Modeling Content and Users: Structured Probabilistic Representation and Scalable Online Inference Algorithms.

Amr Ahmed

CMU-LTI-10-018

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
www.lti.cs.cmu.edu

Thesis Committee:

Eric P. Xing (Chair)
John Lafferty
Tom Mitchell
Zoubin Ghahramani, University of Cambridge
Alexander J. Smola, Yahoo! Research

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
In Language and Information Technologies

MODELING CONTENT AND USERS: STRUCTURED PROBABILISTIC REPRESENTATION AND SCALABLE ONLINE INFERENCE ALGORITHMS

AMR AHMED

THESIS COMMITTEE:

Eric Xing, Chair
John Lafferty
Tom Mitchell
Zoubin Ghahramani, University of Cambridge
Alexander J. Smola, Yahoo! Research

Language Technology Institute
School of Computer Science
Carnegie Mellon University

Jan 2011

ABSTRACT

Online content have become an important medium to disseminate information and express opinions. With the proliferation of online document collections, users are faced with the problem of missing the big picture in a sea of irrelevant and/or diverse content. In this thesis, we addresses the problem of information organization of online document collections, and provide algorithms that create a structured representation of the otherwise unstructured content. We leverage the expressiveness of latent probabilistic models (e.g. topic models) and non-parametric Bayes techniques (e.g. Dirichlet processes), and give online and distributed inference algorithms that scale to terabyte datasets and adapt the inferred representation with the arrival of new documents. Throughout the thesis, we consider two different domains: research publications and social media (news articles and blog posts); and focus on modeling two facets of contnet: temporal dynamics and structural correspondence.

To model the temporal dynamics of document collections, we introduce a non-parametric Bayes model that we call the recurrent Chinese restaurant process (RCRP). RCRP is a framework for modeling complex longitudinal data, in which the number of mixture components at each time point is unbounded. On top of this process, we develop a hierarchical extension and use it to build an infinite dynamic topic model that recovers the timeline of ideas in research publications. Despite the expressiveness of the aforementioned model, it fails to capture the essential element of dynamics in social media: stories. To remedy this, we developed a multi-resolution model that treats stories as a first-citizen object and combines long-term, high-level topics with short-lived, tightly-focused storylines. Inference in the new model is carried out via a sequential Monte Carlo algorithm that processes new documents on real time.

We then consider the problem of structural correspondence in document collections both across modalities and communities. In research publications, this problem arises due to the multi-modalities of research papers and the pressing need for developing systems that can retrieve relevant documents based on any of these modalities (e.g. figures, text, named entities, to name a few). In social media this problem arises due to ideological bias of the document's author that mixes facts with opinions. For both problems we develop a series of factored models. In research publications, the developed model represents ideas across modalities and as such can solve the aforementioned retrieval problem. In social media, the model contrasts the same idea across different ideologies, and as such can explain the bias of a given document on a topical-level and help the user staying informed by providing documents that express alternative views.

Finally, we address the problem of inferring users' intent when they interact with document collections, and how this intent changes over time. The induced user model can then be used in matching users with relevant content.

ACKNOWLEDGMENTS

I wish to thank my advisor Eric Xing for believing in me, and providing me with the *right* amount of freedom through my PhD to explore various research problems while giving guidance and stimulating feedback. Eric taught me a lot of things: how to see the big picture, how to present my research and how to develop a taste for selecting the right problems to work on. Throughout my stay at the SAILING lab, I was really impressed with how dedicated Eric was to his research and to his students. Eric used to tell us that the goal of any advisor is to graduate students who can be at least as successful as their advisor – Eric set the bar very high for his students.

I was very lucky to have an amazing thesis committee comprising: John Lafferty, Tom Mitchell, Zoubin Ghahramani and Alex Smola. John and Tom insights helped me connect the dots between different parts of my thesis. They asked many questions that forced me to look at my work from different angles. In fact, the semi-supervised learning algorithm in Chapter 7 was a response to the question that they both asked independently: "how can you use unlabeled data in this model?". Zoubin was the first person who introduced me to Bayesian machine learning during his lectures at the Machine Learning Summer School in 2005 – a few months before I started my PhD program at CMU. Since then, I always seize the chance to meet him whenever he visits CMU every year to get feedback about my research and ideas. Zoubin always ask deep questions, and give very useful pointers to whatever problem I talked to him about. Alex, in addition to being a member of my thesis committee, was my collaborator on Chapters 5 and 8 of this thesis. I am very grateful to Alex for giving me the chance to visit Yahoo! Research as a summer intern. During this time, I was introduced to large-scale machine learning problems and played with large cluster of computers. Alex in addition to being a great mentor, was also a very good friend who was always generous on giving advises not only on science by also on other things like how to select a digital camera!

