–184, 1995. (pages 13, 78, and 80)

W. Li and A. McCallum. Mixtures of hierarchical topics with pachinko allocation. In *Proceedings of the 24th International Conference on Machine learning*, pages 633–640, 2007. (page 42)

D. J. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003. (page 17)

D. J. MacKay. Bayesian interpolation. *Neural Computation*, 4(3):415–447, 1992. (page 18)

D. J. C. MacKay. Models for dice factories and amino acid probabilities. Technical report, Cavendish Laboratory, University of Cambridge, 1994. (page 130)

D. J. C. MacKay and L. C. B. Peto. A hierarchical Dirichlet language model. *Natural Language Engineering*, 1(3):289–307, September 1995. (pages 13, 14, 16, 19, 24, 26, 29, 41, 42, 43, 44, 45, 78, 79, 128, and 129)

C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*. The MIT Press, 2000. (page 74)

M. Marcus, B. Santorini, and M. Marcinkiewicz. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993. (pages 37 and 88)

D. McAllester and R. E. Schapire. Learning theory and language modeling. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millennium*, chapter 9, pages 271–287. Morgan Kaufmann, 2003. (page 31)

R. McDonald. *Discriminative Training and Spanning Tree Algorithms for Dependency Parsing*. PhD thesis, University of Pennsylvania, 2006. (pages 75, 89, and 91)

D. Mimno and A. McCallum. Organizing the OCA: Learning faceted subjects from a library of digital books. In *Proceedings of the 7th ACM/IEEE joint conference on Digital libraries*, pages 376–385, Vancouver, BC, Canada, 2007. (page 12)

D. Mimno, H. Wallach, and A. McCallum. Community-based link prediction with text. In *Statistical Network Modeling Workshop, held at NIPS 2007*, 2007. (page 15)

T. P. Minka. Estimating a Dirichlet distribution. `http://research.microsoft.com/~minka/papers/dirichlet/`, 2003. (pages 16, 18, 21, and 23)

R. M. Neal. Slice sampling. *Annals of Statistics*, 31:705–767, 2003. (pages 61 and 84)

D. Newman, C. Chemudugunta, P. Smyth, and M. Steyvers. Analyzing entities and topics in news articles using statistical topic models. In *Intelligence and Security Informatics*, Lecture Notes in Computer Science. 2006. (page 12)

M. A. Newton and A. E. Raftery. Approximate Bayesian inference with the weighted likelihood bootstrap. *Journal of the Royal Statistical Society, Series B*, 56:3–48, 1994. (page 64)

K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Learning to classify text from labeled and unlabeled documents. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 792–799, 1998. (page 99)

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 1999. (pages 21 and 29)

J. Pitman and M. Yor. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 25:855–900, 1997. (pages 15 and 78)

L. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition*. Prentice Hall, 1993. (pages 12, 43, and 64)

A. Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 133–142, 1996. (page 88)

K. F. Riley, M. P. Hobson, and S. J. Bence. *Mathematical Methods for Physics and Engineering*. Cambridge University Press, third edition, 2006. (pages 24 and 26)

M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and documents. In D. M. Chickering and J. Y. Halpern, editors, *Proceedings of the 20th Annual Conference on Uncertainty in Artificial Intelligence*, pages 487–494, 2004. (page 117)

R. Rosenfeld. Two decades of statistical language modeling: Where do we go from here? In *Proceedings of the IEEE*, volume 88, pages 1270–1278, 2000. (page 16)

J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994. (page 104)

N. Slonim and N. Tishby. Document clustering using word clusters via the information bottleneck method. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 208–215, 2000. (page 99)

R. Snow, D. Jurafsky, and A. Y. Ng. Learning syntactic patterns for automatic hypernym discovery. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1297–1304. The MIT Press, 2004. (page 73)

M. Steyvers and T. Griffiths. Probabilistic topic models. In T. Landauer, D. McNamara, S. Dennis, and W. Kintsch, editors, *Latent Semantic Analysis: A Road to Meaning*. Laurence Erlbaum, 2007. (pages 12 and 16)

M. Steyvers, P. Smyth, M. Rosen-Zvi, and T. Griffiths. Probabilistic author-topic models for information discovery. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 306–315, 2004. (page 117)

Y. W. Teh. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 985–992, 2006. (pages 13, 15, 42, 78, 79, and 80)

Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101:1566–1581, 2006. (pages 53, 56, 78, 100, 101, 102, and 129)

L. Tesnière. *Éléments de Syntaxe Structurale*. Klincksieck, 1959. (page 13)

H. M. Wallach. Topic modeling: Beyond bag-of-words. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 977–984, Pittsburgh, Pennsylvania, 2006. (pages 15 and 18)

H. M. Wallach, C. Sutton, and A. McCallum. Bayesian modeling of dependency trees using hierarchical Pitman-Yor priors. In *Prior Knowledge for Text and Language Processing Workshop*, Helsinki, Finland, July 2008. (page 15)

D. J. Ward. *Adaptive Computer Interfaces*. PhD thesis, University of Cambridge, 2001. (pages 43 and 64)

D. J. Ward, A. F. Blackwell, and D. J. C. MacKay. Dasher – a data entry interface using continuous gestures and language models. In *UIST 2000: The 13th Annual ACM Symposium on User Interface Software and Technology*, 2000. (pages 43 and 64)

I. H. Witten and T. C. Bell. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, 37(4):1085–1094, July 1991. (page 13)

H. Yamada and Y. Matsumoto. Statistical dependency analysis with support vector machines. In *Proceedings of the International Workshop on Parsing Technologies*, 2003. (page 75)

K. Yu, S. Yu, and V. Tresp. Dirichlet enhanced latent semantic analysis. In R. G. Cowell and Z. Ghahramani, editors, *Proceedings of the Tenth International Conference on Artificial Intelligence and Statistics*, pages 437–444. Society for Artificial Intelligence and Statistics, 2005. (page 99)