ABSTRACT

Title of dissertation: EXPRESSIVE KNOWLEDGE RESOURCES IN PROBABILISTIC MODELS

Yuening Hu, Doctor of Philosophy, 2014

Dissertation directed by: Professor Jordan Boyd-Graber
iSchool, Umiacs

Understanding large collections of unstructured documents remains a persistent problem. Users need to understand the themes of a corpus and to explore documents of interest. Topic models are a useful and ubiquitous tool to discover the main themes (namely topics) of the corpus. Topic models have been successfully applied in natural language processing, computer vision, information retrieval, cognitive science, etc. However, the discovered topics are not always meaningful: some topics confuse two or more themes into one topic; two different topics can be near duplicates; and some topics make no sense at all. Adding knowledge resources into topic models can improve the topics. However, how to **encode** knowledge into topic models and where to **find** these knowledge resources remain two scientific challenges. To address these problems, this thesis presents tree-based topic models to encode prior knowledge, a mechanism incorporating knowledge from untrained users, a polylingual tree-based topic model based on existing dictionaries as knowledge resources, an exploration of regularizing spectral methods to encode prior knowledge into topic models, and a model for automatically building hierarchies of prior knowledge for topic models.

To encode knowledge resources into topic models, we first present tree-based topic models, where correlations between word types are modeled as a prior tree and applied to topic models. We also develop more efficient inference algorithms for tree-based topic models. Experiments on multiple corpora show that efficiency is greatly improved on different number of topics, number of correlations and vocabulary size.

Because users decide whether the topics are useful or not, users' feedback is necessary for effective topic modeling. We thus propose a mechanism for giving normal users a voice to topic models by encoding users' feedback as correlations between word types into tree-based topic models. This framework, interactive topic modeling (ITM), allows untrained users to encode their feedback easily and iteratively into the topic models. We validate the framework both with simulated and real users and discuss strategies for improving the user experience to adapt models to what users need.

Existing knowledge resources such as dictionaries can also improve the model. We propose polylingual tree-based topic models based on bilingual dictionaries and apply this model to domain adaptation for statistical Machine Translation. We derive three different inference schemes and evaluate the efficacy of our model on a Chinese to English translation system, and obtain up to 1.2 BLEU improvement over the machine translation baseline.

This thesis further explores an alternative way—regularizing spectral methods for topic models—to encode prior knowledge into topic models. Spectral methods offer scalable alternatives to Markov chain Monte Carlo and expectation maximization. However, these new methods lack the priors that are associated with probabilistic models. We examine Arora et al.'s anchor algorithm for topic models and encode prior knowledge by regularizing the anchor algorithm to improve the interpretability and generalizability of topic models.

Because existing knowledge resources are limited and because obtaining the knowledge from users is expensive and time-consuming, automatic techniques should also be considered to extract knowledge from the corpus. This thesis further presents a Bayesian hierarchical clustering technique with the Beta coalescent, which provides a possible way to build up the prior tree automatically. Because of its computational complexity, we develop new sampling schemes using sequential Monte carlo and Dirichlet process mixture models, which render the inference practical and efficient.

This thesis explores sources of prior knowledge, presents different ways to encode these expressive knowledge resources into probabilistic topic models, and also applies these models in translation domain adaptation. We also discuss further extensions in a bigger picture of interactive machine learning techniques and domain adaptation for downstream tasks.

EXPRESSIVE KNOWLEDGE RESOURCES IN PROBABILISTIC MODELS

by

Yuening Hu

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2014

Advisory Committee:
Professor Jordan Boyd-Graber/Advisor
Professor Hal Daumé III
Professor Héctor Corrada Bravo
Professor Philip Resnik
Professor David M. Blei

Acknowledgments

I feel very lucky to spend five years in the Department of Computer Science, University of Maryland. I owe my gratitude to all the people who have made this thesis possible.

First and foremost, I would like to thank my advisor, Professor Jordan Boyd-Graber for all the advice and help over the past four years. He gives me flexibility to choose interesting projects and is very open to hear my ideas and thoughts; he works closely with me for all the deadlines, no matter it is weekends or midnight; he always makes himself available whenever I have questions and difficulties; he helps me on the practice talks again and again, thus I could give good conference talks and job talks; and he also encourages and gives me opportunities to mentor junior students. Moreover, his passion to research—being creative, hard-working, responsible and seeking for perfection—greatly influences me. I feel very fortunate to work with and learn from such an extraordinary advisor!

I would also like to thank Professor Hal Daumé III for his help on the coalescent project, the great advice on my talks and jobs, and writing countless reference letters for me! Thanks are also due to Professor Philip Resnik, Professor Héctor Corrada Bravo and Professor David Blei for agreeing to serve on my thesis committee and spending their invaluable time reviewing my thesis!

I was fortunate to be part of the Computational Linguistics and Information Processing Lab (**CLIP**), where all the professors and students are open-minded, creative and willing to collaborate and help. Viet-An Nguyen, a great and responsible collaborator and a close friend; Alison Smith and Brianna Satinoff, two creative collaborators who helped and made my favorite interactive topic modeling paper possible; Thang Nguyen, a smart junior who I collaborated and who I firstly mentored as a graduate student; Vladimir Edelman, a patient senior who helped me with my first experience with Machine Translation.

There are a lot of other colleagues that I should give a special mention. Thank Professor Jimmy Lin, Professor Louiqa Raschid, Professor Naomi Feldman and Professor Douglas W. Oard for their helpful comments and suggestions; thank Joe Webster for a lot of technical support and help; thank Leonardo Claudino, Mohit Iyyer, Alvin Grissom II, John Morgan, Ke Wu, Junhui Li, He He, Xi Chen, Chang Liu, Hua He, Jiarong Jiang, Taesun Moon, Amit Goyal, Jagadeesh Jagarlamudi, Snigdha Chaturvedi, Greg Sanders, Arvind Agarwal, and Sudha Rao for their help and friendship. I would also like to acknowledge help and support from some of the staff members: Jennifer Story, Fatima Bangura and Arlene E. Schenk.

This thesis cannot be done without the help from outside of Maryland. I also want to take this chance to thank my mentors Jianfeng Gao, Michael Auli and Qin Gao at Microsoft Research Redmond, Sinead Williamson at University of Texas at Austin, Jason Chuang at University of Washington, Julia Lane and Evgeny Klochikhin at American Institutes for Research.

My dearest friends have enriched my graduate life in many ways and also deserve a special mention: Teng Long, Qian Zhou, Xuan Yao, Yemei Han, Taotao Liu, Mingyang Ji, Yijing Lu, Bo Sun, Yangwen Liu, Kangyuan Zhu, Zheng Zhang, Xiang He, Xiaojie Cong, Dandan Li and Xiqun Chen. Thank them all for their care and support and for always being there for me.

Finally, I owe my deepest thanks to my family - my mother, father, sister, my other relatives, particularly my husband Ke Zhai. Ke is not only my closest friend in life but also my best partner at work. Words cannot express my gratitude to Ke and how fortunate that I am to have Ke. It would be very hard to finish this thesis without his support. Thank Ke and my whole family for their support!

Table of Contents

List of Tables

List of Figures

Chapter 1

The Need for Prior Knowledge in Probabilistic Models

Understanding large collections of unstructured text remains a persistent problem. Unlike information retrieval, where users know what they are looking for, sometimes users need to *understand* the high-level themes of a corpus and *explore* documents of interest. Probabilistic models have been widely applied to explore the large corpora, especially the topic models, one of the most popular probabilistic models, offer a formalism for exposing a collection's themes, and have been applied to aid information retrieval (Wei and Croft, 2006), understand scientific ideas (Hall et al, 2008), and discover political perspectives (Paul and Girju, 2010). In addition, topic models have also been applied outside text to learn natural scene categories in computer vision (Li Fei-Fei and Perona, 2005); discover patterns in population genetics (Shringarpure and Xing, 2008); and understand the connection between Bayesian models and cognition (Landauer et al, 2006; Griffiths et al, 2007).

Topic models are attractive because they discover groups of words that often appear together in documents. These are the namesake "topics" of topic models and are multinomial distributions over the vocabulary words.[1] The word types which have the highest probability in a topic evince what a topic is about. In addition, each document can be represented as an admixture of these topics, a low-dimensional

[1] To avoid confusion, we will henceforth avoid using "words". We adopt the linguistic convention of referring to we refer the Platonic exemplars of words as "types" or "word types" and particular instances of words in a document as "tokens".

shorthand for representing what a document is about.

The strength of topic models is that they are unsupervised. They do not require any *a priori* annotations. The inputs that a topic model requires are only the text divided into documents and the number of topics you want it to discover, although there are models and heuristics for selecting the number of topics automatically (Teh et al, 2006).

This is clearer with an example. A “technology”[2] topic has high probability for the words “computer, technology, system, service, site, phone, internet, machine”; a topic about “arts” has high probability for the words “play, film, movie, theater, production, star, director, stage”. These topics are “good” topics since they have semantic coherence, which means the top words are meaningful enough for users to understand the main theme of this topic. The judgment of “good” topics is not entirely idiosyncratic. Chang et al (2009) showed that people mostly agree on what makes a good topic.

Despite what you see in topic modeling papers, the topics discovered by topic modeling do not always make sense to ostensible end users. From the users' perspective, there are often “bad” topics. These bad topics can confuse two or more themes into one topic; two different topics can be (near) duplicates; and some topics make no sense at all. This is a fundamental problem, as even if everything went perfectly, the objective function that topic models optimize does not always correlate with human judgments of topic quality (Chang et al, 2009). A closely related issue

[2] Topic models do not name the topics that they discover; for discussion, it's convenient to talk about topics as if they were named. Automatic techniques for labeling (Lau et al, 2011) exist, however. In addition, we italicize the topic names to distinguish these abstractions from the actual words in a topic, which are in quotes

is that topic models—with their bag-of-words vision of the world—simply lack the information needed to create the topics end-users expect.

One way to solve this fundamental problem is to add expressive knowledge resources, which are also considered as **prior knowledge**, into the probabilistic topic models. There has been a thriving cottage industry to make topic models better fit users' expectations by adding various information to topic models, including modeling perspective (Paul and Girju, 2010; Lin et al, 2006), syntax (Gruber et al, 2007; Wallach, 2006), authorship (Rosen-Zvi et al, 2004; Dietz et al, 2007), etc.

While topic models consider documents as "bag of words", ignoring the correlations between words, one type of important prior knowledge is the **correlations** between words within a topic (Petterson et al, 2010), which when appropriately constructed, can create topics that improve topic coherence (Newman et al, 2009, 2010; Mimno et al, 2011).

To consider the correlations between words in topic models, there are two main problems: how to model the correlation prior into topic models; and where to get the prior knowledge of word correlations.

1.1 Encoding Correlations into a Tree Prior for Topic Models

Trees are intuitive methods for encoding prior knowledge. For example, WordNet (Miller, 1990), an online lexical reference system, can be treated as a large tree structure, where word types that have similar meaning are put in the same synset, which implies the correlations between words. Using the tree from WordNet, Abney and Light (1999) used tree-structured multinomials to model selectional restrictions, which was later put into a Bayesian context for topic modeling (Boyd-Graber et al, 2007).

While correlations are mathematically impossible to represent with a symmetric Dirichlet prior, in this dissertation, we organize these correlations into a tree structure (Boyd-Graber et al, 2007; Andrzejewski et al, 2009) and apply the tree prior to topic models, which is called **tree-based topic models**. In the tree-based topic models, good topics can be encouraged by having correlations that link together words the model separated; and a bad topic can be discouraged by correlations that split topics that combine words that should not have been together.

Note that these correlations only form "soft" constraints for the topic model, which means the results will match the correlations if and only if the correlations supported by the underlying statistical model of the text. Also, tree prior preserves conjugacy, making inference using Gibbs sampling (Heinrich, 2004) straightforward.

However, while inference is indeed straightforward, it is not cheap. In Chapter 3, we develop an efficient tree-based topic modeling framework, which dramatically improves the efficiency of inference.

1.2 Obtaining Prior Knowledge from Users to Improve Topics

The next question is where to obtain the correlations. Generally, users of topic models are the ultimate judge of whether a topic is "good" or "bad". A user might echo one of the many complaints lodged against topic models: these documents should have similar topics but do not (Daumé III, 2009); this topic should have syntactic coherence (Gruber et al, 2007; Boyd-Graber and Blei, 2008); this topic makes no sense at all (Newman et al, 2010); this topic shouldn't be associated with this document but is (Ramage et al, 2009); these words shouldn't be the in same topic but are (Andrzejewski et al, 2009); or these words should be in the same topic but are not (Andrzejewski et al, 2009).

These complaints or user feedback can be considered as the sources of correlations, which can be modeled as the tree prior so that the results of topic models would better match with users' expectation. Notice, unlike Andrzejewski et al (2009), these correlations need not come from experts.

However, Andrzejewski et al (2009)'s assumption of *a priori* correlations is inefficient. Users' attempting to curate coherent topics need to see where the topic models go wrong before they can provide useful feedback. This suggests that topic models should be **interactive**—users should be able to see the output, give feedback, and continue refining the output. This whole process should be simple enough to allow these non-expert users to craft models that make sense for them.

Then the next question is **how do we support interactivity** in a principled way? In Chapter 4, we create Interactive Topic Models (ITM), a framework that does not throw away the previous effort users have put in the topic models but that can also adapt to the new feedback users supply.

1.3 Using Existing Prior Knowledge to Assist Machine Translation

In addition to the prior knowledge from users, there are various existing resources which can be used as prior knowledge to improve the topic models. For example, dictionaries group words in similar meaning as synonyms, while group words with opposite meanings as antonyms. These group information can also be used as prior knowledge.

These resources are not limited to one language. They can cross languages. For example, bilingual dictionaries connect words in different languages, and the word pairs, which mean the same thing but in different languages, can also be used as correlations. If bilingual dictionary tells "电脑" in Chinese means "computer" in English, these two words form one correlation which can connect Chinese and English together. As a result, we can also build up tree-based topic models across languages to extract multilingual topics, which enable multilingual applications.

However, the tree-based topic models with correlations across languages only connect multiple languages on the word level. The polylingual topic models proposed by Mimno et al (2009) connect multiple languages on the document level, since they assume the parallel documents share the same topic distributions. Based on these two models, we develop a new topic model, polylingual tree-based topic models that combine the insights of both models and connect multiple languages on both word level and document level.

One possible application for this new model is domain adaptation in statistical machine translation (Koehn, 2009, SMT). Domains are genres that group documents with the similar style. Translations within one domain are better than translations across domains since they vary dramatically in their word choices and style. A correct translation in one domain, may be inappropriate in another domain. For example, if you see “潜水” in newspaper, it usually means “underwater diving”. If you see it on social media, it means a non-contributing “lurker” in online forums.

Training a SMT system using the data from various different domains is referred as the problem of *domain adaptation*. Early efforts focused on building separate models (Foster and Kuhn, 2007) and adding features (Matsoukas et al, 2009) to model domain information. Chiang et al (2011) combines these approaches by directly optimizing genre and collection features by computing separate translation tables for domain. However, all these approaches consider domains as a hard constraint, which are imposed externally and hand-labeled. Obtaining such domain information may be expensive or even unreasonable.

Eidelman et al (2012) used the vanilla topic models as unsupervised approaches for domain adaptation, where they treat the topics as soft domain labels, thus the document topic distribution defines a soft assignment of a document to domains. However, Eidelman et al (2012) induced these topic model domains from source documents *only*, thus they ignored a wealth of information from the target documents that could improve topic models’ ability for discovering domains to help machine translation. As a result, in Chapter 5, this dissertation applies the new multilingual topic models, and learns domains from both source and target documents to improve the SMT results.

1.4 Alternative Spectral Methods to Encode Prior Knowledge

Traditional posterior inference for topic models use MCMC (Griffiths and Steyvers, 2004) or variational EM (Blei et al, 2003b), which can be viewed as local optimization: searching for the latent variables that maximize the data likelihood.

An exciting vein of new research provides provable polynomial-time alternatives. These approaches provide solutions to hidden Markov models (Anandkumar et al, 2012c), mixture models (Kannan et al, 2005), and latent variable grammars (Cohen et al, 2013). The key insight is not to directly optimize observation likelihood but to instead discover latent variables that can reconstruct the moments of the assumed generative model. Unlike search-based methods, which can be caught in local minima, these techniques are guaranteed to find global optima.

These general techniques can be improved by making reasonable assumptions about the models. For example, Arora et al (2012b)'s approach for inference in topic models assume that each topic has a unique "anchor" word (thus, we call this approach **anchor**). This approach is fast and effective; because it only uses word co-occurrence information, it can scale to much larger datasets than MCMC or EM alternatives.

Despite their advantages, these techniques are not a panacea. They do not accommodate the rich priors that modelers have come to expect. The other shortcoming is that these models have not been scrutinized using standard NLP evaluations. Because these approaches emerged from the theory community, **anchor**'s evaluations, when present, typically use training reconstruction.

In Chapter 6, we regularize the **anchor** method to trade-off the fidelity of the reconstruction with the penalty terms that mimic Gaussian and Dirichlet priors. We also evaluate these spectral methods against the traditional MCMC and variational EM by held-out likelihood (Blei et al, 2003b) and topic interpretability (Chang et al, 2009; Newman et al, 2010). We also show that our extension to the **anchor** method enables new applications: for example, using an informed priors to discover concepts of interest.

1.5 Learning Prior Knowledge Automatically using Beta Coalescent

While obtaining prior knowledge from users are expensive and time-consuming, automatic technique to extract prior knowledge from the corpus should also be considered. One possible direction is to construct prior trees automatically by hierarchical clustering techniques, which group similar nodes together into a hierarchy. These nodes sharing the same parent node are similar to each other, which can reflect the correlation between words, if we treat each node as a word. Thus this automatically built tree can be used as prior tree for topic models. Hierarchical clustering has been applied various areas, including natural language processing (Brown et al, 1990), computer vision (Bergen et al, 1992), graphics (Yvart et al, 2005), and network analysis (Girvan and Newman, 2002). In all of these cases, it is crucial to be able to model hierarchies that are non-binary (have a branching factor > 2).

Recent hierarchical clustering techniques have been incorporated inside statistical models; this requires formulating clustering as a statistical—often Bayesian—

problem. Heller and Ghahramani (2005) build binary trees based on the marginal likelihoods, extended by Blundell et al (2010) to trees with arbitrary branching structure. Adams et al (2010) propose a tree-structured stick-breaking process to generate trees with unbounded width and depth, which supports data observations at leaves *and* internal nodes.[3] However, these models do not distinguish edge lengths, an important property in distinguishing how "tight" the clustering is at particular nodes.

Hierarchical models can be divided into complementary "fragmentation" and "coagulation" frameworks (Pitman, 1999). Both produce hierarchical partitions of a dataset. Fragmentation models start with a single partition and divide it into ever more specific partitions until only singleton partitions remain. Coagulation frameworks repeatedly merge singleton partitions until only one partition remains. Pitman-Yor diffusion trees (Knowles and Ghahramani, 2011), a generalization of Dirichlet diffusion trees (Neal, 2003a), are an example of a bushy fragmentation model, and they model edge lengths and build non-binary trees.

Instead, our focus is on bottom-up coalescent models (Berestycki, 2009), one of the coagulation models and complementary to diffusion trees, which can also discover hierarchies and edge lengths. One appealing example is Kingman's coalescent (Teh et al, 2008; Görür and Teh, 2009; Görür et al, 2012), a standard genealogical population model (Kingman, 1982; Berestycki, 2009). However, it is limited to binary trees. In Chapter 7, we generalize Kingman's coalescent to the

[3] This is appropriate where the entirety of a population is known—both ancestors and descendants. We focus on the case where only the descendants are known.

beta coalescent (Pitman, 1999; Berestycki, 2009), which is used as a prior over trees with arbitrary branching factors, and propose a more efficient inference in the belief propagation framework (Teh et al, 2008) using the beta coalescent.

While this hierarchical clustering can be applied in various areas which support a multi-branch hierarchical structure, it can also be used as the tree prior for topic models. Since it is a fully statistical Bayesian model, it gives the option to jointly learning the tree structures and downstream tasks.

1.6 Structure

Given the problems with the existing probabilistic topic models, this dissertation discusses where to get the prior knowledge, how to provide the prior knowledge to topic models efficiently, and how to evaluate these models. This dissertation is organized as follows:

- Chapter 2 reviews the vanilla topic models and tree-based topic models, and show how they can be used to model the correlations between word types;
- Chapter 3 presents a more efficient inference scheme for tree-based topic models, and shows the proposed model dramatically improve the efficiency of inference by showing the empirical comparison on two different datasets;
- Chapter 4 creates a framework for non-expert end users to provide feedback and update the topics interactively and iteratively; we validate the framework both with simulated and real users, and discuss the strategies for improving the user experience to adapt models to what users need;

- Chapter 5 introduces the polylingual tree-based topic models based on the existing knowledge resource, and develops different inference schemes for this model; we further apply this model into domain adaptation in statistical machine translation, and improve the translation BLEU scores;

- Chapter 6 explores spectral methods for topic models with regularization, an alternative way to encode prior knowledge to topic models; we regularize the **anchor** spectral method to trade-off the fidelity of the reconstruction with the penalty terms that mimic Gaussian and Dirichlet priors;

- Chapter 7 presents a Bayesian hierarchical clustering technique using the beta coalescent to construct the prior knowledge automatically; we generalize an existing belief propagation framework for the beta coalescent, and develop new sampling schemes using sequential Monte Carlo and Dirichlet process mixture models to reduce the computational complexity;

- Chapter 8 concludes this dissertation and discusses the possible applications and future extensions of the research work in this dissertation.

Chapter 2

Existing Probabilistic Models for Encoding Prior Knowledge

This chapter begins by reviewing Latent Dirichlet Allocation (LDA, Blei et al (2003b)), one of the most popular topic models. We refer this LDA as vanilla LDA to distinguish it from more complicated models later. However, vanilla LDA is a bag-of-words model, which ignores the correlations between word types. This lacuna can be addressed through tree-based topic models (Boyd-Graber et al, 2007; Andrzejewski et al, 2009), which we also review in this chapter. We describe the generative process for tree-based topic models and the detailed inference scheme. Based on this tree-based topic models, we further develop an efficient inference scheme in Chapter 3, which is further used as the backend model for interactive topic modeling in Chapter 4.

2.1 Vanilla LDA

Topic models, exemplified by Latent Dirichlet Allocation (LDA, Blei et al (2003b)), discovers a set of topics, which are distributions of all words. These topics are usually semantically coherent, and describe the main themes of a corpus. In addition, it also discovers the topic proportion–a distribution over all topics– for each document, from which we know the main themes of a document.

Currently, topic models have been widely applied to aid information retrieval (Wei and Croft, 2006), understand scientific ideas (Hall et al, 2008), discover political perspectives (Paul and Girju, 2010), learn natural scene categories in computer vision (Li Fei-Fei and Perona, 2005), and understand the connection between Bayesian models and cognition (Landauer et al, 2006; Griffiths et al, 2007).

2.1.1 Generative Process

We first review the generative process of vanilla LDA with K topics of V words (Blei et al (2003b) and Griffiths and Steyvers (2004) offer more thorough reviews):

- For each topic $k = \{1 \dots K\}$
  - draw a $V-$dimensional multinomial distribution over all words: $\pi_k \sim \mathrm{Dir}(\beta)$
- For each document d
  - draw a $K-$dimensional multinomial distribution that encodes the probability of each topic appearing in the document: $\theta_d \sim \mathrm{Dir}(\alpha)$
  - for each token $w_{d,n}$ of this document d
    - draw a topic $z_{d,n} \sim \mathrm{Mult}(\theta_d)$
    - draw a token $w_{d,n} | z_{d,n} = k, \pi \sim \mathrm{Mult}(\pi_k)$

where α and β are the hyperparameters of the Dirichlet distribution. Given observed documents, posterior inference discovers the latent variables that best explain an observed corpus.

However, the generative process makes it clear that LDA does not know what individual words *mean*. Tokens are mere draws from a multinomial distribution.

LDA, by itself, has no way of knowing that a topic that gives high probability to "dog" and "leash" should also, all else being equal, give high probability to "poodle". That LDA discovers interesting patterns in its topic is possible due to document co-occurrence. Any semantic or syntactic information not captured in document co-occurrence patterns will be lost to LDA.

2.1.2 Inference

We use Gibbs sampling for posterior inference (Neal, 1993; Resnik and Hardisty, 2010) to uncover the latent variables that best explain observed data. For vanilla LDA, the latent variables of interest are the topic assignment of each token $z_{d,n}$, the per-document distribution over topics θ_d, and the topic distributions over words π_k. The joint probability is

$$p(\mathbf{Z}, \mathbf{W}, \pi, \theta; \alpha, \beta) = \prod_k p(\pi_k|\beta) \Big[\prod_d p(\theta_d|\alpha) \Big[\prod_n \underbrace{p(z_{d,n}|\theta_d)}_{\text{assignment}} \underbrace{p(w_{d,n}|\pi, z_{d,n})}_{\text{token}} \Big]\Big], \quad (2.1)$$

where d is the document index; n is the token index in document d; $\mathbf{Z}$ are the topic assignments for all tokens; $\mathbf{W}$ represents all tokens; α is Dirichlet prior for document-topic distributions θ; and β is Dirichlet prior for topic-word distributions π.

Because the conjugate prior of the multinomial is the Dirichlet, we can integrate out the topic-word distributions π and the per-document topic distributions θ in the conditional distribution

$$
\begin{aligned}
p(z_{d,n} = k | \mathbf{Z}_{-}, \mathbf{W}; \alpha, \beta) = {} & \frac{p(z_{d,n} = k, \mathbf{Z}_{-}, w_{d,n}; \alpha, \beta)}{p(\mathbf{Z}_{-}; \alpha, \beta)} \\
= {} & \underbrace{\frac{\int_{\theta} p(z_{d,n} = k, \mathbf{Z}_{-} | \theta) p(\theta; \alpha) d\theta}{\int_{\theta} p(\mathbf{Z}_{-} | \theta) p(\theta; \alpha) d\theta}}_{\text{topic assignment}} \underbrace{\frac{\int_{\pi} p(w_{d,n}, \mathbf{W}_{-} | z_{d,n}, \mathbf{Z}_{-}, \pi) p(\pi; \beta) d\pi}{\int_{\pi} p(\mathbf{W}_{-} | \mathbf{Z}_{-}, \pi) p(\pi; \beta) d\pi}}_{\text{token}} \\
= {} & \underbrace{p(z_{d,n} = k | \mathbf{Z}_{-}; \alpha)}_{\text{topic assignment}} \underbrace{p(w_{d,n} | \mathbf{W}_{-}, \mathbf{Z}_{-}, z_{d,n}; \beta)}_{\text{token}}
\end{aligned}
\tag{2.2}
$$

where $\mathbf{Z}_{-}$ are the topic assignments excluding the topic assignment for the current word $z_{d,n}$, $\mathbf{W}_{-}$ represents the tokens excluding the current token $w_{d,n}$. Using conjugacy, the two integrals—canceling Gamma functions from the Dirichlet normalizer that appear in both numerator and denominator—are

$$
p(w_{d,n} | \mathbf{W}_{-}, \mathbf{Z}_{-}, z_{d,n}; \beta) = \frac{\frac{\Gamma(n_{w_{d,n}|k} + \beta + 1)}{\Gamma(n_{\cdot|k} + \beta V + 1)}}{\frac{\Gamma(n_{w_{d,n}|k} + \beta)}{\Gamma(n_{\cdot|k} + \beta V)}} \tag{2.3}
$$

$$
p(z_{d,n} = k | \mathbf{Z}_{-}; \alpha) = \frac{\frac{\Gamma(n_{k|d} + \alpha_k + 1)}{\Gamma\left(\sum_{k'} (n_{k'|d} + \alpha_{k'}) + 1\right)}}{\frac{\Gamma(n_{k|d} + \alpha_k)}{\Gamma\left(\sum_{k'} (n_{k'|d} + \alpha_{k'})\right)}} \tag{2.4}
$$

where $n_{k|d}$ is topic k's count in the document d; $\mathbf{Z}_{-}$ are the topic assignments excluding the current token $w_{d,n}$; $n_{w_{d,n}|k}$ is the count of tokens with word $w_{d,n}$ assigned to topic k; V is the vocabulary size, and $n_{\cdot|k}$ is the count of all tokens assigned to topic k.

With additional cancellations, we can remove the remaining Gamma functions to obtain the conditional distribution

$$p(z_{d,n} = k|\mathbf{Z}_{-}, \mathbf{W}; \alpha, \beta) \propto (\alpha_k + n_{k|d})\frac{\beta + n_{w_{d,n}|k}}{\beta V + n_{\cdot|k}}. \tag{2.5}$$

To sample a topic for a token in a document, we compute the probability of all topics according to Equation 2.5, and then randomly sample a topic according to the probability mass. This process can be continued for each token till the model is converged.

2.1.3 Example

Now let's run this vanilla LDA on a corpus of about 2000 New York Times editorials from 1987 to 1996 and see what topics we can get. We start by finding 20 initial topics from this corpus.

The topics that we get are shown in Table 2.1. As we can see, these 20 topics mostly capture the main themes of this corpus, including "Security", "politics", "economy", etc. However, there are two topics both deal with "Russia" (broadly construed). Topic 20 is about the "Soviet Union", but Topic 1 focuses on the development of "Russian Federation" after the collapse of the Soviet Union.

This might be acceptable for some users; other users, however, may view both as part of a single historical narrative, so may want everything related with Russia to appear in the same topic.

Vanilla LDA cannot fix this problem, since it treats each document as a bag of words and no relations between words are considered. We will return to this "Russian" example in later sections after we review tree-based topic models in Section 2.2, which will give us the tools we need to solve this problem in Chapter 4.

Topic	Words
1	election, yeltsin, russian, political, party, democratic, russia, president, democracy, boris, country, south, years, month, government, vote, since, leader, presidential, military
2	new, york, city, state, mayor, budget, giuliani, council, cuomo, gov, plan, year, rudolph, dinkins, lead, need, governor, legislature, pataki, david
3	nuclear, arms, weapon, defense, treaty, missile, world, unite, yet, soviet, lead, secretary, would, control, korea, intelligence, test, nation, country, testing
4	president, bush, administration, clinton, american, force, reagan, war, unite, lead, economic, iraq, congress, america, iraqi, policy, aid, international, military, see
	⋮
20	soviet, lead, gorbachev, union, west, mikhail, reform, change, europe, leaders, poland, communist, know, old, right, human, washington, western, bring, party

Table 2.1: Five topics from 20 topics extracted by vanilla LDA on the editorials from the New York times. The Russian words (colored in red) and Soviet words (colored in blue) appear in Topic 1 and Topic 20 respectively.

2.2 Tree-based Topic Models

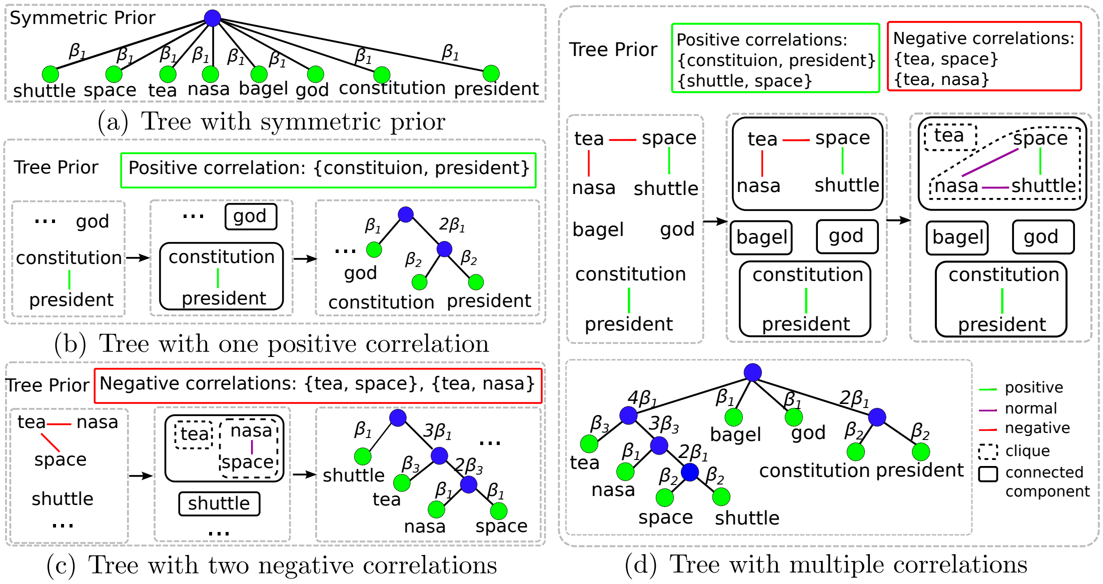
To consider the correlations between words, we turn to tree-structured distributions. Trees are a natural formalism for encoding lexical information. WordNet (Miller, 1990), a resource ubiquitous in natural language processing, is organized as a tree based on psychological evidence that human lexical memory is also represented as a tree. The structure of WordNet inspired Abney and Light (1999) to use tree-structured distributions to model selectional restrictions. Later, Boyd-Graber et al (2007) also used WordNet as a tree structure and put it into a fully Bayesian model for word sense disambiguation. Andrzejewski et al (2009) extended this statistical formalism to encode "must link" (positive correlations) and "cannot link" (negative correlations) correlations from domain experts.

We adopt Andrzejewski et al (2009)'s formalism for these two correlations to encode feedback to topic models. The first are positive correlations (PC), which encourage words to appear in the same topic; the second are the negative correlations (NC), which push words into different topics. In the remaining part of this section, we assume these correlations are from WordNet or dictionaries, and we introduce how to encode the correlations into a tree structure, the generative process and also the inference scheme. In Chapter 4, we discuss how to get these correlations from real users of topic models.

2.2.1 Encoding Correlations in a Tree

Given the priors, the first question is how do we go from positive or negative correlations to a tree? Any correlations (either positive or negative) without overlapping words can easily be encoded in a tree. Imagine the symmetric prior of the vanilla LDA as a very flat tree, where all words are children of the root directly with the same prior, as shown in Figure 2.1(a). To encode one correlation, replace all correlated words with a single new child node of the root, and then attach all the correlated words as children of this new node. Figure 2.1(b) shows how to encode a positive correlation between "constitution" and "president".

When words overlap in different correlations, one option is to treat them separately and create a new internal node for each correlation. This creates multiple paths for each word. This is often useful, as it reflects that tokens of the same word in different contexts can have different meanings. For example, when "drive" means a journey in a vehicle, it is associated with "ride", but when "drive" refers to the


Symmetric Prior
β_1
shuttle space tea nasa bagel god constitution president
Tree Prior
Positive correlation: {constituion, president}
god
constitution
president
β_1
$2\beta_1$
β_2
Tree Prior
Negative correlations: {tea, space}, {tea, nasa}
tea
nasa
space
shuttle
$3\beta_1$
β_3
$2\beta_3$
Tree Prior
Positive correlations: {constituion, president} {shuttle, space}
Negative correlations: {tea, space} {tea, nasa}
bagel
$4\beta_1$
$3\beta_3$
$2\beta_1$
positive
normal
negative
clique
connected component

(a) Tree with symmetric prior

(b) Tree with one positive correlation

(c) Tree with two negative correlations

(d) Tree with multiple correlations

Figure 2.1: Given a flat tree (2.1(a)) as in vanilla LDA (with hyperparameter $\beta = 0.01$ for the uncorrelated words), examples for adding single/multiple positive(in green)/negative(in red) correlations into the tree: generate a graph; detect the connected components; flip the negative edges of components; then detect the cliques; finally construct a prior tree. Figure 2.1(b) add one positive correlation, connect "constitution" and "president" to a new internal node, and then link this new node to the root, set $\beta_2 = 100$; Figure 2.1(c): add two negative correlations, set $\beta_3 = 10^{-6}$ to push "tea" away from "nasa" and "space", and use $\beta_1 = 0.01$ as the prior for the uncorrelated words "nasa" and "space". Figure 2.1(d) add two positive correlations and two negative correlations. A positive correlation between "shuttle" and "space" while a negative correlation between "tea" and "space", implies a negative correlation between "shuttle" and "tea", so "nasa", "space", "shuttle" will all be pushed away from "tea"; {"space", "shuttle"} and {"constitution", president"} are pulled together by β_2.

act of applying force to propel something, it is more connected with "thrust". As in lexical resources like WordNet (Miller, 1990), the path of a word in our tree represents its meaning; when a word appears in multiple correlations, this implies that it has multiple meanings. Each token's path represents its meaning.

Another option is to merge correlations. For positive correlations, this means that correlations are transitive. For negative correlations, it is a little more complex.

If we view negative correlations as completely transitive—placing all words under a sparse internal node—that would mean that only one of the words could appear in a topic. Taken to the illogical extreme where every word is involved in a negative correlation, each topic could only have one word.

Instead, for negative correlations, Andrzejewski et al (2009) view the negative correlations among words as a **correlation graph**; each word is a node and an edge is a negative correlation. To create a tree from this graph, we find all the connected components (Harary, 1969; Hopcroft and Tarjan, 1973) in the graph, and then for each component, we **flip** the edges between each pair of nodes: keep the positive edges, remove the negative edges, and add normal edges between any pair of nodes. [1] Then we run the Bron and Kerbosch (1973) algorithm to find all the cliques (a clique is a subset of vertices where every two vertices are connected by an edge) on the complement of the component (where nodes without an edge in the primal graph are connected in the complement and *vice versa*). To construct the tree, each component will be a child of the root and each clique will be child of the component.

This whole process is shown with examples in Figure 2.1(c): if we want to split "tea" and "space", "tea" and "nasa" at the same time, one graph component has three nodes for "tea", "space" and "nasa" and two negative edges between "tea" and "space", "tea" and "nasa". After flipping the edges of this component, there is only one edge left, which is between "space" and "nasa". So there are two cliques in this component: one includes "space" and "nasa", and the other is "tea", and the tree

[1] The normal edges between "A" and "B" should not be added, if there is a negative edge between "A" and "C", and a positive edge between "B" and "C". For example, in Figure 2.1(d), there is not a normal edge between "tea" and "shuttle", because there is a negative edge between "tea" and "shuttle", and a positive edge between "space" and "shuttle".

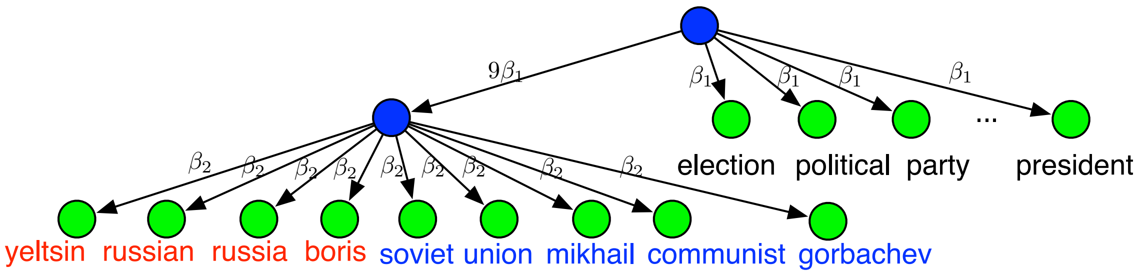


Figure 2.2: Prior tree for Russia Example: one positive correlation is added between the Russia words (in red) and Soviet words (in blue), and the constructed prior tree based on this correlation encourages the red words and blue words to appear in the same topic.

can be constructed as Figure 2.1(c).

Figure 2.1(d) shows a more complex example when there are overlapping words between positive and negative correlations. We first extract all the connected components; for components with negative edges, we do a flip (remove the negative edges, and add all possible normal edges); remove the unnecessary edges: there should be a normal edge between "tea" and "shuttle", but "tea" should be away from "space", while "space" and "shuttle" have a positive edge, so the edge between "tea" and "shuttle" is removed; then we detect all the cliques, and construct the prior tree as shown in Figure 2.1(d).

Now it is easy to fix the problem in "Russian" example in Section 2.1.3. To encourage the words related with "Russian Federation" (in red) and the words related with "Soviet Union" (in blue) to appear together in the same topic, we can simply add one positive correlation between the red words and blue words, and the tree prior can be constructed as Figure 2.2.

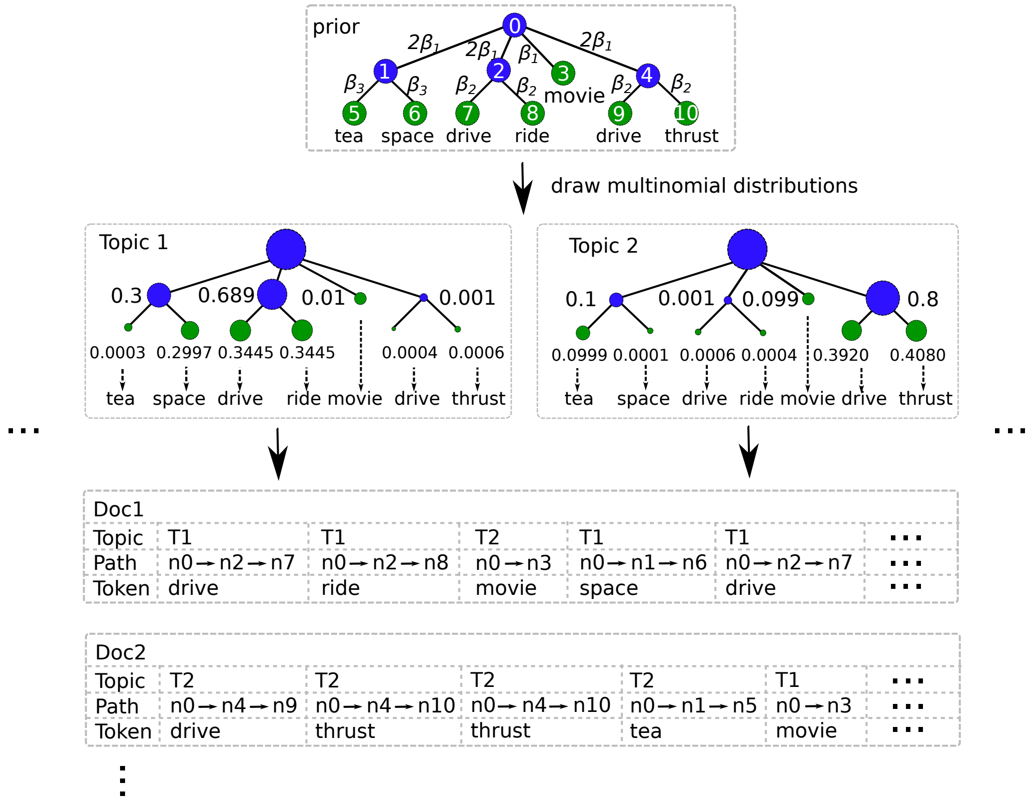
2.2.2 Generative Process

Given the prior tree structure that encodes correlations, words in positive correlations have high probability to be selected in the same topic. For example, in Figure 2.3, "drive" and "ride" are positively correlated and they both have high probability to be selected in Topic 1, while "drive" and "thrust" are both likely to be drawn in Topic 2. On the other hand, if two words are negatively correlated, when one word has high probability to appear in one topic ("space" in Topic 1 in Figure 2.3), the other word turns to unlikely be selected in the same topic ("tea" in Topic 1 in Figure 2.3).

These tree-structured distributions replace multinomial distributions for each of the K topics. Instead, we now have a set of multinomial distributions arranged in a tree (we will go deeper and describe the complete generative process that creates these distributions soon). Each leaf node is associated with a word, and each of the V words in vocabulary must appear in at least (and possibly more than) one leaf node.

One tree structure that satisfies this definition is a very flat tree with only one internal node with V leaf nodes, a child for every word. This is identical to vanilla LDA, as there is only a one V-dimensional multinomial distribution for each topic. The generative process for a token is simple: generate a word according to $w_{d,n} \sim \text{Mult}(\pi_{z_{d,n},\text{root}})$.

Now consider the non-degenerate case. To generate a token $w_{d,n}$ from topic k, we traverse a tree path $l_{d,n}$, which is a list of nodes starting at the root: we start at the



Doc1

Topic	T1	T1	T2	T1	T1	...
Path	n0→n2→n7	n0→n2→n8	n0→n3	n0→n1→n6	n0→n2→n7	...
Token	drive	ride	movie	space	drive	...

Doc2

Topic	T2	T2	T2	T2	T1	...
Path	n0→n4→n9	n0→n4→n10	n0→n4→n10	n0→n1→n5	n0→n3	...
Token	drive	thrust	thrust	tea	movie	...