During my PhD, I was fortunate to have other senior mentors. I am very grateful for having the chance to collaborate with Bob Murphy and William Cohen on modeling and understating biological literature as part of the SLIF project – Chapter 6 was based on our collaboration. Outside CMU, it was a pleasure to work with Kai Yu on deep learning during my internship at NEC Research Labs.

I would like to thank all my collaborators and co-authors: Mohamed Aly, Andrew Arnold, Luis Pedro Coelho, William Cohen, Jacob Eisenstein, Yihong Gong, Qirong Ho, Vanja Josifovski, Joshua Kangas, Yuchen Low, Mladen Kolar, Robert F. Murphy, Ramesh Nallapati, Abdul-Saboor Sheikk, Alexander J. Smola, Le Song, Choon Hui Teo, Wei Xu, Eric P. Xing, Kai Yu and Jun Zhu. This thesis would not have been possible without you.

My friends in CMU and Pittsburgh deserve a special thanks. I would like to thank Yanxin Shi and Jon Elsas for being my first officemates, Pradipta Ray and Henry Lin for being my second officemates, and Manas Pathak for being an officemate and

a close friend in the last two years of my PhD (Switching to using Linux was his fault). Current (and former) members of the SAILING lab deserve special thanks for many stimulating discussions and fun parties: Ross Curtis, Jacob Eisenstein, Wenjie Fu,Anuj Goyal, Fan Guo, Steve Hanneke, Qirong Ho, Hetunandan Kamisetty, Seyoung Kim, Gunhee Kim, Mladen Kolar, Seunghak Lee, Henry Li, Andre Martins, Ankur Parikh, Kriti Puniyani, Pradipta Ray, Suyash Shringarpure, Kyung-Ah Sohn, Le Song, Selen Uguroglu, Matt Wytock, Jing Xiang, Wei Xu,Bin Zhao and Jun Zhu (I hope I didn't miss anyone). I would like also to thank all my other CMU and LTI friends. In addition, I would like to thank Ahmad Adam,Mohamed Noamany, Sherief Khattab, Khalid EL-Arini and Sajid Siddiqi for being good friends, among many others, during my stay at Pittsburgh.

Finally, I am deeply indebted to my dear mother and sister for their love and strong support during my graduate study. They have both been with me through the whole process with its ups and downs. While my father is not here to celebrate this moment with me, I would like to thank him for all his support and encouragement during my life – I am sure he would have been happy and proud of me.

CONTENTS

LIST OF TABLES

LISTINGS

ACRONYMS

Part I

INTRODUCTORY MATERIAL

1 INTRODUCTION

1.1 MOTIVATION AND THESIS STATEMENT

Online document collections have become an important medium to disseminate information and express opinions. Examples of online document collections are abound in different domains. For example, online conference proceedings, journal transactions, and the arxiv, to name a few, are currently the main source of information dissemination in the scientific community. In social media, news portals, and blogs are used on a daily basis to broadcast new stories and express opinions about mainstream events. With the proliferation of such collections, users are faced with the problem of missing the big picture in a sea of irrelevant and/or diverse content[1]. In this thesis, we addresses the problem of information organization of online document collections, and provide algorithms that create a structured representation of the otherwise unstructured collection. The induced structured representation can then be used in guiding the users during their information search episodes.

To endow the user with the complete big picture, we focus on modeling two important aspects of document collections: temporal dynamics, and structural correspondence. Documents in online collections are always time-stamped with the publication time. Models that ignore this time-stamp and feature a *static* representation prevent the users from zooming in and out through time to understand how the story they are looking for develop over time, or how the research topic they are interested in evolve across the years. Furthermore, documents are often accompanied with side information such as the author of a blog post, or an illustrative figure in a scientific paper. Models that feature a *flat* representation restrict the users to use only one modality in searching for relevant papers, or give users only one side of the coin by overwhelming them with stories written from a single ideology.