Figure 2.3: Example of the generative process for drawing topics (first row to second row) and then drawing token observations from topics (second row to third row). In the second row, the size of the children nodes represents the probability in a multinomial distribution drawn from the parent node with the prior in the first row. Different hyperparameter settings shape the topics in different ways. For example, the node with the children "drive" and "ride" has a high transition prior β_2, which means that a topic will always have nearly equal probability for both (if the edge from the root to their internal node has high probability) or neither (if the edge from the root to their internal node has low probability). However, the node for "tea" and "space" has a small transition prior β_3, which means that a topic can only have either "tea" or "space", but not both. To generate a token, for example the first token of doc1 with topic indicator $z_{0,0} = 1$, we start at the root of topic 1: first draw a child of the root n_0, and assume we choose the child n_2; then continue to draw a child of node n_2, and we reach n_7, which is a leaf node and associated with word "drive".

root $l_{d,n}[0]$, select a child $l_{d,n}[1] \sim \text{Mult}(\pi_{k,l_{d,n}[0]})$; if the child is not a leaf node, select a child $l_{d,n}[2] \sim \text{Mult}(\pi_{k,l_{d,n}[1]})$; we keep selecting a child $l_{d,n}[i] \sim \text{Mult}(\pi_{k,l_{d,n}[i-1]})$ until we reach a leaf node, and then emit the leaf's associated word. This walk along the tree, which we represent using the latent variable $l_{d,n}$, replaces the single draw from a topic's multinomial distribution over words. The rest of the generative process remains the same as vanilla LDA with θ_d, the per-document topic multinomial, and $z_{d,n}$, the topic assignment for each token. The topic assignment $z_{d,n}$ selects which tree generates a token.

The multinomial distributions themselves are also random variables. Each transition distribution $\pi_{k,i}$ is drawn from a Dirichlet distribution parameterized by β_i, where i is the index of internal nodes in the tree structure. Choosing these Dirichlet priors specifies the direction (i.e., positive or negative) and strength of correlations that appear. Dirichlet distributions produce sparse multinomials when their parameters are small (less than one). This encourages an internal node to prefer few children. When the Dirichlet's parameters are large (all greater than one), it favors more uniform distributions.

Take Figure 2.3 as an example. To generate a topic, we draw multinomial distributions for each internal node. The whole tree gives a multinomial distribution over all paths for each topic. For any single multinomial distribution, the smaller the Dirichlet hyperparameter is, the more sparsity we get among children. Thus, any paths that pass through such a node face a "bottleneck", and only one (or a few) child will be possible choices once a path touches a node. For example, if we set β_3 to be very small, "tea" and "space" will not appear together in a topic. Conversely, for nodes with a large hyperparameter, the path is selected almost as if it were chosen uniformly at random. For example, "drive" and "thrust" are nearly equally likely once a path reaches Node 4, if β_2 is very large.

2.2.3 Relationship to Other Topic Models

Unlike hierarchical LDA (Blei et al, 2010)—which models structure over topics and each topic is still a distribution over all words—tree-based topic models add additional structure to topics' distribution over words, allowing subsets of the vocabulary to cluster together and expose recurring subtopics.

Tree-based topic models can be viewed as a special case of arbitrary pachinko allocation models (Li and Mccallum, 2006, PAM), Li and Mccallum (2006), however, focus on another special case of pachinko allocation models—a fully-connected hierarchy (four-level PAM). The four-level PAM is very different from tree-based topic models, because each topic is a distribution over all subtopics, and each subtopic is distribution over the entire vocabulary. However, tree-based topic models add structure on individual groups of words (not the entire vocabulary) and are derived from an external knowledge source.

This same distinction separates tree-based topic models from Polya urn model (Mimno et al, 2011), although their goals are similar. While Polya urn model does provide additional positive correlations, these positive correlations are learned from data and are not restricted to sets of words. In addition, Polya urn model cannot handle negative correlations.

While the prior tree of tree-based topic models is constructed given the positive and negative correlations, Polya trees (Lavine, 1992) also provide a prior tree structure, which imposes a Beta prior on the children and generates binary trees. However, the prior tree of tree-based topic models allows a "bushier" tree structure, which can also be automatically learned by the beta coalescent model discussed in Chapter 7.

Adding correlations to topic models can come to either the topic distribution over words or the document distribution over topics, as in correlated topic models (Blei and Lafferty, 2005; Mimno et al, 2008) and Markov random topic fields (Daumé III, 2009). While correlated topic models add a richer correlation structure, they have been shown not to improve perceived topic coherence (Chang et al, 2009).

2.2.4 Inference

Like the inference for vanilla LDA, as introduced in Section 2.1.2, we use Gibbs sampling for posterior inference (Neal, 1993; Resnik and Hardisty, 2010) to uncover the latent variables that best explain observed data.

For inference in tree-based topic models, the joint probability is

$$p(\mathbf{Z}, \mathbf{L}, \mathbf{W}, \pi, \theta; \alpha, \beta) = \tag{2.6}$$
$$\prod_k \prod_i \underbrace{p(\pi_{k,i}|\beta_i)}_{\text{transition}} \Big[\prod_d p(\theta_d|\alpha) \Big[\prod_n \underbrace{p(z_{d,n}|\theta_d)}_{\text{assignment}} \underbrace{p(l_{d,n}|\pi, z_{d,n})}_{\text{path}} \underbrace{p(w_{d,n}|l_{d,n})}_{\text{token}} \Big]\Big],$$

where i is an internal node in the tree. The probability of a path, $p(l_{d,n}|\pi, z_{d,n})$, is the tree structured walk described in Section 2.2, and the probability of a token being generated by a path, $p(w_{d,n}|l_{d,n})$ is one if the path $l_{d,n}$ terminates at leaf node with word $w_{d,n}$ and zero otherwise.

Because the conjugate prior of the multinomial is the Dirichlet, we can integrate out the transition distributions π and the per-document topic distributions θ in the conditional distribution

$$
\begin{aligned}
& p(z_{d,n} = k, l_{d,n} = \lambda | \mathbf{Z}_{-}, \mathbf{L}_{-}, \mathbf{W}; \alpha, \beta) && (2.7) \\
& = \frac{p(z_{d,n} = k, l_{d,n} = \lambda, \mathbf{Z}_{-}, \mathbf{L}_{-}, w_{d,n}; \alpha, \beta)}{p(\mathbf{Z}_{-}, \mathbf{L}_{-}; \alpha, \beta)} \\
& = p(w_{d,n}|\lambda) \underbrace{\frac{\int_{\theta} p(z_{d,n} = k, \mathbf{Z}_{-}|\theta) p(\theta; \alpha) d\theta}{\int_{\theta} p(\mathbf{Z}_{-}|\theta) p(\theta; \alpha) d\theta}}_{\text{topic assignment}} && (2.8) \\
& \qquad \underbrace{\prod_{(i \to j) \in \lambda} \frac{\int_{\pi_{k,i}} p((i \to j) \in \lambda, \mathbf{L}_{-}|\mathbf{Z}_{-}, z_{d,n}, \pi_{k,i}) p(\pi_{k,i}; \beta_i) d\pi_{k,i}}{\int_{\pi_{k,i}} p(\mathbf{L}_{-}|\mathbf{Z}_{-}, \pi_{k,i}) p(\pi_{k,i}; \beta_i) d\pi_{k,i}}}_{\text{path}} \\
& = \mathbb{I}\left[\Omega(\lambda) = w_{d,n}\right] \underbrace{p(z_{d,n} = k | \mathbf{Z}_{-}; \alpha)}_{\text{topic assignment}} \underbrace{p(l_{d,n} = \lambda | \mathbf{L}_{-}, \mathbf{Z}_{-}, z_{d,n}; \beta)}_{\text{path}}
\end{aligned}
$$

where $\mathbf{Z}_{-}$ are the topic assignments, $\mathbf{L}_{-}$ are the path assignments excluding the current token $w_{d,n}$, and the indicator function ensures that the path $\lambda_{d,n}$ ends in a path consistent with the token $w_{d,n}$. Using conjugacy, the two integrals—canceling Gamma functions from the Dirichlet normalizer that appear in both numerator and denominator—are

$$
p(l_{d,n} = \lambda | \mathbf{L}_{-}, z_{d,n} = k, w_{d,n}; \beta) = \frac{\prod_{(i,j) \in \lambda} \frac{\Gamma(n_{i \to j|k} + \beta_{i \to j} + 1)}{\Gamma\left(\sum_{j'} (n_{i \to j'|k} + \beta_{i \to j'}) + 1\right)}}{\prod_{(i,j) \in \lambda} \frac{\Gamma(n_{i \to j|k} + \beta_{i \to j})}{\Gamma\left(\sum_{j'} (n_{i \to j'|k} + \beta_{i \to j'})\right)}} \tag{2.9}
$$

$$
p(z_{d,n} = k | \mathbf{Z}_{-}; \alpha) = \frac{\frac{\Gamma(n_{k|d} + \alpha_k + 1)}{\Gamma\left(\sum_{k'} (n_{k'|d} + \alpha_{k'}) + 1\right)}}{\frac{\Gamma(n_{k|d} + \alpha_k)}{\Gamma\left(\sum_{k'} (n_{k'|d} + \alpha_{k'})\right)}} \tag{2.10}
$$

where $\beta_{i \to j}$ is the prior for edge $i \to j$, $n_{i \to j|k}$ is the number of paths that go from node i to node j in topic k. All the other terms are the same as in vanilla LDA: $n_{k|d}$

is topic k's count in the document d, and α is the per-document Dirichlet parameter.

With additional cancellations, we can remove the remaining Gamma functions to obtain the conditional distribution[2]

$$p(z = k, l_w = \lambda | \mathbf{Z}_-, \mathbf{L}_-, w) \propto (\alpha_k + n_{k|d}) \prod_{(i \to j) \in \lambda} \frac{\beta_{i \to j} + n_{i \to j|k}}{\sum_{j'} (\beta_{i \to j'} + n_{i \to j'|k})}. \tag{2.11}$$

2.2.5 Problems

Now we can come back to the problem in the 'Russian" example that we have discussed in Section 2.1.3. To fix the problem, the first question is **where we can get the correlations to construct the prior tree in Figure 2.2?** In this dissertation, we discuss to get these correlations (prior knowledge) from actual users of topic models (Chapter 4), existing knowledge resources (Chapter 5) , or automatically learning the prior knowledge from data directly (Chapter 7).

Given the prior tree as in Figure 2.2, the second problem is **how we can learn topics efficiently?** We can certainly run tree-based topic models we just discussed to learn new topics. However, the complexity of computing the sampling distribution is increased to $O(KLS)$ for models with K topics, paths at most L nodes long, and at most S paths per word. In contrast, computing the analogous conditional sampling distribution for vanilla LDA has complexity $O(K)$. As a result, tree-based topic models consider correlations between words at the cost of more complex inference. To do this more efficiently, this dissertation presents an efficient tree-based topic

[2] In this and future equations we will omit the indicator function that ensures paths end in the required token $w_{d,n}$ by using l_w instead of l. In addition, we omit the subscript $_{d,n}$ from z and l, as all future appearances of these random variables will be associated with single token.

models in Chapter 3; we discuss how to start from the initial topics (Table 2.1) and update the topics given the new prior (Chapter 4); and we also propose more efficient spectral methods to consider prior knowledge in topic models (Chapter 6).

The updated topics of this "Russian" example will be given in Table 4.1 in Chapter 4. Next we discuss about efficient inference for tree-based topic models in the next chapter (Chapter 3).

Chapter 3

Efficient Inference for Tree-based Topic Models

Tree-based topic models consider correlations between words but result in more complex inference, thus it is not sufficient to serve users in an interactive setting, which we will introduce in Chapter 4. Any model involving interaction with users should be as **computationally efficient** as possible to minimize users' waiting time. In particular, Thomas and Cook (2005), summarizing a decade of human-computer interaction research, argue that a system should respond to a user on the order of one second for a simple response, such as clicking a web link, and on the order of ten seconds for a response from a complex user-initiated activity. This requirement demands that we develop more efficient inference for tree-based topic models.

Two widely used inference for topic models are *Gibbs sampling* (Neal, 1993) and *variational Bayesian inference* (Blei et al, 2003b). While both frameworks produce good approximations of the posterior, the latter, though with larger throughput, has higher latency. In addition, Gibbs sampling's sparsity is another attractive property, which can be further used to improve efficiency (Yao et al, 2009). As a result, we focus on improving the efficiency of Gibbs sampling.

SparseLDA (Yao et al, 2009) is an efficient Gibbs sampling algorithm for LDA based on a refactorization of the conditional topic distribution. However, it is not directly applicable to tree-based topic models. In this chapter, we first review

SparseLDA (Yao et al, 2009) and provide a factorization for tree-based models within a broadly applicable inference framework that improves inference efficiency.

3.1 Sparse LDA

The SparseLDA (Yao et al, 2009) scheme for speeding inference begins by rearranging vanilla LDA's sampling equation (Equation 2.5) as

$$\begin{aligned} p(z=k|Z_-, w) &\propto (\alpha_k + n_{k|d})\frac{\beta + n_{w|k}}{\beta V + n_{\cdot|k}} \\ &= \underbrace{\frac{\alpha_k \beta}{\beta V + n_{\cdot|k}}}_{s_{\text{LDA}}} + \underbrace{\frac{n_{k|d}\beta}{\beta V + n_{\cdot|k}}}_{r_{\text{LDA}}} + \underbrace{\frac{(\alpha_k + n_{k|d})n_{w|k}}{\beta V + n_{\cdot|k}}}_{q_{\text{LDA}}}. \end{aligned} \tag{3.1}$$

Following their lead, we call these three terms "buckets". A bucket is the *total* probability mass marginalizing over latent variable assignments (i.e., $s_{\text{LDA}} \equiv \sum_k \frac{\alpha_k \beta}{\beta V + n_{\cdot|k}}$, similarly for the other buckets). The three buckets are: a smoothing-only bucket s_{LDA} with Dirichlet prior α_k and β which contributes to every topic in every document; a document-topic bucket r_{LDA}, which is only non-zero for topics that appear in a document (non-zero $n_{k|d}$); and the topic-word bucket q_{LDA}, which is only non-zero for words that appear in a topic (non-zero $n_w|k$). The three buckets sum to one, so this is a "reshuffling" of the original conditional probability distribution.

Then the inference is changed to a two-step process: instead of computing the probability mass for each topic, we first compute the probability for each topic in each of the three buckets; then we randomly sample *which* bucket we need and then (and only then) select a topic *within* that bucket, as shown in Figure 3.1. Because

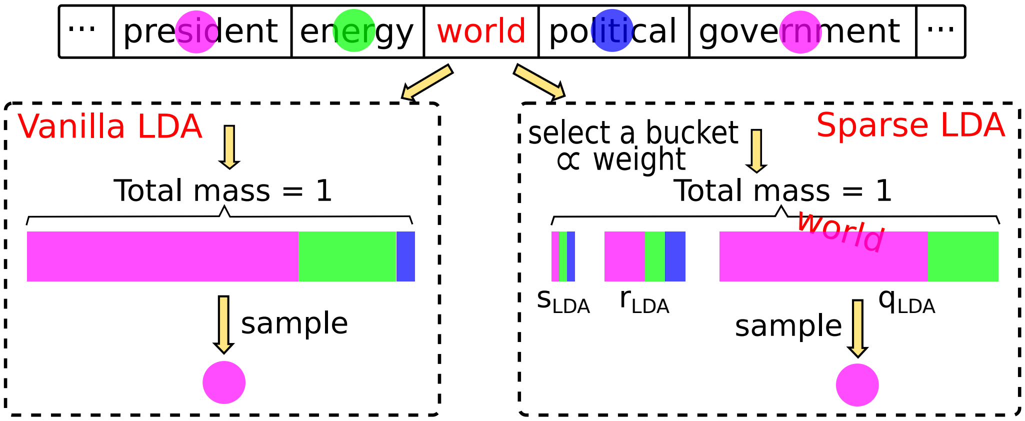


Figure 3.1: Comparison of inference between vanilla LDA and sparse LDA: vanilla LDA computes the probability for each topic (sums to 1) and sample a topic; sparse LDA computes the probability for each topic in the three buckets separately (total still sums to 1), so it will select a bucket proportional to its weight, and then sample a topic within the selected bucket. Because s_{LDA} is shared by all tokens and r_{LDA} is shared by all tokens in a document, both of them can be cached to save computation. q_{LDA} only includes topics with nonzero count (only the pink and green topics in this example), which is very sparse in practice, so it saves computation greatly.

we are still sampling from the same conditional distribution, this does not change the underlying sampling algorithm.

At this point, it might seem counterintuitive that this scheme should improve inference, as we sample from $3K$ (three buckets for each topic) possible outcomes rather than K outcomes as before. However, while all topics have non-zero contribution to s_{LDA}, this smoothing-only bucket s_{LDA} is shared across the corpus, thus we only need to compute it once, cache it, and then efficiently update the bucket total probabilities in constant time (in contrast to $O(K)$ construction of the conditional distribution in traditional LDA sampling schemes). Similarly, the document-topic r_{LDA} is shared by all words across a document, and we only need to consider the topics that have non-zero contributions in this document, thus it can be computed efficiently and cached for each document and updated easily. The topic-word bucket

q_{LDA} has to be computed specifically for each token, but only for the (typically) few topics that a word has non-zero counts, which is very sparse. Because q_{LDA} often has the largest mass and has few non-zero terms, this speeds inference.

Yao et al (2009) propose to further speedup by sampling topics within a bucket in *descending* probability. The information needed to compute a probability within a bucket is stored in an array in decreasing order of probability mass. Thus, on average, after selecting one of the three buckets, only a handful of topics need to be explicitly considered. To maintain $(topic, count)$ tuples in sorted order within a bucket more efficiently, the topic and the count are packed into one integer (count in higher-order bits and topic in lower-order bits). Because a count change is only a small shift in the overall ordering, a bubble sort (Astrachan, 2003) returns the array to sorted order in $O(n)$.

3.2 Efficient Sampling for Tree-based Topic Models

While tree-based topic models are more complicated than vanilla LDA, our model enjoys much of the same sparsity: each topic has a limited number of words seen in a corpus, and each document has only a handful topics. In this section, we take advantage of that sparsity to extend the sampling techniques for SparseLDA to the tree-based topic models.

To match the form of Equation 3.1, we first define

$$\begin{aligned}
N_{k,\lambda} &= \prod_{(i \to j) \in \lambda} \sum_{j'} (\beta_{i \to j'} + n_{i \to j'|k}) \\
S_{\lambda} &= \prod_{(i \to j) \in \lambda} \beta_{i \to j} \\
O_{k,\lambda} &= \prod_{(i \to j) \in \lambda} (\beta_{i \to j} + n_{i \to j|k}) - \prod_{(i \to j) \in \lambda} \beta_{i \to j}.
\end{aligned} \tag{3.2}$$

We call $N_{k,\lambda}$ the normalizer for path λ in topic k, S_λ the smoothing factor for path λ, and $O_{k,\lambda}$ the observation for path λ in topic k. Notice $N_{k,\lambda}$ and $O_{k,\lambda}$ are path and topic specific, and S_λ is specified for each path. Then we refactor Equation 2.11 as in Equation 3.3, yielding buckets analogous to SparseLDA's,

$$\begin{aligned}
p(z = k, l = \lambda | Z_-, L_-, w) &\propto (\alpha_k + n_{k|d}) \prod_{(i \to j) \in \lambda} \frac{\beta_{i \to j} + n_{i \to j|k}}{\sum_{j'} (\beta_{i \to j'} + n_{i \to j'|k})} \qquad (3.3) \\
&\propto (\alpha_k + n_{k|d}) N_{k,\lambda}^{-1} [S_\lambda + O_{k,\lambda}] \\
&\propto \underbrace{\frac{\alpha_k S_\lambda}{N_{k,\lambda}}}_{s} + \underbrace{\frac{n_{k|d} S_\lambda}{N_{k,\lambda}}}_{r} + \underbrace{\frac{(\alpha_k + n_{k|d}) O_{k,\lambda}}{N_{k,\lambda}}}_{q}.
\end{aligned}$$

where the buckets s, r and q are,

$$s \equiv \sum_{k,\lambda} \frac{\alpha_k S_\lambda}{N_{k,\lambda}} \qquad r \equiv \sum_{k,\lambda} \frac{n_{k|d} S_\lambda}{N_{k,\lambda}} \qquad q \equiv \sum_{k,\lambda} \frac{(\alpha_k + n_{k|d}) O_{k,\lambda}}{N_{k,\lambda}} \tag{3.4}$$

We use the same bucket names without the subscript "LDA" from SparseLDA. Unlike SparseLDA, each bucket sums over the probability of not only the topics but also paths. However, the sampling process is much the same as for SparseLDA:

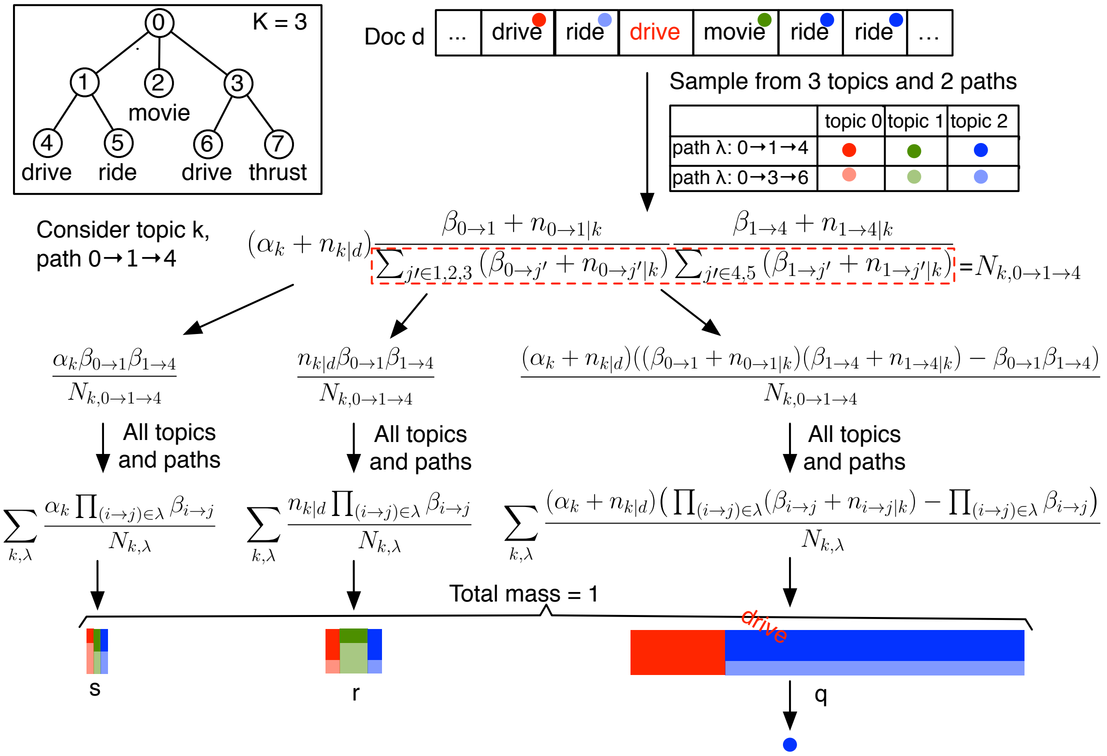

K = 3
0
1
2
3
4
5
6
7
movie
drive
ride
drive
thrust
Doc d
...
drive
ride
drive
movie
ride
ride
...
Sample from 3 topics and 2 paths
topic 0
topic 1
topic 2
path λ: 0→1→4
path λ: 0→3→6
Consider topic k, path 0→1→4
$(\alpha_k + n_{k|d}) \frac{\beta_{0\to1} + n_{0\to1|k}}{\sum_{j'\in1,2,3}(\beta_{0\to j'} + n_{0\to j'|k})} \frac{\beta_{1\to4} + n_{1\to4|k}}{\sum_{j'\in4,5}(\beta_{1\to j'} + n_{1\to j'|k})}$
$=N_{k,0\to1\to4}$
$\frac{\alpha_k \beta_{0\to1}\beta_{1\to4}}{N_{k,0\to1\to4}}$
$\frac{n_{k|d} \beta_{0\to1}\beta_{1\to4}}{N_{k,0\to1\to4}}$
$\frac{(\alpha_k + n_{k|d})((\beta_{0\to1} + n_{0\to1|k})(\beta_{1\to4} + n_{1\to4|k}) - \beta_{0\to1}\beta_{1\to4})}{N_{k,0\to1\to4}}$
All topics and paths
All topics and paths
All topics and paths
$\sum_{k,\lambda} \frac{\alpha_k \prod_{(i\to j)\in\lambda} \beta_{i\to j}}{N_{k,\lambda}}$
$\sum_{k,\lambda} \frac{n_{k|d} \prod_{(i\to j)\in\lambda} \beta_{i\to j}}{N_{k,\lambda}}$
$\sum_{k,\lambda} \frac{(\alpha_k + n_{k|d})\left(\prod_{(i\to j)\in\lambda}(\beta_{i\to j} + n_{i\to j|k}) - \prod_{(i\to j)\in\lambda} \beta_{i\to j}\right)}{N_{k,\lambda}}$
Total mass = 1
drive
s
r
q

Figure 3.2: An example of efficient inference for tree-based topic models: color denotes different topics; and the shade denotes the paths; like SparseLDA, we need to compute the three buckets, but instead of just considering all topics, we need to consider all topics and paths. First select a bucket proportional to the probability mass, and then sample a topic and a path within the selected bucket. The normalizer N_{k_λ} changes for each path, as s and r are not shared by multiple tokens. Because r only includes the terms where $n_{k|d}$ is non-zero, and q only includes the terms where any of $n_{i\to j|k}$ along path λ is non-zero, which implies the part $\prod_{(i\to j)\in\lambda}(\beta_{i\to j} + n_{i\to j|k}) - \prod_{(i\to j)\in\lambda}\beta_{i\to j}$ is non-zero (only the red and blue topics in this example). Both are sparse in practice, so it reduces computation time.

select *which* bucket and then select a topic and path combination *within* the bucket. The resulting algorithm is Algorithm 1. Figure 3.2 shows a specific example of this proposed inference. However, the correlations introduced by the tree-based structure complicate inference.

One of the benefits of SparseLDA was that the smoothing-only bucket s is shared across tokens in a corpus and thus need not be recomputed. This is no longer possible, as $N_{k,\lambda}$ is distinct for each path in tree-based LDA. This negates the benefit of caching the smoothing-only bucket s, but we recover some of the benefits by caching and updating the normalizer $N_{k,\lambda}$ rather than the bucket s. We split the normalizer to two parts: the "root" normalizer from the root node (shared by all paths) and the "downstream" normalizer,

$$N_{k,\lambda} = \underbrace{\sum_{j'} (\beta_{root \to j'} + n_{root \to j'|k})}_{\text{root normalizer}} \cdot \underbrace{\prod_{(i \to j) \in \lambda'} \sum_{j'} (\beta_{i \to j'} + n_{i \to j'|k})}_{\text{downstream normalizer}} \tag{3.5}$$

where λ' denotes the path excluding the root. The root normalizer only considers the children of root, and it is shared by all tokens. As a result, we can cache it and update it in constant time. The downstream normalizer considers the remaining part of the normalizer, and it is needed only for correlated words (i.e., words that have been placed in correlations); in many situations it is reasonable to assume that these are relatively few (compared to the overall size of the vocabulary). This normalizer splitting saves memory and improves computation efficiency.

A second problem is that the normalizer $N_{k,\lambda}$ is coupled; changing transition count $n_{i \to j|k}$ in one path changes the normalizers of all cousin paths (paths that share at least one node i). Take Figure 2.1 (left middle) as an example: the paths for "constitution" and "president" are coupled, because they share an edge. When we change the count for each edge along the path of "constitution", the count of the shared edge is changed, so that both downstream normalizers will be changed. For this problem, we precompute which paths share downstream normalizers; all paths are partitioned into cousin sets, defined as sets for which changing the count of one member of the set changes the downstream normalizer of other paths in the set. Thus, when updating the counts for path λ, we only recompute $N_{k,\lambda'}$ for all λ' in the cousin set.

In addition, SparseLDA's computation of q, the topic word bucket, benefits from topics with unobserved (i.e., zero count) words. In our case, any non-zero path—a path with *any* non-zero edge—contributes to the probability mass of bucket q (notice a path might have zero path count but non-zero edges). To quickly determine whether a path contributes, we introduce an **edge-masked count** (EMC) for each path. Higher order bits encode whether edges have been observed and lower order bits encode the number of times the path has been observed. For example, in Figure 2.1 (left bottom), if we use 8 bits for EMC and observed the path ending in "space" seven times and "nasa" zero times, the EMC for "space" is $\overline{111}00111$, and the EMC for "nasa" is $\overline{11}000000$, since the first two edges of the path ending at "nasa" have been traversed.

3.3 Sorting Paths

Encoding the path using EMC allows us to extend SparseLDA's sorting strategy to consider latent variable assignments in *decreasing* order of probability mass. Unlike SparseLDA, our latent space is richer; we must sample both a path l and a topic z. Considering fewer possible assignments can speed sampling at the cost of the overhead of maintaining sorted data structures.

Sorting topic and path prominence for a word (sT) can improve our ability to sample from q. If we rank the topic and path pairs for a word in the decreasing order of edge-masked count (EMC), the order serves as a proxy of ranking the topic and path pairs by their probability mass. That is, when sampling a topic and path from q, we sample based on the decreasing EMC, which roughly correlates with path probability. Thus, we will on average choose our sample from the conditional distribution more quickly.

Recall that SparseLDA packs the topic and count into one integer to sort more efficiently. We cannot directly extend this because we need to pack topic, path, and EMC together, and EMC is already a packed integer. Instead, we pack topic and path into one integer, and sort an integer pair (EMC, topic-path integer) together according to the value of EMC.

Using Figure 2.1(left bottom) as example, if we use 8 bits for EMC and 8 bits for packing topic and path, and assume we observe the path of "space" (path index 3) seven times and "nasa" (path index 4) zero times in topic 5, the integer pair for "space" is $(\overline{111}00111, \overline{0101}0011)$ and for "nasa" is $(\overline{110}00000, \overline{0101}0100)$. Like SparseLDA, since we only need to update the count by either increasing one or decreasing one, we can use bubble sort to maintain sorted order.

Sorting topics' prominence within a document (sD) can improve sampling from the document-topic bucket r; when we need to sample within a bucket, we consider paths in decreasing order of the document-topic count $n_{k|d}$, so we can identify a topic and path more quickly if the bucket r is selected.

Algorithm 1 Efficient sampling

```
for token w in this document do
    sample = rand() $*(s + r + q)$
    if sample $< s$ then
        return topic $k$, path $\lambda$ sampled from $s$
    sample $- = s$
    if sample $< r$ then
        return topic $k$, path $\lambda$ sampled from $r$
    sample $- = r$
    return topic $k$, path $\lambda$ sampled from $q$
```

Algorithm 2 Efficient CRB sampling

```
for token w in this document do
    sample = rand() $*(s' + r + q)$
    if sample $< s'$ then
        compute $s$
        sample $* = (s + r + q)/(s' + r + q)$
        if sample $< s$ then
            return topic $k$, path $\lambda$ sampled from $s$
        sample $- = s$
    sample $- = s'$
    if sample $< r$ then
        return topic $k$, path $\lambda$ sampled from $r$
    sample $- = r$
    return topic $k$, path $\lambda$ sampled from $q$
```

3.4 Efficient Sampling with Coarse-to-Refined Buckets

While refactoring and caching the normalizers as described in Section 3.2 improves efficiency, the gains are disappointing. This is because while the smoothing-only bucket s is small, recomputing it is expensive because it requires us to consider all topics and paths (Equation 3.4). This is not a problem for SparseLDA because s is shared across all tokens.

However, when the counts of each edge per topic are all zero, the prior on bucket s gives an obvious upper bound,

$$s = \sum_{k,\lambda} \frac{\alpha_k \prod_{(i \to j) \in \lambda} \beta_{i \to j}}{\prod_{(i \to j) \in \lambda} \sum_{j'} (\beta_{i \to j'} + n_{i \to j'|k})} \leq \sum_{k,\lambda} \frac{\alpha_k \prod_{(i \to j) \in \lambda} \beta_{i \to j}}{\prod_{(i \to j) \in \lambda} \sum_{j'} \beta_{i \to j'}} = s'. \tag{3.6}$$

A sampling algorithm can take advantage of this upper bound by not explicitly calculating s, which we call sampling with Coarse-to-Refined Bucket (CRB). Instead, we use a larger s' as proxy, and only compute the smaller refined bucket s if and only if we hit the coarse bucket s' (Algorithm 2). No accuracy is sacrificed for efficiency in this algorithm.

As shown in Algorithm 2, when we sample a bucket, if it is not the coarse bucket s', we sample a topic and a path based on the other two buckets (these are always explicitly computed, but their sparsity helps); when we choose the coarse bucket s', we will explicitly compute the refined bucket s and sample based on the correct probabilities in the refine bucket s. This approximation does not sacrifice accuracy, as we always sample from the true distribution if our sample lands in the approximation gap $s' - s$, but we gain efficiency as samples often do not land in the smoothing-only bucket s or even in its coarse approximation s'. This whole process is shown in Figure 3.3.

3.5 Measuring Inference Time Efficiency

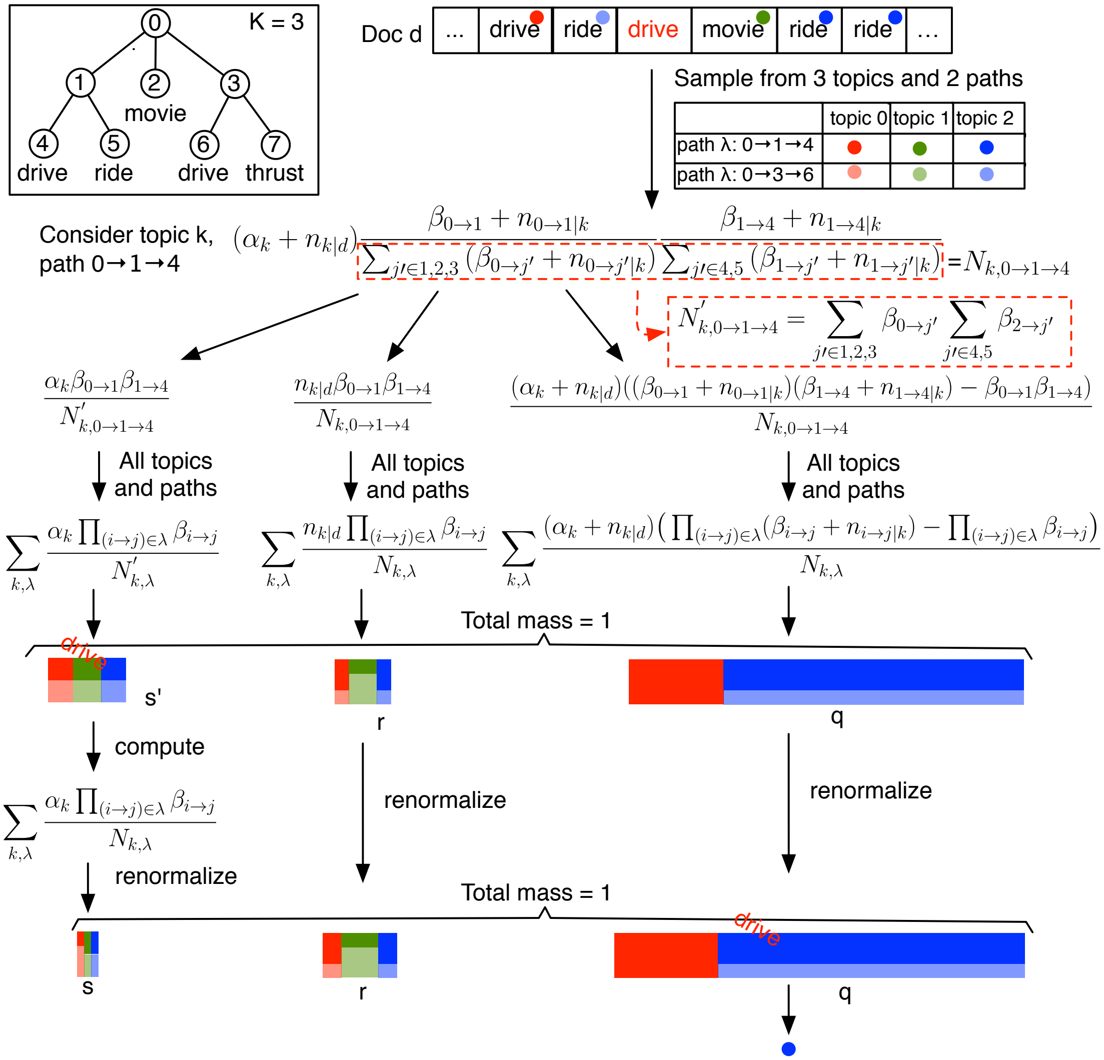
In this section, we compare the running time[1] of our proposed sampling algorithms Fast and Fast-CRB against the unfactored Gibbs sampler (Naïve) and in addition examine the effect of sorting.

The first corpus we use is the 20 Newsgroups corpus (20News),[2] which contains 18770 documents (originally 18846 documents, short documents are deleted) divided into 20 constituent newsgroups, 9743 words, and 632032 tokens. In addition, we use editorials from the New York Times (NYT) from 1987 to 1996, including 13284 documents, 41554 words, and 2714634 tokens.

For both datasets, we rank words by average tf-idf and choose the top V words as the vocabulary. Tokenization, lemmatization, and stopword removal was

[1] Mean of five chains on a 6-Core 2.8-GHz CPU, 16GB RAM.

[2] http://people.csail.mit.edu/jrennie/20Newsgroups/


K = 3
0
1
2
3
4
5
6
7
movie
drive
ride
drive
thrust
Doc d
...
drive
ride
drive
movie
ride
ride
...
Sample from 3 topics and 2 paths
Consider topic k, path 0→1→4
$(\alpha_k + n_{k|d})\frac{\beta_{0\to1} + n_{0\to1|k}}{\sum_{j'\in 1,2,3}(\beta_{0\to j'} + n_{0\to j'|k})}\frac{\beta_{1\to4} + n_{1\to4|k}}{\sum_{j'\in 4,5}(\beta_{1\to j'} + n_{1\to j'|k})} = N_{k,0\to1\to4}$
$N'_{k,0\to1\to4} = \sum_{j'\in 1,2,3}\beta_{0\to j'}\sum_{j'\in 4,5}\beta_{2\to j'}$
$\frac{\alpha_k\beta_{0\to1}\beta_{1\to4}}{N'_{k,0\to1\to4}}$
$\frac{n_{k|d}\beta_{0\to1}\beta_{1\to4}}{N_{k,0\to1\to4}}$
$\frac{(\alpha_k + n_{k|d})((\beta_{0\to1} + n_{0\to1|k})(\beta_{1\to4} + n_{1\to4|k}) - \beta_{0\to1}\beta_{1\to4})}{N_{k,0\to1\to4}}$
All topics and paths
All topics and paths
All topics and paths
$\sum_{k,\lambda}\frac{\alpha_k\prod_{(i\to j)\in\lambda}\beta_{i\to j}}{N'_{k,\lambda}}$
$\sum_{k,\lambda}\frac{n_{k|d}\prod_{(i\to j)\in\lambda}\beta_{i\to j}}{N_{k,\lambda}}$
$\sum_{k,\lambda}\frac{(\alpha_k + n_{k|d})(\prod_{(i\to j)\in\lambda}(\beta_{i\to j} + n_{i\to j|k}) - \prod_{(i\to j)\in\lambda}\beta_{i\to j})}{N_{k,\lambda}}$
Total mass = 1
drive
s'
r
q
compute
renormalize
renormalize
$\sum_{k,\lambda}\frac{\alpha_k\prod_{(i\to j)\in\lambda}\beta_{i\to j}}{N_{k,\lambda}}$
renormalize
Total mass = 1
drive
s
r
q

	topic 0	topic 1	topic 2
path λ: 0→1→4	●	●	●
path λ: 0→3→6	●	●	●

Figure 3.3: An example of sampling with coarse-to-refined buckets. Computing the exact smoothing-only s bucket in Figure 3.2 needs to go over all topics and paths, which is time-consuming. Instead, we use an upper bound of s initially. We call this the coarse bucket s'; if the current token doesn't land in this coarse bucket, we can just sample a topic and a path in the other two buckets as before; only when the token lands in this coarse bucket do we compute the actual bucket s. We compute the true normalized distribution then resample a topic and a path.

performed using the Natural Language Toolkit (Loper and Bird, 2002). We use WordNet 3.0 to generate correlations between words. WordNet organizes words into sets of synonyms called synsets. For each synset, we generate a subtree with

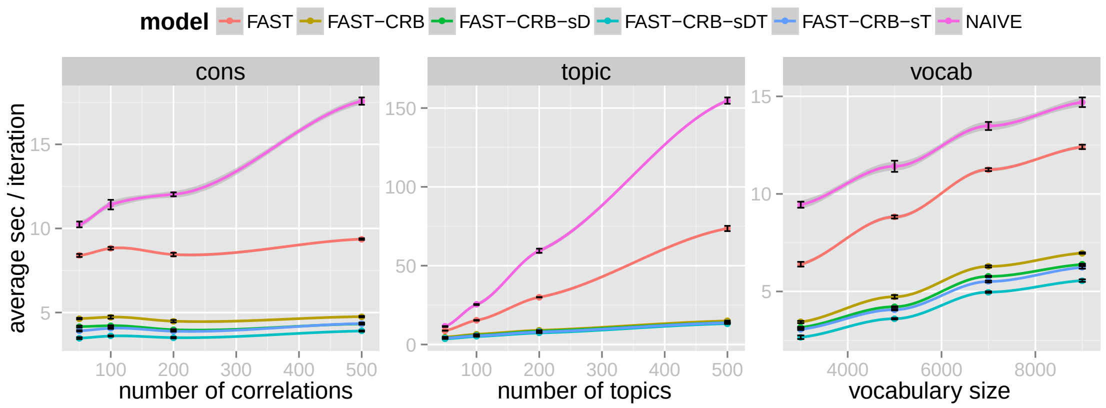


Figure 3.4: 20 newsgroups' average running time per iteration (Sec) over 100 iterations, averaged over 5 seeds. Experiments begin with 50 topics, 100 correlations, vocab size 5000 and then vary one dimension: number of correlations (left), number of topics (middle), and vocabulary size (right).

all words in the synset—that are also in our vocabulary—as leaves connected to a common parent. This subtree's common parent is then attached to the root node. The generated correlations include {"drive", "ride", "riding"}, {"drive", "repel"}, etc., which represents different senses of word "drive".

The hyperparameters for all experiments are $\alpha = 0.1$, $\beta = 0.01$ for uncorrelated words, $\beta = 100$ for the correlated words. However, sampling hyperparameters often (but not always) undoes the correlations (by making β for correlations comparable to β for uncorrelated words), so we keep the hyperparameters fixed.

We compared the FAST and FAST-CRB against NAÏVE (Figure 3.4 and Figure 3.5) on different numbers of topics, various vocabulary sizes and different numbers of correlations. For both datasets, FAST is consistently faster than NAÏVE and FAST-CRB is consistently faster than FAST. Their benefits are clearer as distributions become sparse (e.g., the first iteration for FAST is slower than later iterations). Gains

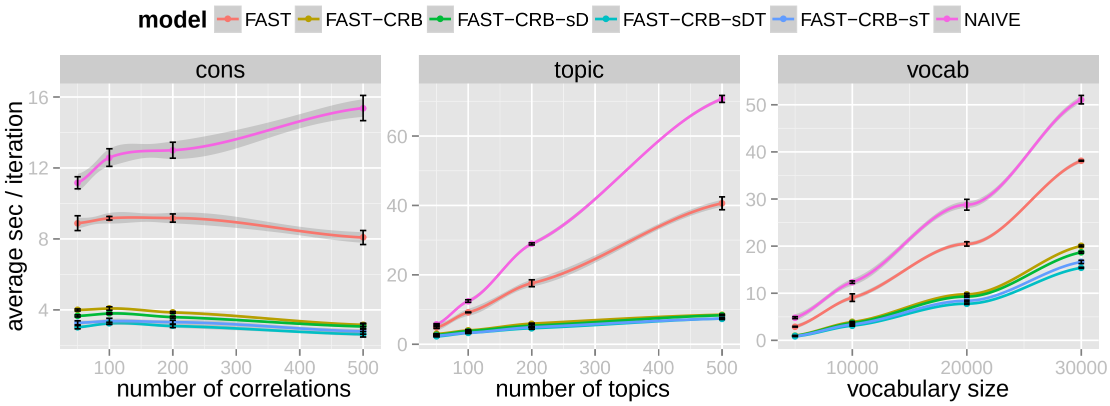


Figure 3.5: New York Times' average running time per iteration (Sec) over 100 iterations, averaged over 5 seeds. Experiments begin with 100 topics, 100 correlations, vocab size 10000 and then vary one dimension: number of correlations (left), number of topics (middle), and vocabulary size (right).

grow as the topic number increases, but diminish with larger vocabulary size. While both sorting strategies reduce time, sorting topics and paths for a word (sT) helps more than sorting topics in a document (sD), and combining the two is (with one exception) better than either alone.

Although 20News is smaller than NYT, inference on 20News is slower than on NYT for different number of topics and correlations. It is because NYT has a lot of words with high tf-idf score but low frequency. When we filter the dataset using the vocabulary ranked by tf-idf, a lot of high frequency words are filtered out, resulted in less remaining tokens in NYT than in 20News. Also, 20News has much more words with multiple paths, and this sometimes prevents the techniques of this section from speeding inference.

As more correlations are added, Naïve's time increases while that of Fast-CRB decreases in NYT dataset (Figure 3.5). This is because the number of non-zero

	C50	C100	C200	C500
correlated	1.306	1.494	1.904	1.735
uncorrelated	14.419	14.294	13.858	11.516

Table 3.1: The total number of non-zero paths for correlated words averaged over the number of tokens with correlated words (first row), and the same value for uncorrelated words (second row), as the number of correlations increases. When the number of correlations increases, the averaged value for correlated words doesn't change much while the averaged value for uncorrelated words decreases. It is because the number of non-zero paths for uncorrelated words decreases as more correlations are added to the model.

paths for uncorrelated words decreases as more correlations are added to the model. Since our techniques save computation for every zero path, the overall computation decreases as correlations push uncorrelated words to a limited number of topics (Table 3.1).

3.6 Summary

This chapter presents an efficient inference scheme for tree-based topic models, which consider the correlations between words efficiently into topic models. Given such as an efficient backend model, we expose this model to users and propose interactive topic modeling in Chapter 4, where we will get the feedback from users and use the feedback as correlations between words to improve the topics according to users' needs. We also extend this model and apply them in domain adaptation for statistical machine translation in Chapter 5.

While this chapter utilizes the sparsity of topic models to improve the efficiency of Gibbs sampling inference, we also explore another polynomial alternative model—spectral method for topic models—in Chapter 6, which potentially can also encode word correlations, but is less sensitive to local optima and more efficient.

Chapter 4

Interactive Topic Modeling

While efficient tree-based topic models—discussed in Chapter 3, encode the correlations between words in topic models—the next question is where we can get these correlations. One option is to use existing knowledge resources such as WordNet in Chapter 3.5 to generate the correlations, but it is not always what users want. Because users are the ultimate judge to decide whether a topic is "good" or "bad", this chapter describes how to get these correlations from users.

While users do not need to know the details of the backend models, it is essential that users can see the output topics, give feedback, and continue to update the topics iteratively. This suggests that topic models should be **interactive**. In this chapter, we describe interactive topic modeling (ITM), which combines the efficient tree-based topic model described in the previous chapter, with machine learning techniques to incorporate user feedback in an interactive exploration of a corpus. We validate the framework both with simulated and real users. Moreover, we conduct a user study using a legislative corpus, and compare how users, armed with either ITM or vanilla topic models, explore a legislative dataset to answer questions about political policies. In addition, we also build on heuristics proposed for topic coherence to suggest correlations automatically.

4.1 How Users Can Benefit from Interactive Topic Models

Users of topic models are the ultimate judge of whether a topic is "good" or "bad". In this section, we discuss what it means to have effective topic models, effective topic modeling software, and how these technologies interact with users. After identifying a set of goals that interactive topic models should fulfill, we present two real scenarios where interactive topic models can help a user cope with a surfeit of data.

Are you a good topic or a bad topic? A user might echo one of the many complaints lodged against topic models: these documents should have similar topics but do not (Daumé III, 2009); this topic should have syntactic coherence (Gruber et al, 2007; Boyd-Graber and Blei, 2008); this topic makes no sense at all (Newman et al, 2010); this topic shouldn't be associated with this document but is (Ramage et al, 2009); these words shouldn't be the in same topic but are (Andrzejewski et al, 2009); or these words should be in the same topic but are not (Andrzejewski et al, 2009).

Many of these complaints can be corrected by encouraging topic models through correlations (Chapter 2.2). Good topics can be encouraged by having correlations that link together words the model separated. A bad topic can be discouraged by correlations that split topic words that should not have been together. This is the approach of Andrzejewski et al (2009), who used tree-based priors (Boyd-Graber et al, 2007) to encode expert correlations on topic models. A nice property of these correlations is that they form "soft" constraints for the model, which means the results will match users' expectations if and only if the correlations are supported by the underlying statistical model of the text (For an example of when data override correlations, see the "Macintosh" vs. "Windows" discussion in Section 4.3.2).

Moreover, Andrzejewski et al (2009)'s assumption of *a priori* correlations is inefficient. Users attempting to curate coherent topics need to see where the topic models go wrong before they can provide useful feedback. This suggests that topic models should be **interactive**—users should be able to see the output, give feedback, and continue refining the output.

Another reason to prefer an interactive process is that users invest effort in understanding the output of a topic model. They might look at every topic to determine which topics are good and which are bad. After a user gives feedback, a model should not throw away the topics the user did not have a problem with. This saves user cognitive load (Ceaparu et al, 2004), will lead to a faster end-result by not relearning acceptable topics, and can perhaps improve overall quality by maintaining topics that are acceptable to users.

Additionally, any model involving interaction with users should be as **computationally efficient** as possible to minimize users' waiting time. The efficient tree-based topic models, discussed in Chapter 3, can serve as an efficient backend model in this interactive setting. This requirement motivates the work for efficient tree-based topic models as in Chapter 3.

To summarize, we want a system that can help users who may or may not have prior knowledge of topic modeling, such as a news reporter or political analyst, to obtain or update topics easily, and this system should be:

- Simple enough to allow novices to encode feedback and update topics
- Fast enough to get the updates quickly
- Flexible enough to update the topics iteratively
- “Smart” enough to keep the “good” topics and improve the “bad” topics

We use efficient tree-based topic models discussed in Chapter 3 to meet the first two requirements, and combine it with a framework called interactive topic modeling (ITM) to satisfy the other two requirements.

Before discussing the details of how interactive topic modeling (ITM) works, we begin with two examples of a system that meets the above requirements. While we gloss over the details of our interactive topic modeling system for now, these examples motivate why a user might want an interactive system.

4.1.1 Example A: Joining Two Topics with Similar Content

For the first task, we recall the “Russian” problem that we have discussed in Section 2.1.3. We ran vanilla LDA for a corpus of New York Times editorials and obtained 20 topics. However, as shown in Table 4.1 (left) Topic 1 and Topic 20 both deal with “Russia” (broadly construed). Topic 20 is about the “Soviet Union”, but Topic 1 focuses on the development of “Russian Federation” after the collapse of the Soviet Union. This might be acceptable for some users; other users, however, may view both as part of a single historical narrative.

At this point, two things could happen. If the user was not a machine learning expert, they would throw up their hands and assume that topic modeling is not an appropriate solution. A machine learning expert would sit down and come up with a new model that would capture these effects, such as a model where topics evolve over time (Wang et al, 2008).

However, both of these outcomes are suboptimal for someone trying to understand what's going on in this corpus *right now*. Our system for creating interactive topic models allows a user to create a positive correlation with all of the clearly "Russian" or "Soviet" words ({"boris", "communist", "gorbachev", "mikhail", "russia", "russian", "soviet", "union", "yeltsin"}, shown in red and blue in Table 4.1). This yields the topics in Table 4.1 (right).[1] The two "Russia" topics were combined into Topic 20. This combination also pulled in other relevant words that are not near the top of either topic before: "moscow" and "relations"(green in Topic 20, right). Topic 20 concerns the entire arc of the transition from the "Soviet Union" to the "Russian Federation", and Topic 1 is now more about "democratic movements" in countries other than Russia. The other 18 topics stay almost the same, allowing our hypothetical user to continue their research.

4.1.2 Example B: Splitting a Topic with Mixed Content

The National Institutes of Health (NIH), America's foremost health-research funding agency, also has challenging information needs. They need to understand the research that they are funding, and they use topic modeling as one of their tools (Talley et al, 2011). After running a 700-topic topic model, an informant from

[1] Technical details for this experiment that will make sense later: this runs inference forward 100 iterations with tree-based topic model (Chapter 2.2 and 3) and **doc** ablation strategy discussed in chapter 4.2.

Topic	Words	Topic	Words
1	election, yeltsin, russian, political, party, democratic, russia, president, democracy, boris, country, south, years, month, government, vote, since, leader, presidential, military	**1**	election, democratic, south, country, president, party, africa, lead, even, democracy, leader, presidential, week, politics, minister, percent, voter, last, month, years
2	new, york, city, state, mayor, budget, giuliani, council, cuomo, gov, plan, year, rudolph, dinkins, lead, need, governor, legislature, pataki, david	2	new, york, city, state, mayor, budget, council, giuliani, gov, cuomo, year, rudolph, dinkins, legislature, plan, david, governor, pataki, need, cut
3	nuclear, arms, weapon, defense, treaty, missile, world, unite, yet, soviet, lead, secretary, would, control, korea, intelligence, test, nation, country, testing	3	nuclear, arms, weapon, treaty, defense, war, missile, may, come, test, american, world, would, need, lead, get, join, yet, clinton, nation
4	president, bush, administration, clinton, american, force, reagan, war, unite, lead, economic, iraq, congress, america, iraqi, policy, aid, international, military, see	4	president, administration, bush, clinton, war, unite, force, reagan, american, america, make, nation, military, iraq, iraqi, troops, international, country, yesterday, plan
	⋮		⋮
20	soviet, lead, gorbachev, union, west, mikhail, reform, change, europe, leaders, poland, communist, know, old, right, human, washington, western, bring, party	**20**	soviet, union, economic, reform, yeltsin, russian, lead, russia, gorbachev, leaders, west, president, boris, moscow, europe, poland, mikhail, communist, power, relations

Table 4.1: Five topics from 20 topics extracted topic model on the editorials from the New York times before adding a positive correlation (left) and after (right). After the correlation (words in red and blue on left) was added, which encourage Russian and Soviet terms to be in the same topic (in red and blue), non-Russian terms gain increased prominence in Topic 1 (in green), and "Moscow" (which was not part of the correlation) appeared in Topic 20 (in green).

the NIH reported that Topic 318 conflated two distinct concepts: the "urinary system" and the "nervous system", shown on the left of Table 4.2.

This was unsatisfactory to the users, as these are discrete systems and should not be combined. To address this error, we added a negative correlation between "bladder" and "spinal_cord" and applied our model to update their results. Then, Topic 318 was changed to a topic about "motor nerves" only (as shown on the right of Table 4.2): in addition to "bladder", other words associated with the urinary system disappeared; the original words related with "spinal_cord" all remained in the same topic; and more related words (in green)—"injured", "plasticity" and "locomotor"—appeared in this topic. The updated topic matched with NIH experts' needs.

Topic	Words	Topic	Words
318	bladder, sci, spinal_cord, spinal_cord_injury, spinal, urinary, urinary_tract, urothelial,injury, motor, recovery, reflex, cervical, urothelium, functional_recovery	**318**	sci, spinal_cord, spinal_cord_injury, spinal, injury, recovery, motor, reflex, urothelial, injured, functional_recovery, plasticity, locomotor, cervical, locomotion

Table 4.2: One topic from 700 topics extracted by topic model on the NIH proposals before (left) adding a negative correlation (between "bladder" and "spinal_cord") and after (right). After the correlation was added to push urinary system terms (red) away from the motor nerves terms (blue), most urinary terms went away (red), and some new terms related with motor nerves appeared (green).

4.1.3 Example C: Joining and Splitting

Researchers at United States Department of Agriculture (USDA) also stuck at a similar problem as NIH researchers, when they used topic models to explore the main themes of their grants. After running a topic model with 50 topics, they found that Topic 25 was a mixture of the "crops" related words (red) and "bees" related words (blue), as shown on the left of Table 4.3.

They want to separate the two concepts while keeping each concept's related words together. At this point, we added one positive correlation among the "crops" words, one positive correlation among the "bees" words, and one negative correlation between "wheat" (one "crops" word) and "bee" (one "bees" word).

Table 4.3 shows the updated results with applying the three correlations while keeping the "bees" words in Topic 25.[2] With the correlations, the "crops" words now appear in Topic 47, while "bees" words are still in Topic 25. The related words (green) with each topic respectively also show up. The remaining 48 topics do not

[2] To do this, we remember the topic assignments of the blue words if it is Topic 25, while unassign the topic assignments at other cases. More details about unassign is discussed in Section 4.2.

change much.

In addition, we also tried to apply the three correlations while keeping the "crops" words in Topic 25, as shown in Table 4.4. Again, topic models successfully updated the topics as the correlations: the "crops" words remain in Topic 25, with related words (green) showing up; and the "bees" words now in Topic 46. All the other topics remain the same.

Topic	Words	Topic	Words
25	wheat grain barley sorghum quality bee bees winter honey rust resistance cereal oat spring grains lines flour head hard	**25**	quality resistance bee bees rust honey winter lines dakota pollination spring fhb head hard colonies breeding varieties soft cultivars
47	plant plants species hawaii seed seeds native tropical collection sep resources accessions collections culture growth seedlings tissue invasive propagation	**47**	wheat plant grain plants sorghum barley hawaii species seed grains cereal accessions tropical seeds oat flour collection crops resources

Table 4.3: Two topics from 50 topics extracted by topic models on the USDA proposals before (left) adding multiple correlations and after (right). The correlations include: one positive correlation between red words, one positive correlation between blue words, and one negative correlation between "wheat" and "bee". In addition, users want the blue words to stay in Topic 25, and move the red words to a different topic (not specified). The correlations push the red words to Topic 47, and the blue words remained in Topic 25. Some related words also appeared (green). All the other topics remain the same.

Topic	Words	Topic	Words
25	wheat grain barley sorghum quality bee bees winter honey rust resistance cereal oat spring grains lines flour head hard	**25**	wheat grain sorghum barley quality grains resistance winter rust cereal lines oat flour spring varieties kansas fhb head breeding
46	strains fungal fungi plant bacterial pathogens pathogen host isolates pcr bacteria strain aflatoxin dna disease molecular microbial species fungus	**46**	strains fungi fungal plant pathogens bacterial pathogen host isolates pcr aflatoxin bacteria bee disease strain bees dna microbial species honey

Table 4.4: Two topics from 50 topics extracted by topic models on the USDA proposals before (left) adding multiple correlations and after (right). The correlations include: one positive correlation between red words, one positive correlation between blue words, and one negative correlation between "wheat" and "bee". In addition, users want the red words to stay in Topic 25, and move the blue words to a different topic (not specified). The correlations push the blue words to Topic 46, and the red words remained in Topic 25. Some related words also appeared (green). All the other topics remain the same.

These three real-world examples show what is possible with ITM. More specifically, they show that topic models do not always satisfy users' needs; effective topic modeling requires us to provide frameworks to allow users to improve the outputs of topic models.

4.1.4 Improvement or Impatience?

The skeptical reader might wonder if the issues presented above are *problems* that are being solved by interactive topic modeling. It could be merely that users are impatient and are looking at topic models before they are fully converged. If this is the case, then interactive topic modeling is only a placebo that makes users feel better about their model. The result is that they run inference longer, and end up at the same model. In fact, interactive topic models can do more.

From users' perspective, topic models are often used before the models converge: not only because users despise waiting for the results of inference, but also because normal users, non-machine learning experts, lack the intuition and tools to determine whether a model has converged (Evans, 2013). Thus interactive topic modeling might encourage them to more effectively "buy in" to the resulting analysis, as users have more patience when they are actively involved in the process (Bendapudi and Leone, 2003; Norman, 1993). As machine learning algorithms enter mainstream use, it is important not to overlook the **human factors** that connect to usage and adoption.

From models' perspective, interactive topic modeling allows models to **converge faster** than they would otherwise. As we show in Section 4.5.2, interactivity can improve ill-formed topics faster than through additional rounds of inference alone.

In addition, interactive topic models can also allow users to escape from **local minima**. For example, the example from Section 4.1.2 was obtained from an inference run after tens of thousands of iterations that, by all traditional measures of convergence, was the best answer that could be hoped for. By adding correlations, however, we discover topics that are more coherent and escape from local minima (Section 4.1.2).

4.2 Making Topic Models Interactive

As we argued in Section 4.1, there is a need for interactive topic models. Traditional topic models do not offer a way for non-experts to tweak the models, and those that do are "one off" interactions that preclude fine-grained adjustments and tweaks that solve users' problems but leave the rest of the model unmolested. This section proposes a framework for interactive topic refinement, interactive topic modeling (ITM).

Figure 4.1 shows the process at a high level: start with vanilla LDA (without any correlations), show users topics, solicit feedback from users, encode the feedback as correlations between words, and then do topic modeling with the corresponding tree-based prior. Each cycle is Figure 4.1 is called one **round**. Users can do as many rounds as they want until they are satisfied.

Since it takes some effort for users to understand the topics and figure out the "good" topics and "bad" topics, to save users' effort and time, ITM should be smart

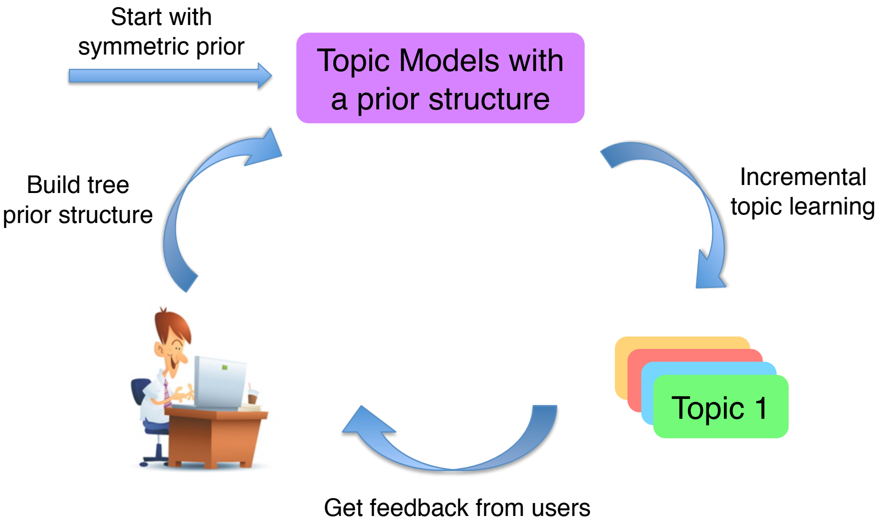


Figure 4.1: Interactive topic modeling: start with a vanilla LDA with symmetric prior, get the initial topics. Then repeat the following process till users are satisfied: show users topics, get feedback from users, encode the feedback into a tree prior, update topics with tree-based LDA.

enough to remember the "good" topics while improving the "bad" topics. In this section, we detail how interactively changing correlations can be accommodated in ITM.

A central tool that we will use is the strategic unassignment of states, which we call ablation (distinct from **feature** ablation in supervised learning). The state of a Markov Chain in MCMC inference stores the topic assignment of each token. In the implementation of a Gibbs sampler, unassignment is done by setting a token's topic assignment to an invalid topic (e.g., -1, as we use here) and decrementing any counts associated with that token.

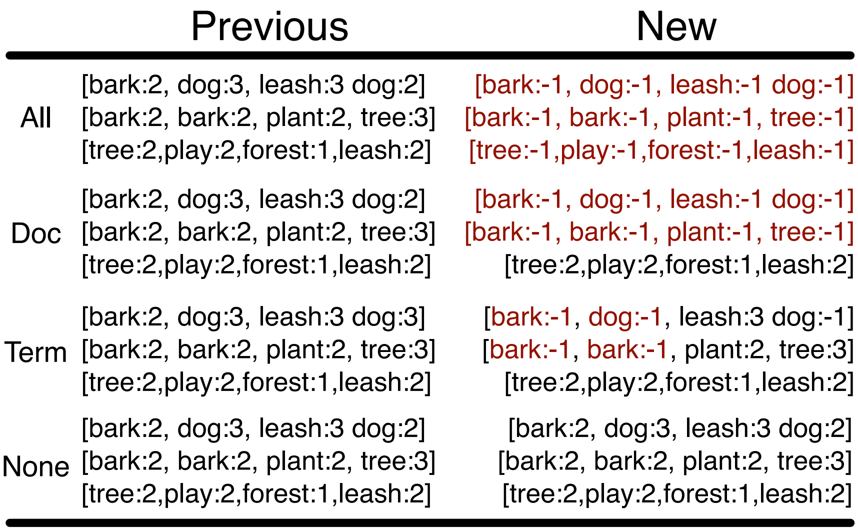
The correlations created by users implicitly signal that the model put certain words in the wrong place. In other models, this input is sometimes used to "fix", i.e., deterministically hold constant topic assignments (Ramage et al, 2009). Instead, we change the underlying model, using the current topic assignments as a starting position for a new Markov chain with some states strategically unassigned. How much of the existing topic assignments we use leads to four different options, which are illustrated in Figure 4.2.

An equivalent (and equally important) way to think about how ablation works is as technique to handle the inertia of inference. Inference schemes for topic models can become caught in local optima [3] (Section 4.1.4); because of the way topic models are used, users can often diagnose these local optima. Ablation allows the errors that trap inference in local optima to be forgotten, while retaining the unobjectionable parts of the model. Without ablation, inertia would keep inference trapped in a local optimum.

All We could revoke all state assignments, essentially starting the sampler from scratch. This does not allow *interactive* refinement, as there is nothing to enforce that the new topics will be in any way consistent with the existing topics. Once the topic assignments of all states are revoked, all counts will be zero, retaining no information about the state the user observed.

Doc Because topic models treat the document context as exchangeable, a document is a natural context for partial state ablation. Thus if a user adds a set of words $\boldsymbol{S}$ to correlations, then we have reason to suspect that all documents containing any one of $\boldsymbol{S}$ may have incorrect topic assignments. This is reflected in the state of the sampler by performing the UNASSIGN (Algorithm 3) operation for each token in any

[3] We discuss algorithms less sensitive to local optima in Chapter 6.



	Previous	New
All	[bark:2, dog:3, leash:3 dog:2]	[bark:-1, dog:-1, leash:-1 dog:-1]
	[bark:2, bark:2, plant:2, tree:3]	[bark:-1, bark:-1, plant:-1, tree:-1]
	[tree:2,play:2,forest:1,leash:2]	[tree:-1,play:-1,forest:-1,leash:-1]
Doc	[bark:2, dog:3, leash:3 dog:2]	[bark:-1, dog:-1, leash:-1 dog:-1]
	[bark:2, bark:2, plant:2, tree:3]	[bark:-1, bark:-1, plant:-1, tree:-1]
	[tree:2,play:2,forest:1,leash:2]	[tree:2,play:2,forest:1,leash:2]
Term	[bark:2, dog:3, leash:3 dog:3]	[bark:-1, dog:-1, leash:3 dog:-1]
	[bark:2, bark:2, plant:2, tree:3]	[bark:-1, bark:-1, plant:2, tree:3]
	[tree:2,play:2,forest:1,leash:2]	[tree:2,play:2,forest:1,leash:2]
None	[bark:2, dog:3, leash:3 dog:2]	[bark:2, dog:3, leash:3 dog:2]
	[bark:2, bark:2, plant:2, tree:3]	[bark:2, bark:2, plant:2, tree:3]
	[tree:2,play:2,forest:1,leash:2]	[tree:2,play:2,forest:1,leash:2]

Figure 4.2: Four different strategies for state ablation after the words "dog" and "bark" are added to the correlation {"leash", "puppy"} to make the correlation {"dog", "bark", "leash", "puppy"}. The state is represented by showing the current topic assignment after each word (e.g. "leash" in the first document has topic 3, while "forest" in the third document has topic 1). On the left are the assignments before words were added to correlations, and on the right are the ablated assignments. Unassigned tokens are given the new topic assignment -1 and are highlighted in red.

document containing a word added to a correlation. This is equivalent to the Gibbs2 sampler of Yao et al (2009) for incorporating new documents in a streaming context. Viewed in this light, a user is using words to select documents that should be treated as "new" for this refined model.

Term Another option is to perform ablation only on the topic assignments of tokens which have been added to a correlation. This applies the unassignment operation (Algorithm 3) only to tokens whose corresponding word appears in added correlations (i.e. a subset of the **Doc** strategy). This makes it less likely that other tokens in similar contexts will follow the words explicitly included in the correlations to new topic assignments.

Algorithm 3 UNASSIGN(doc d, token w)

```
Get the topic of token $w$: $k$
Decrement topic count: $n_{k|d} - -$
**for** path $\lambda$ of $w$ in previous prior tree **do**
    **for** edge $e$ of path $\lambda$ **do**
        Decrement edge count: $n_{e|k} - -$
Forget the topic of token $w$
```

Algorithm 4 MOVE(doc d, token w)

```
Get the topic of token $w$: $k$
**for** path $\lambda'$ of $w$ in previous prior tree **do**
    **for** edge $e'$ of path $\lambda'$ **do**
        Decrement edge count: $n_{e'|k} - -$
**for** path $\lambda$ of $w$ in current prior tree **do**
    **for** edge $e$ of path $\lambda$ **do**
        Increment edge count: $n_{e|k} + +$
```

None The final option is to move words into correlations but keep the topic assignments fixed, as described in Algorithm 4. This is arguably the simplest option, and in principle is sufficient, as the Markov chain should find a stationary distribution regardless of the starting position. However, when we "move" a token's count (Algorithm 4) for word that changes from uncorrelated to correlated, it is possible that there is a new ambiguity in the latent state: we might not know the path. We could either merge the correlations to avoid this problem (as discussed in Section 2.2.1), restricting each token to a unique path, or sample a new path. These issues make this ablation scheme undesirable.

The **Doc** and **Term** ablation schemes can both be viewed as online inference (Yao et al, 2009; Hoffman et al, 2010). Both of them view the correlated words or some documents as unseen documents and then use the previously seen documents (corresponding to the part of the model a user was satisfied with) in conjunction with the modified model to infer the latent space on the "new" data. Regardless of what ablation scheme is used, after the state of the Markov chain is altered, the next step is to actually run inference forward, sampling assignments for the unassigned tokens for the "first" time and changing the topic assignment of previously assigned tokens. How many additional iterations are required after adding correlations is a delicate tradeoff between interactivity and effectiveness, which we investigate further in Section 4.3.

The interactive topic modeling framework described here fulfills the requirements laid out in Section 4.1: it is simple enough that untrained users can provide feedback and update topics; it is flexible enough to incorporate that feedback into the resulting models; and it is "smart" enough—through ablation—to retain the good topics while correcting the errors identified by users. Interactive topic modeling could serve the goals of our hypothetical political scientist to explore corpora to identify trends and topics of interest.

4.3 Users in the Loop

In this section, we describe evaluations of our ITM system. First, we describe fully automated experiments to help select how to build a system that can learn and adapt from users' input but also is responsive enough to be usable. This requires selecting ablation strategies and determining how long to run inference after ablation (Section 4.3.1).

Next, we perform an open-ended evaluation to explore what untrained users do when presented with an ITM system. We expose our system to users on a crowdsourcing platform and explore users' interactions, and investigate what correlations users created and how these correlations were realized on a social media corpus (Section 4.3.2).

4.3.1 Simulated Users

In this section, we use the 20 Newsgroup corpus (20News) introduced in Section 3.5. We use the default split for training and test set, and the vocabulary contains the most frequent 5000 words.

Refining the topics with ITM is a process where users try to map their mental topics with the topics from topic models. Category information is one possible way that users form topics in their mind. For the 20News corpus, users might have some category information in mind, such as, "politics", "economies", "energy", "technologies", "entertainments", "sports", "arts", etc. They might have some words associated with each category. For example, the words "government", "president" for "politics", and "gas", "oil" for "energy". Probably at the beginning the word list associated with each category is not complete, that is, they have limited number of words in mind, but they might come up with more words for each category later.

This whole process can be simulated by ranking words in each category by their information gain (IG).[4] Sorting words by IG discovers words that should be correlated with a classification label. If we believe that vanilla LDA lacks these correlations (because of a deficiency of the model), topics that have these correlations should better represent the collection (as measured by classification accuracy). Intuitively, these words represent a user thinking of a concept they believe is in the collection (e.g., "Christianity") and then attempting to think of words they believe should be connected to that concept.

To simulate this process, we start with the words with high IG for each category,

[4] Computed by Rainbow toolbox, http://www.cs.umass.edu/~mccallum/bow/rainbow/

and gradually consider more words according to the ranking to simulate the whole process. We treat the words in each category as a positive correlation and add one more word each round to refine the topics.

More concretely, for the 20News dataset, we rank the top 200 words for each class by IG, and delete words associated with multiple labels to prevent correlations for different labels from merging. The smallest class had 21 words remaining after removing duplicates (due to high overlaps of 125 overlapping words between "talk.religion.misc" and "soc.religion.christian", and 110 overlapping words between "talk.religion.misc" and "alt.atheism"), so the top 21 words for each class were the ingredients for our simulated correlations. For example, for the class "soc.religion.christian," the 21 correlated words include "catholic, scripture, resurrection, pope, sabbath, spiritual, pray, divine, doctrine, orthodox." We simulate a user's ITM session by adding a word to each of the 20 positive correlations until each of the correlations has 21 words.

We evaluate the quality of the topic models through an extrinsic classification task. We represent a document's features as the topic vector (the multinomial distribution θ in Chapter 2.2) and learn a mapping to one of the twenty newsgroups using a supervised classifier (Hall et al, 2009). As the topics form a better lower-dimensional representation of the corpus, the classification accuracy improves.

Our goal is to understand the phenomenon of ITM, not classification, so the classification results are well below state of the art. However, adding interactively selected topics to state of the art features (tf-idf unigrams) gives a relative error reduction of 5.1%, while adding topics from vanilla LDA gives a relative error reduction of 1.1%. Both measurements were obtained without tuning or weighting features, so presumably better results are possible.

We set the number of topics to be the same as the number of categories and hope the topics can capture the categories as well as additional related information. While this is not a classification task, and it is not directly comparable with state of the art classifiers like SVM, it performs better than the **Null** baseline (running with comparable iterations without any correlations) in Figure 4.3 and Figure 4.4.

This experiment is structured as a series of rounds. Each round adds an additional correlation for each newsgroup (thus 20 per round). After a correlation is added to the model, we ablate topic assignments according to one of the strategies described in Section 4.2, run inference for some number of iterations, extract the new estimate of the per-document topic distribution, learn a classifier on the training data, and apply that classifier to the test data. We do 21 rounds in total, and the following sections investigate the choice of number of iterations and ablation strategy. The number of LDA topics is set to 20 to match the number of newsgroups. The hyperparameters for all experiments are $\alpha = 0.1$, $\beta = 0.01$ for uncorrelated words, $\beta = 100$ for positive correlations and $\beta = 10^{-6}$ for negative correlations.

We start the process after only 100 iterations of inference using a vanilla LDA model. At 100 iterations, the chain has not converged, but such small numbers of iterations is a common practice for impatient users initially investigating a dataset (Evans, 2013; Carbone, 2012).[5] After observing initial topics, the user then

[5] A machine learning purist would argue that such usage is incorrect, as you only want samples from a converged Markov chain. Without commenting on this debate, this experiment reflects the reality of how topic models are used for analyzing text.

gradually updates the topics, allowing inference to continue.

Moreover, while the patterns shown in Figure 4.4 were broadly consistent with larger numbers of iterations, such configurations sometimes had too much inertia to escape from local extrema. More iterations make it harder for the correlations to influence the topic assignment, another reason to start with smaller numbers of initial iterations.

Investigating Ablation Strategies

First, we investigate which ablation strategy best incorporates correlations. Figure 4.3 shows the classification accuracy of six different ablation strategies for each of 21 rounds. Each result is averaged over five different chains using 10 additional iterations of Gibbs sampling per round (other numbers of iterations are discussed in Section 4.3.1). As the number of words per correlation increases, the accuracy increases as models gain more information about the classes.

To evaluate whether our model works better, we first compare our model against a baseline without any correlations. This is to test whether the correlations help or not. This baseline is called **Null**, and it runs inference for a comparable number of iterations for fair comparison. While **Null** sees no correlations, it serves as a lower baseline for the accuracy but shows the effect of additional inference. Figure 4.3 shows that the **Null** strategy has a lower accuracy than interactive versions, especially with more correlations.

We also compare our model with non-interactive baselines, which are **All Initial** and **All Full** with all correlations known *a priori*. **All Initial** runs the model for

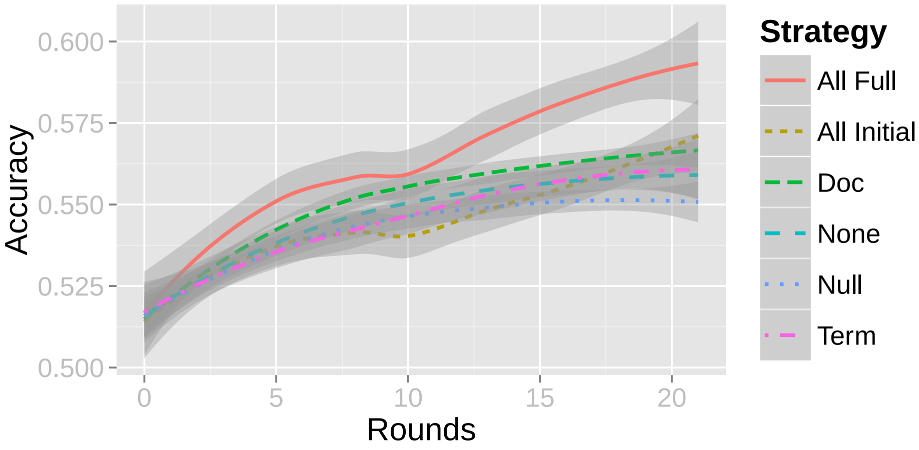


Figure 4.3: Accuracy (y-axis) using different ablation strategies as additional correlations are added (x-axis). We start with 100 iterations, then for each round, add one more word for each of the 20 positive correlations, and run 10 additional iterations. **Null** represents standard LDA, as the lower baseline. **All Initial** and **All Full** are non-interactive baselines, and **All Full** is the upper baseline. The results of **None**, **Term**, **Doc** are more stable (as denoted by the bars), and the accuracy is increased gradually as more correlated words are added.

the only the initial number of iterations (100 iterations in this experiment), while **All Full** runs the model for the total number of iterations added for the interactive version. (That is, if there were 21 rounds and each round of interactive modeling added 10 iterations, **All Full** would have 210 iterations more than **All Initial**). **All Full** is an upper baseline for the accuracy since it both sees the correlations at the beginning and also runs for the maximum number of total iterations. **All Initial** sees the correlations before the other ablation techniques but it has fewer total iterations.

In Figure 4.3, both **All Initial** and **All Full** show a larger variance (as denoted by bands around the average trends) than the interactive schemes. This can be viewed as akin to simulated annealing, as the interactive settings have more freedom to explore in early rounds. For topic models with **Doc** or **Term** ablation, this freedom is limited to only correlated words or words related with correlated words. Since the model is less free to explore the entire space, these ablation strategies result in much lower variance.

All Full has the highest accuracy; this is equivalent to where users know all correlations *a priori*. This strategy corresponds to an omniscient and infinitely patient user. Neither of these properties are realistic. First, it is hard for users to identify and fix all problems at once. Often smaller problems are not visible until larger problems have been corrected. This requires multiple iterations of inspection and correction. Second, this process requires a much longer waiting time, as all inference must be rerun from scratch after every iteration.

The accuracy of each interactive ablation strategy is (as expected) between the lower and upper baselines. Generally, the correlations will influence not only the topics of the correlated words, but also the topics of the correlated words' context in the same document. **Doc** ablation gives more freedom for the correlations to overcome the inertia of the old topic distribution and move towards a new one influenced by the correlations.

How Many Iterations do Users Have to Wait?

For a fixed corpus and computational environment, the number of iterations is the primary factor that determines how long a user has to wait. While more iterations can get closer to convergence, it also implies longer waiting time. So we need to balance convergence and waiting time.

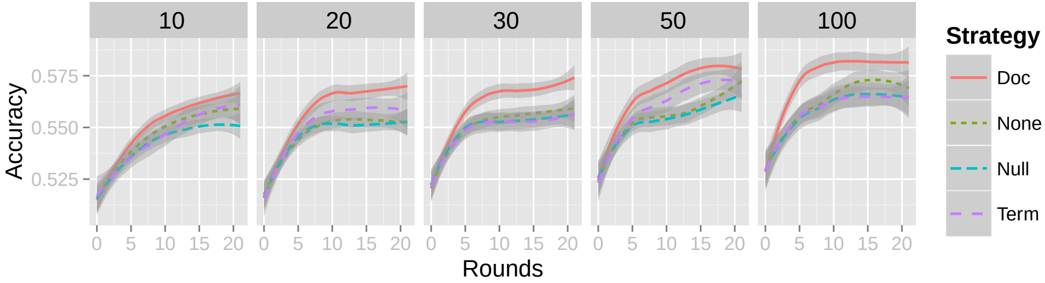


Figure 4.4: Classification accuracy by strategy and number of iterations between rounds. We start with 100 iterations, then for each round, add one more word for each of the 20 positive correlations, and run additional 10 iterations. The **Doc** ablation strategy performs best, suggesting that the document context is important for ablation correlations. While more iterations are better, there is a tradeoff with interactivity.

Figure 4.4 shows the effect of using different numbers of Gibbs sampling iterations between rounds. For each of the ablation strategies, we run $10, 20, 30, 50,$ 100 additional Gibbs sampling iterations for each round. As expected, more iterations increase accuracy, although improvements diminish beyond 100 iterations. With more correlations, additional iterations help less, as the model has more *a priori* knowledge to draw upon.

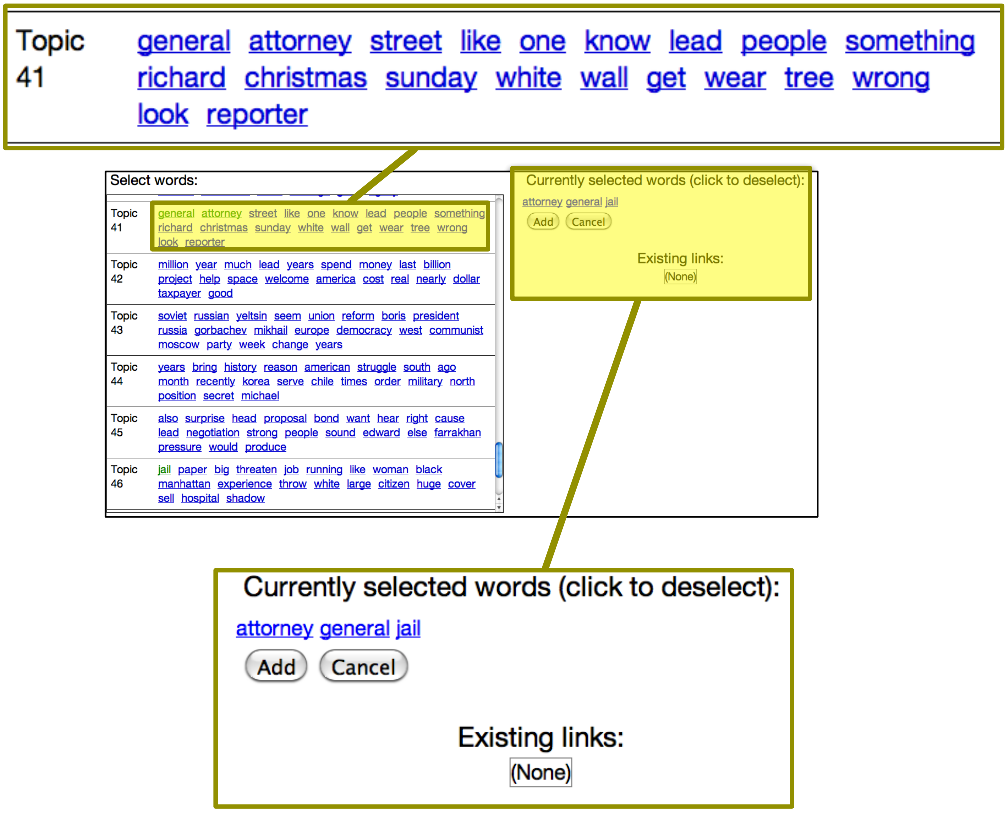
For all numbers of additional iterations, while the **Null** serves as the lower baseline for accuracy in all cases, the **Doc** ablation clearly outperforms the other ablation schemes, consistently yielding a higher accuracy. Thus, there is a benefit when the model has a chance to relearn the document context when correlations are added, and **Doc** provides the flexibility for topic models to overcome the inertia of the old topic distribution but does not throw away the old distribution entirely. The difference is greater with more iterations, suggesting **Doc** needs more iterations to "recover" from unassignment.

The number of additional iterations per round is directly related to users' waiting time. According to Figure 4.4, more iterations for each round achieves higher accuracy, while increasing wait time. This is a tradeoff between latency and model quality, and may vary based on users, applications, and data size.

However, the luxury of having hundreds or thousands of additional iterations for each correlation would be impractical. For even moderately sized datasets, even one iteration per second can tax the patience of individuals who want to use the system interactively. Studies have shown that a long waiting time may affect cognitive load, making it harder for a user to recall what they were doing or the context of the initial task (Ceaparu et al, 2004). Based on these results and an *ad hoc* qualitative examination of the resulting topics, we found that 30 additional iterations of inference was acceptable; this is used in later experiments, though this number can vary in different settings.

4.3.2 Real Users from Mechanical Turk

To move beyond using simulated users adding the same words regardless of what topics were discovered by the model, we needed to expose the model to human users. We solicited approximately 200 judgments from Mechanical Turk, a popular crowd-sourcing platform that has been used to gather linguistic annotations (Snow et al, 2008), measure topic quality (Chang et al, 2009; Stevens et al, 2012), and supplement traditional inference techniques for topic models (Chang, 2010). After presenting our interface for collecting judgments, we examine the results from these



Select words:

Topic 41	general attorney street like one know lead people something richard christmas sunday white wall get wear tree wrong look reporter
Topic 42	million year much lead years spend money last billion project help space welcome america cost real nearly dollar taxpayer good
Topic 43	soviet russian yeltsin seem union reform boris president russia gorbachev mikhail europe democracy west communist moscow party week change years
Topic 44	years bring history reason american struggle south ago month recently korea serve chile times order military north position secret michael
Topic 45	also surprise head proposal bond want hear right cause lead negotiation strong people sound edward else farrakhan pressure would produce
Topic 46	jail paper big threaten job running like woman black manhattan experience throw white large citizen huge cover sell hospital shadow

Currently selected words (click to deselect):

attorney general jail

Existing links:

(None)

Figure 4.5: Interface for Mechanical Turk experiments. Users see the topics discovered by the model and select words (by clicking on them) to build correlations to be added to the model.

ITM sessions both quantitatively and qualitatively.

Figure 4.5 shows the interface used in the Mechanical Turk tests. The left side of the screen shows the current topics in a scrollable list, with the top 30 words displayed for each topic.

Users create correlations by clicking on words from the topic word lists. The word lists use a color-coding scheme to help the users keep track of which words they are already in correlations. The right side of the screen displays the existing correlations. Users can click on icons to edit or delete each one. The correlation being built is also shown in its own panel. Clicking on a word will remove that word

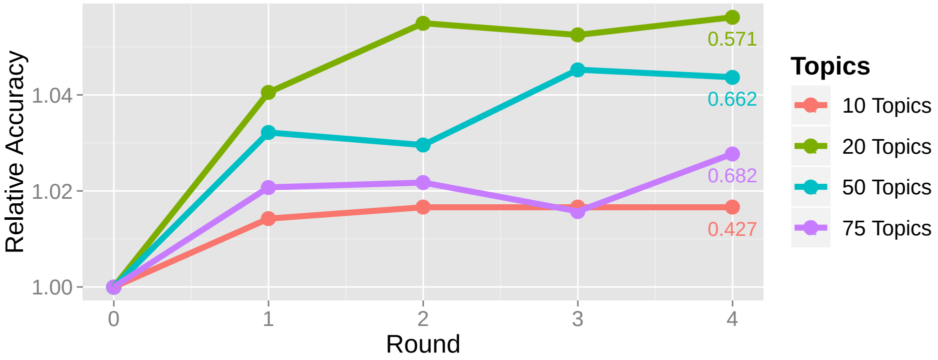


Figure 4.6: The relative accuracy improvement (using round 0 as a baseline) of the best Mechanical Turk user session for each of the four numbers of topics, with the actual accuracy marked for the last round. While the 10-topic model does not provide enough flexibility to create good correlations, the best users could clearly improve classification with more topics.

from the current correlation.