In this thesis, we leverage the expressiveness of latent probabilistic models (e.g. topic models) and non-parametric Bayes techniques (e.g. Dirichlet processes) to build a dynamic and multi-faceted representation of document collections. To illustrate the power, expressiveness and flexibility of these techniques, throughout the thesis, we apply our models to two different domains: research publications and social media (news articles and blog posts). Furthermore, we give online and distributed inference algorithms that scale to terabyte datasets and adapt the inferred representation with the arrival of new documents.

Finally, rather then passively relying on the user to find their information based on the induced representation, we address the problem of modeling users intents when they interact with document collections and how these intents change over time. The induced user model can then be used in pro-actively matching users with relevant content.

1 We use the terms document collection and content ex changeably in this thesis

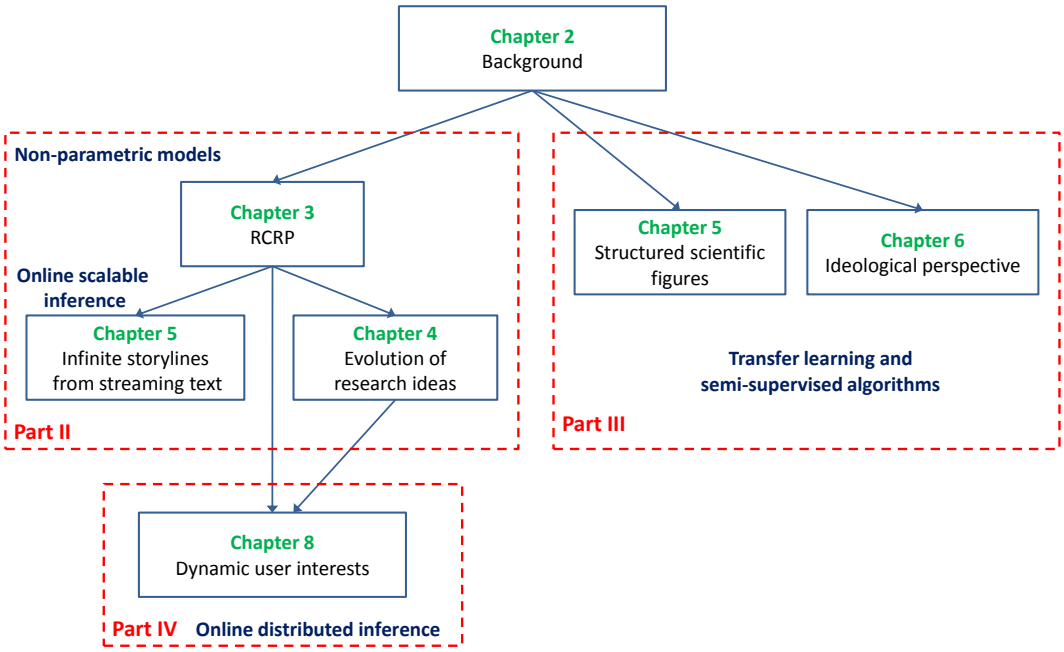


Figure 1.1: Dependency relationship between the content of this thesis

1.2 THESIS ORGANIZATION

After this introductory chapter, we review important and relevant concepts for this thesis in Chapter 2. The rest of this thesis is then divided into three parts: modeling dynamic content (Chapter 3,4 and 5), Modeling multi-faceted content (Chapter 6 and 7) and modeling users (chapter 8). The content of these chapters is summarized below and their inter-dependency is depicted in Figure 1.1.

- Part II: Modeling dynamic content: in this part we design dynamic non-parametric models of time-evolving content in both research publications and social media.
  - In Chapter 3, we introduce a *non-parametric Bayes* model that we call the recurrent Chinese restaurant process (RCRP). RCRP is a framework for modeling complex longitudinal data, in which the number of mixture components at each time point is unbounded. We illustrate the utility of this process via dynamic mixtures models in which documents are assumed to be generated from a single component.
  - In Chapter 4, we consider the problem of modeling research publications to understand the evolution of research ideas and identify paper that spawn new fields. To achieve this goal, we relax the assumption made in Chapter 3 and allow each document to address multiple topics using a dynamic hierarchical model built on top of the RCRP. We also give an efficient Gibbs sampling algorithm for this new process.
  - In Chapter 5, we give a unified model for streaming news that jointly solves the three major problems associated with news articles: clustering documents into stories, classifying stories into topics, and trend

analysis. We propose a multi-resolution model that treats stories as a first-citizen objects and combines long-term, high-level topics with short-lived, tightly-focused storylines. Inference in the new model is carried out via an *online scalable*, sequential Monte Carlo algorithm that processes new documents on real time (one document at a time).

- Part III: Modeling multi-faceted content: Textual content is always accompanied with illustrative figures (as in research publications) or tagged with the author's ideology (as in social media such as blogs). In this part we focus on modeling multi-faceted content and give models that can leverage unlabeled and partially labeled content using *transfer learning and semi-supervised inference algorithms*.
  - In Chapter 6, we address the problem of modeling figures in biological literature. In biological articles, a typical figure often comprises multiple panels, accompanied by either scoped or global captioned text. Moreover, the text in the caption contains important semantic entities such as protein names, gene ontology, tissues labels, etc., relevant to the images in the figure. We present a new structured probabilistic topic model built on a realistic figure generation scheme to model the structurally annotated biological figures. The resulting program constitutes one of the key IR engines in our SLIF system that has recently entered the final round (4 out 70 competing systems) of the Elsevier Grand Challenge on Knowledge Enhancement in the Life Science. In this chapter we also given a transfer learning framework that leverages the power of partially labeled data.
  - In Chapter 7, we address a similar problem to the one addressed in Chapter 7 but in the social media domain. The problem arises due to ideological bias of the document's author that mixes facts with opinions. We develop a factored model that contrasts the same idea across different ideologies, and as such can explain the bias of a given document on a topical-level and help the user staying informed by providing documents that express alternative views. We also give a semi-supervised inference algorithm that harnesses the power of unlabeled data.
- Part IV: Modeling users: the structured representation inferred by the models discussed in this thesis empower the user with the necessary tools to find what they are looking for and give users the big picture of dynamic, multi-faceted content. In this part we take a pro-active approach and model users' intents and how these intents change over time.
  - In Chapter 8, we design a multi-scale dynamic model for user modeling. The proposed model captures both the short-term and long-term intents of the user. Moreover, we design an *online, distributed* inference algorithm that scales to tens of million of users and adapts the inferred representation with the arrival of new users' interactions. The model presented in this Chapter utilizes a finite-dimensional approximation to the processes given in Part II.

1.3 THESIS CONTRIBUTION AND RELATED PUBLICATIONS

The contribution of this thesis, along with related publications, can be classified along the following three dimensions:

- Models:
  - The recurrent Chinese restaurant process for modeling complex longitudinal data[9].
- Inference algorithms:
  - Collapsed Gibbs sampling inference algorithms for variants of the (hierarchical) RCRP [9, 12].
  - An online sequential monto Carlo inference algorithm for a variant of the RCRP [4].
  - An online distributed inference algorithm for a fixed-dimension variant of the RCRP[5].
  - A tight approximate inference algorithm for non-conjugate admixture models [7]
  - A Metropolis-Hasting based semi-supervised learning algoirthm for inference in tightly-couple latent variable models[11]
- Applications:
  - Research Publications: Modeling the evolution of ideas [12] and scientific figures [13]
  - Social Media: unified analysis of news stories [5] and detection of politcal bias in the blogshpere [11]
  - Personalizations: time-sensitive and multi-scale learning of user interests which is then used in delivering personalized content [5].
  - Systems: an IR system for biological figures[2, 3, 38] which was finalist (4 out of 70 teams) of the Elsevier Grand Challenge on Knowledge Enhancement in the Life Science.

2

BACKGROUND

Clustering is a popular unsupervised technique used to explore and visualize a document collection. Out of the many available clustering models, the mixture of uni-grams model is most-often used for document clustering. This model assumes that each document is generated from a single component (cluster or topic)[1] and that each cluster is a uni-gram distribution over a given vocabulary. This assumption limits the expressive power of the model, and does not allow for modeling documents as a mixture of topics. For example, a document might address health and sport but the mixture of uni-gram model has to generate this document from a single component and as such the model can not discover salient, pure clusters.