Users were not given a specific goal; instead, they were instructed to add correlations between words so that the topics (we called them "word groups" in the instructions) made more sense. This was intentionally underspecified, as we wanted to see what would happen when ITM was placed in the hands of untrained users.

As in Section 4.3.1, we can compute the classification accuracy for users as they add words to correlations. The best users, who seemed to understand the task well, were able to increase the classification accuracy (Figure 4.6). The median user, however, had an accuracy improvement indistinguishable from zero. Despite this, we can examine the users' behavior to better understand their goals and how they interact with the system.

The correlation sizes ranged from one word to over forty. The more words in the correlation, the more likely it was to noticeably affect the topic distribution.

This observation makes sense given our updating method. A correlation with more words will probably cause the topic assignments to be reset for more documents.

Most of the large correlations (more than ten words) corresponded to the themes of the individual newsgroups. Some common themes for large correlations were:

- Themes that matched a single newsgroup: religion, space exploration, health, foreign countries, cars, motorcycles, graphics, encryption
- Themes that spanned multiple related newsgroups: sports, government, computers, cars/motorcycles
- Themes that probably matched a sub-topic of a single newsgroup: homosexuality, Israel, computer programming

Some users created correlations with both "baseball" and "hockey" words, while others separated them. ("baseball" and "hockey" are in separate newsgroups.) The separate correlations often contained overlapping words. Even so, the choice of combined vs. separate correlations almost always determined whether baseball and hockey would be in the same topic in the model. A similar situation occurred with "cars" and "motorcycles", which are also discussed in separate newsgroups.

Some users created inscrutable correlations, like {"better", "people", "right", "take", "things"} and {"fbi", "let", "says"}. They may have just clicked random words to finish the task quickly. While subsequent users could delete poor correlations, most chose not to. Because we wanted to understand broader behavior we made no effort to squelch such responses.

The two-word correlations illustrate an interesting contrast. Some pairs are linked together in the corpus, like {“jesus”, “christ”, “solar”, “sun”}. With others, like {“even”, “number”} and {“book”, “list”}, the users seem to be encouraging collocations to be in the same topic. However, the collocations may not be present in any document in this corpus.

Not all sensible correlations led to successful topic changes. Many users grouped “mac” and “windows” together, but they were almost never placed in the same topic. The corpus includes separate newsgroups for Macintosh and Windows hardware, and divergent contexts of “mac” and “windows” overpowered the prior distribution.

Other correlations led to topic changes that were not necessarily meaningful. For example, one user created a correlation consisting of male first names. A topic did emerge with these words, but the rest of the words in that topic seemed random. This suggests that the set of male first names aren’t associated with each other in the corpus. Preliminary experiments on newspaper articles had similar correlations that created a more meaningful topic associated with obituaries and social announcements.

Finally, many correlations depend on a user’s background and perspective, showing the flexibility of this approach. Some users grouped “israeli”, “jewish”, “arab”, and “muslim” with international politics words, and others with religion words. On the other hand, “christian” was always grouped with religion words. The word “msg” appears to have two different interpretations. Some users grouped it with computer words (reading it as a message), while others grouped it with health words (reading it as a food additive).

As mentioned in Section 2.2, topic models with a tree-based prior can represent situations where words have multiple meanings. In previous work, the paths in the tree—provided by WordNet—correspond to the distinct meanings of a word (Boyd-Graber et al, 2007). Users found the formalism intuitive enough to build their own small WordNets to distinguish the different meanings of "msg".

4.4 User Study

New systems for information access are typically investigated through task-based user studies to determine whether the new approach allows users to complete specific tasks as well as with current systems. Wacholder and Liu (2008), for example, compared traditional paper-based book indices with full-text search for answering questions in large text collections. Following their lead, we compare the information-seeking effectiveness using both interactive and non-interactive topic modeling.

We asked users to fill the role of the running example a political scientist attempting to find legislation relevant to "immigration and refugee issues" (among other topics). Using full-text search aided by either vanilla topic models or interactive topic models (ITM), users were asked to answer questions based content in a collection of legislative debates.

We found that users were able to answer the questions equally well in both group with ITM (experimental group) and group without ITM (control group). However, users in the group using ITM had radically different strategies for how they found information in the corpus. Rather than relying on full-text search, users used topic models to find relevant information.

4.4.1 Legislative Corpus

In the process of becoming a law, potential US legislation is sponsored by a congressperson and introduced for debate by a committee in either the US House of Representatives (lower chamber) or the US Senate (upper chamber). Once introduced, the bill is debated within the chamber where it was introduced. Our corpus contains transcripts of these debates for the 109^{th} congress, which served during the 2005 and 2006 calendar years.

The corpus is available online from GovTrack.[6] Each page is associated with a bill and a vote. Uninteresting procedural bills, with less than 20% "Yea" votes or less than 20% "Nay" votes, are removed. We selected a subset of this congressional debate dataset that includes ten bills and their associated debates. Each debate has multiple turns (a single uninterrupted speech by a unique congressperson), and we use each turn as a document for topic modeling. This yields 2,550 documents in total; we ignore all temporal, speaker-related, or legislative organization. While this is somewhat unrealistic for a real-world study of legislative information, we will use some of this discarded information to aid evaluation. The subset includes bills on immigration, the estate (death) tax, stem cell research, and others. Detailed information can be found in Appendix A.

[6] http://www.govtrack.us/data/us/109/

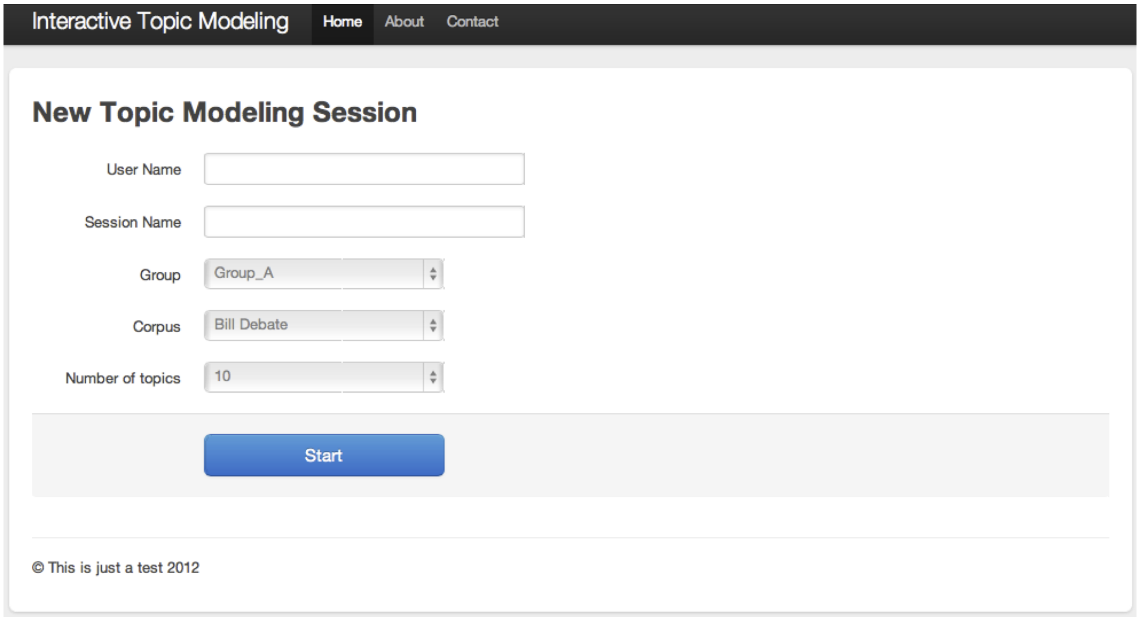
4.4.2 Introduction of ITM Interface

The ITM interface is a web-based application.[7] It provides a workflow for users to select model parameters (corpus and number of topics), create an initial topic model, name the topics, and refine the topics using ITM. The interface also provides multiple ways for a user to explore the corpus: a full-text search over all documents, a full-text search within a single topic, a listing of documents associated with each topic, and links to access specific documents. We walk through this workflow in detail below.

From the initial screen (Figure 4.7), users specify the session information, such as user name, corpus, number of topics, etc. Once users click "start", the interface loads the initial set of topics, including the top topic words and related documents, as shown in Figure 4.8. The top topic words are displayed such that the size of a word is proportional to the probability of this word appearing in the topic.

After clicking on the topic, users can view additional information and, most importantly, edit the topic (editing is disabled for the control group). After clicking on a topic, four "bins" are visible: "all", "ignore", "important" and "trash". Initially, all of the topic words are in the "all" bin. As shown in Figure 4.9, users can drag words to different bins based on their importance to the topic: words that are important to the topic to the"important" bin, words that should be ignored in this topic to the "ignored" bin, and words that should be stopwords in the whole corpus to "trash". Users can also add new words to this topic by typing the word and

[7] This ITM interface is implemented with a HTML and jQuery (http://jquery.com/) front end, connected via Ajax and JSON.



New Topic Modeling Session

User Name

Session Name

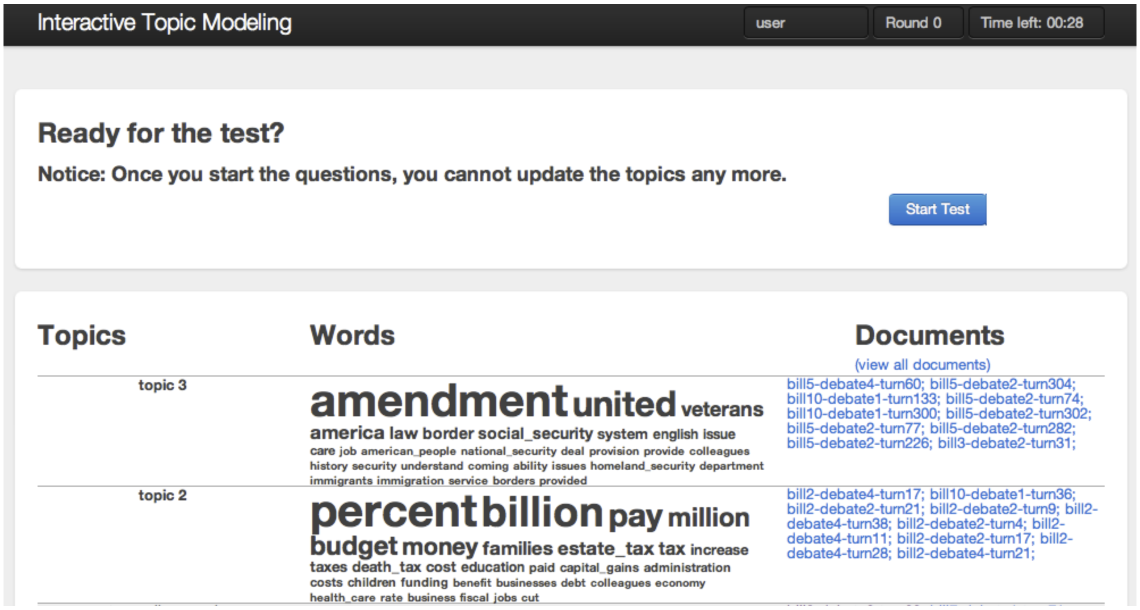
Group: Group_A

Corpus: Bill Debate

Number of topics: 10

© This is just a test 2012

Figure 4.7: The start page of the ITM interface: users specify the user name, session name, corpus, number of topics, and experimental group (Group A: control group (LDA only); Group B: experimental group (ITM).)

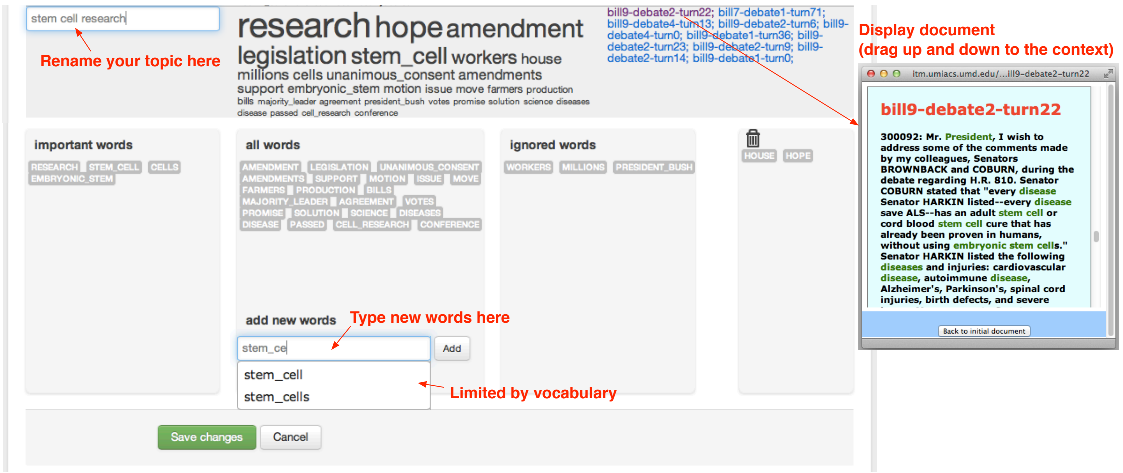


Ready for the test?

Notice: Once you start the questions, you cannot update the topics any more.

Topics	Words	Documents
topic 3	amendment united veterans america law border social_security system english issue care job american_people national_security deal provision provide colleagues history security understand coming ability issues homeland_security department immigrants immigration service borders provided	bill5-debate4-turn60; bill5-debate2-turn304; bill10-debate1-turn133; bill5-debate2-turn74; bill10-debate1-turn300; bill5-debate2-turn302; bill5-debate2-turn77; bill5-debate2-turn282; bill5-debate2-turn226; bill3-debate2-turn31;
topic 2	percent billion pay million budget money families estate_tax tax increase taxes death_tax cost education paid capital_gains administration costs children funding benefit businesses debt colleagues economy health_care rate business fiscal jobs cut	bill2-debate4-turn17; bill10-debate1-turn36; bill2-debate2-turn21; bill2-debate2-turn9; bill2-debate4-turn38; bill2-debate2-turn4; bill2-debate4-turn11; bill2-debate2-turn17; bill2-debate4-turn28; bill2-debate4-turn21;

Figure 4.8: Two topics displayed in the ITM interface. The most probable words in each topic are displayed with the size proportional to the probability of a word appearing in this topic. The documents most associated with each topic are shown in each topic's panel. The user can view all documents by selecting "view all documents".


stem cell research
Rename your topic here
research hope amendment legislation stem_cell workers house millions cells unanimous_consent amendments support embryonic_stem motion issue move farmers production bills majority_leader agreement president_bush votes promise solution science diseases disease passed cell_research conference
bill9-debate2-turn22; bill7-debate1-turn71; bill9-debate4-turn13; bill9-debate2-turn6; bill9-debate4-turn0; bill9-debate1-turn36; bill9-debate2-turn23; bill9-debate2-turn9; bill9-debate2-turn14; bill9-debate1-turn0;
Display document
(drag up and down to the context)
important words
RESEARCH
STEM_CELL
CELLS
EMBRYONIC_STEM
all words
AMENDMENT
LEGISLATION
UNANIMOUS_CONSENT
AMENDMENTS
SUPPORT
MOTION
ISSUE
MOVE
FARMERS
PRODUCTION
BILLS
MAJORITY_LEADER
AGREEMENT
VOTES
PROMISE
SOLUTION
SCIENCE
DISEASES
DISEASE
PASSED
CELL_RESEARCH
CONFERENCE
ignored words
WORKERS
MILLIONS
PRESIDENT_BUSH
HOUSE
HOPE
add new words
Type new words here
stem_ce
Add
stem_cell
stem_cells
Limited by vocabulary
Save changes
Cancel
itm.umiacs.umd.edu/...ill9-debate2-turn22
Back to initial document

bill9-debate2-turn22

300092: Mr. President, I wish to address some of the comments made by my colleagues, Senators BROWNBACK and COBURN, during the debate regarding H.R. 810. Senator COBURN stated that "every disease Senator HARKIN listed--every disease save ALS--has an adult stem cell or cord blood stem cell cure that has already been proven in humans, without using embryonic stem cells." Senator HARKIN listed the following diseases and injuries: cardiovascular disease, autoimmune disease, Alzheimer's, Parkinson's, spinal cord injuries, birth defects, and severe

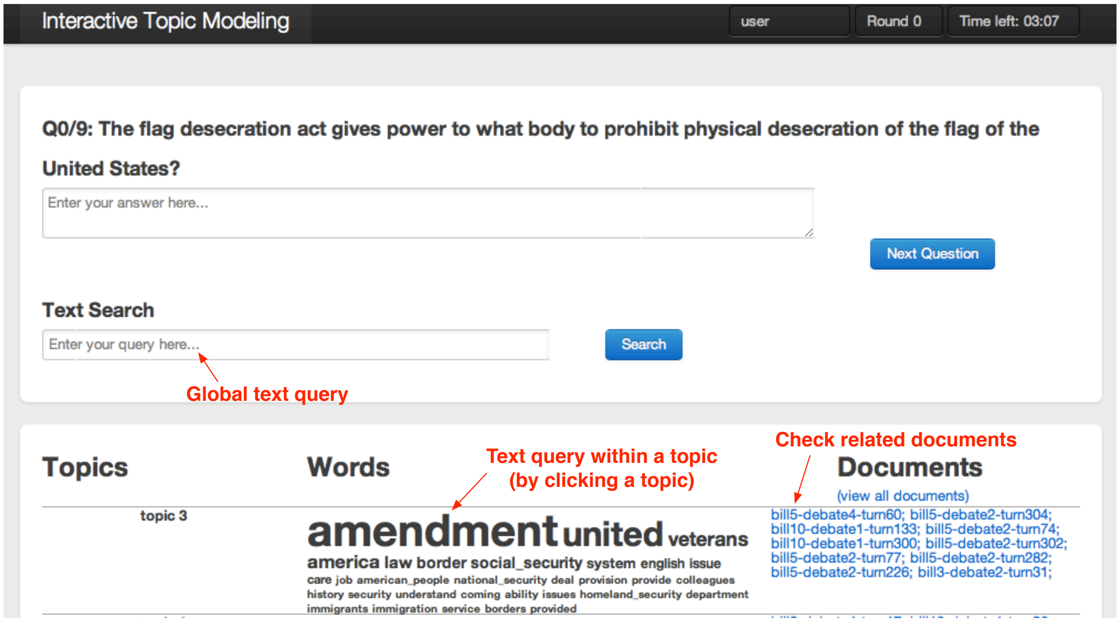
Figure 4.9: ITM interface for refining a topic. Users can put words into different "bins", name topics, and add new words to the topic.

clicking "add".[8]

Once the user has finished editing a topic, changes are committed by pressing the "Save" button. The backend then receives the users' feedback. The model adds a positive correlation between all words in the "important" bin, a negative correlation between words in the "ignored" bin and words in the "important" bin, and removes words in the "trash" from the model. With these changes to the model, the ITM relearns the topics and updates the topics. While in principle users may update the topics as many times as they wish, our study limited a user's exploration and modification of topics to fifteen minutes. Then, the users entered the next phase of the study, answering questions about the corpus.

In the question answering phase (Figure 4.10), users have three options to explore the data to answer the questions: by reading through related documents associated with a topic, searching through all of the documents through full-text search,

[8] Only words present in the model's vocabulary can be added; this constraint is enforced via an autocomplete function.



Q0/9: The flag desecration act gives power to what body to prohibit physical desecration of the flag of the United States?

amendment united veterans america law border social_security system english issue care job american_people national_security deal provision provide colleagues history security understand coming ability issues homeland_security department immigrants immigration service borders provided

bill5-debate4-turn60; bill5-debate2-turn304; bill10-debate1-turn133; bill5-debate2-turn74; bill10-debate1-turn300; bill5-debate2-turn302; bill5-debate2-turn77; bill5-debate2-turn282; bill5-debate2-turn226; bill3-debate2-turn31;

Figure 4.10: Test page of the ITM interface. Users will see one question each time, and they can answer the question by, searching keywords globally, checking the related topics or topic documents, or narrowing query results down by topics, that is, after a global query, click a topic, and the query results will be filtered by the relevance to this topic, displayed below the topic you clicked. Click "Next question" to proceed, and users are not allowed to go back to previous questions.

or via a text search *restricted to a single topic*. The full-text search is important because it is a commonly used means of finding data within large corpora (Shneiderman et al, 1997) and because it has been used in previous previous information-seeking studies (Wacholder and Liu, 2008).[9] Initial studies, where access to the data was restricted to only topic model information, were too difficult. We expect users to use topics when they are useful and use full-text search when topics are less useful in answering a question. After each question, users click "Next question" to proceed; users cannot return to previous questions.

[9] Some examples where the websites for accessing the legislative data have full-text search: http://thomas.loc.gov/home/LegislativeData.php?n=BillText; http://www.senate.gov/ pagelayout/legislative/g_three_sections_with_teasers/legislative_home.htm.

4.4.3 User Population

To evaluate the effectiveness of ITM for information-seeking tasks, we compare the performance of users in two groups: the experimental group (ITM) and a control group (vanilla LDA).

For the experimental group, users start with an initial set of topics and can refine the topics using ITM for up to fifteen minutes. They then start the test phase for thirty minutes. They are provided with the refined topics for use during the test.

The control group also has access to the initial topics, but they cannot refine the topics. They are given up to fifteen minutes to check the topics, rename the topics, and review documents associated with the topics. This is to avoid experimental differences caused by the experimental group benefiting from *exploring* the corpus rather than from *interactive topic modeling*. After spending up to fifteen minutes exploring the corpus, the control group also has thirty minutes to answer the test questions.

The study participants are randomly assigned to a group. Each participant views a video explaining how to use the interface and do the test. During the study, the system logs the related information of each user. After the study, participants complete a survey on their educational/technical background and familiarity with legislation or topic models.

The study had twenty participants (ten for each group). All of the users are fluent in English. Participants are either students pursuing a degree in Computer Science, Information Science, Linguistics, or working in a related field. A post-test user survey revealed that most users have little or no knowledge about congressional debates and that users have varied experience with topic models.

We designed ten free response questions by exploring this legislation corpus, including questions regarding legislation which deals with taxes, the US-Mexico border, and other issues. The full text of the questions appears in Appendix B.

4.4.4 User Study Analysis

We examined two aspects of the experiment: how well the experimental group's final topics replicated ground-truth annotations (below, we refer to this metric as **refine**) and how well both the groups answered the questions (**test**).

Our experiment views the corpus as an unstructured text collection (a typical use case of topic models); however, each turn in the dataset is associated with a single bill. We can view this association as the true clustering of the dataset. We compare this clustering against the clustering produced by assigning each document to a cluster corresponding to its highest-probability topic.

We compare these reference clusters to the clusters produced by ITM using **variation of information** (Meilă, 2007). This score has a range from zero to infinity and represents the information-theoretic "distance" between two partitions (lower is better). Using this information, we compute the variation of information (Meilă, 2007) between the true labels (the 10 underlying bills) and the topic modeling clusters. While we have a good initial set of topics (the initial variation of information score is low), users in the experimental group—who reported little knowledge about the

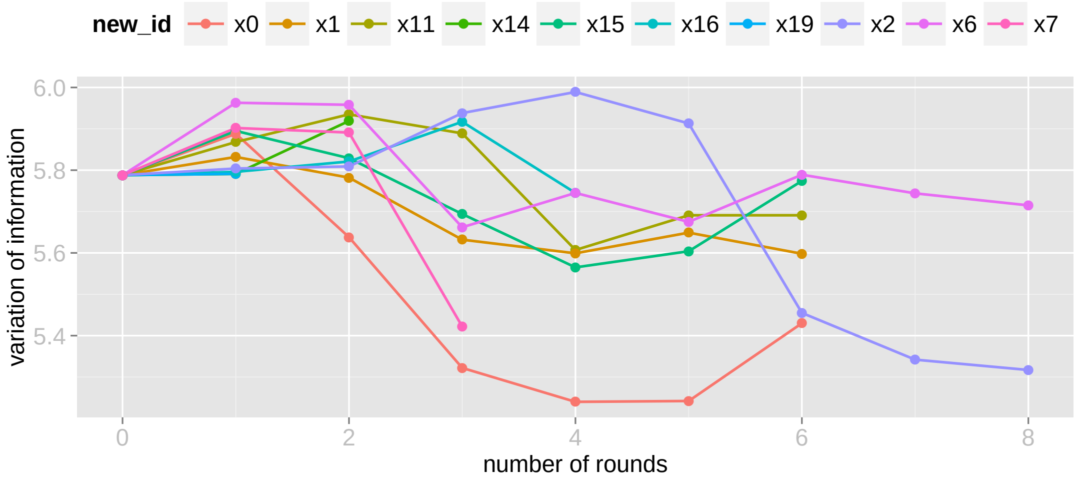


Figure 4.11: Distance to true labels (measured by Variation of information) changes as different users in experimental group refine topics. Users are labeled from x0 to x19. Ten of them are randomly assigned to experimental group. All ten users start with the same initial topics and are able to refine the topics for the extent of the refinement phase. Most users in the experimental group successfully reduced the variation of information (where lower is better).

legislative process—still can reduce this score by refining the topics. To avoid bias from users, users do not know that their topics will be evaluated by variation of information.

As shown in Figure 4.11, ten users in the experimental group started with the same initial topics and refined the topics for multiple rounds. In the given fifteen minutes, some users played with ITM for up to eight rounds while one user only tried two rounds. Although users occasionally increased the variation of information, by the end of the refinement phase a majority of users successfully reduced the variation of information of the topics.

User “x2” provides an example of a successful ITM round. This user saw a topic mixing “energy”-related words with other words. To make a coherent topic about “energy”, they put “oil”, “natural gas”, “gas”, “production” and “resources” in the important bin, and put “patriot_act”, “federal_government”, “tax_cuts”, “stem_cell” into the ignored bin. After updating, this topic became a coherent topic about “energy”. After refining topics for eight rounds, they successfully made other topics more coherent; he named these topics “homeland security”, “immigration”, “abortion”, “energy”, “flag burning”, etc., which match well with the corpus’s true clusters. Thus this user successfully reduced the variation of information as shown in Figure 4.11.

In addition to evaluating the variation of information for the experimental group, we also evaluated the users’ answers to content-specific questions. While the difference between the groups’ performance was not statistically significant, ITM changed the *usage pattern* to favor topic models over full text search.

To evaluate the **test**, we graded the answers and compared the scores of users in two groups. Of the 20 participants, two didn’t use their session name correctly, meaning the interface didn’t store their answers properly, and one user encountered an issue and wasn’t able to finish the questions. Thus we have complete answers for 17 participants. Each question was graded by two graders with Scott’s π agreement 0.839 (Artstein and Poesio, 2005). While there is no significant difference between the two groups’ test scores, the scores for experimental group had a much smaller variance compared to the control group.

To better understand how users answer the questions, the ITM system logs the number of full-text searches that include words from any of the topics (queried-topic-words) and the number of times that users used topics to filter query results

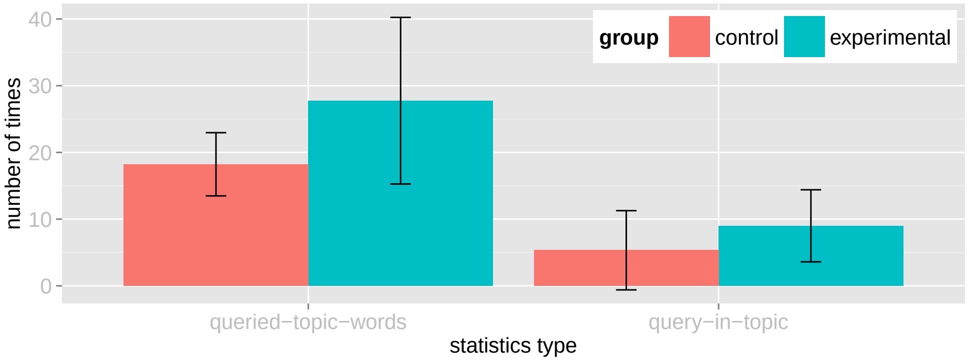


Figure 4.12: Statistics show that users' search strategies during the user study used topics more than the control group.

(query-in-topic).

The process of modifying topics inspired users in the experimental group to use queries that included words from the topics (Figure 4.12); this may be because users learned more key terms while exploring and refining the topics. These topic words are helpful to answer questions: users in experimental group queried topic words an average of 27.8 times, while the control group queried topic words 18.2 times on average. Users in the experimental group also used "query-in-topic" (restricting a full text search *within* a topic) more than the users in control group. This is probably because those users work with refined topics that are better aligned with the underlying bills (several questions were about specific bills).

We also found that users in both groups click topics much more when the question is about the general understanding of the data set, for example, "Name 5 of the debated legislation in this data set.". For more detailed questions like "The Gulf of Energy Security act will provide revenue streams for which fund?", users in both groups prefer to query text directly.

However, Figure 4.12 shows a large variance, so we should not overstate these results. In the conclusion, we discuss additional studies that can untangle the usefulness of topic models for evaluating information-seeking from other effects such as how familiar users are to topic models, whether they understand the task clearly, and whether they are effective consumers of information.

Some users in the control group also performed very well. For example, user “x5” in the control group obtained a high score. During the initial fifteen minute exploration phase, this user clicked on topics to review documents 71 times, substantially more than any user in either the control group or the experimental group. Users such as “x5”, who are topic model-savvy have better intuitions about how topic models work and how they can be used to help explore a corpus. In the post-session survey, the user reported that the interface, designed to facilitate ITM (but with interactive inference disabled for the control group) helped them understand the corpus and answer the questions.

Not all users in the experimental group performed well on the task. One user only refined two topics, and some users failed to improve the topics (failed to reduce the variation of information). Some users complained that they weren’t given enough time to update the topics.

In general, most reported liking the interface. Users from both the experimental group and the control group commented that the topics helped them answer some of the questions. Some users also commented that some of the questions were too detailed, suggesting that perhaps additional methods to search the corpus may be helpful.

This study provides evidence that the ITM interface assists users in exploring a large corpus and that topic modeling is helpful for users attempting to understand legislative documents. Users used ITM to improve the initial clusters; this is especially promising, as these users had little background knowledge of congressional debates and few had familiarity with topic models.

4.5 Automatically Suggesting Correlations

While we have demonstrated that ITM can encode correlations into topic models interactively, our pilot with users showed that it is often difficult, particularly for untrained users, to decide how to guide interactive topic models. This is because there are many possible choices: if the vocabulary size is V, there are about V^2 possible pair correlations (let alone higher order correlations). To save users' effort, we can either build the tree hierarchy of correlations automatically by hierarchical clustering techniques (as discussed in Chapter 7), or build on heuristics proposed for topic coherence to suggest correlations automatically as discussed in this section.

4.5.1 Generating New Correlations

Newman et al (2010) argues that topics whose words often appear close together in a reference corpus make more sense to users. They measure this through pointwise mutual information (PMI) averaged over all word pairs present in the top words of a topic (sorted in order of decreasing probability, as is usually done to show topics to a user). Thus, for a topic's top words T,

$$\mathrm{PMI}_{\mathcal{C}}(T) \equiv \frac{\sum_{(w_i,w_j):w_i \neq w_j} \mathrm{PMI}_{\mathcal{C}}(w_i, w_j)}{|T|(|T|-1)}, \tag{4.1}$$

where $\mathcal{C}$ is the corpus to compute PMI, and PMI is computed within a small local window. Topics that have a high score tend to be judged as "making sense" to users, and those that have lower scores tend to be judged as not making sense to users.

Based on this strategy, ITM could seek to improve this score by suggesting positive correlations for pairs with high PMI score and negative correlations for pairs with low PMI score. To ensure that we only suggest meaningful correlations, we weight suggestions by the tf-idf (Salton, 1968) for each word (by taking the max over all documents). This focuses correlations toward pairs that are highly relevant to at least some subset of the documents (this prevents correlations capturing syntactic or other dependencies). Combining the PMI and tf-idf score, we rank word pairs by

$$\mathrm{PC}(X, Y) = \max_d(\text{tf-idf}(X, d)) \cdot \max_d(\text{tf-idf}(Y, d)) \cdot \mathrm{PMI}_{\mathcal{C}}(X, Y) \tag{4.2}$$

for positive correlations (PC) and by

$$\mathrm{NC}(X, Y) = \frac{\max_d(\text{tf-idf}(X, d)) \cdot \max_d(\text{tf-idf}(Y, d))}{\mathrm{PMI}_{\mathcal{C}}(X, Y)} \tag{4.3}$$

for negative correlations (NC). The pairs with the highest scores for these metrics become the suggestions. Since the number of word pairs is very large, we only consider the word pairs from different topics for PC and the word pairs from the same topic for NC.

Although we have an automatic technique for selecting correlations, this does not replace a fully interactive system. This is because PMI cannot distinguish different meanings of the same word. For example, “msg” in Section 4.3.2 has two meanings “message” or “a food additive”, which belong to different topics. However, PMI treats “msg” as having a single meaning. In addition, while Newman et al (2010) argues that PMI matches well with users’ expectation, it may diverge in some cases, and automatic techniques cannot handle user-specific information needs (e.g., the examples in Section 4.1).

4.5.2 Human Evaluation over Automatically Generated Correlations

We use the 20 Newsgroups corpus (20News, described in Section 3.5) in this experiment. The topic number is set to be 20, and 100 iterations produce initial topics. Four rounds of interaction are performed with 50 iterations each round.

In this experiment, we have two different topic models: one uses automatically generated correlations based on PMI_{20news} (five positive and five negative correlations each round) and the other group runs for same number of iterations without any correlations. We name the two models as correlation group (ITM) and non-correlation group (LDA), respectively.

For evaluation, we showed the resulting topics to users on Mechanical Turk and asked whether they preferred the correlated topics (ITM), the control topics (LDA), or they looked equally coherent (Equal). Four users compared each pair. The

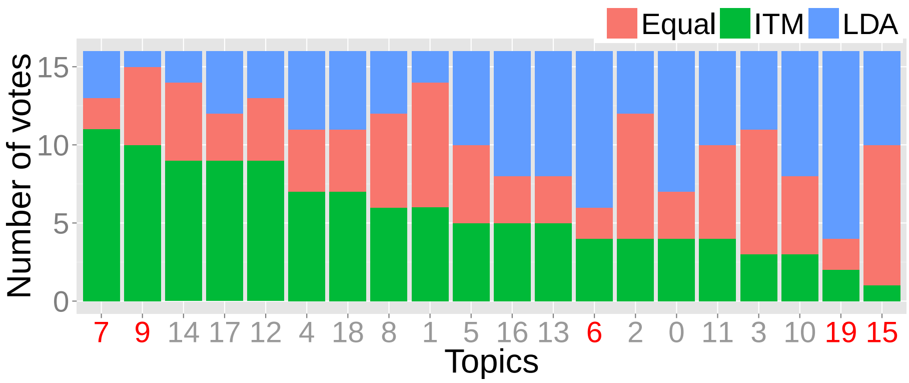


Figure 4.13: The total number of votes across rounds for each topic by users on Mechanical Turk (16 votes for each topic). x-axis is in a decreasing order of the number of votes for ITM. The red colored topics are significantly different from a uniform vote distribution, while the others are not (tested by χ^2-test). In general, there is no clear preference from users between models with correlations (ITM) and models without (LDA), which is the result in imbalanced attention being focused on some topics more than others.

positioning (i.e., left vs. right) and order of words within a topic were shuffled to avoid bias.

We first compare the votes for each group in each topic, as shown in Figure 4.13, in decreasing order of the number of votes for the correlated group (ITM). Users show significant preference in five of the total 20 topics (colored in red, by χ^2-test). Users did not have a clear preference overall; this was counter-intuitive, as the correlations were changing the topics dramatically. There were three reasons that the correlations did not always favor the ITM group:

- first, topics that are most confusing (as measured by Equation 4.2 and 4.3) get the correlations to improve coherence; thus the correlations affect some topics more than others
- second, because some topics have more correlations, other topics have fewer

correlations; thus those topics are similar to the uncorrelated case

- finally, some topics are ignored by correlations **and** have side-effects of correlations in the other topics; these are not always good for the coherence of the ignored topics

We describe each of these scenarios with examples, below.

Confusing Topics Get Correlations Some topics did show substantial improvement, however. For example, Figure 4.14 shows how Topic 7 changes as correlations are added (we only show the correlations related with Topic 7). Initially, the topic mixes "transportation" and "health", in the first round, a negative correlation between "car" and "cancer" pushed "car" away. Though "cars" remains, additional relevant terms—"medicine", "pain", "aids" and "effects" appear. In the second round, when a negative correlation between "cancer" and "msg" is added, "cars" disappears and words in correlated group (ITM) are mostly related with "health". The non-correlated group (LDA) also stabilizes but much more slowly than the correlated group (ITM); after the fourth round, however, users do not have a clear preference.

Ignored Topics Stay the Same While Topic 7 was improved, Topic 5, initially including words "software, data, set, graphics, code, used, sun, user, following, text" etc., stayed the same to users by the end of four rounds, as shown in Table 4.5. In the second round of Topic 5, one positive correlation {"window", "widget"} was added in ITM, but LDA already had these two correlated words in its topic, so

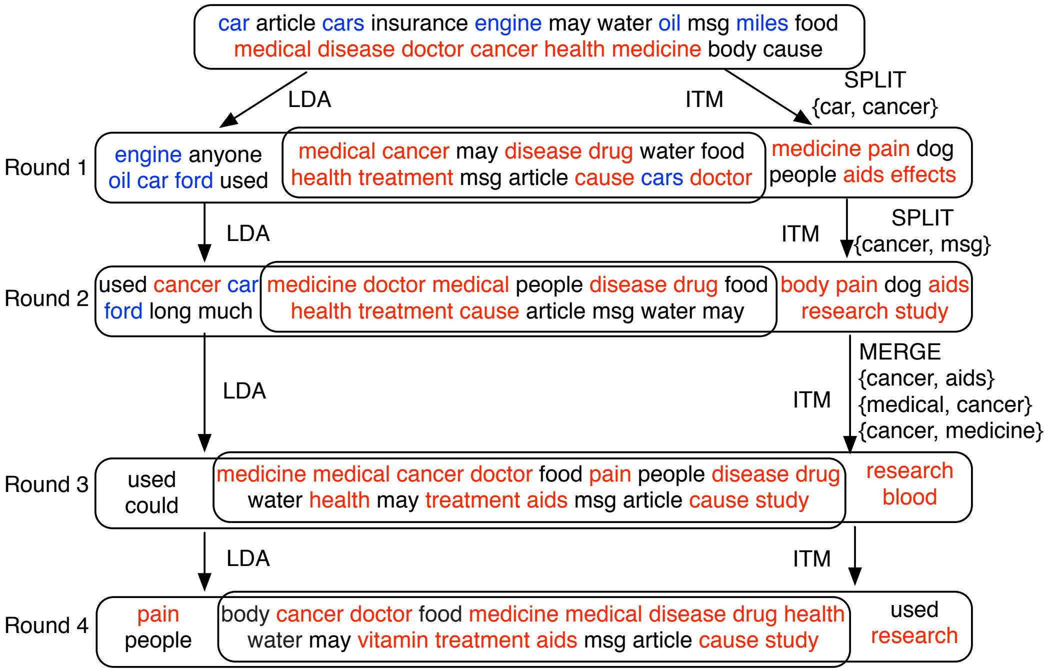

car article cars insurance engine may water oil msg miles food
medical disease doctor cancer health medicine body cause
LDA
ITM
SPLIT
{car, cancer}
Round 1
engine anyone
oil car ford used
medical cancer may disease drug water food
health treatment msg article cause cars doctor
medicine pain dog
people aids effects
LDA
ITM
SPLIT
{cancer, msg}
Round 2
used cancer car
ford long much
medicine doctor medical people disease drug food
health treatment cause article msg water may
body pain dog aids
research study
LDA
ITM
MERGE
{cancer, aids}
{medical, cancer}
{cancer, medicine}
Round 3
used
could
medicine medical cancer doctor food pain people disease drug
water health may treatment aids msg article cause study
research
blood
LDA
ITM
Round 4
pain
people
body cancer doctor food medicine medical disease drug health
water may vitamin treatment aids msg article cause study
used
research

Figure 4.14: Comparison of the "evolution" of Topic 7 between LDA and ITM for four rounds. Correlations related with this topic is shown and we compared the topic words between LDA (left) and ITM (right). At the beginning, Topic 7 is a mixed topic between "transportation" (blue) and "health" (red). While adding more correlations and running for additional iterations, this topic becomes a pure "health" topic. While after four rounds, LDA and ITM get similar result, ITM successfully converges to a good coherent topic much faster.

the correlation had little effect on the topic. In next round, two related negative correlations {"max", "font"} and {"max", "window"} were added to ITM, which improved the topic, so users prefer the ITM. In first and last round, users have no preference between the two groups.

Ignored Topics Suffer Side-effects Another case is that users sometimes decisively prefer the non-correlated group (LDA). For example, Topic 19, starts with words "system, good, power, much, research, first, large, data, systems, work" etc., only had one relevant correlation, a negative correlation {"max", "model"} added in the final round. However, the result of correlations in previous rounds negatively impacted Topic 19, which had words “stolen” from it by other topics. This shows that improving some topics (e.g., Topic 7) sometimes comes at the expense of others; Table 4.5 shows that users preferred the version of the topic left untouched by correlations.

Round	Topic 5			Topic 19		
	ITM	Equal	LDA	ITM	Equal	LDA
R1	2	1	1	0	2	2
R2	0	0	4	0	0	4
R3	3	0	1	1	0	3
R4	0	4	0	1	0	3

Table 4.5: The round votes for Topic 5 and Topic 19. For Topic 5, while users have no preference in R1 and R4, they prefer LDA in R2: one correlation “MERGE {window, widget}” was added in ITM, but LDA also had the two correlated words in its topic, so there was no clear improvement; in R3, two related correlations (“SPLIT {max, font}” and “SPLIT {max, window}”) were added to ITM, and users prefer the improved topic in ITM. For Topic 19, no related correlations were added to ITM in the first three rounds; there was one unimportant correlation “SPLIT {max, model}” added in R4, which resulted in no clear improvement. So users clearly prefer LDA for Topic 19.

Figure 4.14 shows that for LDA and ITM, Topic 7 converged to the same “health” topic, but ITM helped it converge the faster. We discuss the distinction between improvements and impatience in Section 4.1.4, which also gives an example of a topic that remains problematic even in a fully converged topic model.

4.6 Summary

This chapter proposes to obtain prior knowledge from users and presents interactive topic modeling, which updates topics iteratively based on users’ feedback. While we have developed two types of interface and evaluated this framework with real users, better visualization can further aid users and save users' time, for example, instead of displaying each topic as a bag of words, the collocation of words in a document can also be displayed for each topic (Smith et al, 2014), which hopefully can help users to provide feedback.

While better visualization can assist users, whether the whole interactive framework are helpful for users to understand the corpus must be evaluated with real users. While this chapter has already done user study in an information seeking task, our user population is too diverse and small. Broadening the number of users would allow us to draw stronger conclusions.

This chapter focuses on extracting flat topics from an interactive setting, which can be further extended to modeling hierarchical topics (Blei et al, 2003a) and dynamic topics (Blei and Lafferty, 2006) interactively and iteratively. This framework is also flexible enough to be applied in multilingual topic models, which is used in domain adaptation for statistical machine translation in the next chapter (Chapter 5).

Interactive topic modeling in general is a new framework for using and understanding statistical models that empowers users of topic models to evaluate and refine topics based on their unique expertise and perspective. These models offer new opportunities for wider and richer use of topic models, suggest new probabilistic models and methods for machine learning, and can serve as an exemplar for allowing users to improve statistical models.