Recently, mixed membership models, also known as admixture models, have been proposed to remedy the aforementioned deficiency of mixture models. Statistically, an object $\mathbf{w}$ is said to be derived from an *admixture* if it consists of a bag of elements, say $\{w_1, \ldots, w_N\}$, each sampled independently or coupled in some way, from a mixture model, according to an *admixing coefficient vector* θ, which represents the (normalized) fraction of contribution from each of the mixture component to the object being modeled. In a typical text modeling setting, each document corresponds to an object, the words thereof correspond to the elements constituting the object, and the document-specific admixing coefficient vector is often known as a *topic vector*. Since the *admixture formalism* enables an object to be synthesized from elements drawn from a mixture of multiple sources, it is also known as *mixed membership model* in the statistical community [48]. Special instances of admixture models have been used for population genetics [107], text modeling [29], and network mining [15]. In the text modeling case, which is the focus of this thesis, it is named as the latent Dirichlet allocation (LDA) model due to the choice of a Dirichlet distribution as the prior for the topic vector θ [29]. A precursor to mixed membership models is the probabilistic latent semantic analysis model [65] (pLSA), however, pLSA is not a fully generative models as it models the document topic vector as a parameter and as such it is susceptible to over-fitting [29].

Since mixed membership models are central to this thesis, we give a more in depth overview of topic models (pioneered by the LDA model) in section 2.1 and then in section 2.2 we explore Dirichlet processes and how they are used to implement mixed membership models with potentially infinite number of components.

1 Throughout the thesis, we will use the words topic, mixture and component interchangeably to denote the same concept.

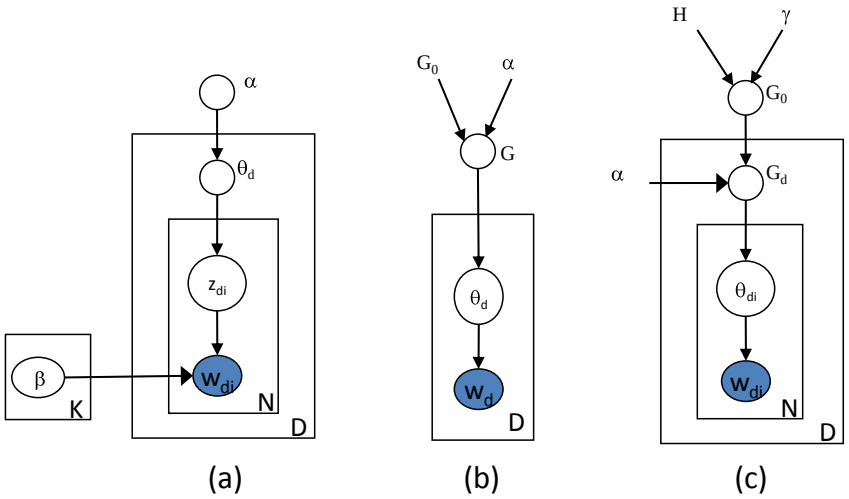


Figure 2.1: Basic mixture and (infinite) mixed membership models. (a) LDA, (b) DPM and (c) HDP mixture (an infinite mixed membership model)

2.1 TOPIC MODELS

In this section we review recent advances in topic models. Topic models is an are of research that was pioneered by the latent Dirichlet allocation model (LDA) [29]. The graphical representation of LDA is given in figure 2.1.(a). We first begin by detailing the generative process of LDA. LDA employs a semantic entity known as *topic* to drive the generation of the document in question. Each topic is represented by a topic-specific word distribution which is modeled as a multinomial distribution over words, denoted by Multi(β). The generative process of LDA proceeds as follows:

1. Draw topic proportions $\theta_d|\alpha \sim \text{Dir}(\alpha)$.
2. For each word
   (a) Draw a topic $z_{di}|\theta_d \sim \text{Mult}(\theta_d)$.
   (b) Draw a word $w_{di}|z_{di}, \beta \sim \text{Multi}(\beta_{z_{di}})$.

In step 1 each document d samples a topic-mixing vector θ_d from a Dirichlet prior. The component $\theta_{d,k}$ of this vector defines how likely topic k will appear in document d. For each word in the document w_{di}, a topic indicator z_{di} is sampled from θ_d, and then the word itself is sampled from a topic-specific word distribution specified by this indicator. Recent research in topic models seek to improve the expressiveness of one the following: topic vector, topic distribution, and document representation.