Chapter 5

Polylingual Tree-based Topic Models for SMT Domain Adaptation

While we have introduced the tree-based topic models in Chapter 3 and applied this model in an interactive framework in Chapter 4, this chapter extends the tree-based topic models to polylingual tree-based topic models and uses them to aid *statistical machine translation* (Koehn, 2009, SMT). This chapter directly uses the existing knowledge resources such as dictionaries or WordNet to improve SMT, but it can also be further extended to an interactive setting and obtain prior knowledge from users as in Chapter 4.

In this chapter, we review existing topic models for discovering topics in multilingual datasets and discuss how they can improve SMT (Section 5.1); we create a model—polylingual tree-based topic models (ptLDA)—that use information from both external dictionaries and document alignments simultaneously (Section 5.2), and derive both MCMC and variational inference for this new topic model (Section 5.3); we further evaluate our model on the task of SMT using aligned datasets, show that ptLDA offers better domain adaptation than other topic models for machine translation (Section 5.4), and discuss how these topic models improve SMT with detailed examples (Section 5.5).

5.1 Topic Models for Machine Translation

Modern machine translation systems use millions of examples of translations to learn translation rules in local (phrase) context, while ignoring the global (document) context. These systems work best when the training corpus has consistent global context, including genre, register, and topic. Systems that are robust to systematic variation in the training set are said to exhibit *domain adaptation*.

Topic models are a promising solution for automatically discovering the global context—for example, domain knowledge—in machine translation corpora. However, past work either relies solely on monolingual source-side models (Eidelman et al, 2012; Hasler et al, 2012; Su et al, 2012), or limited modeling of the target side (Xiao et al, 2012). In contrast, machine translation uses inherently multilingual data: an SMT system must translate a phrase or sentence from a *source* language to a different *target* language, so existing applications of topic models (Eidelman et al, 2012) are ignoring available information on the target side that could aid domain discovery.

This is not for a lack of multilingual topic models. Topic models bridge the chasm between languages using document connections (Mimno et al, 2009), dictionaries (Boyd-Graber and Resnik, 2010), and word alignments (Zhao and Xing, 2006). However, no models combine multiple bridges between languages. Before introducing a new topic model that connects different languages using multiple bridges, we briefly review lexical weighting and domain adaptation for SMT.

5.1.1 Statistical Machine Translation

Statistical machine translation casts machine translation as a probabilistic process (Koehn, 2009). For a parallel corpus of aligned source and target sentences $(\mathcal{F}, \mathcal{E})$, a phrase $\bar{f} \in \mathcal{F}$ is translated to a phrase $\bar{e} \in \mathcal{E}$ according to a distribution $p_w(\bar{e}|\bar{f})$. One popular method to estimate the probability $p_w(\bar{e}|\bar{f})$ is using lexical weighting features.

Lexical Weighting In phrase-based SMT, lexical weighting features estimate the phrase pair quality by combining lexical translation probabilities of words in a phrase (Koehn et al, 2003). Lexical conditional probabilities $p_w(e|f)$ are maximum likelihood estimates from relative lexical frequencies $c(f, e)/\sum_e c(f, e)$, where $c(f, e)$ is the count of observing lexical pair (f, e) in the training dataset. Given a word alignment a, the lexical weight for this phrase pair $p_w(\bar{e}|\bar{f}; a)$ is the normalized product of lexical probabilities of the aligned word pairs within that phrase pair (Koehn et al, 2003):

$$p_w(\bar{e}|\bar{f}; a) = \prod_{i=1}^{n} \frac{1}{\{|j|(i,j) \in a\}|} \sum_{\forall (i,j) \in a} p_w(e_i|f_j) \tag{5.1}$$

where i and j are the word positions in target phrase $\bar{e}$ and source phrase $\bar{f}$ respectively. In Section 5.1.2, we create topic-specific lexical weighting features.

Cross-Domain SMT A SMT system is usually trained on documents with the same genre (e.g., sports, business) from a similar style (e.g., newswire, blog-posts). These are called *domains.* Translations within one domain are better than translations across domains since they vary dramatically in their word choices and style. A correct translation in one domain may be inappropriate in another domain. For example, “潜水” in a newspaper usually means “underwater diving”. On social media, it means a non-contributing “lurker”.

Domain Adaptation for SMT Training a SMT system using diverse data requires *domain adaptation*. Early efforts focus on building separate models (Foster and Kuhn, 2007) and adding features (Matsoukas et al, 2009) to model domain information. Chiang et al (2011) combine these approaches by directly optimizing genre and collection features by computing separate translation tables for each domain.

However, these approaches treat domains as hand-labeled, constant, and known *a priori*. This setup is at best expensive and at worst infeasible for large data. Topic models provide a solution where domains can be automatically induced from raw data: treat each topic as a domain.[1]

5.1.2 Inducing Domains with Topic Models

Topic models take the number of topics K and a collection of documents as input, where each document is a bag of words. They output two distributions: a distribution over topics for each document d; and a distribution over words for each topic. If each topic defines a SMT domain, the document’s topic distribution is a soft domain assignment for that document.

Given the soft domain assignments, Eidelman et al (2012) extract lexical

[1] Henceforth we will use the term “topic” and “domain” interchangeably: “topic” to refer to the concept in topic models and “domain” to refer to SMT corpora.

weighting features conditioned on the topics, optimizing feature weights using the *Margin Infused Relaxed Algorithm* (Crammer et al, 2006, MIRA). The topics come from source documents *only* and create topic-specific lexical weights from the per-document topic distribution $p(k \mid d)$. The lexical probability conditioned on the topic is expected count $e_k(e, f)$ of a word translation pair under topic k,

$$\hat{c}_k(e, f) = \textstyle\sum_d p(k|d) c_d(e, f), \tag{5.2}$$

where $c_d(\bullet)$ is the number of occurrences of the word pair in document d. The lexical probability conditioned on topic k is the unsmoothed probability estimate of those expected counts

$$p_w(e|f; k) = \hat{c}_k(e, f) / \textstyle\sum_e \hat{c}_k(e, f), \tag{5.3}$$

from which we can compute the lexical weight of this phrase pair $p_w(\bar{e}|\bar{f}; a, k)$ given a word alignment a(Koehn et al, 2003):

$$p_w(\bar{e}|\bar{f}; a, k) = \prod_{i=1}^{n} \frac{1}{\{|j|(i, j) \in a\}|} \sum_{\forall(i,j)\in a} p_w(e_i|f_j; k) \tag{5.4}$$

where i and j are the word positions in target phrase $\bar{e}$ and source phrase $\bar{f}$ respectively.

For a test document d, the document topic distribution $p(k \mid d)$ is inferred based on the topics learned from training data. The lexical weight feature of a phrase pair

$(\bar{e}, \bar{f})$ is,

$$f_k(\bar{e}|\bar{f}) = -\log\left\{p_w(\bar{e}|\bar{f};k) \cdot p(k|d)\right\}, \tag{5.5}$$

a combination of the topic dependent lexical weight and the topic distribution of the document, from which we extract the phrase.

Given the topic-adapted features, Eidelman et al (2012) compute the resulting model score by combining these adapted features in a linear model with other standard SMT features and optimizing the weights:

$$\underbrace{\sum_p \lambda_p f_p(\bar{e}, \bar{f})}_{\text{standard features}} + \underbrace{\sum_k \lambda_k f_k(\bar{e}|\bar{f})}_{\text{adapted features}} \tag{5.6}$$

These adapted features allow us to bias the translations according to the topics. For example, if topic k is dominant in a test document, the feature $f_k(\bar{e}|\bar{f})$ will be large, which may bias the decoder to a translation that has small value of the standard feature $f_p(\bar{e}|\bar{f})$. In addition, combining the adapted features with the standard features makes this model more flexible. For a test document with less clear topics, the topic distribution will tend toward being fairly uniform. In this case, the topic features will contribute less to the translation results and the standard features will dominate the translation results.

Conceptually, this approach is just reweighting examples. The probability of a topic given a document is never zero. Every translation observed in the training set will contribute to $p_k(e|f)$; many of the expected counts, however, will be less than one. This obviates the explicit smoothing used in other domain adaptation systems (Chiang et al, 2011).

We adopt this framework in its entirety. Our contribution are topics that capture *multilingual* information and thus better capture the domains in the parallel corpus. To extract better multilingual topics, we use the existing multilingual knowledge resources such as bilingual dictionaries as the prior knowledge, which can also be obtained from users in an interactive setting.

5.1.3 Beyond Vanilla Topic Models

Eidelman et al (2012) ignore a wealth of information that could improve topic models and help machine translation. Namely, they only use monolingual data from the source language, ignoring all target-language data and available lexical semantic resources between source and target languages.

Different languages complement each other to reduce ambiguity. For example, “木马” in a Chinese document can be either “hobbyhorse” in a children’s topic, or “Trojan virus” in a technology topic. A short Chinese context obscures the true topic. However, these terms are unambiguous in English, revealing the true topic.

While vanilla topic models (LDA) can only be applied to monolingual data, there are a number of topic models for parallel corpora: Zhao and Xing (2006) assume aligned word pairs share same topics; Mimno et al (2009) connect different languages through comparable documents. These models take advantage of word or document *alignment information* and infer more robust topics from the aligned dataset.

On the other hand, *lexical information* can induce topics from multilingual corpora. For instance, orthographic similarity connects words with the same meaning in related languages (Boyd-Graber and Blei, 2009), and dictionaries are a more general source of information on which words share meaning (Boyd-Graber and Resnik, 2010).

These two approaches are not mutually exclusive, however; they reveal different connections across languages. In the next section, we combine these two approaches into a polylingual tree-based topic model.

5.2 Polylingual Tree-based Topic Models

In this section, we bring existing tree-based topic models (Boyd-Graber et al, 2007, tLDA) and polylingual topic models (Mimno et al, 2009, pLDA) together and create the polylingual tree-based topic model (ptLDA) that incorporates both word-level correlations and document-level alignment information.

Word-level Correlations As introduced in Chapter 2, tree-based topic models incorporate the positive correlations between words by encouraging words that appear together in a **concept** to have similar probabilities given a topic.[2] These concepts can come from WordNet (Boyd-Graber and Resnik, 2010), domain experts (Andrzejewski et al, 2009), or user constrains (Hu et al, 2013, Chapter 4). When we gather concepts from bilingual resources, these concepts can connect different languages. For example,

[2] One positive correlation is built on words in a concept. We use “concept” instead of “positive correlation” in this chapter.

if a bilingual dictionary defines "电脑" as "computer", we combine these words in a concept.

These concepts (positive correlations) are organized into a **prior tree** structure as in Chapter 2. As Figure 5.1 shows, words in the same concept share a common parent node, and then that concept becomes one of many children of the root node. Words that are not in any concept—**uncorrelated words**—are directly connected to the root node.

When this tree serves as a prior for topic models, words in the same concept are positively correlated in topics (Chapter 2). For example, if "电脑" has high probability in a topic, so will "computer", since they share the same parent node. With the tree priors, each topic is no longer a distribution over word types; instead, it is a distribution over paths, and each path is associated with a word type. The same word could appear in multiple paths, and each path represents a unique sense of this word.

Document-level Alignments Lexical resources connect languages and help guide the topics. However, these resources are sometimes brittle and may not cover the whole vocabulary. Aligned document pairs provide a more corpus-specific, flexible association across languages.

Landauer and Littman (1990) connect documents in different languages by projecting both documents to a shared latent semantic indexing space. Similarly, polylingual topic models (Mimno et al, 2009) assume that the aligned documents in different languages share the same topic distribution and each language has a unique topic distribution over its word types. This level of connection between languages is flexible: instead of requiring the exact matching on words and sentences, only a coarse document alignment is necessary, as long as the documents discuss the same topics.

Combine Words and Documents We introduce polylingual tree-based topic models (ptLDA), which connect information across different languages by incorporating both word correlation (as in tLDA) and document alignment information (as in pLDA). We initially assume a given tree structure, deferring the tree's provenance to the end of this section.

Generative Process As in LDA, each word token is associated with a topic. However, tree-based topic models introduce an additional step of selecting a concept in a topic responsible for generating each word token. This is represented by a path $y_{d,n}$ through the topic's tree.

The probability of a path in a topic depends on the transition probabilities in a topic. Each concept i in topic k has a distribution over its children nodes is governed by a Dirichlet prior: $\pi_{k,i} \sim \mathrm{Dir}(\beta_i)$. Each path ends in a word (i.e., a leaf node) and the probability of a path is the product of all of the transitions between topics it traverses. Topics have correlations over words because the Dirichlet parameters can encode positive or negative correlations (Andrzejewski et al, 2009).

With these correlated in topics in hand, the generation of documents is very similar to LDA. For every document d, we first sample a distribution over topics θ_d

Algorithm 5 Generative Process for ptlda

```
for topic $k \in 1, \cdots, K$ do
    for each internal node $n_i$ do
        draw a distribution $\pi_{ki} \sim \mathrm{Dir}(\beta_i)$
for document set $d \in 1, \cdots, D$ do
    draw a distribution $\theta_d \sim \mathrm{Dir}(\alpha)$
    for each word in documents $d$ do
        choose a topic $z_{dn} \sim \mathrm{Mult}(\theta_d)$
        sample a path $y_{dn}$ with probability $\prod_{(i,j) \in y_{dn}} \pi_{z_{dn},i,j}$
        $y_{dn}$ leads to word $w_{dn}$ in language $l_{dn}$
        append token $w_{dn}$ to document $d_{l_{dn}}$
```

from a Dirichlet prior $\mathrm{Dir}(\alpha)$. For every token in the documents, we first sample a topic z_{dn} from the multinomial distribution θ_d, and then sample a path y_{dn} along the tree according to the transition distributions specified by topic z_{dn}. Because every path y_{dn} leads to a word w_{dn} in language l_{dn}, we append the sampled word w_{dn} to document $d_{l_{dn}}$. Aligned documents have words in both languages; monolingual documents only have words in a single language.

The full generative process is shown in Algorithm 5.

If we use a flat symmetric Dirichlet prior instead of the tree prior, we recover pLDA; and if all documents are monolingual (i.e., with distinct distributions over topics θ), we recover tLDA. ptLDA connects different languages on both the word level (using the word correlations) and the document level (using the document alignments). We compare these models' machine translation performance in Section 5.4.

Build Prior Tree Structures One remaining question is the source of the word-level connections across languages for the tree prior. We consider two resources to build trees that correlate words across languages. The first is multilingual dictionaries (*dict*), which match words with the same meaning in different languages

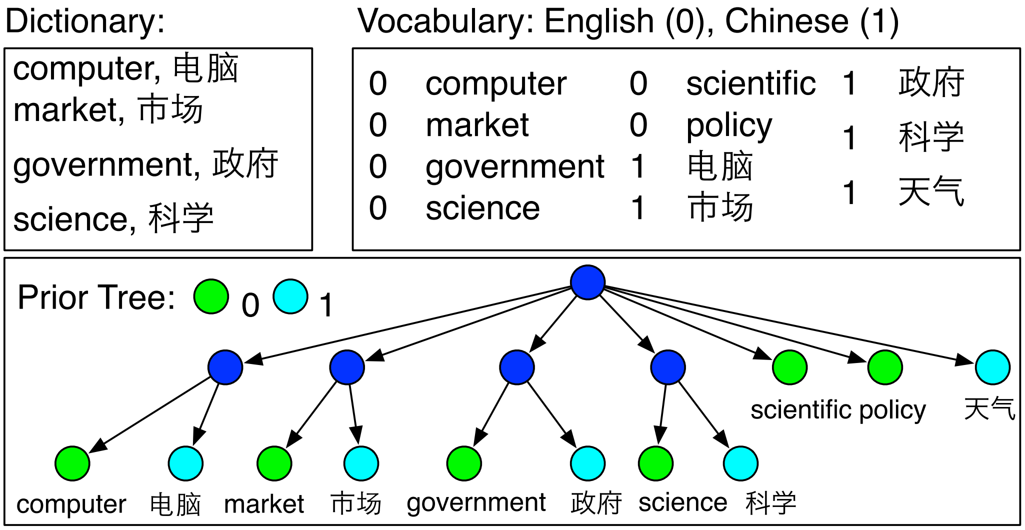


Figure 5.1: An example of constructing a prior tree from a bilingual dictionary: word pairs with the same meaning but in different languages are concepts; we create a common parent node to group words in a concept, and then connect to the root; uncorrelated words are connected to the root directly. Each topic uses this tree structure as a prior.

together. These relations between words are used as the concepts (Bhattacharya, 2006) in the prior tree (Figure 5.1).

In addition, we extract the word alignments from aligned sentences in a parallel corpus. The word pairs define concepts for the prior tree (*align*). We use both resources for our models (denoted as ptLDA-*dict* and ptLDA-*align*) in our experiments (Section 5.4) and show that they yield comparable performance in SMT.

5.3 Inference

Inference of probabilistic models discovers the posterior distribution over latent variables. The first choice is *Gibbs sampling* (Neal, 1993), which is basically the same as what we have discussed in Chapter 2 and Chapter 3, thus it is omitted in this chapter to avoid duplication.

A second choice is *variational Bayesian inference* (Blei et al, 2003b, VB), which also produces good approximations of the posterior mode (Asuncion et al, 2009) as *Gibbs sampling.* In addition, Mimno et al (2012) propose hybrid inference that takes advantage of parallelizable variational inference for global variables (Wolfe et al, 2008) while enjoying the sparse, efficient updates for local variables (Neal, 1993).

The variational inference and the hybrid inference for polylingual tree-based topic models are derived and developed in collaboration with Ke Zhai, who also parallelized variational inference using hadoop MapReduce. The equations for variational inference are attached in Appendix C. More details can be found in Hu et al (2014).

5.4 Experiments

We evaluate our new topic model, ptLDA, and existing topic models—LDA, pLDA, and tLDA—on their ability to induce domains for machine translation and the resulting quality of the translations using standard machine translation metrics.

Dataset and SMT Pipeline We use the NIST MT Chinese-English parallel corpus (NIST), excluding non-UN and non-HK Hansards portions as our training dataset. It contains 1.6M sentence pairs, with 40.4M Chinese tokens and 44.4M English tokens. We replicate the SMT pipeline of Eidelman et al (2012): word segmentation (Tseng et al, 2005), align (Och and Ney, 2003), and symmetrize (Koehn et al, 2003) the data. We train a modified Kneser-Ney trigram language model on English (Chen and Goodman, 1996). We use CDEC (Dyer et al, 2010) for decoding, and MIRA (Crammer et al, 2006) for parameter training. To optimize SMT system, we tune the parameters on NIST MT06, and report results on three test sets: MT02, MT03 and MT05.[3]

Topic Models Configuration We compare our polylingual tree-based topic model (ptLDA) against tree-based topic models (tLDA), polylingual topic models (pLDA) and vanilla topic models (LDA).[4] We also examine different inference algorithms—Gibbs sampling (**gibbs**), variational inference (**variational**) and a hybrid approach (**variational-hybrid**)—on the effects of SMT performance. In all experiments, we set the per-document Dirichlet parameter $\alpha = 0.01$ and the number of topics to 10, as used in Eidelman et al (2012).

Resources for Prior Tree To build the tree for tLDA and ptLDA, we extract the word correlations from a Chinese-English bilingual dictionary (Denisowski, 1997).[5] We filter the dictionary using the NIST vocabulary, and keep entries mapping single Chinese and single English words. The prior tree has about 1000 word pairs (*dict*).

We also extract the bidirectional word alignments between Chinese and English using GIZA++ (Och and Ney, 2003). We then remove the word pairs appearing more than 50K times or fewer than 500 times and construct a second prior tree with about 2500 word pairs (*align*).

We apply both trees to tLDA and ptLDA, denoted as tLDA-*dict*, tLDA-*align*, ptLDA-*dict*, and ptLDA-*align*. However, tLDA-*align* and ptLDA-*align* do worse than

[3] The NIST datasets contain 878, 919, 1082 and 1664 sentences for MT02, MT03, MT05 and MT06 respectively.

[4] For Gibbs sampling, we use implementations available in Hu and Boyd-Graber (2012) for tLDA; and Mallet (McCallum, 2002) for LDA and pLDA.

[5] This is a two-level tree structure. However, one could build a more sophisticated tree prior with a hierarchical dictionary such as multilingual WordNet.

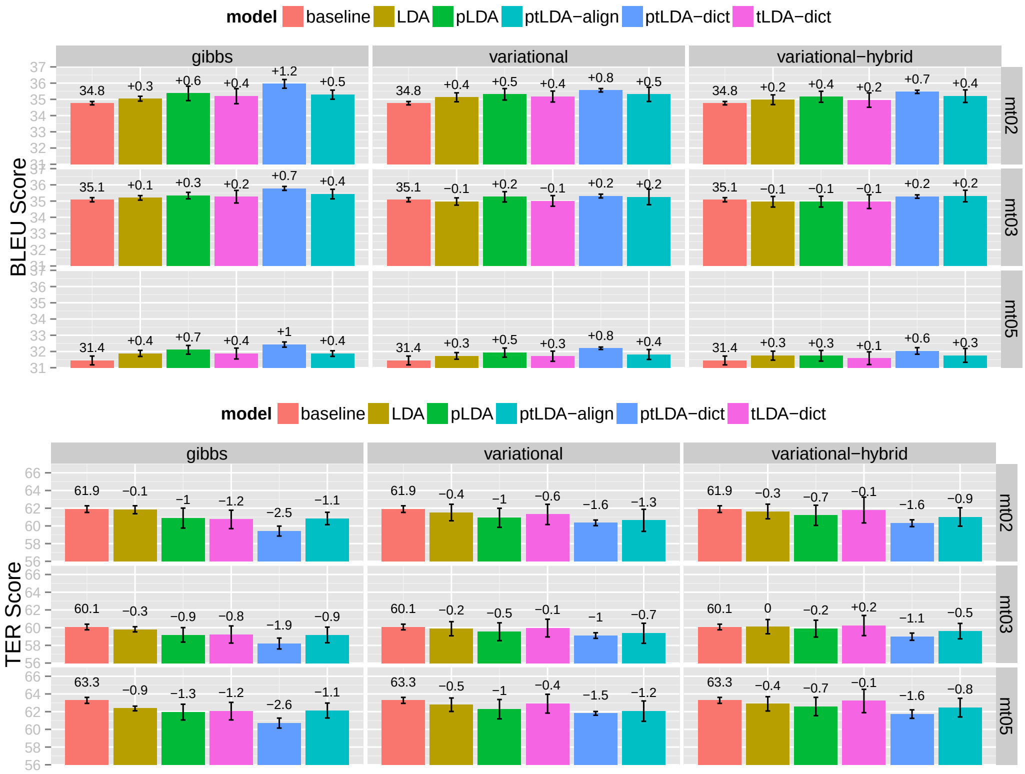


Figure 5.2: Machine translation performance for different models and inference algorithms against the baseline, on BLEU (top, higher the better) and TER (bottom, lower the better) scores. Our proposed ptLDA performs best. Results are averaged over 5 random runs. For model ptLDA-*dict* with different inference schemes, the BLEU improvement on three test sets is mostly significant with $p = 0.01$, except the results on MT03 using variational and variational-hybrid inferences.

tLDA-*dict* and ptLDA-*dict*, so we omit tLDA-*align* in the results.

Domain Adaptation using Topic Models We examine the effectiveness of using topic models for domain adaptation on standard SMT evaluation metrics—BLEU (Papineni et al, 2002) and TER (Snover et al, 2006). We report the results on three different test sets (Figure 5.2), and all SMT results are averaged over five runs.

We refer to the SMT model without domain adaptation as **baseline**.[6] LDA marginally improves machine translation (less than half a BLEU point). Polylingual

[6] Our replication of Eidelman et al (2012) yields slightly higher baseline performance, but the trend is consistent.

topic models pLDA and tree-based topic models tLDA-*dict* are consistently better than LDA, suggesting that incorporating additional bilingual knowledge improves topic models. These improvements are not redundant: our new ptLDA-*dict* model, which has aspects of both models yields the best performance among these approaches—up to a 1.2 BLEU point gain (higher is better), and -2.6 TER improvement (lower is better). The BLEU improvement is significant (Koehn, 2004) at $p = 0.01$,[7] except on MT03 with variational and variational-hybrid inference.

While ptLDA-*align* performs better than **baseline** SMT and LDA, it is worse than ptLDA-*dict*, possibly because of errors in the word alignments, making the tree priors less effective.

One thing that we didn't compare directly in Figure 5.2 is to treat the bilingual dictionary as additional training data or knowledge base, and then train the translation system. However, knowledge-based machine translation systems (Nirenburg, 1989) do not work as well as statistical machine translation systems. In addition, the ptLDA-*align*, which doesn't have extra information other than training data, also performs better than the current **baseline**. This shows that topic models do improve translation results, even without extra dictionary information.

Scalability While **gibbs** has better translation scores than **variational** and **variational-hybrid**, it is less scalable to larger datasets. With 1.6M NIST training sentences, **gibbs** takes nearly a week to run 1000 iterations. In contrast, the parallelized **variational** and **variational-hybrid** approaches, which we implement in

[7] Because we have multiple runs of each topic model (and thus different translation models), we select the run closest to the average BLEU for the translation significance test.

MapReduce (Dean and Ghemawat, 2004; Wolfe et al, 2008; Zhai et al, 2012), take less than a day to converge.

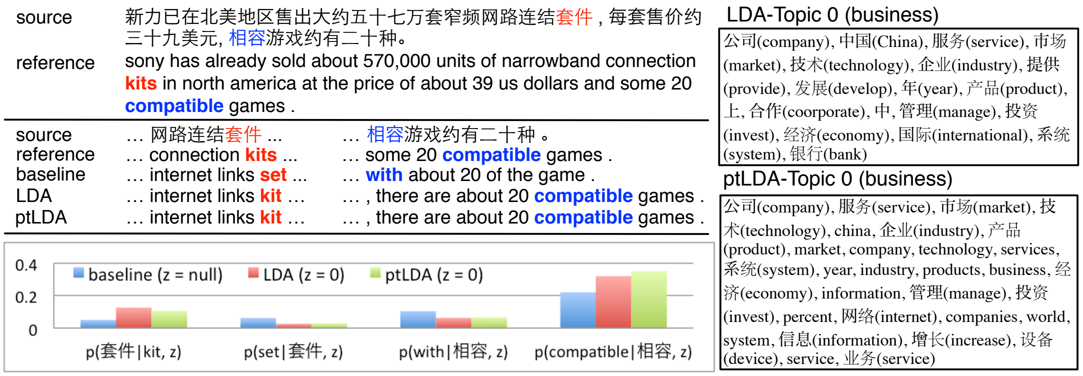
5.5 Discussion

In this section, we qualitatively analyze the translation results and investigate how ptLDA and its cousins improve SMT. We also discuss other approaches to improve unsupervised domain adaptation for SMT.

5.5.1 How do Topic Models Help SMT?

We present two examples of how topic models can improve SMT. The first example shows both LDA and ptLDA improve the **baseline**. The second example shows how LDA introduce biases that mislead SMT and how ptLDA's bilingual constraints correct these mistakes.

Figure 5.3 shows a sentence about a company introducing new technology gadgets where both LDA and ptLDA improve translations. The **baseline** translates "套件" to "set" (red), and "相容" to "with" (blue), which do not capture the reference meaning of a *add-on device* that works with *compatible* games. Both LDA and ptLDA assign this sentence to a business domain, which makes the translations probabilities shift toward correct translations: the probability of translating "相容" to "compatible" and the probability of translating "套件" to "kit" in the business domain are both significantly larger than without the domain knowledge; and the probabilities of translating "相容" to "with" and the probability of translating "set" to "套件" in



source	新力已在北美地区售出大约五十七万套窄频网路连结套件 , 每套售价约三十九美元, 相容游戏约有二十种。
reference	sony has already sold about 570,000 units of narrowband connection **kits** in north america at the price of about 39 us dollars and some 20 **compatible** games .

source	… 网路连结套件 ...	… 相容游戏约有二十种 。
reference	… connection **kits** ...	… some 20 **compatible** games .
baseline	… internet links **set** ...	… **with** about 20 of the game .
LDA	… internet links **kit** …	… , there are about 20 **compatible** games .
ptLDA	… internet links **kit** …	… , there are about 20 **compatible** games .

LDA-Topic 0 (business)

公司(company), 中国(China), 服务(service), 市场(market), 技术(technology), 企业(industry), 提供(provide), 发展(develop), 年(year), 产品(product), 上, 合作(coorporate), 中, 管理(manage), 投资(invest), 经济(economy), 国际(international), 系统(system), 银行(bank)

ptLDA-Topic 0 (business)

公司(company), 服务(service), 市场(market), 技术(technology), china, 企业(industry), 产品(product), market, company, technology, services, 系统(system), year, industry, products, business, 经济(economy), information, 管理(manage), 投资(invest), percent, 网络(internet), companies, world, system, 信息(information), 增长(increase), 设备(device), service, 业务(service)

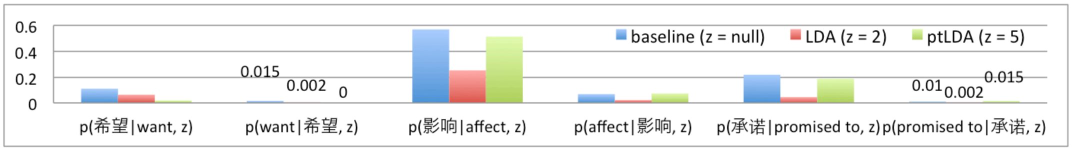
Figure 5.3: Better SMT result using topic models for domain adaptation. Top row: the source sentence and its reference translation. Middle row: the highlighted translations from different approaches. Bottom row: the change of relevant translation probabilities after incorporating the domain knowledge from LDA and ptLDA. Right: most-probable words of the topic the source sentence is assigned to under LDA (top) and ptLDA (bottom). The Chinese translations are in parenthesis.

the business domain decrease.

The second example (Figure 5.4) illustrates how ptLDA offers further improvements over LDA. The source sentence discusses foreign affairs. The **baseline** correctly translates the word “影响” to “affect”. However, LDA—which only takes monolingual information from the source language—assigns this sentence to economic development. This misleads SMT to lower the probability for the correct translation “affect”; it chooses “impact” instead. In contrast, ptLDA—which incorporates bilingual constraints—successfully labels this sentence as foreign affairs and produces a softer, more nuanced translation that better matches the reference. The translation of “承诺” is very similar, except in this case, both the **baseline** and LDA produce the incorrect translation “the commitment of”. This is possible because the probabilities of translating “承诺” to “promised to” and translating “promised to” to “承诺” (the correct translation, in both directions) increase when conditioned on ptLDA’s correct topic but decrease when conditioned on LDA’s incorrect topic.

source	消息指出，韩国使馆人员向中方官员表示，韩国方面并没有支持朝鲜人以这种方法前往韩国，韩国并不希望这类事件再次发生，以免对中国和朝鲜半岛双方间的关系带来影响，韩国方面并向中国方面承诺，愿意协助中国管理好在京的韩国居民
reference	sources said rok embassy personnel told chinese officials that rok has not backed any dpr koreans to get to rok in such a manner and rok would not **like** **such** things happen again to **affect** relationship between china and the two sides of the korean peninsula . rok also **promised to** assist china in the administration of koreans in beijing .

source	… 不希望 ...	… 以免对…关系带来影响…	… 韩国方面并向中国方面承诺…
reference	… would not **like** ...	… to **affect** the relationship …	… rok also **promised to** the chinese side ...
baseline	… does not **want** ...	… so as not to **affect** the relations…	… south korea and **the commitment of** the chinese side ...
LDA	… does not **hope that** ...	… so as to avoid **impact** the relations…	… the rok side , and **the commitment of** the chinese side ...
ptLDA	… does not **hope that** ...	… so as not to **affect** the relations…	… south korea has **promised to** the chinese side ...



LDA-Topic 5 (economic development)	ptLDA-Topic 2 (foreign affairs)
发展(develop), 国(country), 两(two), 中国(China), 关系(relation), 中, 合作(cooperate), 经济(economy), 人民(people), 友好(friendly), 国家(country), 新(new), 问题(problem), 上, 加强(emphasize), 重要(important), 和平(peace), 共同(together), 建设(build), 世界(world)	china, 问题(issue), military, united, president, 国家(country), 地区(area), minister, 伊拉克(Iraq), 和平(peace), nuclear, people, 总统(president), peace, security, 联合国(UN), 军事(military), 以色列(Israel), iraq, foreign, international, 部队(army), beijing, world, defense, south, 安 全(security), war, 协议(agreement), 会议(conference)

Figure 5.4: Better SMT result using ptLDA compared to LDA and the baseline. Top row: the source sentence and a reference translation. Second row: the highlighted translations from different models. Third row: the change of relevant translation probabilities after incorporating domain knowledge from LDA and ptLDA. Bottom row: most-probable words for the topics the source sentence is assigned to under LDA (left) and ptLDA (right). The meanings of Chinese words are in parenthesis.

5.5.2 Other Approaches

Other approaches have used topic models for machine translation. Xiao et al (2012) present a topic similarity model based on LDA that produces a feature that weights grammar rules based on topic compatibility. They also model the source and target side of rules and compare the target similarity during decoding by projecting the target distribution into the source space. Hasler et al (2012) use the source-side topic assignments from *hidden topic Markov models* (Gruber et al, 2007, HTMM) which model documents as a Markov chain and assign one topic to the whole sentence, instead of a mixture of topics. Su et al (2012) also apply HTMM to monolingual data and apply the results to machine translation. To our knowledge, however, this is the first work to use *multilingual* topic models for domain adaptation in machine translation.

In addition to topic models, Kuhn et al (2010) cluster the phrases in both source and target languages based on the information in the phrase table, and compute the phrase probabilities based on the probabilities of phrase clusters and the phrase probability within that cluster. However, this is still limited to local context. Our model provides a more flexible to connect source and target languages and capture global context information.

5.5.3 Improving Language Models

Topic models capture document-level properties of language, but a critical component of machine translation systems is the language model, which provides local constraints and preferences. Domain adaptation for language models (Bellegarda, 2004; Wood and Teh, 2009) is an important avenue for improving machine translation. Models that simultaneously discover global document themes as well as local, contextual domain-specific information (Wallach, 2006; Boyd-Graber and Blei, 2008) may offer further improvements.

5.5.4 External Data

The topic models presented here only require weak alignment between documents at the document level. Extending to larger datasets for learning topics is straightforward in principle. For example, ptLDA could learn domains from a much larger corpus like Wikipedia and then apply the extracted domains to machine translation data. However, this presents further challenges, as Wikipedia's domains are not representative of newswire machine translation datasets; a flexible hierarchical topic model (Teh et al, 2006) would better distinguish useful domains from extraneous ones.

5.6 Summary

Topic models generate great interest, but their use in “real world” applications still lags; this is particularly true for multilingual topic models. As topic models become more integrated in commonplace applications, their adoption, understanding, and robustness will improve.

This chapter contributes to the deeper integration of topic models into critical applications by presenting a new multilingual topic model, ptLDA, comparing it with other multilingual topic models on a machine translation task, and showing that these topic models improve machine translation. Further improvement is possible by incorporating topic models deeper in the decoding process and adding domain knowledge to the language model (Botha et al, 2012).

In this chapter, ptLDA models both source and target data to induce domains from using existing knowledge resources like bilingual dictionaries or data alignments. This can be further extended to an interactive setting and obtain there prior knowledge (concepts knowledge) from users as in Chapter 4. These users are not limited to bilingual speaker only, and the prior knowledge from monolingual users can be combined with existing knowledge to aid SMT.

So far, we assume that we can get the prior knowledge either from users or from existing knowledge resources and then apply the prior knowledge to probabilistic models. When these prior knowledge sources are not available, we can automatically learn the prior tree structure by Bayesian hieararchical clustering techniques as introduced in next Chapter (Chapter 7).

Chapter 6

Regularized Anchor Methods for Topic Models to Encode Priors

All the topics models that we have discussed so far are formulated as latent variable models, including the vanilla topic models and tree-based topic models in Chapter 2, efficient tree-based topic models (Chapter 3), interactive topic modeling (Chapter 4), and polylingual tree-based topic models (Chapter 5). Posterior inference discovers the hidden variables that best explain a dataset. Typical solutions use MCMC (Griffiths and Steyvers, 2004, Chapter 2, 3) or variational EM (Blei et al, 2003b), which can be viewed as local optimization: searching for the latent variables that maximize the data posterior distributions.

An exciting vein of new research provides provable polynomial-time alternatives. These approaches provide solutions to hidden Markov models (Anandkumar et al, 2012c), mixture models (Kannan et al, 2005), latent variable grammars (Cohen et al, 2013), and topic models (Arora et al, 2012b; Ding et al, 2013b). The key insight is not to directly optimize observation likelihood but to instead discover latent variables that can reconstruct statistics of the assumed generative model. Unlike search-based methods, which can be caught in local minima, these techniques are often guaranteed to find global optima.

Despite their advantages, these techniques are not a panacea. They do not accommodate the rich priors that modelers have come to expect. Priors can improve performance (Wallach et al, 2009), provide domain adaptation (Daumé III, 2007; Finkel and Manning, 2009), and guide models to reflect users' needs (Chapter 4).

Another shortcoming is that these models have not been scrutinized using standard NLP evaluations. Because these approaches emerged from the theory community, these evaluations, when present, typically use training reconstruction.

In this chapter, we regularize the **anchor** method (Arora et al, 2012b) to trade-off the reconstruction fidelity with penalty terms that mimic Gaussian and Dirichlet priors. We show that our regularized models can generalize to previously unseen data—as measured by held-out likelihood (Blei et al, 2003b)—and are more interpretable (Chang et al, 2009; Newman et al, 2010), which is the same as the goal of adding prior knowledge to topics models. We also show that our extension to the **anchor** method enables new applications: for example, using informed priors to discover concepts of interest.

6.1 Anchor Words: Scalable Topic Models

Arora et al (2012b) propose a new inference scheme for topic models by assuming that each topic has a unique "anchor" word (thus, we call this approach **anchor**). This approach is fast and obtains a global optimum with theoretical guarantees; because it only uses word co-occurrence information, it can scale to much larger datasets than MCMC or EM alternatives.

In this section, we briefly review the **anchor** method, which includes selecting anchor words and recovering topics. Once we have established the **anchor** objective

K	number of topics
V	vocabulary size
M	document frequency: minimum documents an anchor word candidate must appear in
$\boldsymbol{Q}$	word co-occurrence matrix $Q_{i,j} = p(w_1 = i, w_2 = j)$
$\bar{\boldsymbol{Q}}$	conditional distribution of Q $\bar{Q}_{i,j} = p(w_1 = j \,\vert\, w_2 = i)$
$\bar{\boldsymbol{Q}}_{i,\cdot}$	row i of $\bar{Q}$
$\boldsymbol{A}$	topic matrix, of size $V \times K$ $A_{j,k} = p(w = j \,\vert\, z = k)$
$\boldsymbol{C}$	anchor coefficient of size $K \times V$ $C_{j,k} = p(z = k \,\vert\, w = j)$
$\mathcal{S}$	set of anchor word indexes $\{s_1, \dots s_K\}$
λ	regularization weight

Table 6.1: Notation used. Matrices are in bold $(\boldsymbol{Q}, \boldsymbol{C})$, sets are in script $\mathcal{S}$. Note that we change the topic matrix notation from π to A in this chapter.

function, in the next section we regularize the objective function.

Rethinking Data: Word Co-occurrence Inference in topic models can be viewed as a black box: given a set of documents, discover the topics that best explain the data. The difference between **anchor** and conventional inference is that while conventional methods take a collection of documents as input, **anchor** takes *word co-occurrence* statistics. Given a vocabulary of size V, we represent this joint distribution as $\boldsymbol{Q}_{i,j} = p(w_1 = i, w_2 = j)$, where each cell represents the probability of words appearing together in a document.

Like other topic modeling algorithms, the output of the **anchor** method is the topic word distributions $\boldsymbol{A}$ with size $V * K$, where K is the total number of topics desired, a parameter of the algorithm. The k^{th} column of A will be the topic distribution over all words for topic k, and $\boldsymbol{A}_{w,k}$ is the probability of observing type w given topic k.

Anchors: Topic Representatives The **anchor** method (Arora et al, 2012a) is based on the separability assumption (Donoho and Stodden, 2003), which assumes that each topic contains at least one namesake "anchor word" that has non-zero probability only in that topic. Intuitively, this means that each topic has unique, specific word that, when used, identifies that topic. For example, while "run", "base", "fly", and "shortstop" are associated with a topic about baseball, only "shortstop" is unambiguous, so it could serve as this topic's anchor word.

Thus the first step of the **anchor** method is to select the anchor words. Given infinite data, the convex hull of the rows in $\bar{\boldsymbol{Q}}$ will be a simplex where the vertices of this simplex correspond to the anchor words (Arora et al, 2012a). Given V data points in $\bar{\boldsymbol{Q}}$ which define a simplex, the goal is to find an approximation to the vertices of this simplex. Arora et al (2012a) propose to iteratively find the furthest point from the subspace spanned by the anchor words so far to approximate the anchor words. This algorithm avoids the linear programming altogether and efficiently finds the approximation of the anchor words. More details can be found in Arora et al (2012a).

The next step of the **anchor** method is to recover the topic matrix A given the anchor words. Let's assume that we knew what the anchor words were: a set $\mathcal{S}$ that indexes rows in $\boldsymbol{Q}$. Now consider the **conditional distribution** of word i, the probability of the rest of the vocabulary given an observation of word i; we represent this as $\bar{\boldsymbol{Q}}_{i,\cdot}$, as we can construct this by normalizing the rows of $\boldsymbol{Q}$. For an anchor word $s_a \in \mathcal{S}$, this will look like a topic; $\bar{\boldsymbol{Q}}_{\text{"shortstop"},\cdot}$ will have high probability for words associated with baseball.

The key insight of the **anchor** algorithm is that the conditional distribution of polysemous non-anchor words can be reconstructed as a linear combination of the conditional distributions of anchor words. For example, $\bar{\boldsymbol{Q}}_{\text{``fly''},\cdot}$ could be reconstructed by combining the anchor words "insecta", "boeing", and "shortshop". We represent the coefficients of this reconstruction as a matrix $\boldsymbol{C}$, where $C_{i,k} = p(z = k \,|\, w = i)$. Thus, for any word i,

$$\bar{\boldsymbol{Q}}_{i,\cdot} \approx \sum_{s_k \in \mathcal{S}} C_{i,k} \bar{\boldsymbol{Q}}_{s_k,\cdot}. \tag{6.1}$$

The coefficient matrix is **not** the usual output of a topic modeling algorithm (For example, Algorithm 1 and Algorithm 2 in Chapter 3). The usual output is the probability of a word *given a topic*. The coefficient matrix C is the probability of a topic *given a word*. We use Bayes rule to recover the topic distribution

$$p(w = i | z = k) \equiv A_{i,k} \propto p(z = k | w = i) p(w = i) = C_{i,k} \sum_j \bar{Q}_{i,j} \tag{6.2}$$

where $p(w)$ is the normalizer of $\boldsymbol{Q}$ to obtain $\bar{\boldsymbol{Q}}_{w,\cdot}$.

As a result, the problem in second step is changed from recovering topic matrix A to finding the coefficients C that best reconstruct the data $\bar{Q}$ (Equation 6.1). Arora et al (2012a) chose the C that minimizes the KL divergence between $\bar{Q}_{i,\cdot}$ and the reconstruction based on the anchor word's conditional word vectors $\sum_{s_k \in \mathcal{S}} C_{i,k} \bar{\boldsymbol{Q}}_{s_k,\cdot}$,

$$\boldsymbol{C}_{i,\cdot} = \text{argmin}_{\boldsymbol{C}_{i,\cdot}} D_{\text{KL}}\Big(\bar{\boldsymbol{Q}}_{i,\cdot} \,||\, \sum_{s_k \in S} C_{i,k} \bar{\boldsymbol{Q}}_{s_k,\cdot}\Big). \tag{6.3}$$

Recovering the coefficients C is very efficient, as it only depends on the size of the vocabulary once the co-occurrence statistics $\boldsymbol{Q}$ are obtained, then we can easily recover the topic matrix $\boldsymbol{A}$ using Equation 6.2.

Problems While the **anchor** algorithm is very efficient, it does not support rich priors for topic models, while MCMC (Griffiths and Steyvers, 2004) and variational EM (Blei et al, 2003b) methods can. This prevents models from using priors to guide the models to discover particular themes (Zhai et al, 2012), or to encourage sparsity in the models (Yao et al, 2009).

In the rest of this chapter, we correct this lacuna by adding regularization to find better coefficient matrix C in the **anchor** algorithm. This is inspired by Bayesian priors. We keep the first step—the anchor selection subroutine (Arora et al, 2012a)—unchanged. The difference in our approach is in how we discover the anchor coefficients C.

Related Work Besides traditional topic models such as Latent Dirichlet Allocation (Blei et al, 2003b; Griffiths and Steyvers, 2004), different spectral algorithms for topic models have also been developed recently.

Anandkumar et al (2012a) develop a novel spectral algorithm for topic models based on moments, and Anandkumar et al (2012b) further improve the algorithm by doing two singular decompositions based on second and third moments. This algorithm is expensive to find the higher order of moments and compute singular decomposition for large matrices. However, the anchor method by Arora et al (2012a) only takes the co-occurrence matrix (second moment) as the input, and avoids singular decomposition by doing a reconstruction optimization.

Ding et al (2013b) discover topics also based on the separability assumption as Arora et al (2012a). Instead of taking the co-occurrence matrix (size $V \times V$) as inputs, they associate each word with a D-dimensional vector, where D is the number of documents, and use the matrix X (size $V \times D$)—made by these D-dimensional vectors for V words—as inputs. Then they use a clustering algorithm to figure out K groups of anchor words (named as novel words in Ding et al (2013b)), and recover the topics by reconstructing word vectors using the vectors of anchor words. One issue with their algorithm is D can be very large in practice, which may cause computation problems, thus Ding et al (2014) further scale it up. Another problem is, like the anchor method, Ding et al (2013b) do not consider priors in optimization, and Ding et al (2013a) only consider the L_∞ regularization to encourage the sparsity. We discuss different regularization in this chapter for the anchor methods, which can also be used in Ding et al (2013b).

6.2 Adding Regularization

In this section, we add regularizers to the **anchor** objective (Equation 6.3). We briefly review regularizers and then add two regularizers, inspired by Gaussian (L_2, Section 6.2.1) and Dirichlet priors (Beta, Section 6.2.2), to the **anchor** objective function (Equation 6.3).

Regularization terms are ubiquitous. They typically appear as an additional term in an optimization problem. Instead of optimizing a function just of the data x and parameters β, $f(x, \beta)$, one optimizes an objective function that includes a regularizer that is only a function of parameters: $f(w, \beta) + r(\beta)$. Regularizers are critical in staid methods like linear regression (Ng, 2004), in workhorse methods such as maximum entropy modeling (Dudík et al, 2004), and also in emerging fields such as deep learning (Wager et al, 2013).

In addition to being useful, regularization terms are appealing theoretically because they often correspond to probabilistic interpretations of parameters. For example, if we are seeking the MLE of a probabilistic model parameterized by β, $p(x|\beta)$, adding a regularization term $r(\beta) = \sum_{i=1}^{L} \beta_i^2$ corresponds to adding a Gaussian prior

$$f(\beta_i) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{\beta_i^2}{2\sigma^2}\right\} \tag{6.4}$$

and maximizing log probability of the posterior (ignoring constant terms) (Rennie, 2003).

6.2.1 L_2 Regularization

The simplest form of regularization we can add is L_2 regularization. This is similar to assuming that probability of a word given a topic comes from a Gaussian distribution. While the distribution over topics is typically Dirichlet, Dirichlet distributions have been replaced by logistic normals in topic modeling applications (Blei and Lafferty, 2005) and for probabilistic grammars of language (Cohen and Smith, 2009).

Augmenting the **anchor** objective with an L_2 penalty yields

$$\boldsymbol{C}_{i,\cdot} = \text{argmin}_{C_{i,\cdot}} D_{\text{KL}}\left(\bar{\boldsymbol{Q}}_{i,\cdot} \,||\, \sum_{s_k \in S} C_{i,k} \bar{\boldsymbol{Q}}_{s_k,\cdot}\right) + \lambda \|\boldsymbol{C}_{i,\cdot} \; - \mu_{i,\cdot}\|_2^2, \tag{6.5}$$

where regularization weight λ balances the importance of a high-fidelity reconstruction against the regularization, which encourages the anchor coefficients to be close to the vector μ. When the mean vector μ is zero, this encourages the topic coefficients to be zero. In Section 6.3.3, we use a non-zero mean μ to encode an informed prior to encourage topics to discover specific concepts.

6.2.2 Beta Regularization

The more common prior for topic models is a Dirichlet prior (Minka, 2000). However, we cannot apply this directly because the optimization is done on a row-by-row basis of the anchor coefficient matrix $\boldsymbol{C}$, optimizing $\boldsymbol{C}$ for a fixed word w for and all topics. If we want to model the probability of a word, it must be the probability of word w in a topic versus all other words.

Modeling this dichotomy (one versus all others in a topic) is possible. The constructive definition of the Dirichlet distribution (Sethuraman, 1994) states that if one has a V-dimensional multinomial $\theta \sim \text{Dir}(\alpha_1 \ldots \alpha_V)$, then the marginal distribution of θ_w follows $\theta_w \sim \text{Beta}(\alpha_w, \sum_{i \neq w} \alpha_i)$. This is the tool we need to consider the distribution of a single word's probability.

This requires including the topic matrix as part of the objective function. The topic matrix is a linear transformation of the coefficient matrix (Equation 6.2). The objective for beta regularization becomes

$$\boldsymbol{C}_{i,\cdot} = \mathrm{argmin}_{C_{i,\cdot}} D_{\mathrm{KL}}\Big(\bar{\boldsymbol{Q}}_{i,\cdot} \,||\, \sum_{s_k \in S} C_{i,k} \bar{\boldsymbol{Q}}_{s_k,\cdot}\Big) - \lambda \sum_{s_k \in S} \log\left(\mathrm{Beta}(A_{i,k}; a, b)\right), \tag{6.6}$$

where λ again balances reconstruction against the regularization. To ensure the tractability of this algorithm, we enforce a convex regularization function, which requires that $a > 1$ and $b > 1$. If we enforce a uniform prior—$\mathbb{E}_{\mathrm{Beta}(a,b)}\left[A_{i,k}\right] = \frac{1}{V}$—and that the *mode* of the distribution is also $\frac{1}{V}$,[1] this gives us the following parametric form for a and b:

$$a = \frac{x}{V} + 1, \text{ and } b = \frac{(V-1)x}{V} + 1 \tag{6.7}$$

for real x greater than zero.

6.2.3 Initialization and Convergence

Equation 6.5 and Equation 6.6 are optimized using L-BFGS gradient optimization (Galassi et al, 2003). We initialize $\boldsymbol{C}$ randomly from $\mathrm{Dir}(\alpha)$ with $\alpha = \frac{60}{V}$ (Wallach et al, 2009). We update $\boldsymbol{C}$ after optimizing all V rows. The newly updated $\boldsymbol{C}$ replaces the old topic coefficients. We track how much the topic coefficients $\boldsymbol{C}$ change between two consecutive iterations i and $i+1$ and represent it as $\Delta\boldsymbol{C} \equiv \|\boldsymbol{C}^{i+1} - \boldsymbol{C}^{i}\|_2$. We stop optimization when $\Delta\boldsymbol{C} \leq \delta$. When $\delta = 0.1$, the L_2 and unregularized anchor algorithm converges after a single iteration, while beta regularization typically converges after fewer than ten iterations (Figure 6.4).

[1] For $a, b < 1$, the expected value is still the uniform distribution but the mode lies at the boundaries of the simplex. This corresponds to a sparse Dirichlet distribution, which our optimization cannot at present model.

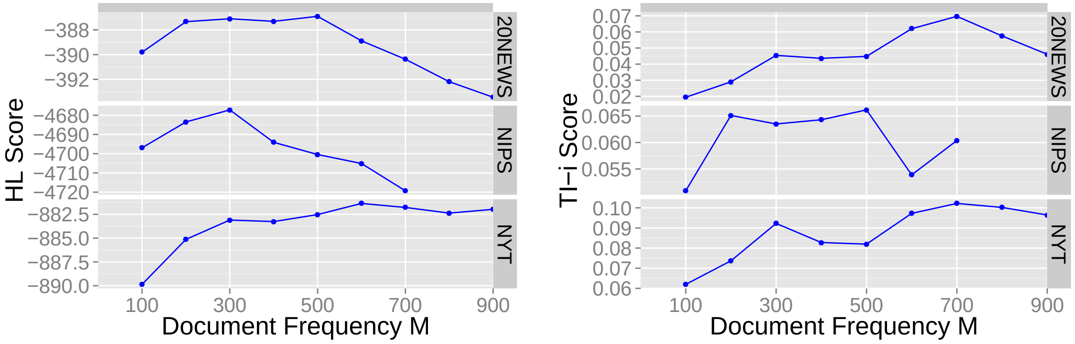


Figure 6.1: Grid search for document frequency M for our datasets with 20 topics (other configurations not shown) on development data. The performance on both HL and TI score indicate that the unregularized **anchor** algorithm is very sensitive to M. The M selected here is applied to subsequent models. For both HL and TI, higher is better.

Corpus	Train	Dev	Test	Vocab
NIPS	1231	247	262	12182
20NEWS	11243	3760	3726	81604
NYT	9255	2012	1959	34940

Table 6.2: The number of documents in the train, development, and test folds in our three datasets.

6.3 Regularization Improves Topic Models

In this section, we measure the performance of our proposed regularized anchor word algorithms. We will refer to specific algorithms in bold. For example, the original anchor algorithm is **anchor**. Our L_2 regularized variant is **anchor-L_2**, and our beta regularized variant is **anchor-beta**. To provide conventional baselines, we also compare our methods against topic models from variational inference (Blei et al, 2003b, **variational)** and MCMC (Griffiths and Steyvers, 2004; McCallum, 2002, **mcmc)**.

We apply these inference strategies on three diverse corpora: scientific articles from the Neural Information Processing Society (NIPS),[2] Internet newsgroups postings (20NEWS),[3] and New York Times editorials (Sandhaus, 2008, NYT). Statistics for the datasets are summarized in Table 6.2. We split each dataset into a training fold (70%), development fold (15%), and a test fold (15%): the training data are used to fit models; the development set are used to select parameters (anchor threshold M, document prior α, regularization weight λ); and final results are reported on the test fold.

We use two evaluation measures, held-out likelihood (Blei et al, 2003b, **hl)** and topic interpretability (Chang et al, 2009; Newman et al, 2010, **ti)**. Held-out likelihood measures how well the model can reconstruct held-out documents that the model has never seen before. This is the typical evaluation for probabilistic models. Topic interpretability is a more recent metric to capture how useful the topics can be to human users attempting to make sense of a large datasets.

Held-out likelihood cannot be computed with existing **anchor** algorithms, so we use the topic distributions learned from **anchor** as input to a reference variational inference implementation (Blei et al, 2003b) to compute **hl**. This requires an additional parameter, the Dirichlet prior α for the per-document distribution over topics. We select α using grid search on the development set.

To compute **ti** and evaluate topic coherence, we use normalized pairwise mutual information (NPMI) (Lau et al, 2014) over topics' twenty most probable words. Topic coherence is computed against the NPMI of a reference corpus. For

[2] `http://cs.nyu.edu/~roweis/data.html`

[3] `http://qwone.com/~jason/20Newsgroups/`

coherence evaluations, we use both intrinsic and extrinsic text collections to compute NPMI. Intrinsic coherence (TI-i) is computed on training and development data at development time and on training and test data at test time. Extrinsic coherence (TI-e) is computed from English Wikipedia articles, with disjoint halves (1.1 million pages each) for distinct development and testing TI-e evaluation.

6.3.1 Grid Search for Parameters on Development Set

Anchor Threshold A good anchor word must have a unique, specific context but also explain other words well. A word that appears only once will have a very specific cooccurence pattern but will explain other words' coocurrence poorly because the observations are so sparse. As discussed in Section 6.1, the **anchor** method uses document frequency M as a threshold to only consider words with robust counts.

Because all regularizations benefit equally from higher-quality anchor words, we use cross-validation to select the document frequency cutoff M using the unregularized **anchor** algorithm. Figure 6.1 shows the performance of **anchor** with different M on our three datasets with 20 topics for our two measures HL and TI-i.

Regularization Weight Once we select a cutoff M for each combination of dataset, number of topics K and a evaluation measure, we select a regularization weight λ on the development set. Figure 6.2 shows that **beta** regularization framework improves topic interpretability TI-i on all datasets and improved the held-out likelihood HL on 20NEWS. The L_2 regularization also improves held-out likelihood HL for the 20NEWS corpus (Figure 6.2).

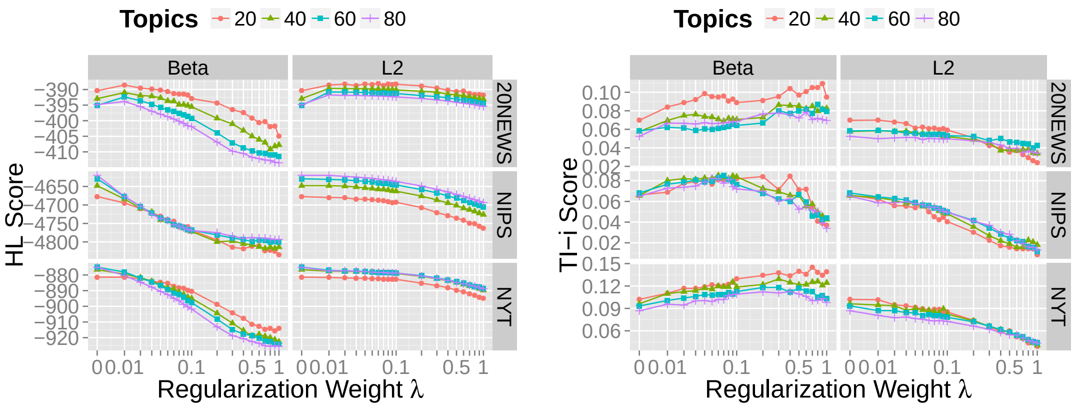


Figure 6.2: Selection of λ based on HL and TI scores on the development set. For both HL and TI, higher is better. The value of $\lambda = 0$ is equivalent to the original **anchor** algorithm; regularized versions find better solutions as the regularization weight λ becomes non-zero.

In the interests of brevity, we do not show the figures for selecting M and λ using TI-e, which is similar to TI-i: **anchor-beta** improves TI-e score on all datasets, **anchor-L_2** improves TI-e on 20NEWS and NIPS with 20 topics and NYT with 40 topics.

6.3.2 Evaluating Regularization

With document frequency M and regularization weight λ selected from the development set, we compare the performance of those models on the test set. We also compare with standard implementations of Latent Dirichlet Allocation: Blei's LDAC (**variational**) and Mallet (**mcmc**). We run 100 iterations for LDAC and 5000 iterations for Mallet.

Each result is averaged over three random runs and appears in Figure 6.3. The highly-tuned, widely-used implementations uniformly have better held-out likelihood than **anchor**-based methods, but the much faster **anchor** methods are often

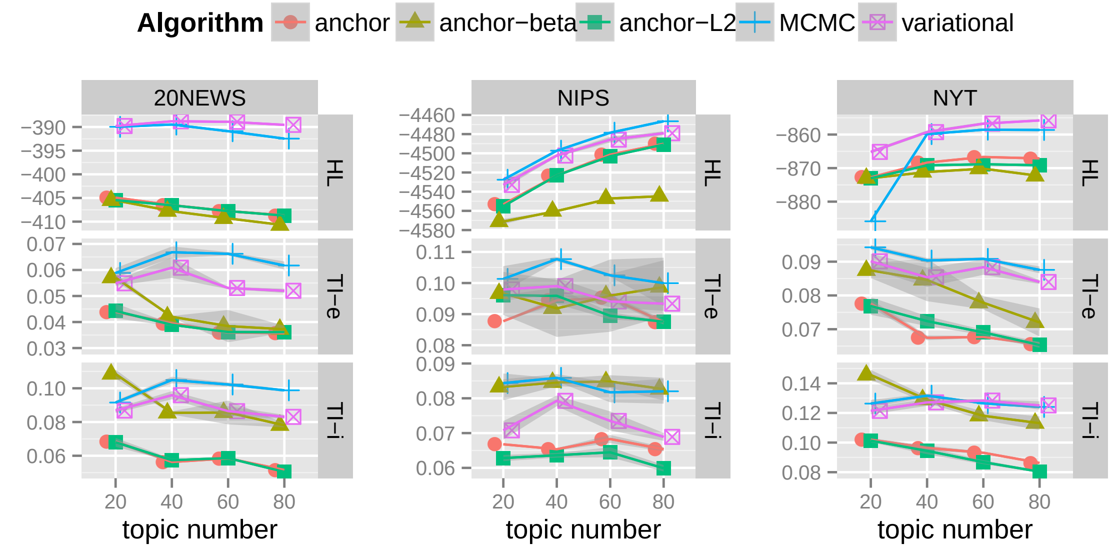


Figure 6.3: Comparing **anchor-beta** and **anchor-L_2** against the original **anchor** and the traditional **variational** and **MCMC** on HL score and TI score. For both HL and TI, higher is better. **variational** and **mcmc** provide the best held-out generalization. **anchor-beta** sometimes gives the best TI score and is consistently better than **anchor**. The specialized vocabulary of NIPS causes high variance for the extrinsic interpretability evaluation (TI-e).

comparable. Within **anchor**-based methods, L_2-regularization offers comparable held-out likelihood as unregularized **anchor**, while **anchor-beta** often has better interpretability. Because of the mismatch between the specialized vocabulary of NIPS and the general-purpose language of Wikipedia, TI-e has a high variance.

6.3.3 Informed Regularization

A frequent use of priors is to add information to a model. This is not possible with the existing **anchor** method. An informed prior for topic models seeds a topic with words that describe a topic of interest. In a topic model, these seeds will serve as a "magnet", attracting similar words to the topic (Zhai et al, 2012).

We can achieve a similar goal with **anchor-L_2**. Instead of encouraging anchor coefficients to be zero in Equation 6.5, we can instead encourage word probabilities to be close to an arbitrary mean $\mu_{i,k}$. This vector can reflect expert knowledge.

One example of a source of expert knowledge is Linguistic Inquiry and Word Count (Pennebaker and Francis, 1999, LIWC), a dictionary of keywords related to a set of psychological concepts such as positive emotions, negative emotions, and death. For example, it associates "excessive, estate, money, cheap, expensive, living, profit, live, rich, income, poor, etc." for the concept materialism.

We associate each anchor word with its closest LIWC category based on the cooccurrence matrix $\boldsymbol{Q}$. This is computed by greedily finding the anchor word that has the highest cooccurrence score for any LIWC category: we define the score of a category to anchor word w_{s_k} as $\sum_i Q_{s_k,i}$, where i ranges over words in this category; we compute the scores of all categories to all anchor words; then we find the highest score and assign the category to that anchor word; we greedily repeat this process until all anchor words have a category.

Given these associations, we create a goal mean $\mu_{i,k}$. If there are L_i anchor words associated with LIWC word i, $\mu_{i,k} = \frac{1}{L_i}$ if this keyword i is associated with anchor word w_{s_k} and zero otherwise.

We apply **anchor-L_2** with informed priors on NYT with twenty topics and compared the topics against the original topics from **anchor**. Table 6.3 shows that the topic with anchor word "soviet", when combined with LIWC, draws in the new words "bush" and "nuclear"; reflecting the threats of force by Bush during the cold war. For the topic with topic word "arms", when associated with the LIWC category with the terms "agree" and "agreement", draws in "clinton", who represented a

Topic	Shared Words	Original (Top, green) vs. Informed L_2 (Bottom, orange)
soviet	american make president **soviet** union **war** years	gorbachev moscow russian force economic world europe political communist lead reform germany country **military** state **service** washington bush **army** unite **chief troops officer** nuclear time week
district	**assembly** board city **county district member** state york	representative manhattan brooklyn queens election bronx council island local incumbent housing municipal **people party group social** republican year make years **friend** vote **compromise** million
peace	american force government israel **peace** political president state unite washington	war military country minister leaders nation world palestinian israeli election **offer justice aid deserve** make bush years **fair** clinton **hand**
arms	**arms** bush congress force iraq make north nuclear president state washington weapon	administration treaty missile defense war military korea reagan **agree agreement** american **accept** unite **share** clinton years
trade	**administration** america american country **economic** government make president state **trade** unite washington	world market japan foreign china policy price political **business economy** congress year years clinton bush **buy**

Table 6.3: Examples of topic comparison between **anchor** and informed **anchor-L_2**. A topic is labeled with the anchor word for that topic. The **bold** words are the informed prior from LIWC. With an informed prior, relevant words appear in the top words of a topic; this also draws in other related terms (red).

more conciliatory foreign policy compared to his republican predecessors.

6.4 Discussion

Having shown that regularization can improve the **anchor** topic modeling algorithm, in this section we discuss *why* these regularizations can improve the model and the implications for practitioners.

Efficiency Efficiency is a function of the number of iterations and the cost of each iteration. Both **anchor** and **anchor-L_2** require a single iteration, although the

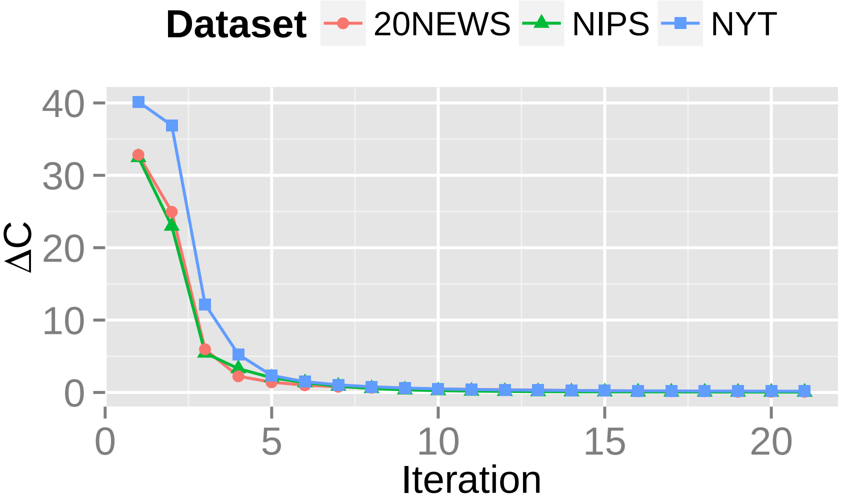


Figure 6.4: Convergence of anchor coefficient C for **anchor-beta**. ΔC is the difference of current C from the C at the previous iteration. C is converged within ten iterations for all three datasets.

latter's iteration is slightly more expensive. For **beta**, as described in Section 6.2.2, we update anchor coefficients C row by row, and then repeat the process over several iterations until it converges. However, it often converges within ten iterations (Figure 6.4) on all three datasets: this requires much fewer iterations than MCMC or variational inference, and the iterations are less expensive. In addition, since we optimize each row $C_{i,\cdot}$ independently, the algorithm can be easily parallelized.

Sensitivity to Document Frequency While the original **anchor** is sensitive to the document frequency M (Figure 6.1), adding regularization makes this less critical. Both **anchor-L_2** and **anchor-beta** are less sensitive to M than **anchor**.

To highlight this, we compare the topics of **anchor** and **anchor-beta** when $M = 100$. As Table 6.4 shows, the words "article", "write", "don" and "doe" appear in most of **anchor**'s topics. While **anchor-L_2** also has some bad topics, it

Topic	anchor	anchor-beta
frequently	article write don **doe** make time people good file question	article write don **doe** make people time good email file
debate	write article people make don **doe** god key government time	people make god article write don **doe** key point government
wings	game team write wings article win red play hockey year	game team wings win red hockey play season player fan
stats	player team write game article stats year good play **doe**	stats player season league baseball fan team individual playoff nhl
compile	program file write email **doe** windows call problem run don	compile program code file ftp advance package error windows sun

Table 6.4: Topics from **anchor** and **anchor-beta** with $M = 100$ on 20NEWS with 20 topics. Each topic is identified with its associated anchor word. When $M = 100$, the topics of **anchor** suffer: the four colored words appear in almost every topic. **anchor-beta**, in contrast, is less sensitive to suboptimal M.

still can find reasonable topics, demonstrating **anchor-beta**'s greater robustness to suboptimal M.

L_2 (Sometimes) Improves Generalization As Figure 6.2 shows, **anchor-L_2** sometimes improves held-out development likelihood for the smaller 20NEWS corpus. However, the λ selected on development data does not always improve test set performance. In Figure 6.3, **anchor-beta** closely tracks **anchor**. Thus, L_2 regularization does not hurt generalization while imparting expressiveness and robustness to parameter settings.

Beta Improves Interpretability Figure 6.3 shows that **anchor-beta** improves topic interpretability (TI) compared to unregularized anchor methods. In this section, we try to understand why.

We first compare the topics from the original **anchor** against **anchor-beta** to analyze the topics qualitatively. Table 6.5 shows that **beta** regularization promotes

Topic	Shared Words	**anchor** (Top, green) vs. **anchor-beta** (Bottom, orange)
computer	computer means science screen	system phone university problem doe work windows internet software chip mac set fax technology information data
		quote mhz pro processor ship remote print devices complex cpu electrical transfer ray engineering serial reduce
power	power play period supply ground light battery engine	car good make high problem work back turn control current small time
		circuit oil wire unit water heat hot ranger input total joe plug
god	god jesus christian bible faith church life christ belief religion hell word lord truth love	people make things true doe
		sin christianity atheist peace heaven
game	game team player play win fan hockey season baseball red wings score division league goal leaf cup toronto	run good
		playoff trade
drive	drive disk hard scsi controller card floppy ide mac bus speed monitor switch apple cable internal port meg	problem work
		ram pin

Table 6.5: Comparing topics—labeled by their anchor word—from **anchor** and **anchor-beta**. With beta regularization, relevant words are promoted, while more general words are suppressed, improving topic coherence.

rarer words within a topic and demotes common words. For example, in the topic about hockey with the anchor word game, "run" and "good"—ambiguous, polysemous words—in the unregularized topic are replaced by "playoff" and "trade" in the regularized topic. These words are less ambiguous and more likely to make sense to a consumer of topic models.

Figure 6.5 shows why this happens. Compared to the unregularized topics from **anchor**, the beta regularized topic steals from the rich and creates a more uniform distribution. Thus, highly frequent words do not as easily climb to the top of the distribution, and the topics reflect topical, relevant words rather than globally frequent terms.

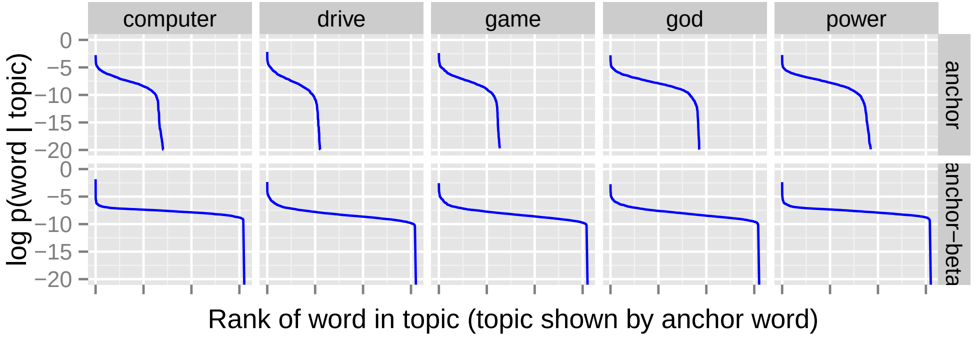


Figure 6.5: How beta regularization influences the topic distribution. Each topic is identified with its associated anchor word. Compared to the unregularized **anchor** method, **anchor-beta** steals probability mass from the "rich" and prefers a smoother distribution of probability mass. These words often tend to be unimportant, polysemous words common across topics.

6.5 Summary

This chapter introduces two different regularizations that offer users more interpretable models and the ability to inject prior knowledge without sacrificing the speed and generalizability of the underlying approach. However, one sacrifice that this approach does make is the beautiful theoretical guarantees of previous work. An important piece of future work is a theoretical understanding of generalizability in extensible, regularized models.

Incorporating other regularizations could further improve performance or unlock new applications. Our regularizations do not explicitly encourage sparsity; applying other regularizations such as L_1 could encourage true sparsity (Tibshirani, 1994).

Another possible direction is to consider word correlations (Chapter 2) as priors in the regularization of the **anchor** methods. For example, to encode a positive correlation between word w_i and word w_j, we can find the optimal $\boldsymbol{C}_{i,\cdot}$ and $\boldsymbol{C}_{j,\cdot}$ together while encouraging $\boldsymbol{C}_{i,\cdot}$ and $\boldsymbol{C}_{j,\cdot}$ to be similar to each other; similarly for negative correlation between w_i and w_j, we can regularize the objective to encourage $\boldsymbol{C}_{i,\cdot}$ and $\boldsymbol{C}_{j,\cdot}$ to be different from each other. This would be an alternative way for replacing tree-based topic models (Chapter 2), and it will be much faster than sampling-based inference (Chapter 2), even the efficient factorized inference (Chapter 3). Once it is done, this new spectral models with considering word correlations can also be applied to the interactive setting (Chapter 4) or real applications (Chapter 5).

In general, these regularizations could improve spectral algorithm for latent variables models, improving the performance for other NLP tasks such as latent variable PCFGs (Cohen et al, 2013) and HMMs (Anandkumar et al, 2012c), combining the flexibility and robustness offered by priors with the speed and accuracy of new, scalable algorithms.

Chapter 7

Automatically Building Hierarchical Prior Trees from Data

In the previous chapters, the prior tree treats words as leaves and posits a hierarchical structure to represent the relationship between words. As obtaining this prior knowledge from users is expensive and time-consuming, this necessitates automatic techniques to extract prior knowledge from corpora.

Hierarchical clustering is one possible way to get this tree structure automatically. It treats observations as leaf nodes, and builds up the hierarchy according to the similarity of different nodes. Nodes sharing the same parent node are closest to each other, and the similarity of nodes in the same subtree is reflected in the hierarchy of this subtree. This hierarchical tree structure matches what we want for the prior tree, and it is built automatically.

Traditional hierarchical clustering techniques (Duda and Hart, 1973) build up the hierarchical trees on an agglomerative bottom-up way based on various distance measures among nodes, for example, the Brown clustering (Brown et al, 1992). However, there are some limitations for these traditional techniques: it is hard to figure out the correct number of clusters; it is difficult to choose a distance measure; and there is no probabilistic model to measure whether the clusters are good not. As a result, we are more interested in Bayesian hierarchical clustering methods (Heller and Ghahramani, 2005; Knowles and Ghahramani, 2011), which are can be further integrated into tree-based topic models 2.

In this chapter, we present a Bayesian hierarchical clustering technique with the Beta coalescent, which provides an alternative way to build the prior tree automatically. Because it is expensive to build up multi-branch trees automatically, we develop new sampling schemes using sequential Monte Carlo and Dirichlet process mixture models, which render the inference practical and efficient.

7.1 Bayesian Clustering Approaches

Recent hierarchical clustering techniques have been incorporated inside statistical models; this requires formulating clustering as a statistical—often Bayesian—problem. Heller and Ghahramani (2005) build binary trees based on the marginal likelihoods, extended by Blundell et al (2010) to trees with arbitrary branching structure. Adams et al (2010) propose a tree-structured stick-breaking process to generate trees with unbounded width and depth, which supports data observations at leaves *and* internal nodes.[1] However, these models do not distinguish edge lengths, an important property in distinguishing how "tight" the clustering is at particular nodes.

Hierarchical models can be divided into complementary "fragmentation" and "coagulation" frameworks (Pitman, 1999). Both produce hierarchical partitions of a dataset. Fragmentation models start with a single partition and divide it into ever more specific partitions until only singleton partitions remain. Coagulation

[1] This is appropriate where the entirety of a population is known—both ancestors and descendants. We focus on the case where only the descendants are known. For a concrete example, see Section 7.5.2.

frameworks repeatedly merge singleton partitions until only one partition remains. Pitman-Yor diffusion trees (Knowles and Ghahramani, 2011), a generalization of Dirichlet diffusion trees (Neal, 2003a), are an example of a bushy fragmentation model, and they model edge lengths and build non-binary trees.

Instead, our focus is on bottom-up coalescent models (Berestycki, 2009), one of the coagulation models and complementary to diffusion trees, which can also discover hierarchies and edge lengths. In this model, n nodes are observed (we use both *observed* to emphasize that nodes are known and *leaves* to emphasize topology). These observed nodes are generated through some unknown tree with latent edges and unobserved internal nodes. Each node (both observed and latent) has a single parent. The convention in such models is to assume our observed nodes come at time $t = 0$, and at time $-\infty$ all nodes share a common ur-parent through some sequence of intermediate parents.

Consider a set of n individuals observed at the present (time $t = 0$). All individuals start in one of n singleton sets. After time t_i, a set of these nodes coalesce into a new node. Once a set merges, their parent replaces the original nodes. This is called a **coalescent event**. This process repeats until there is only one node left, and a complete tree structure π (Figure 7.1) is obtained.

Different coalescents are defined by different probabilities of merging a set of nodes. This is called the coalescent **rate**, defined by a general family of coalescents: the lambda coalescent (Pitman, 1999; Sagitov, 1999). We represent the rate via the symbol λ_n^k, the rate at which k out of n nodes merge into a parent node. From a collection of n nodes, $k \leq n$ can coalesce at some coalescent event (k can be different for different coalescent events). The rate of a fraction γ of the nodes coalescing is given by $\gamma^{-2}\Lambda(d\gamma)$, where $\Lambda(d\gamma)$ is a finite measure on $[0, 1]$. So k nodes merge at rate

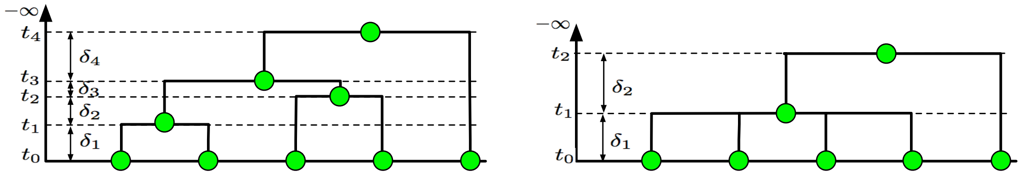
$$\lambda_n^k = \int_0^1 \gamma^{k-2}(1-\gamma)^{n-k}\Lambda(d\gamma) \qquad (2 \leq k \leq n). \tag{7.1}$$

Choosing different measures yields different coalescents. A degenerate Dirac delta measure at 0 results in Kingman's coalescent (Kingman, 1982), where λ_n^k is 1 when $k = 2$ and zero otherwise. Because this gives zero probability to non-binary coalescent events, this only creates binary trees.

Alternatively, using a beta distribution $\text{BETA}(2 - \alpha, \alpha)$ as the measure Λ yields the beta coalescent. When α is closer to 1, the tree is bushier; as α approaches 2, it becomes Kingman's coalescent. If we have n_{i-1} nodes at time t_{i-1} in a beta coalescent, the rate $\lambda_{n_{i-1}}^{k_i}$ for a children set of k_i nodes at time t_i and the total rate $\lambda_{n_{i-1}}$ of *any* children set merging—summing over all possible mergers—is

$$\lambda_{n_{i-1}}^{k_i} = \frac{\Gamma(k_i - \alpha)\Gamma(n_{i-1} - k_i + \alpha)}{\Gamma(2-\alpha)\Gamma(\alpha)\Gamma(n_{i-1})} \quad \text{and} \quad \lambda_{n_{i-1}} = \sum_{k_i=2}^{n_{i-1}} \tbinom{n_{i-1}}{k_i} \lambda_{n_{i-1}}^{k_i}. \tag{7.2}$$

Each coalescent event also has an edge length—**duration**—δ_i. The duration of an event comes from an exponential distribution, $\delta_i \sim \exp(\lambda_{n_{i-1}})$, and the parent node forms at time $t_i = t_{i-1} - \delta_i$. Shorter durations mean that the children more closely resemble their parent (the mathematical basis for similarity is specified by a transition kernel, Section 5.2).



(a) Kingman's coalescent (b) the beta coalescent

Figure 7.1: The beta coalescent can merge four similar nodes at once, while Kingman's coalescent only merges two each time.

Analogous to Kingman's coalescent, the prior probability of a complete tree π is the product of all of its constituent coalescent events $i = 1, \ldots m$, merging k_i children after duration δ_i,

$$p(\pi) = \prod_{i=1}^{m} \underbrace{p(k_i|n_{i-1})}_{\text{Merge } k_i \text{ nodes}} \cdot \underbrace{p(\delta_i|k_i, n_{i-1})}_{\text{After duration } \delta_i} = \prod_{i=1}^{m} \lambda_{n_{i-1}}^{k_i} \cdot \exp(-\lambda_{n_{i-1}}\delta_i). \tag{7.3}$$

7.2 Beta Coalescent Belief Propagation

The beta coalescent prior only depends on the topology of the tree. In real clustering applications, we also care about a node's *children* and *features*. In this section, we define the nodes and their features, and then review how we use message passing to compute the probabilities of trees.

An internal node ρ_i is defined as the merger of other nodes. The children **set** of node ρ_i, $\rho_{\vec{c}_i}$, coalesces into a new node $\rho_i \equiv \cup_{b \in \vec{c}_i} \rho_b$. This encodes the identity of the nodes that participate in specific coalescent events; Equation 7.3, in contrast, only considers the number of nodes involved in an event. In addition, each node is associated with a multidimensional feature vector y_i.

Two terms specify the relationship between nodes' features: an **initial** distribution $p_0(y_i)$ and a **transition kernel** $\kappa_{t_i t_b}(y_i, y_b)$. The initial distribution can be viewed as a prior or regularizer for feature representations. The transition kernel encourages a child's feature y_b (at time t_b) to resemble feature y_i (formed at t_i); shorter durations $t_b - t_i$ increase the resemblance.

Intuitively, the transition kernel can be thought as a similarity score; the more similar the features are, the more likely nodes are. For Brownian diffusion (discussed in Section 7.4.4), the transition kernel follows a Gaussian distribution centered at a feature. The covariance matrix Σ is decided by the mutation rate μ (Felsenstein, 1973; Teh et al, 2008), the probability of a mutation in an individual. Different kernels (e.g., multinomial, tree kernels) can be applied depending on modeling assumptions of the feature representations.

To compute the probability of the beta coalescent tree π and observed data $\mathbf{x}$, we generalize the belief propagation framework used by Teh et al (2008) for Kingman's coalescent; this is a more scalable alternative to other approaches for computing the probability of a Beta coalescent tree (Birkner et al, 2011). We define a subtree structure $\theta_i = \{\theta_{i-1}, \delta_i, \rho_{\vec{c}_i}\}$, thus the tree θ_m after the final coalescent event m is a complete tree π. The message for node ρ_i marginalizes the features of the nodes in its children set.[2] The total message for a parent node ρ_i is

$$M_{\rho_i}(y_i) = Z_{\rho_i}^{-1}(\mathbf{x}|\theta_i) \prod_{b \in \vec{c}_i} \int \kappa_{t_i t_b}(y_i, y_b) M_{\rho_b}(y_b) dy_b. \tag{7.4}$$

[2] When ρ_b is a leaf, the message $M_{\rho_b}(y_b)$ is a delta function centered on the observation.

where $Z_{\rho_i}(\mathbf{x}|\theta_i)$ is the local normalizer, which can be computed as the combination of initial distribution and messages from a set of children,

$$Z_{\rho_i}(\mathbf{x}|\theta_i) = \int p_0(y_i) \prod_{b \in \vec{c}_i} \left(\int \kappa_{t_i t_b}(y_i, y_b) M_{\rho_b}(y_b) dy_b \right) dy_i. \tag{7.5}$$

Recursively performing this marginalization through message passing provides the joint probability of a complete tree π and the observations $\mathbf{x}$. At the root,

$$Z_{-\infty}(\mathbf{x}|\theta_m) = \int p_0(y_{-\infty}) \kappa_{-\infty,t_m}(y_{-\infty}, y_m) M_{\rho_m}(y_m) dy_m dy_{-\infty} \tag{7.6}$$

where $p_0(y_{-\infty})$ is the initial feature distribution and m is the number of coalescent events. This gives the marginal probability of the whole tree,

$$p(\mathbf{x}|\pi) = Z_{-\infty}(\mathbf{x}|\theta_m) \prod_{i=1}^{m} Z_{\rho_i}(\mathbf{x}|\theta_i), \tag{7.7}$$

The joint probability of a tree π combines the prior (Equation 7.3) and likelihood (Equation 7.7),

$$p(\mathbf{x}, \pi) = Z_{-\infty}(\mathbf{x}|\theta_m) \prod_{i=1}^{m} \lambda_{n_{i-1}}^{k_i} \exp(-\lambda_{n_{i-1}} \delta_i) \cdot Z_{\rho_i}(\mathbf{x}|\theta_i). \tag{7.8}$$

7.3 Sequential Monte Carlo Inference

Sequential Monte Carlo (SMC)—often called particle filters—estimates a structured sequence of hidden variables based on observations (Doucet et al, 2001). For coalescent models, this estimates the posterior distribution over tree structures given observations $\mathbf{x}$. Initially ($i = 0$) each observation is in a singleton cluster;[3] in subsequent particles ($i > 0$), points coalesce into more complicated tree structures θ_i^s, where s is the particle index and we add superscript s to all the related notations to distinguish between particles. We use sequential importance resampling (Gordon et al, 1993, SIR) to weight each particle s at time t_i, denoted as w_i^s.

The weights from SIR approximate the posterior. Computing the weights requires a conditional distribution of data given a latent state $p(\mathbf{x}|\theta_i^s)$, a transition distribution between latent states $p(\theta_i^s|\theta_{i-1}^s)$, and a proposal distribution $f(\theta_i^s|\theta_{i-1}^s, \mathbf{x})$. Together, these distributions define weights

$$w_i^s = w_{i-1}^s \frac{p(\mathbf{x} \,|\, \theta_i^s) p(\theta_i^s \,|\, \theta_{i-1}^s)}{f(\theta_i^s \,|\, \theta_{i-1}^s, \mathbf{x})}. \tag{7.9}$$

Then we can approximate the posterior distribution of the hidden structure using the normalized weights, which become more accurate with more particles.

To apply SIR inference to belief propagation with the beta coalescent prior, we first define the particle space structure. The s^{th} particle represents a subtree θ_{i-1}^s at time t_{i-1}^s, and a transition to a new subtree θ_i^s takes a set of nodes $\rho_{\vec{c}_i}^s$ from θ_{i-1}^s, and merges them at t_i^s, where $t_i^s = t_{i-1}^s - \delta_i^s$ and $\theta_i^s = \{\theta_{i-1}^s, \delta_i^s, \rho_{\vec{c}_i}^s\}$. Our proposal distribution must provide the duration δ_i^s and the children set $\rho_{\vec{c}_i}^s$ to merge based on the previous subtree θ_{i-1}^s.

[3] The relationship between time and particles is non-intuitive (following Teh et al (2008)). Time t goes backward with subsequent particles. When we use time-specific adjectives for particles, this is with respect to *inference*.

We propose the duration δ_i^s from the exponential distribution as in the beta's coalescent and propose a children set from the posterior distribution based on the local normalizers. [4] This is the "priorpost" method in Teh et al (2008).

However, this approach is intractable. Given n_{i-1} nodes at time t_i, we must consider all possible children sets $\binom{n_{i-1}}{2} + \binom{n_{i-1}}{3} + \cdots + \binom{n_{i-1}}{n_{i-1}}$. The computational complexity grows from $O(n_{i-1}^2)$ (Kingman's coalescent) to $O(2^{n_{i-1}})$ (beta coalescent).

7.4 Efficiently Finding Children Sets with DPMM

We need a more efficient way to consider possible children sets. Even for Kingman's coalescent, which only considers *pairs* of nodes, Görür et al (2012) do not exhaustively consider all pairs. Instead, they use data structures from computational geometry to select the R closest pairs as their restriction set, reducing inference to $O(n \log n)$. While finding closest pairs is a traditional problem in computational geometry (Shamos and Hoey, 1975), discovering arbitrary-sized sets is less studied.

In this section, we describe how we use a Dirichlet process mixture model (Antoniak, 1974, DPMM) to discover a restriction set Ω, integrating DPMMs into the SMC proposal. We first briefly review what DPMMs are, describe why they are attractive, and then describe how we incorporate DPMMs in SMC inference.

[4] This is a special case of Section 7.4.3's algorithm, where the restriction set Ω_i is all possible subsets.

7.4.1 DPMM

The DPMM is defined by a concentration β and a base distribution G_0. A distribution over mixtures is drawn from a Dirichlet process (DP): $G \sim \mathrm{DP}(\beta, G_0)$. Each observation x_i is assigned to a mixture component μ_i drawn from G. Because the Dirichlet process is a discrete distribution, observations i and j can have the same mixture component ($\mu_i = \mu_j$). When this happens, points are said to be in the same partition. Posterior inference can discover a distribution over partitions.

Given the Brownian diffusion kernel, a natural choice for the base distribution of the DP in the DPMM is a Gaussian. We review Gibbs sampling for this model (Neal, 2000), which provides distributions over partitions that become the restriction set.

We initialize partitions randomly and then repeatedly resample which partition each node is in. This is possible through the exchangeablility of the Dirichlet process.

Let x_n be the current node and $\mathbf{x}_{-n}$ all other nodes, z_n the current node's cluster assignment, $\mathbf{z}_{-n}$ all other nodes' cluster assignments, n_k is the number of nodes assigned to cluster k, and N is the total number of observations. As before, β is the Dirichlet process concentration parameter. We assume that the base distribution G_0 is a Gaussian distribution with mean μ_0 and covariance Σ_0 and that each cluster has known covariance Σ_k, thus the conditional distribution is

$$p(z_n = k | \mathbf{z}_{-n}, \mathbf{x}, \mu, \beta) = \begin{cases} \frac{n_k \mathcal{N}(x_n; \hat{\mu}_k, \hat{\Sigma}_k)}{\beta + N - 1} & k \text{ is old} \\ \\ \frac{\beta \mathcal{N}(x_n; \hat{\mu}_k, \hat{\Sigma}_k)}{\beta + N - 1} & k \text{ is new,} \end{cases} \tag{7.10}$$

where

$$\hat{\mu}_k = \frac{\mu_0 \Sigma_0^{-1} + \sum_{i \neq n} \mathbb{I}\left[z_i = k\right] x_i \cdot \Sigma_k^{-1}}{\Sigma_0^{-1} + \sum_{i \neq n} \mathbb{I}\left[z_i = k\right] \cdot \Sigma_k^{-1}}, \qquad \hat{\Sigma}_k = \frac{1 + \Sigma_0^{-1} \Sigma_k + \sum_{i \neq n} \mathbb{I}\left[z_i = k\right]}{\Sigma_0^{-1} + \sum_{i \neq n} \mathbb{I}\left[z_i = k\right] \cdot \Sigma_k^{-1}}$$

This is also called the infinite Gaussian mixture model (IGMM) (Rasmussen, 2000), which clusters nodes with similar feature values, providing useful candidates for the coalescent to merge.

7.4.2 Attractive Properties of DPMMs

DPMMs **and Coalescents** Berestycki (2009) showed that the distribution over partitions in a Dirichlet process is equivalent to the distribution over coalescents' allelic partitions—the set of members that have the same feature representation—when the mutation rate μ of the associated kernel is half of the Dirichlet concentration β (Section 5.2). For Brownian diffusion, we can connect DPMM with coalescents by setting the kernel covariance $\Sigma = \mu \mathbf{I}$ to $\Sigma = \beta/2\mathbf{I}$.

The base distribution G_0 is also related with nodes' feature. The base distribution G_0 of a Dirichlet process generates the probability measure G for each block, which generates the nodes in a block. As a result, we can select a base distribution which fits the distribution of the samples in coalescent process. For example, if we use Gaussian distribution for the transition kernel and prior, a Gaussian is also appropriate as the DPMM base distribution.

Effectiveness as a Proposal The necessary condition for a valid proposal (Cappe et al, 2007) is that it should have support on a superset of the true posterior. In our case, the distribution over partitions provided by the DPMM considers all possible children sets that could be merged in the coalescent. Thus the new proposal with DPMM satisfies this requirement, and it is a valid proposal.

In addition, Chen (2003) gives a set of desirable criteria for a good proposal distribution: accounts for outliers, considers the likelihood, and lies close to the true posterior. The DPMM fulfills these criteria. First, the DPMM provides a distribution over all partitions. Varying the concentration parameter β can control the length of the tail of the distribution over partitions. Second, choosing the base distribution of the DPMM appropriately models the feature likelihood; i.e., ensuring the DPMM places similar nodes together in a partition with high probability. Third, the DPMM qualitatively provides reasonable children sets when compared with exhaustively considering all children sets (Figure 7.3).

7.4.3 Incorporating DPMM in SMC Proposals

To address the inference intractability in Section 5.3, we use the DPMM to obtain a distribution over partitions of nodes. Each partition contains clusters of nodes, and we take a union over all partitions to create a restriction set $\Omega_i = \{\omega_{i1}, \omega_{i2}, \cdots\}$, where each ω_{ij} is a subset of the n_{i-1} nodes. A standard Gibbs sampler provides these partitions (see supplemental).

With this restriction set Ω_i, we propose the duration time δ_i^s from the exponential distribution and propose a children set $\rho^s_{\vec{c}_i}$ based on the local normalizers

$$f_i(\delta^s_i) = \lambda^s_{n_{i-1}} \exp(-\lambda^s_{n_{i-1}} \delta^s_i) \tag{7.11}$$

$$f_i(\rho^s_{\vec{c}_i} | \delta^s_i, \theta^s_{i-1}) = \frac{Z_{\rho_i}(\mathbf{x} | \theta^s_{i-1}, \delta^s_i, \rho^s_{\vec{c}_i})}{Z_0} \cdot \mathbb{I}\left[\rho^s_{\vec{c}_i} \in \Omega^s_i\right], \tag{7.12}$$

where Ω^s_i restricts the candidate children sets, $\mathbb{I}$ is the indicator, and we replace $Z_{\rho_i}(\mathbf{x}|\theta^s_i)$ with $Z_{\rho_i}(\mathbf{x}|\theta^s_{i-1}, \delta^s_i, \rho^s_{\vec{c}_i})$ since they are equivalent here. The normalizer is

$$Z_0 = \sum\nolimits_{\rho'_{\vec{c}}} Z_{\rho_i}(\mathbf{x}|\theta^s_{i-1}, \delta^s_i, \rho'_{\vec{c}}) \cdot \mathbb{I}\left[\rho'_{\vec{c}} \in \Omega^s_i\right] = \sum\nolimits_{\rho'_{\vec{c}} \in \Omega^s_i} Z_{\rho_i}(\mathbf{x}|\theta^s_{i-1}, \delta^s_i, \rho'_{\vec{c}}). \tag{7.13}$$

Applying the true distribution (the i^{th} multiplicand from Equation 7.8) and the proposal distribution (Equation 7.11 and Equation 7.12) to the SIR weight update (Equation 7.9),

$$w^s_i = w^s_{i-1} \frac{\lambda^{|\rho^s_{\vec{c}_i}|}_{n_{i-1}} \cdot \sum_{\rho'_{\vec{c}} \in \Omega^s_i} Z_{\rho_i}(\mathbf{x}|\theta^s_{i-1}, \delta^s_i, \rho'_{\vec{c}})}{\lambda^s_{n_{i-1}}}, \tag{7.14}$$

where $|\rho^s_{\vec{c}_i}|$ is the size of children set $\rho^s_{\vec{ci}}$; parameter $\lambda^{|\rho^s_{\vec{c}_i}|}_{n_{i-1}}$ is the rate of the children set $\rho^s_{\vec{c}_i}$ (Equation 7.2); and $\lambda^s_{n_{i-1}}$ is the rate of all possible sets given a total number of nodes n_{i-1} (Equation 7.2).

We can view this new proposal as a coarse-to-fine process: DPMM proposes candidate children sets; SMC selects a children set from DPMM to coalesce. Since the coarse step is faster and filters "bad" children sets, the slower finer step considers fewer children sets, saving computation time (Algorithm 6). If Ω_i has all children sets,

Algorithm 6 MCMC inference for generating a tree

```
for Particle $s = 1, 2, \cdots, S$ do
    Initialize $n^s = n$, $i = 0$, $t_0^s = 0$, $w_0^s = 1$.
    Initialize the node set $V^s = \{\rho_0, \rho_1, \cdots, \rho_n\}$.
while $\exists s \in \{1 \cdots S\}$ where $n^s > 1$ do
    Update $i = i + 1$.
    for Particle $s = 1, 2, \cdots, S$ do
        if $n^s == 1$ then
            Continue.
        Propose a duration $\delta_i^s$ by Equation 7.11.
        Set coalescent time $t_i^s = t_{i-1}^s - \delta_i^s$.
        Sample partitions $p_i^s$ from DPMM.
        Propose a set $\rho_{\vec{c}_i}^s$ according to Equation 7.12.
        Update weight $w_i^s$ by Equation 7.14.
        Update $n^s = n^s - |\rho_{\vec{c}_i}^s| + 1$.
        Remove $\rho_{\vec{c}_i}^s$ from $V^s$, add $\rho_i^s$ to $V^s$.
    Compute effective sample size **ESS** (Neal, 1998).
    if **ESS** $< S/2$ then
        Resample particles  (Fearhhead, 1998).
```

it recovers exhaustive SMC. We estimate the effective sample size (Neal, 1998) and resample (Fearhhead, 1998) when needed. For smaller sets, the DPMM is sometimes impractical (and only provides singleton clusters). In such cases it is simpler to enumerate all children sets.

7.4.4 Example Transition Kernel: Brownian Diffusion

This section uses Brownian diffusion as an example for message passing framework. The initial distribution $p_0(y)$ of each node is $\mathcal{N}(0, \infty)$; the transition kernel $\kappa_{t_i t_b}(y, \cdot)$ is a Gaussian centered at y with variance $(t_i - t_b)\Sigma$, where b is a child node index ($b \in \vec{c}_i$), $\Sigma = \mu\mathbf{I}$, $\mu = \beta/2$ and β is the concentration parameter of DPMM. Then the local normalizer $Z_{\rho_i}(\mathbf{x}|\theta_i)$ is

$$Z_{\rho_i}(\mathbf{x}|\theta_i) = \int \mathcal{N}(y_i; 0, \infty) \prod\nolimits_{b \in \vec{c}_i} \mathcal{N}(y_i; \hat{y}_b, \Sigma(v_{\rho_b} + t_b - t_i)) dy_i, \tag{7.15}$$

and the node message $M_{\rho_i}(y_i)$ is normally distributed $M_{\rho_i}(y_i) \sim \mathcal{N}(y_i; \hat{y}_{\rho_i}, \Sigma v_{\rho_i})$, where

$$v_{\rho_i} = \left(\sum\nolimits_{b \in \vec{c}_i} (v_{\rho_b} + t_b - t_i)^{-1}\right)^{-1}, \qquad \hat{y}_{\rho_i} = \left(\sum\nolimits_{b \in \vec{c}_i} \frac{\hat{y}_{\rho_b}}{v_{\rho_b} + t_b - t_i}\right) v_{\rho_i}.$$

7.5 Experiments: Finding Bushy Trees

In this section, we compare trees built by the beta coalescent (**beta**) against those built by Kingman's coalescent (**kingman**) and hierarchical agglomerative clustering (Eads, 2007, **hac**) on both synthetic and real data. We show **beta** performs best and can capture data in interpretable, bushier trees.

Setup The parameter α for the beta coalescent is between 1 and 2. The closer α is to 1, bushier the tree is, and we set $\alpha = 1.2$.[5] We set the mutation rate as 1, thus the DPMM parameter is initialized as $\beta = 2$, and updated using slice sampling (Neal, 2003b). All experiments use 100 initial iterations of DPMM inference with 30 more iterations after each coalescent event (forming a new particle).

Metrics We use three metrics to evaluate the quality of the trees discovered by our algorithm: purity, subtree and path length. The dendrogram **purity** score (Powers, 1997; Heller and Ghahramani, 2005) measures how well the leaves in a subtree belong to the same class. For any two leaf nodes, we find the least common subsumer node s and—for the subtree rooted at s—measure the fraction of leaves with same class

[5] With DPMM proposals, while the Beta coalescent prior is very important, α has a negligible effect, so we elide further analysis for different α values.

labels. The **subtree** score (Teh et al, 2008) is the ratio between the number of internal nodes with all children in the same class and the total number of internal nodes. The path **length** score is the average difference—over all pairs—of the lowest common subsumer distance between the true tree and the generated tree, where the lowest common subsumer distance is the distance between the root and the lowest common subsumer of two nodes. For **purity** and **subtree**, higher is better, while for **length**, lower is better. Scores are in expectation over particles and averaged across chains.

7.5.1 Synthetic Hierarchies

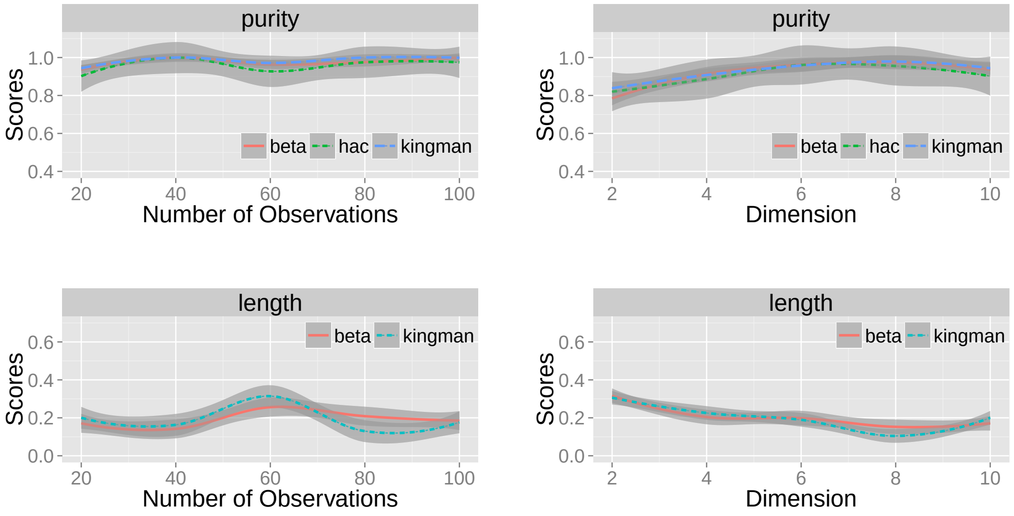
To test how well the different methods capture hierarchical data, we generate synthetic hierarchical data with a known structure and test whether our model can recover the hierarchy; we also assume each child of the root has a unique label and the descendants also have the same label as their parent node (except the root node).

According to Berestycki (2009), given n_{i-1} nodes at time t_{i-1} and $t_i = t_{i-1} - \delta_i$, the expected number of nodes that merge at time t_i is

$$1 + \delta_i \left(\sum_{k_i=2}^{n_{i-1}} (k_i - 1) \tbinom{n_{i-1}}{k_i} \lambda_{n_{i-1}}^{k_i} \right). \tag{7.16}$$

Therefore we start with n_0 nodes, sample a duration time δ_i, and compute the expected number of nodes to be merged at time t_i; we then merge that number of nodes and repeat until there is only one node.

Next we generate the features for nodes from a Gaussian kernel. We start with



(a) Increasing observations

(b) Increasing dimension

Figure 7.2: Comparing **beta** against **kingman** and **hac** on different dimensions: Figure 7.2(a) shows the effect of changing the underlying data size; Figure 7.2(b) shows the effect of increasing number of dimension. The results of the three models are comparable, which shows that introducing DPMM to the proposal does not hurt the inference accuracy.

the root node as a multivariate Gaussian distribution $\mathcal{N}(\mu_0, \Sigma_0)$, where the mean $\mu_0 = (0, \cdots, 0)$ and $\Sigma_0 = \rho_0 \mathbf{I}$ ($\mathbf{I}$ is the identity matrix). For each child, we sample the feature vector y_c from the parent Gaussian $\mathcal{N}(y_p, \Sigma_p)$, and set $\Sigma_c = \frac{1}{n}\rho_p \mathbf{I}$. In this experiment, we generate the data with parameter $\rho_0 = 10$. Labels are assigned based on the root's children; each subtree rooted at a child of the root receives the same label. This class label is used to calculate the metrics defined above.

Given the generated synthetic data, we compare **beta** against **kingman** and **hac** by varying the number of observations (Figure 7.2(a)) and feature dimensions (Figure 7.2(b)). In both cases, **beta** is comparable to **kingman** and **hac** (no edge **length**). While increasing the feature dimension improves both scores, more observations do not: for synthetic data, a small number of observations suffice to

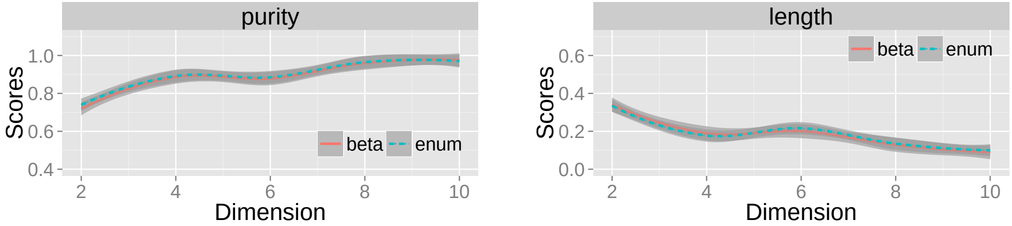


Figure 7.3: Comparing the inference schemes using DPMM proposal for children sets against an exhaustive enumeration of all possible children sets (**enum**). The results show that DPMM does not sacrifice any accuracy, and is comparable with **enum**.

construct a good tree.

To evaluate the effectiveness of using our DPMM as a proposal distribution, we compare exhaustively enumerating all children set candidates (**enum**) while keeping the SMC otherwise unchanged; this experiment uses ten data points (**enum** is completely intractable on larger data). **Beta** uses the DPMM and achieved similar accuracy (Figure 7.3) while greatly improving efficiency. More qualitative comparison of constructed tree structures between Kingman's coalescent and the beta coalescent can be found in Appendix D.

7.5.2 Human Tissue Development

Our first real dataset is based on the developmental biology of human tissues. As a human develops, tissues specialize, starting from three embryonic germ layers: the endoderm, ectoderm, and mesoderm. These eventually form all human tissues. For example, one developmental pathway is *ectoderm* → *neural crest* → *cranial neural crest* → *optic vesicle* → *cornea*. Because each germ layer specializes into many different types of cells at specific times, it is inappropriate to model this development

	Biological Data			20-newsgroups Data		
	hac	**kingman**	**beta**	**hac**	**kingman**	**beta**
purity ↑	0.453	0.474 ± 0.029	**0.492** ± 0.028	0.465	0.510 ± 0.047	**0.565** ± 0.081
subtree ↑	0.240	0.302 ± 0.033	**0.331** ± 0.050	0.571	0.651 ± 0.013	**0.720** ± 0.013
length ↓	–	0.654 ± 0.041	**0.586** ± 0.051	–	0.477 ± 0.027	**0.333** ± 0.047

Table 7.1: Comparing the three models: **beta** performs best on all three scores.

as a binary tree, or with clustering models lacking path lengths.

Historically, uncovering these specialization pathways is a painstaking process, requiring inspection of embryos at many stages of development; however, massively parallel sequencing data make it possible to efficiently form developmental hypotheses based on similar patterns of gene expression. To investigate this question we use the transcriptome of 27 tissues with known, unambiguous, time-specific lineages (Jongeneel et al, 2005). We reduce the original 182727 dimensions via principle component analysis (Shlens, 2005, PCA). We use five chains with five particles per chain.

Using reference developmental trees, **beta** performs better on all three scores (Table 7.1) because **beta** builds up a bushy hierarchy more similar to the true tree. The tree recovered by **beta** (Figure 7.4) reflects human development. The first major differentiation is the division of embryonic cells into three layers of tissue: endoderm, mesoderm, and ectoderm. These go on to form almost all adult organs and cells. The placenta (magenta), however, forms from a fourth cell type, the trophoblast; this is placed in its own cluster at the root of the tree. It also successfully captures ectodermal tissue lineage. However, mesodermic and endodermic tissues, which are highly diverse, do not cluster as well. Tissues known to secrete endocrine hormones (dashed borders) cluster together.

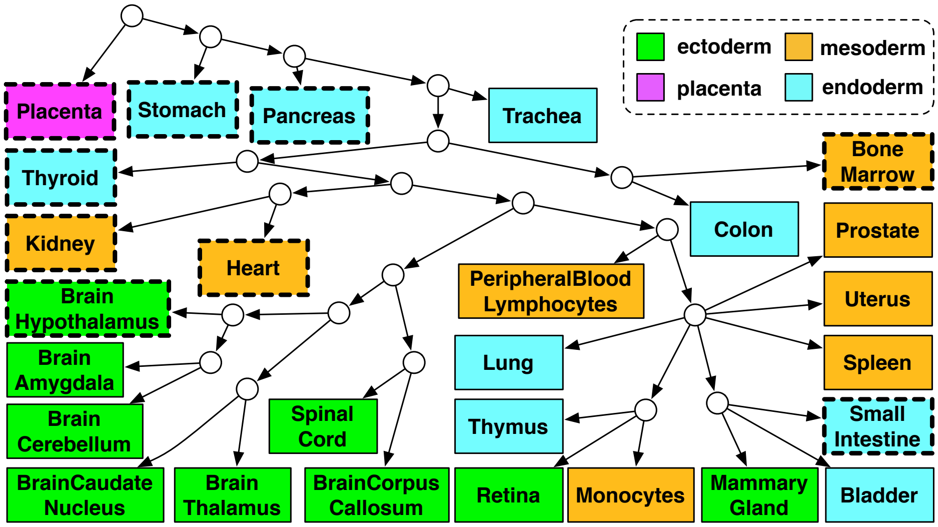


Figure 7.4: One sample hierarchy of human tissue from **beta**. Color indicates germ layer origin of tissue. Dashed border indicates secretory function. While neural tissues from the ectoderm were clustered correctly, some mesoderm and endoderm tissues were commingled. The cluster also preferred placing secretory tissues together and higher in the tree.

7.5.3 Clustering 20-newsgroups Data

Following Heller and Ghahramani (2005), we also compare the three models on 20-newsgroups,[6] a multilevel hierarchy first dividing into general areas (rec, space, and religion) before specializing into areas such as baseball or hockey.[7] This true hierarchy is inset in the bottom right of Figure 7.5, and we assume each edge has the same length. We apply latent Dirichlet allocation (Blei et al, 2003b) with 50 topics to this corpus, and use the topic distribution for each document as the document feature. We use five chains with eighty particles per chain.

As with the biological data, **beta** performs best on all scores for 20-newsgroups. Figure 7.5 shows a bushy tree built by **beta**, which mostly recovered the true

[6] `http://qwone.com/~jason/20Newsgroups/`

[7] We use “rec.autos”, “rec.sport.baseball”, “rec.sport.hockey”, “sci.space” newsgroups but also—in contrast to Heller and Ghahramani (2005)—added “soc.religion.christian”.

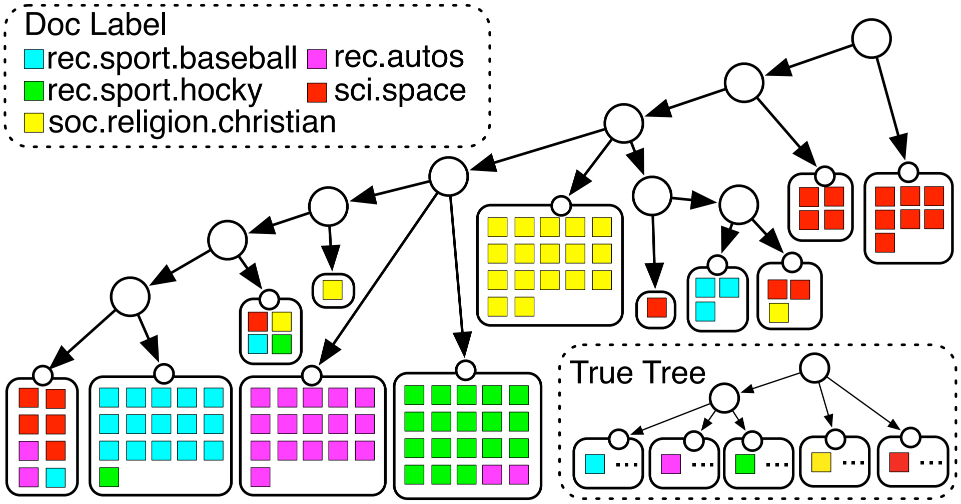


Figure 7.5: One sample hierarchy of the 20newsgroups from **beta**. Each small square is a document colored by its class label. Large rectangles represent a subtree with all the enclosed documents as leaf nodes. Most of the documents from the same group are clustered together; the three "rec" groups are merged together first, and then merged with the religion and space groups.

hierarchy. Documents within a newsgroup merge first, then the three "rec" groups, followed by "space" and "religion" groups. We only use topic distribution as features, so better results could be possible with more comprehensive features.

7.6 Summary

This chapter generalizes Bayesian hierarchical clustering, moving from Kingman's coalescent to the beta coalescent. Our novel inference scheme based on SMC and DPMM make this generalization practical and efficient. This new model provides a bushier tree, often a more realistic view of data.

While we only consider real-valued vectors, which we model through the ubiquitous Gaussian, other likelihoods might be better suited to other applications. For example, for discrete data such as in natural language processing, a multinomial likelihood may be more appropriate. This is a straightforward extension of our model via other transition kernels and DPMM base distributions.

Recent work uses the coalescent as a means of producing a clustering in tandem with a downstream task such as classification (Rai and Daumé III, 2008) or tree-based topic models (Chapter 2). For example, these automatically learned hierarchies can be used as prior trees for tree-based topic models (Chapter 2), which is less expensive than interacting with users (Chapter 4).

In addition, a fully statistical model like the beta coalescent that jointly learns the hierarchy and a downstream task could improve performance in dependency parsing (Koo et al, 2008) (clustering parts of speech) or multilingual sentiment (Boyd-Graber and Resnik, 2010) (finding sentiment-correlated words across languages). Particularly, this joint learning framework can also be used in topic modeling to learn topics and hierarchies jointly and automatically, which may further improve the topic coherence.

Chapter 8

Conclusion and Future Work

Adding expressive prior knowledge to topic models allows users to obtain better topics. It can be treated as semi-supervising topic models or adding a "soft" constraint to topic models. While tree-based topic models meet these requirements, it is important to consider possible sources of these prior knowledge.

In this dissertation, we discuss three important sources: getting the prior knowledge from users in an interactive setting (Chapter 4), using existing knowledge resources (Chapter 5), and extracting the prior knowledge automatically by Bayesian hierarchical clustering techniques (Chapter 7).

Given the prior knowledge, we review and propose various models to incorporate the prior knowledge: we review vanilla topic models and tree-based topic models (Chapter 2), and present three different new models, including interactive topic modeling (Chapter 4), polylingual tree-based topic models (Chapter 5) and Beta coalescent models (Chapter 7).

We also discuss different inference algorithms for these models: efficient Gibbs sampling (Chapter 3), variational inference (briefly, Chapter 5), spectral methods with regularization (Chapter 6), and particle filters (Chapter 7).

In addition, we use various ways to evaluate these models and inference algorithms: automatic measures (Chapter 3, 4, 6, 7), user study (Chapter 4), and a real application in statistical machine translation (Chapter 5).

In this chapter, we discuss directions for future extensions:

Better evaluation with users. The question of whether topic models assist users in information seeking requires more experimentation. While we showed that ITM encouraged users to use topics to help them find information, our population was too diverse and too small to be able to demonstrate that these techniques helped them to *better* or *more quickly* access the information. Broadening the number of users for the user study would allow us to draw stronger conclusions about how interactive topic modeling changes or helps the way users seek out information from large corpora. In addition, with larger populations, a mixed-effects model could potentially untangle the effects of how familiar that users are to topic models, whether they understand the task clearly, their background knowledge of the subject, and whether they understand how to use the interface. Explicitly modeling and measuring these effects would effectively reduce the variance and help explain the interaction between these nuanced facets of user behavior.

New Probabilistic Models and Inference Techniques. In our attempt to minimize the inference latency, we developed new techniques for probabilistic inference that take advantage of sparsity in probabilistic models. While our approach was more complicated than first method designed specifically for latent Dirichlet allocation (Yao et al, 2009), it still offered substantial computational speedup. This suggests that taking advantage of sparsity could also be used in other probabilistic models such as context-free grammars (Johnson et al, 2007; Johnson, 2010) and feature-based models (Monroe et al, 2008; Wu et al, 2010).

In addition to improving traditional inference, finding alternative solutions such as spectral methods (Anandkumar et al, 2012b; Arora et al, 2012a) is another possible way to improve inference. Encoding prior knowledge by regularize these models can further improve the performance. This dissertation encodes Gaussian prior, Dirichlet prior and informed priors to anchor methods for topic models and obtain more coherent topics. It can be further extended to encode L_1 regularization to encourage true sparsity (Tibshirani, 1994), or consider more specific priors into the regulizer, for example, word correlations as in Chapter 2.

Domain Adaptation for Downstream Tasks Topic models provide a natural way for inducing domains, since each topic can be treated as a domain. In this dissertation, we have applied topic models for domain adaptation in translation models in statistical machine translation; in addition, topic models can also be used in domain adaptation for language models (Bellegarda, 2004; Wood and Teh, 2009), another important avenue for improving machine translation. These domain knowledge from topic models can also improve content understanding, and can be potentially helpful in a lot of other downstream tasks such as domain-specific clustering or classification.

Joint Learning Structures and Downstream Tasks Instead of using traditional hierarchical agglomerative clustering techniques such as the Brown clustering (Brown et al, 1992), this dissertation presents a Bayesian hierarchical clustering technique based on the beta coalescent to automatically learn the prior tree structures. This fully statistical model can be encoded into downstream Bayesian tasks and jointly learn the hierarchy and a downstream task to improve performance in dependency parsing (Koo et al, 2008) (clustering parts of speech) or multilingual sentiment (Boyd-Graber and Resnik, 2010) (finding sentiment-correlated words across languages). In particular, this joint learning framework can also be used in topic modeling to learn topics and hierarchies jointly and automatically, which may further improve the topic coherence.

Toward More Interactive Statistical Models While interactive topic modeling can obviate or replace some of the newer topic models, some models seem apt for interactive topic modeling. For example, combining interactivity with dynamic topic modeling (Blei and Lafferty, 2006; Wang et al, 2008) could help to improve historians or social scientists working with datasets over long time periods; supervised topic models could help researchers understand how documents interact with external socioeconomic indicators such as the sentiment (Pang and Lee, 2008; Sayeed et al, 2012), consumer price index (Shoemaker, 2011), stock price (Kogan et al, 2009), or geography (Eisenstein et al, 2010); and topic models that go beyond the bag of words (Wallach, 2006; Boyd-Graber and Blei, 2008) could help understand syntactic patterns and linguistic structure. Learning from users is not just a benefit, but it is an essential goal for machine learning algorithms to be accepted by researchers who are not computer scientists and eventually the broader public. Interactive topic models are an example of tools that can learn from and help users interact with large datasets, an essential tool for modern text-based research.

Appendix A

Titles of the ten bills used in the user study (Chapter 4.4)

- H.R. 6061: **Secure Fence** Act of 2006
- H.R. 8: **Death Tax** Repeal Permanency Act of 2005
- S. 2271: USA PATRIOT Act Additional Reauthorizing **Amendments Act** of 2006
- S. 3711: Gulf of Mexico **Energy Security** Act of 2006
- S. 2611: Comprehensive **Immigration** Reform Act of 2006
- S. 403: **Child** Custody Protection Act
- H.R. 4297: **Tax** Increase Prevention and Reconciliation Act of 2005
- S.J.Res. 12: **Flag Desecration** resolution
- H.R. 810: **Stem Cell** Research Enhancement Act of 2005
- S.Con.Res. 83: **Budget** resolution FY2007

Appendix B

Questions list in the user study (Chapter 4.4)

- The flag desecration act gives power to what body to prohibit physical desecration of the flag of the United States?
- The child custody protection act makes it what type of crime to take minors across state lines to circumvent laws requiring involvement of parents in abortion decisions?
- According to the debate for the secure fence act, what is the length (in miles) of the border that the U.S. shares with Mexico?
- The Gulf of Energy Security act will provide revenue streams for which fund?
- A senator compares the immigration debate to the release of what film?
- Name 5 of the debated legislation in this data set. Give either the full name or the number assigned to the debate.
- Name 2 of the debated legislation in this data set deals with taxes or the budget?
- What is the name of the debated legislation which proposes an amendment to the constitution?
- Name the 2 debated legislation in this data set that discusses illegal immigrants?

Appendix C

Variational Inference for Tree-based Topic Models

The variational inference for tree-based topic models is derived and developed by collaborating with Ke Zhai, who also parallelized variational inference using hadoop MapReduce. The equations for variational inference are attached here for dissertation completeness. More details can be found in (Hu et al, 2014).

For a collection of D documents, each of which contains N_d number of words, the latent variables of ptLDA are: transition distributions $\boldsymbol{\pi}_{ki}$ for every topic k and internal node i in the prior tree structure; multinomial distributions over topics $\boldsymbol{\theta}_d$ for every document d; topic assignments z_{dn} and path y_{dn} for the n^{th} word w_{dn} in document d. The joint distribution of polylingual tree-based topic models is

$$p(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{\pi}; \alpha, \beta) = \prod_k \prod_i p(\boldsymbol{\pi}_{ki} | \beta_i) \cdot \prod_d p(\boldsymbol{\theta}_d | \alpha) \cdot \prod_d \prod_n p(z_{dn} | \boldsymbol{\theta}_d) \\ \cdot \textstyle\prod_d \prod_n \big(p(y_{dn} | z_{dn}, \boldsymbol{\pi}) p(w_{dn} | y_{dn}) \big). \tag{C.1}$$

Exact inference is intractable, so we turn to approximate posterior inference to discover the latent variables that best explain our data. Two widely used approximation approaches are *Markov chain Monte Carlo* (Neal, 2000, MCMC) and *variational Bayesian inference* (Blei et al, 2003b, VB). Both frameworks produce good approximations of the posterior mode (Asuncion et al, 2009).

MCMC (Gibbs Sampling) has been discussed in Chapter 2 and Chapter 3, so we derive the variational Bayesian inference here. In addition, Mimno et al (2012) propose hybrid inference that takes advantage of parallelizable variational inference for global variables (Wolfe et al, 2008) while enjoying the sparse, efficient updates for local variables (Neal, 1993).

C.1 Variational Bayesian Inference

Variational Bayesian inference approximates the posterior distribution with a simplified *variational distribution* q over latent variables: document topic proportions θ, transition probabilities π, topic assignments z, and path assignments y.

Variational distributions typically assume a mean-field distribution over these latent variables, removing all dependencies between the latent variables. We follow this assumption for the transition probabilities $q(\boldsymbol{\pi} \,|\, \boldsymbol{\lambda})$ and the document topic proportions $q(\boldsymbol{\theta} \,|\, \boldsymbol{\gamma})$; both are variational Dirichlet distributions. However, due to the tight coupling between the path and topic variables, we must model this joint distribution as one multinomial, $q(\boldsymbol{z}, \boldsymbol{y} \,|\, \boldsymbol{\phi})$. If word token w_{dn} has K topics and S paths, it has a $K * S$ length variational multinomial ϕ_{dnks}, which represents the probability that the word takes path s in topic k. The complete variational distribution is

$$q(\boldsymbol{\theta}, \boldsymbol{\pi}, \boldsymbol{z}, \boldsymbol{y}|\boldsymbol{\gamma}, \boldsymbol{\lambda}, \boldsymbol{\phi}) = \prod_{d} q(\boldsymbol{\theta}_d|\boldsymbol{\gamma}_d) \cdot \prod_{k} \prod_{i} q(\boldsymbol{\pi}_{ki}|\lambda_{ki}) \cdot \prod_{d} \prod_{n} q(z_{dn}, y_{dn}|\boldsymbol{\phi}_{dn}). \quad \text{(C.2)}$$

Our goal is to find the variational distribution q that is closest to the true posterior, as measured by the *Kullback-Leibler* (KL) divergence between the true posterior p and variational distribution q. This induces an “evidence lower bound” (ELBO, $\mathcal{L}$) as a function of a variational distribution q:

$$\begin{aligned}\mathcal{L} =& \mathbb{E}_q[\log p(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{\pi})] - \mathbb{E}_q[\log q(\boldsymbol{\theta}, \boldsymbol{\pi}, \boldsymbol{z}, \boldsymbol{y})] \\ =& \sum_k \sum_i \mathbb{E}_q[\log p(\boldsymbol{\pi}_{ki}|\beta_i)] + \sum_d \mathbb{E}_q[\log p(\boldsymbol{\theta}_d|\alpha)] \\ &+ \sum_d \sum_n \mathbb{E}_q[\log p(z_{dn}, y_{dn}|\boldsymbol{\theta}_d, \boldsymbol{\pi}) p(w_{dn}|y_{dn})] + \mathbb{H}[q(\boldsymbol{\theta})] + \mathbb{H}[q(\boldsymbol{\pi})] + \mathbb{H}[q(\boldsymbol{z}, \boldsymbol{y})],\end{aligned} \tag{C.3}$$

where $\mathbb{H}[\bullet]$ represents the entropy of a distribution. Optimizing $\mathcal{L}$ using coordinate descent provides the following updates:

$$\phi_{dnkt} \propto \exp\{\Psi(\gamma_{dk}) - \Psi(\textstyle\sum_k \gamma_{dk}) + \sum_{i \to j \in s} \left(\Psi(\lambda_{k,i \to j}) - \Psi(\sum_{j'} \lambda_{k,i \to j'})\right)\}; \tag{C.4}$$

$$\gamma_{dk} = \alpha_k + \textstyle\sum_n \sum_{s \in \Omega^{-1}(w_{dn})} \phi_{dnkt}; \tag{C.5}$$

$$\lambda_{k,i \to j} = \beta_{i \to j} + \textstyle\sum_d \sum_n \sum_{s \in \Omega'(w_{dn})} \phi_{dnkt} \mathbb{I}\left[i \to j \in s\right]; \tag{C.6}$$

where $\Omega'(w_{dn})$ is the set of all paths that lead to word w_{dn} in the tree, and t represents one particular path in this set. $\mathbb{I}\left[i \to j \in s\right]$ is the indicator of whether path s contains an edge from node i to j.

C.2 Hybrid Stochastic Inference

Given the complementary strengths of MCMC and VB, and following hybrid inference proposed by Mimno et al (2012), we also derive hybrid inference for ptLDA.

The transition distributions $\boldsymbol{\pi}$ are treated identically as in variational inference. We posit a variational Dirichlet distribution $\boldsymbol{\lambda}$ and choose the one that minimizes the KL divergence between the true posterior and the variational distribution.

For topic $\boldsymbol{z}$ and path $\boldsymbol{y}$, instead of variational updates, we use a Gibbs sampler within a document. We sample z_{dn} and y_{dn} conditioned on the topic and path assignments of all other document tokens, based on the variational expectation of $\boldsymbol{\pi}$,

$$q(z_{dn} = k, y_{dn} = s | \neg \boldsymbol{z}_{dn}, \neg \boldsymbol{y}_{dn}; \boldsymbol{w}) \propto \tag{C.7}$$

$$(\alpha + \textstyle\sum_{m \neq n} \mathbb{I}\left[z_{dm} = k\right]) \cdot \exp\{\mathbb{E}_q[\log p(y_{dn}|z_{dn}, \boldsymbol{\pi}) p(w_{dn}|y_{dn})]\}.$$

This equation embodies how this is a hybrid algorithm: the first term resembles the Gibbs sampling term encoding how much a document prefers a topic, while the second term encodes the expectation under the variational distribution of how much a path is preferred by this topic,

$$\mathbb{E}_q[\log p(y_{dn}|z_{dn}, \boldsymbol{\pi}) p(w_{dn}|y_{dn})] = \mathbb{I}_{[\Omega(y_{dn}) = w_{dn}]} \cdot \sum_{i \to j \in y_{dn}} \mathbb{E}_q[\log \lambda_{z_{dn}, i \to j}]. \tag{C.8}$$

For every document, we sweep over all its tokens and resample their topic z_{dn} and path y_{dn} conditioned on all the other tokens' topic and path assignments $\neg \boldsymbol{z}_{dn}$ and $\neg \boldsymbol{y}_{dn}$. To avoid bias, we discard the first B burn-in sweeps and take the following M samples. We then use the empirical average of these samples update the global variational parameter $q(\boldsymbol{\pi}|\boldsymbol{\lambda})$ based on how many times we sampled these paths

$$\lambda_{k,i \to j} = \tfrac{1}{M} \textstyle\sum_d \sum_n \sum_{s \in \Omega^{-1}(w_{dn})} \big(\mathbb{I}\left[i \to j \in s \right] \cdot \mathbb{I}\left[z_{dn} = k, y_{dn} = s \right] \big) + \beta_{i \to j}. \quad \text{(C.9)}$$

For our experiments, we use the recommended settings $B = 5$ and $M = 5$ from Mimno et al (2012).

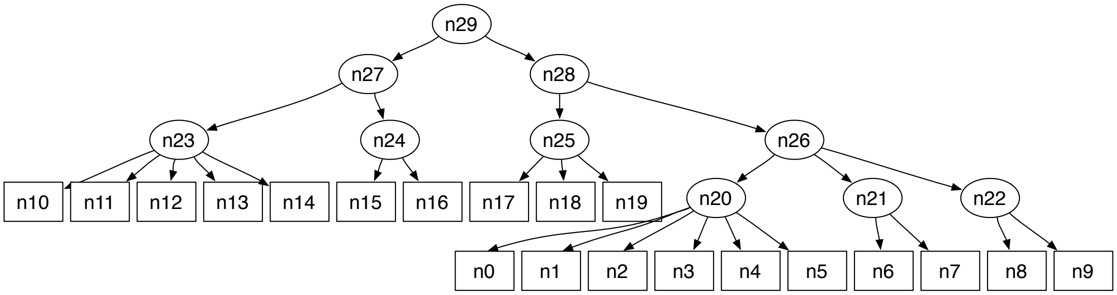
Appendix D

Comparing Coalescent Models on Synthetic Data

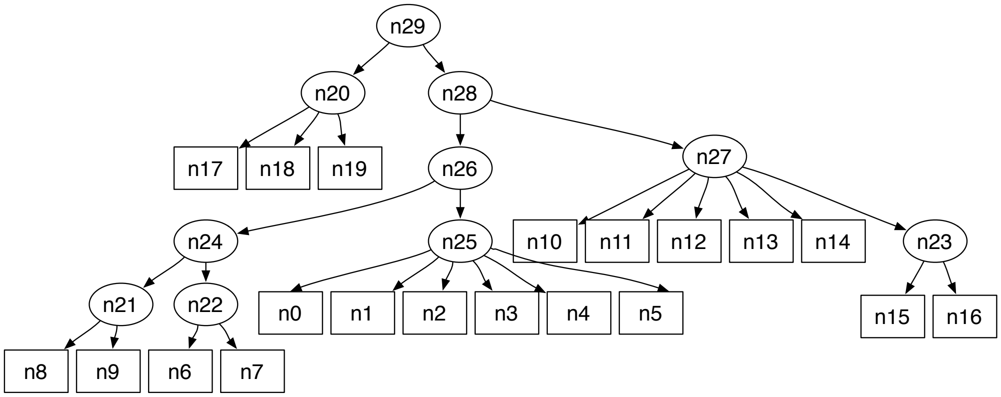
This appendix compares the constructed synthetic trees of Beta coalescent and Kingman's coalescent with the true synthetic trees. For all the following trees, the square nodes are the observed leaf nodes, and the circle nodes are the detected hidden internal nodes.

D.1 Tree1: n = 20

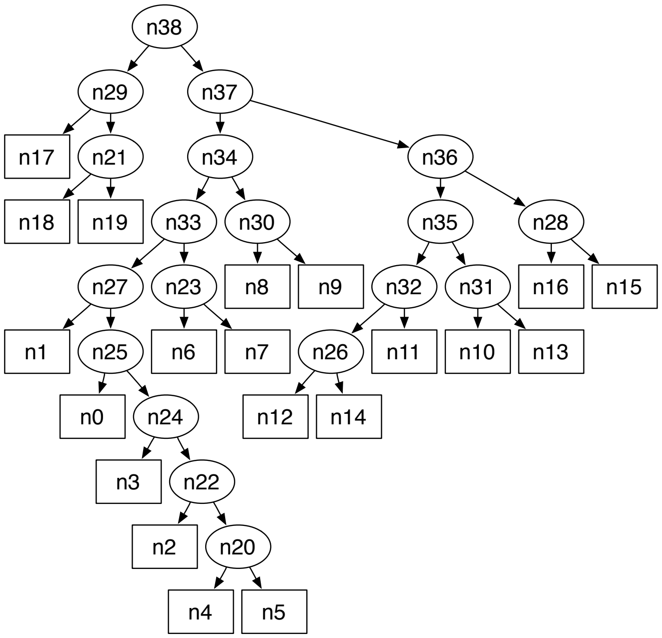
- True synthetic tree



- Constructed tree from Beta coalescent

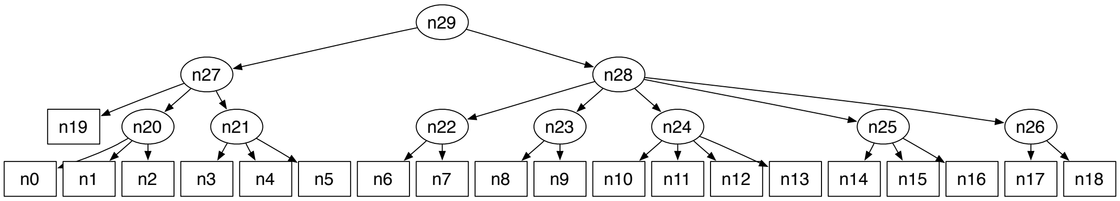


- Constructed tree from Kingman's coalescent

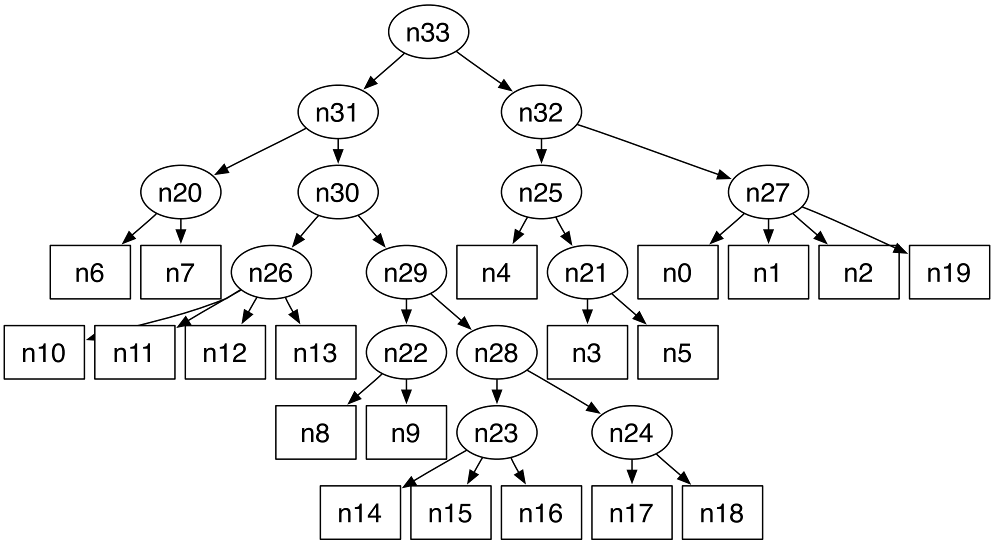


D.2 Tree2: n = 20

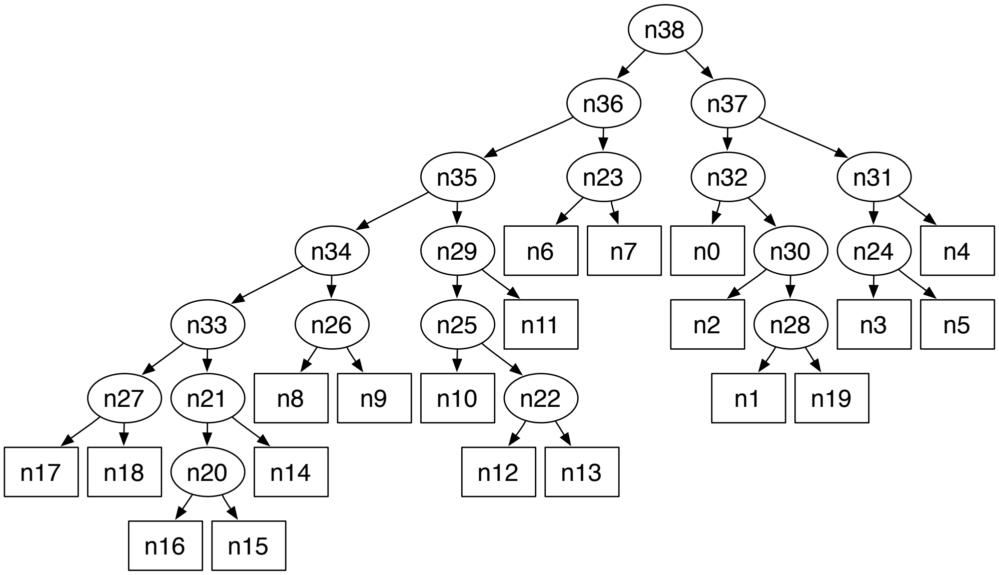
- True synthetic tree



- Constructed tree from Beta coalescent

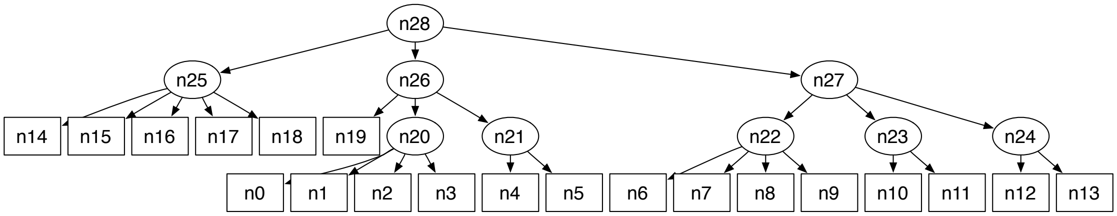


- Constructed tree from Kingman's coalescent

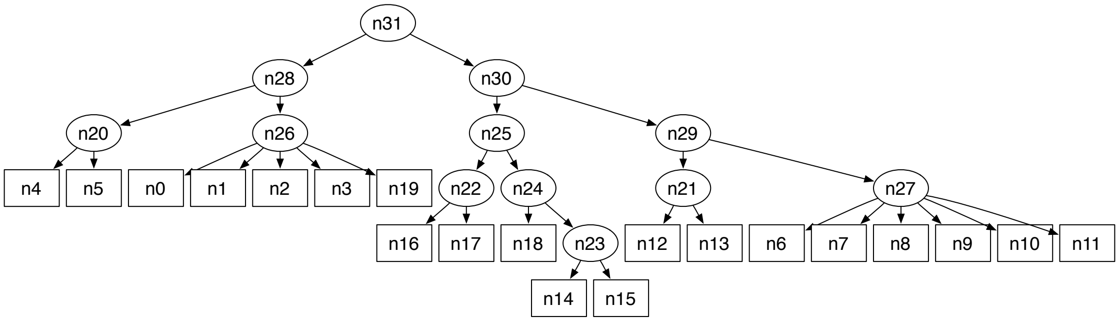


D.3 Tree3: n = 20

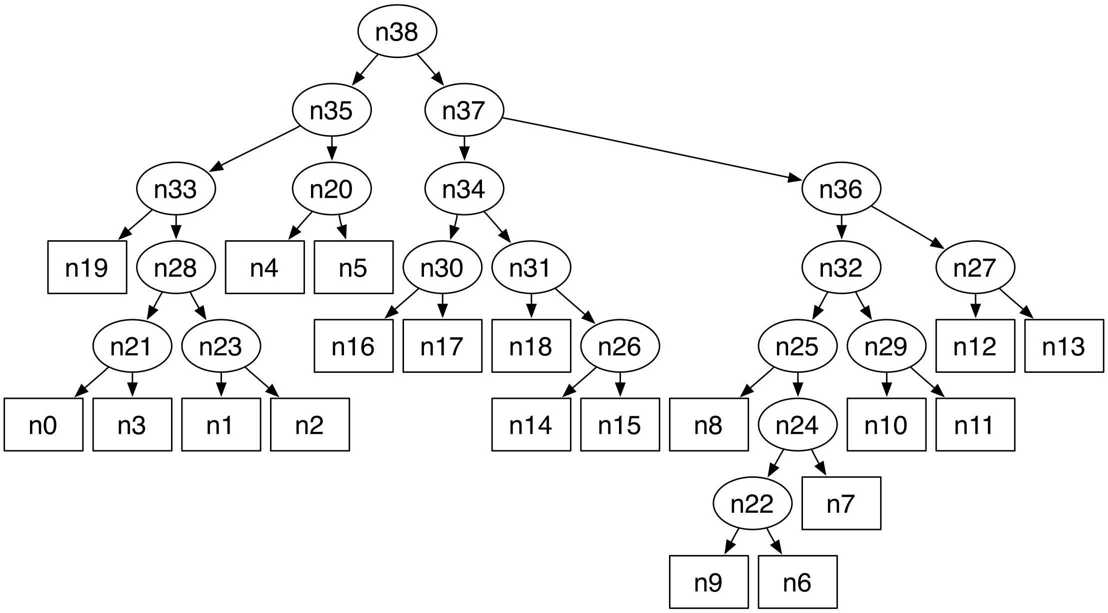
- True synthetic tree



- Constructed tree from Beta coalescent

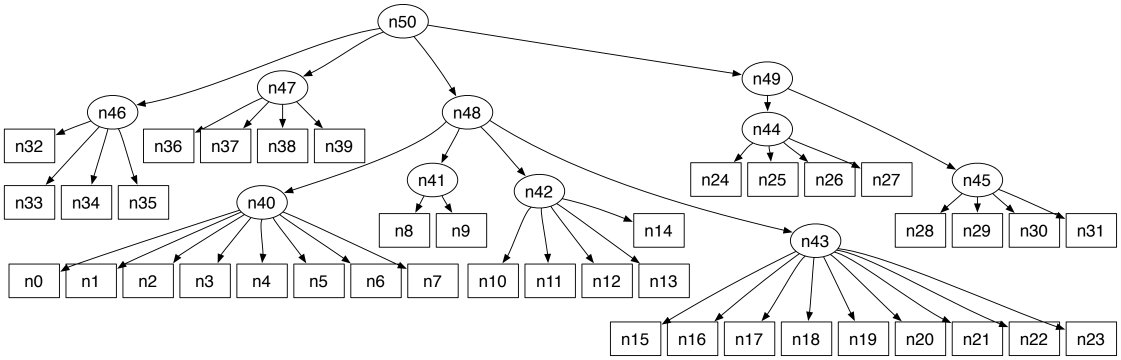


- Constructed tree from Kingman's coalescent

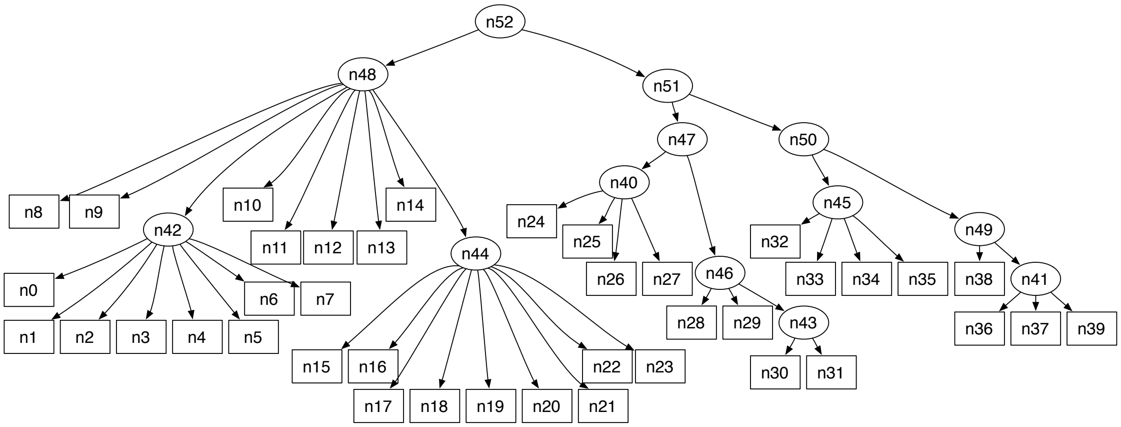


D.4 Tree4: n = 40

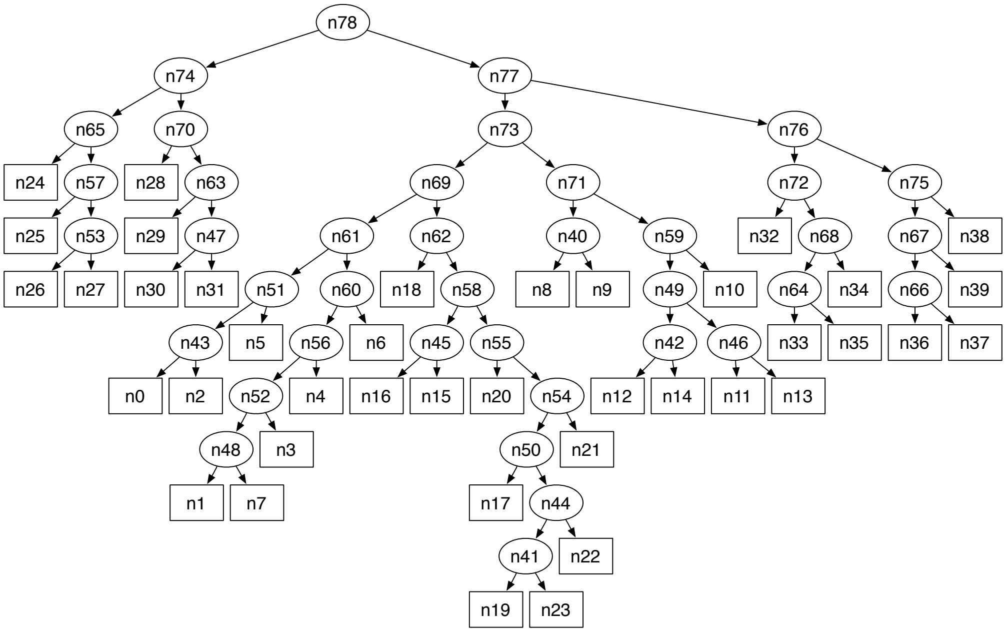
- True synthetic tree



- Constructed tree from Beta coalescent

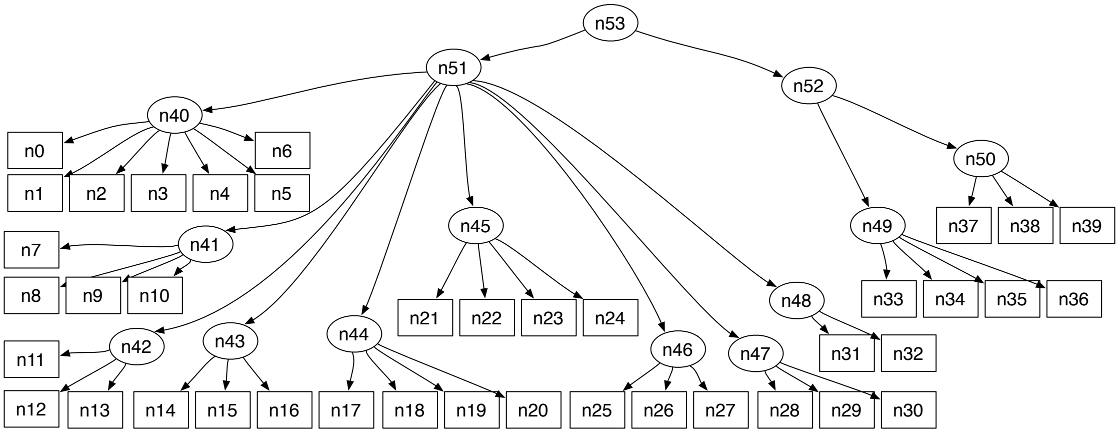


- Constructed tree from Kingman's coalescent

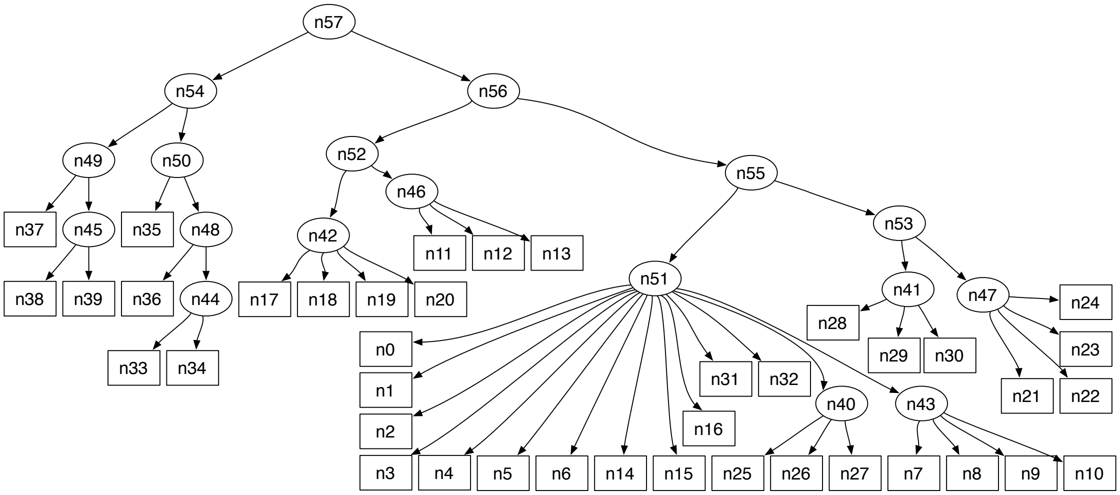


D.5 Tree5: n = 40

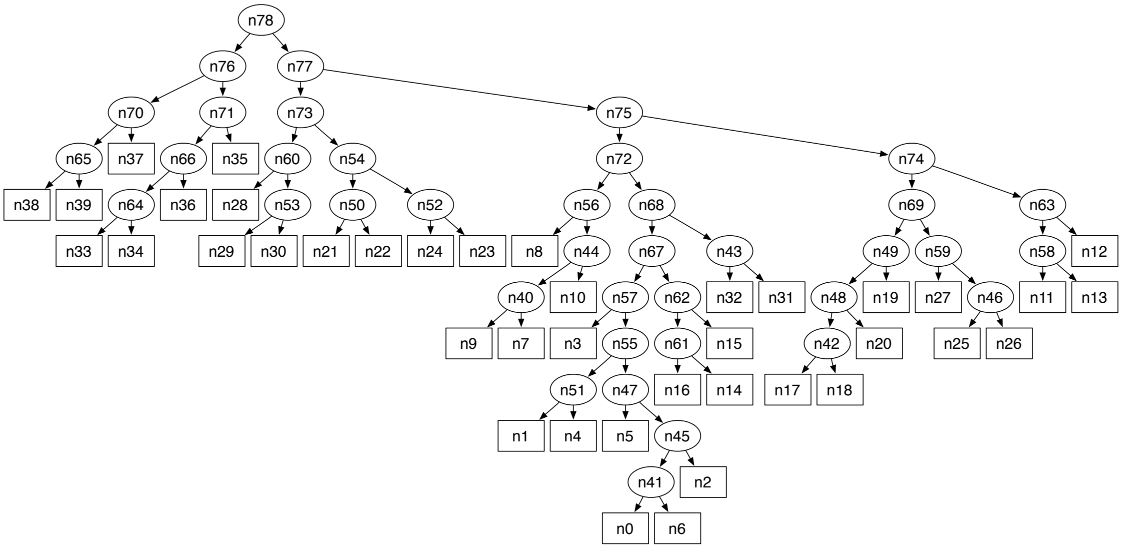
- True synthetic tree



- Constructed tree from Beta coalescent



- Constructed tree from Kingman's coalescent



Bibliography

Abney S, Light M (1999) Hiding a semantic hierarchy in a Markov model. In: Proceedings of the Workshop on Unsupervised Learning in Natural Language Processing, pp 1–8

Adams R, Ghahramani Z, Jordan M (2010) Tree-structured stick breaking for hierarchical data. In: Proceedings of Advances in Neural Information Processing Systems

Anandkumar A, Foster DP, Hsu D, Kakade S, Liu YK (2012a) A spectral algorithm for latent dirichlet allocation. In: NIPS, pp 926–934

Anandkumar A, Foster DP, Hsu D, Kakade SM, Liu YK (2012b) Two svds suffice: Spectral decompositions for probabilistic topic modeling and latent dirichlet allocation. CoRR abs/1204.6703

Anandkumar A, Hsu D, Kakade SM (2012c) A method of moments for mixture models and hidden markov models. In: Proceedings of Conference on Learning Theory

Andrzejewski D, Zhu X, Craven M (2009) Incorporating domain knowledge into topic modeling via Dirichlet forest priors. In: Proceedings of the International Conference of Machine Learning

Antoniak CE (1974) Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. The Annals of Statistics 2(6):1152–1174

Arora S, Ge R, Halpern Y, Mimno DM, Moitra A, Sontag D, Wu Y, Zhu M (2012a) A practical algorithm for topic modeling with provable guarantees. CoRR abs/1212.4777

Arora S, Ge R, Moitra A (2012b) Learning topic models — going beyond svd. CoRR abs/1204.1956

Artstein R, Poesio M (2005) Kappa3 = alpha (or beta). Technical report, University of Essex Department of Computer Science

Astrachan O (2003) Bubble sort: an archaeological algorithmic analysis. In: Proceedings of the 34th SIGCSE technical symposium on computer science education

Asuncion A, Welling M, Smyth P, Teh YW (2009) On smoothing and inference for topic models. In: Proceedings of Uncertainty in Artificial Intelligence

Bellegarda JR (2004) Statistical language model adaptation: review and perspectives. vol 42, pp 93–108

Bendapudi N, Leone RP (2003) Psychological implications of customer participation in co-production. Journal of Marketing 67(1):14–28

Berestycki N (2009) Recent progress in coalescent theory. In: Ensaios Matematicos, vol 16

Bergen J, Anandan P, Hanna K, Hingorani R (1992) Hierarchical model-based motion estimation

Bhattacharya I (2006) Collective entity resolution in relational data. PhD Dissertation, University of Maryland, College Park

Birkner M, Blath J, Steinrucken M (2011) Importance sampling for lambda-coalescents in the infinitely many sites model. Theoretical population biology 79(4):155–73

Blei DM, Lafferty JD (2005) Correlated topic models. In: Proceedings of Advances in Neural Information Processing Systems

Blei DM, Lafferty JD (2006) Dynamic topic models. In: Proceedings of the International Conference of Machine Learning

Blei DM, Griffiths TL, Jordan MI, Tenenbaum JB (2003a) Hierarchical topic models and the nested chinese restaurant process. In: Proceedings of Advances in Neural Information Processing Systems

Blei DM, Ng A, Jordan M (2003b) Latent Dirichlet allocation. Journal of Machine Learning Research 3

Blei DM, Griffiths TL, Jordan MI (2010) The nested Chinese restaurant process and Bayesian nonparametric inference of topic hierarchies. Journal of the ACM 57(2):7:1–7:30

Blundell C, Teh YW, Heller KA (2010) Bayesian rose trees. In: Proceedings of Uncertainty in Artificial Intelligence

Botha JA, Dyer C, Blunsom P (2012) Bayesian language modelling of German compounds. In: Proceedings of the 24th International Conference on Computational Linguistics (COLING)

Boyd-Graber J, Blei DM (2008) Syntactic topic models. In: Proceedings of Advances in Neural Information Processing Systems

Boyd-Graber J, Blei DM (2009) Multilingual topic models for unaligned text. In: Proceedings of Uncertainty in Artificial Intelligence

Boyd-Graber J, Resnik P (2010) Holistic sentiment analysis across languages: Multilingual supervised latent Dirichlet allocation. In: Proceedings of Emperical Methods in Natural Language Processing

Boyd-Graber J, Blei DM, Zhu X (2007) A topic model for word sense disambiguation. In: Proceedings of Emperical Methods in Natural Language Processing

Bron C, Kerbosch J (1973) Algorithm 457: finding all cliques of an undirected graph. Communications of the ACM 16(9):575–577

Brown P, Pietra VJD, deSouza PV, Lai JC, L MR (1992) Class-based n-gram models of natural language. Computational Linguistics 18(4):4467–479

Brown PF, Pietra VJD, deSouza PV, Lai JC, Mercer RL (1990) Class-based n-gram models of natural language. Computational Linguistics 18:18–4

Cappe O, Godsill S, Moulines E (2007) An overview of existing methods and recent advances in sequential Monte Carlo. PROCEEDINGS-IEEE 95(5):899

Carbone K (2012) Topic modeling: Confusion and excitement. Http://dh201.humanities.ucla.edu/?p=502

Ceaparu I, Lazar J, Bessiere K, Robinson J, Shneiderman B (2004) Determining causes and severity of end-user frustration. International journal of human-computer interaction 17(3):333–356

Chang J (2010) Not-so-latent Dirichlet allocation: Collapsed Gibbs sampling using human judgments. In: NAACL Workshop: Creating Speech and Language Data With Amazon'ss Mechanical Turk

Chang J, Boyd-Graber J, Wang C, Gerrish S, Blei DM (2009) Reading tea leaves: How humans interpret topic models. In: Proceedings of Advances in Neural Information Processing Systems

Chen SF, Goodman J (1996) An empirical study of smoothing techniques for language modeling. In: Proceedings of the Association for Computational Linguistics

Chen Z (2003) Bayesian filtering: From kalman filters to particle filters, and beyond. McMaster [Online]

Chiang D, DeNeefe S, Pust M (2011) Two easy improvements to lexical weighting. In: Proceedings of the Human Language Technology Conference

Cohen S, Stratos K, Collins M, Foster DP, Ungar L (2013) Experiments with spectral learning of latent-variable PCFGs. In: Conference of the North American Chapter of the Association for Computational Linguistics

Cohen SB, Smith NA (2009) Shared logistic normal distributions for soft parameter tying in unsupervised grammar induction. In: Conference of the North American Chapter of the Association for Computational Linguistics

Crammer K, Dekel O, Keshet J, Shalev-Shwartz S, Singer Y (2006) Online passive-aggressive algorithms. Journal of Machine Learning Research 7:551–585

Daumé III H (2007) Frustratingly easy domain adaptation. In: Proceedings of the Association for Computational Linguistics

Daumé III H (2009) Markov random topic fields. In: Proceedings of Artificial Intelligence and Statistics

Dean J, Ghemawat S (2004) MapReduce: Simplified data processing on large clusters. In: Symposium on Operating System Design and Implementation

Denisowski P (1997) CEDICT. Http://www.mdbg.net/chindict/

Dietz L, Bickel S, Scheffer T (2007) Unsupervised prediction of citation influences. In: Proceedings of the International Conference of Machine Learning

Ding W, Rohban MH, Ishwar P, Saligrama V (2013a) A new ceometric approach to latent topic modeling and discovery. In: IEEE International Conference on Acoustics, Speech, and Signal Processing

Ding W, Rohban MH, Ishwar P, Saligrama V (2013b) Topic discovery through data dependent and random projections. In: Proceedings of the International Conference of Machine Learning

Ding W, Rohban MH, Ishwar P, Saligrama V (2014) Efficient distributed topic modeling with provable guarantees. In: Proceedings of Artificial Intelligence and Statistics

Donoho D, Stodden V (2003) When does non-negative matrix factorization give correct decomposition into parts? MIT Press, p 2004

Doucet A, De Freitas N, Gordon N (eds) (2001) Sequential Monte Carlo methods in practice

Duda RO, Hart PE (1973) Pattern Classification and Scene Analysis. Wiley

Dudík M, Phillips SJ, Schapire RE (2004) Performance guarantees for regularized maximum entropy density estimation. In: Proceedings of Conference on Learning Theory

Dyer C, Lopez A, Ganitkevitch J, Weese J, Ture F, Blunsom P, Setiawan H, Eidelman V, Resnik P (2010) cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In: Proceedings of ACL System Demonstrations

Eads D (2007) Hierarchical clustering (scipy.cluster.hierarchy). SciPy

Eidelman V, Boyd-Graber J, Resnik P (2012) Topic models for dynamic translation model adaptation. In: Proceedings of the Association for Computational Linguistics

Eisenstein J, O'Connor B, Smith NA, Xing EP (2010) A latent variable model for geographic lexical variation. In: Proceedings of Emperical Methods in Natural Language Processing, pp 1277–1287

Evans P (2013) More fun with topic modeling. Http://mith.umd.edu/engl668k/?p=1595

Fearhhead P (1998) Sequential Monte Carlo method in filter theory. PhD Dissertation, University of Oxford

Felsenstein J (1973) Maximum-likelihood estimation of evolutionary trees from continuous characters. Am J Hum Genet 25(5):471–492

Finkel JR, Manning CD (2009) Hierarchical bayesian domain adaptation. In: Conference of the North American Chapter of the Association for Computational Linguistics, Morristown, NJ, USA

Foster G, Kuhn R (2007) Mixture-model adaptation for smt. In: Proceedings of the Second Workshop on Statistical Machine Translation

Galassi M, Davies J, Theiler J, Gough B, Jungman G, Booth M, Rossi F (2003) Gnu Scientific Library: Reference Manual. Network Theory Ltd.

Girvan M, Newman MEJ (2002) Community structure in social and biological networks. Proceedings of the National Academy of Sciences 99:7821–7826

Gordon N, Salmond D, Smith A (1993) Novel approach to nonlinear/non-Gaussian Bayesian state estimation. IEEE Proceedings F, Radar and Signal Processing 140(2):107–113

Görür D, Teh YW (2009) An efficient sequential Monte Carlo algorithm for coalescent clustering. In: Proceedings of Advances in Neural Information Processing Systems

Görür D, Boyles L, Welling M (2012) Scalable inference on Kingman's coalescent using pair similarity. Journal of Machine Learning Research 22:440–448

Griffiths TL, Steyvers M (2004) Finding scientific topics. Proceedings of the National Academy of Sciences 101(Suppl 1):5228–5235

Griffiths TL, Canini KR, Sanborn AN, Navarro DJ (2007) Unifying rational models of categorization via the hierarchical Dirichlet process. In: Proceedings of the 29th Annual Conference of the Cognitive Science Society

Gruber A, Rosen-Zvi M, Weiss Y (2007) Hidden topic Markov models. In: Artificial Intelligence and Statistics

Hall D, Jurafsky D, Manning CD (2008) Studying the history of ideas using topic models. In: Proceedings of Emperical Methods in Natural Language Processing

Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH (2009) The WEKA data mining software: An update. SIGKDD Explorations 11

Harary F (1969) Graph Theory. Addison-Wesley

Hasler E, Haddow B, Koehn P (2012) Sparse lexicalised features and topic adaptation for SMT. In: Proceedings of IWSLT

Heinrich G (2004) Parameter estimation for text analysis. Tech. rep., http://www.arbylon.net/publications/text-est.pdf

Heller KA, Ghahramani Z (2005) Bayesian hierarchical clustering. In: Proceedings of the International Conference of Machine Learning

Hoffman M, Blei DM, Bach F (2010) Online learning for latent Dirichlet allocation. In: NIPS

Hopcroft H, Tarjan R (1973) Algorithm 447: efficient algorithms for graph manipulation. Communications of the ACM 16(6):372–378

Hu Y, Boyd-Graber J (2012) Efficient tree-based topic modeling. In: Proceedings of the Association for Computational Linguistics

Hu Y, Boyd-Graber J, Satinoff B, Smith A (2013) Interactive topic modeling. Machine Learning Journal

Hu Y, Zhai K, Edelman V, Boyd-Graber J (2014) Polylingual tree-based topic models for translation domain adaptation. In: Proceedings of the Association for Computational Linguistics

Johnson M (2010) PCFGs, topic models, adaptor grammars and learning topical collocations and the structure of proper names. In: Proceedings of the Association for Computational Linguistics

Johnson M, Griffiths TL, Goldwater S (2007) Bayesian inference for PCFGs via Markov chain Monte Carlo. In: Conference of the North American Chapter of the Association for Computational Linguistics

Jongeneel C, Delorenzi M, Iseli C, Zhou D, Haudenschild C, Khrebtukova I, Kuznetsov D, Stevenson B, Strausberg R, Simpson A, Vasicek T (2005) An atlas of human gene expression from massively parallel signature sequencing (mpss). Genome Res 15:1007–1014

Kannan R, Salmasian H, Vempala S (2005) The spectral method for general mixture models. In: Proceedings of Conference on Learning Theory

Kingman JFC (1982) On the genealogy of large populations. Journal of Applied Probability 19:27–43

Knowles D, Ghahramani Z (2011) Pitman-Yor diffusion trees. In: Proceedings of Uncertainty in Artificial Intelligence

Koehn P (2004) Statistical significance tests for machine translation evaluation. In: Proceedings of Emperical Methods in Natural Language Processing

Koehn P (2009) Statistical Machine Translation. Cambridge University Press, URL `http://books.google.com/books?id=D21UAAAACAAJ`

Koehn P, Och FJ, Marcu D (2003) Statistical phrase-based translation. In: Conference of the North American Chapter of the Association for Computational Linguistics

Kogan S, Levin D, Routledge BR, Sagi JS, Smith NA (2009) Predicting risk from financial reports with regression. In: Conference of the North American Chapter of the Association for Computational Linguistics

Koo T, Carreras X, Collins M (2008) Simple semi-supervised dependency parsing. In: Proceedings of the Association for Computational Linguistics

Kuhn R, Chen B, Foster GF, Stratford E (2010) Phrase clustering for smoothing tm probabilities - or, how to extract paraphrases from phrase tables. In: COLING

Landauer TK, Littman ML (1990) Fully automatic cross-language document retrieval using latent semantic indexing. In: Proceedings of the 6 th Annual Conference of the UW Centre for the New Oxford English Dictionary

Landauer TK, McNamara DS, Marynick DS, Kintsch W (eds) (2006) Probabilistic Topic Models. Laurence Erlbaum

Lau JH, Grieser K, Newman D, Baldwin T (2011) Automatic labelling of topic models. In: Proceedings of the Association for Computational Linguistics, pp 1536–1545

Lau JH, Newman D, Baldwin T (2014) Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In: Proceedings of the European Chapter of the Association for Computational Linguistics

Lavine M (1992) Some aspects of Polya tree distributions for statistical modeling. The Annals of Statistics 20(3):1222–1235

Li W, Mccallum A (2006) Pachinko allocation: Dag-structured mixture models of topic correlations. In: International Conference on Machine Learning, pp 577–584

Li Fei-Fei, Perona P (2005) A Bayesian hierarchical model for learning natural scene categories. In: Computer Vision and Pattern Recognition

Lin WH, Wilson T, Wiebe J, Hauptmann A (2006) Which side are you on? identifying perspectives at the document and sentence levels. In: Proceedings of the Conference on Natural Language Learning (CoNLL)

Loper E, Bird S (2002) NLTK: the natural language toolkit. In: Tools and methodologies for teaching

Matsoukas S, Rosti AVI, Zhang B (2009) Discriminative corpus weight estimation for machine translation. In: Proceedings of Emperical Methods in Natural Language Processing

McCallum AK (2002) Mallet: A machine learning for language toolkit, http://www.cs.umass.edu/ mccallum/mallet

Meilă M (2007) Comparing clusterings—an information based distance. Journal of Multivariate Analysis 98(5):873–895

Miller GA (1990) Nouns in WordNet: A lexical inheritance system. International Journal of Lexicography 3(4):245–264

Mimno D, Wallach H, McCallum A (2008) Gibbs sampling for logistic normal topic models with graph-based priors. In: NIPS 2008 Workshop on Analyzing Graphs: Theory and Applications

Mimno D, Wallach H, Naradowsky J, Smith D, McCallum A (2009) Polylingual topic models. In: Proceedings of Emperical Methods in Natural Language Processing

Mimno D, Wallach H, Talley E, Leenders M, McCallum A (2011) Optimizing semantic coherence in topic models. In: Proceedings of Emperical Methods in Natural Language Processing

Mimno D, Hoffman M, Blei D (2012) Sparse stochastic inference for latent Dirichlet allocation. In: Proceedings of the International Conference of Machine Learning

Minka TP (2000) Estimating a dirichlet distribution. Tech. rep., Microsoft, http://research.microsoft.com/en-us/um/people/minka/papers/dirichlet/

Monroe BL, Colaresi MP, Quinn KM (2008) Fightin' Words: Lexical Feature Selection and Evaluation for Identifying the Content of Political Conflict. Political Analysis, Vol 16, Issue 4, pp 372-403, 2008

Neal RM (1993) Probabilistic inference using Markov chain Monte Carlo methods. Tech. Rep. CRG-TR-93-1, University of Toronto

Neal RM (1998) Annealed importance sampling. Technical report 9805, University of Toronto

Neal RM (2000) Markov chain sampling methods for Dirichlet process mixture models. Journal of Computational and Graphical Statistics 9(2):249–265

Neal RM (2003a) Density modeling and clustering using Dirichlet diffusion trees. Bayesian Statistics 7:619–629

Neal RM (2003b) Slice sampling. Annals of Statistics 31:705–767

Newman D, Karimi S, Cavedon L (2009) External evaluation of topic models. In: Proceedings of the Aurstralasian Document Computing Symposium

Newman D, Lau JH, Grieser K, Baldwin T (2010) Automatic evaluation of topic coherence. In: Conference of the North American Chapter of the Association for Computational Linguistics

Ng AY (2004) Feature selection, l1 vs. l2 regularization, and rotational invariance. In: Proceedings of the International Conference of Machine Learning

Nirenburg S (1989) Knowledge-based machine translation. Machine Translation 4:5–24

Norman DA (ed) (1993) Things That Make Us Smart: Defending Human Attributes In The Age Of The Machine. Addison-Wesley, Reading MA

Och F, Ney H (2003) A systematic comparison of various statistical alignment models. In: Computational Linguistics, vol 29(21), pp 19–51

Pang B, Lee L (2008) Opinion Mining and Sentiment Analysis. Now Publishers Inc

Papineni K, Roukos S, Ward T, Zhu WJ (2002) BLEU: a method for automatic evaluation of machine translation. In: Proceedings of the Association for Computational Linguistics, pp 311–318

Paul M, Girju R (2010) A two-dimensional topic-aspect model for discovering multifaceted topics. In: Association for the Advancement of Artificial Intelligence

Pennebaker JW, Francis ME (1999) Linguistic Inquiry and Word Count, 1st edn. Lawrence Erlbaum

Petterson J, Alex S, Caetano T, Buntine W, Shravan N (2010) Word features for latent Dirichlet allocation. In: Neural Information Processing Systems

Pitman J (1999) Coalescents with multiple collisions. The Annals of Probability 27:1870–1902

Powers DMW (1997) Unsupervised learning of linguistic structure an empirical evaluation. International Journal of Corpus Linguistics 2:91–131

Rai P, Daumé III H (2008) The infinite hierarchical factor regression model. In: Proceedings of Advances in Neural Information Processing Systems

Ramage D, Hall D, Nallapati R, Manning C (2009) Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In: Proceedings of Emperical Methods in Natural Language Processing

Rasmussen CE (2000) The infinite Gaussian mixture model. In: Proceedings of Advances in Neural Information Processing Systems

Rennie J (2003) On l2-norm regularization and the Gaussian prior

Resnik P, Hardisty E (2010) Gibbs sampling for the uninitiated. Tech. Rep. UMIACS-TR-2010-04, University of Maryland

Rosen-Zvi M, Griffiths TL, Steyvers M, Smyth P (2004) The author-topic model for authors and documents. In: Proceedings of Uncertainty in Artificial Intelligence

Sagitov S (1999) The general coalescent with asynchronous mergers of ancestral lines. Journal of Applied Probability 36:1116–1125

Salton G (1968) Automatic Information Organization and Retrieval. McGraw Hill Text

Sandhaus E (2008) The New York Times annotated corpus. Http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp? catalogId=LDC2008T19

Sayeed AB, Boyd-Graber J, Rusk B, Weinberg A (2012) Grammatical structures for word-level sentiment detection. In: North American Association of Computational Linguistics

Sethuraman J (1994) A constructive definition of Dirichlet priors. Statistica Sinica 4:639–650

Shamos M, Hoey D (1975) Closest-point problems. In: IEEE Symposium on Foundations of Computer Science

Shlens J (2005) A tutorial on principal component analysis. In: Systems Neurobiology Laboratory, Salk Institute for Biological Studies

Shneiderman B, Byrd D, Croft WB (1997) Clarifying search: A user-interface framework for text searches. D-Lib Magazine 3(1)

Shoemaker OJ (2011) Variance estimates for price changes in the consumer price index. Bureau of Labor Statistics Report

Shringarpure S, Xing EP (2008) mStruct: a new admixture model for inference of population structure in light of both genetic admixing and allele mutations. In: Proceedings of the International Conference of Machine Learning

Smith A, Chuang J, Hu Y, Boyd-Graber J, Findlater L (2014) Quantifying the role of discourse topicality in speakers' choices of referring expressions. In: ACL Workshop on Workshop on Interactive Language Learning, Visualization, and Interfaces

Snover M, Dorr B, Schwartz R, Micciulla L, Makhoul J (2006) A study of translation edit rate with targeted human annotation. In: In Proceedings of Association for Machine Translation in the Americas

Snow R, O'Connor B, Jurafsky D, Ng A (2008) Cheap and fast—but is it good? Evaluating non-expert annotations for natural language tasks. In: Proceedings of Emperical Methods in Natural Language Processing

Stevens K, Kegelmeyer P, Andrzejewski D, Buttler D (2012) Exploring topic coherence over many models and many topics. In: Empirical Methods in Natural Language Processing, vol 20

Su J, Wu H, Wang H, Chen Y, Shi X, Dong H, Liu Q (2012) Translation model adaptation for statistical machine translation with monolingual topic information. In: Proceedings of the Association for Computational Linguistics

Talley EM, Newman D, Mimno D, Herr BW, Wallach HM, Burns GAPC, Leenders AGM, McCallum A (2011) Database of NIH grants using machine-learned categories and graphical clustering. Nature Methods 8(6):443–444

Teh YW, Jordan MI, Beal MJ, Blei DM (2006) Hierarchical Dirichlet processes. Journal of the American Statistical Association 101(476):1566–1581

Teh YW, Daumé III H, Roy DM (2008) Bayesian agglomerative clustering with coalescents. In: Proceedings of Advances in Neural Information Processing Systems

Thomas JJ, Cook KA (2005) Illuminating the path: The research and development agenda for visual analytics. IEEE Computer Society Press

Tibshirani R (1994) Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society, Series B 58:267–288

Tseng H, Chang P, Andrew G, Jurafsky D, Manning C (2005) A conditional random field word segmenter. In: SIGHAN Workshop on Chinese Language Processing

Wacholder N, Liu L (2008) Assessing term effectiveness in the interactive information access process. Information Processing and Management 44(3):1022–1031

Wager S, Wang S, Liang P (2013) Dropout training as adaptive regularization. In: Proceedings of Advances in Neural Information Processing Systems, pp 351–359

Wallach H, Mimno D, McCallum A (2009) Rethinking LDA: Why priors matter. In: Proceedings of Advances in Neural Information Processing Systems

Wallach HM (2006) Topic modeling: Beyond bag-of-words. In: Proceedings of the International Conference of Machine Learning

Wang C, Blei DM, Heckerman D (2008) Continuous time dynamic topic models. In: Proceedings of Uncertainty in Artificial Intelligence

Wei X, Croft B (2006) LDA-based document models for ad-hoc retrieval. In: Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval

Wolfe J, Haghighi A, Klein D (2008) Fully distributed EM for very large datasets. In: Proceedings of the International Conference of Machine Learning, pp 1184–1191, DOI http://doi.acm.org/10.1145/1390156.1390305

Wood F, Teh YW (2009) A hierarchical nonparametric Bayesian approach to statistical language model domain adaptation. In: Proceedings of the International Conference on Artificial Intelligence and Statistics, vol 12

Wu X, Yu K, Wang H, Ding W (2010) Online streaming feature selection. In: International Conference on Machine Learning, pp 1159–1166

Xiao X, Xiong D, Zhang M, Liu Q, Lin S (2012) A topic similarity model for hierarchical phrase-based translation. In: Proceedings of the Association for Computational Linguistics

Yao L, Mimno D, McCallum A (2009) Efficient methods for topic model inference on streaming document collections. In: Knowledge Discovery and Data Mining

Yvart A, Hahmann S, Bonneau GP (2005) Hierarchical triangular splines. ACM Trans Graph 24(4):1374–1391

Zhai K, Boyd-Graber J, Asadi N, Alkhouja M (2012) Mr. LDA: A flexible large scale topic modeling package using variational inference in mapreduce. In: Proceedings of World Wide Web Conference

Zhao B, Xing EP (2006) BiTAM: Bilingual topic admixture models for word alignment. In: Proceedings of the Association for Computational Linguistics