Much of the expressiveness of topic models lies in the choice of the prior for the topic vectors of documents. While the choice of a Dirichlet prior is mathematically convenient due to the conjugacy between the Dirichlet and the Multinomial distributions, it fails to capture non-trivial correlations among the elements of the topic vector θ. Several alternatives have been proposed to replace the Dirichlet

distribution by a Logistic normal distribution [26, 7], a directed acyclic graph [82] or a linear sparse Gaussian prior [109]. All of these models seek to uncover the structure of correlation between topics and as such can model the fact that two topics like sports and health are more likely to occur in a document as opposed to sports and politics. Another direction of research seeks to utilize the side-information of the document to influence the document's topic vector. For example, one would assume that all the documents written by a single author would address the same topics, and as such would have similar topic vectors. There are two main techniques to incorporate this intuition: upstream and downstream conditioning. In upstream models, the topic vector is sampled conditioned on the side information [91, 111, 77], while in downstream models, the side information is generated based on the topic vector of the document [28, 139]. Upstream models are often trained using the conditional maximum likelihood principle (CMLE), while downstream models can be trained either using the MLE principle [28] or the max-margin principle [139]. Finally, several authors proposed to use the inter-document relationships to smooth the topic vectors across related documents. For example, one would expect that if document A cites document B, then both documents are likely to speak about similar topics [95, 34].

Another direction of research seeks to improve the representation of the topic distribution β. Instead of using a uni-gram distribution, several authors proposed to use N-gram models [123, 128] to capture popular and salient phrases in each topic. On the other hands, other authors propose to keep the uni-gram distribution because its computational advantages and use statistical tests as a post-processing step to recover salient N-gram phrases for each topic [41]. Since the topic distributions are a corpus-wide parameters which are shared across all documents in the collection, several authors proposed to smooth those parameters across related collections. This relationship can encode temporal dependency [27, 126] or citation graph [125]. Finally several authors proposed to depart from the flat topic relationship, and arrange topics in a hierarchy [25, 24, 90].

Mostly, topic models are applied to documents represented as a bag of words, however, some recent work departed from this representation by representing documents as a sequence of words [123, 128], a sequence of phrases [34], a sequence of words and part of speech tags [57],or a set of dependency parse trees [30]. Several other applications of topic models consider documents represented by more than one modality such as text and images [22].

2.2 DIRICHLET PROCESSES

In section 2.1 we assumed that the number of topics (mixtures) in the model is fixed. Selecting the number of topics is a model selection problem that can be addresses using for instance cross validation. Alternatively, one can consider non-parametric techniques that adapt the number of topics to the data set at hand. The power of non-parametric techniques is not limited to model selection, but they endow the designer with necessary tools to specify priors over sophisticated (possibly infinite) structures like trees and sequences, and provide a principled way of learning these structures form data. We first give an overview of Dirichlet processes (DPs) and how they can be used to instantiate a clustering with infinite

number of components via the Dirichlet process mixture model (DPM) - DPM is the infinite equivalent of the mixture of uni-gram model. Then, we discuss Hierarchical Dirichlet processes (HDPs) and show how they can be used to instantiate an infinite mixed membership model.

The Dirichlet process (DP), see Figure 2.1.(b), is a distribution over distributions [50]. A DP denoted by $DP(G_0, \alpha)$ is parameterized by a base measure G_0 and a concentration parameter α. We write $G \sim DP(G_0, \alpha)$ for a draw of a distribution G from the Dirichlet process. G itself is a distribution over a given parameter space θ, therefore we can draw parameters $\theta_{1:N}$ from G. Integrating out G, the parameters θ follow a Polya urn distribution [19], also knows as the Chinese restaurant process (CRP), in which the previously drawn values of θ have strictly positive probability of being redrawn again, thus making the underlying probability measure G discrete with probability one. More formally,

$$\theta_i|\theta_{1:i-1}, G_0, \alpha \sim \sum_k \frac{m_k}{i-1+\alpha}\delta(\phi_k) + \frac{\alpha}{i-1+\alpha} G_0. \tag{2.1}$$

where $\phi_{1:k}$ denotes the distinct values among the parameters θ, and m_k is the number of parameters θ having value ϕ_k. By using the DP at the top of a hierarchical model, one obtains the Dirichlet process mixture model, DPM [16]. The generative process thus proceeds as follows:

$$G|\alpha, G_0 \sim DP(\alpha, G_0), \quad \theta_d|G \sim G, \quad \mathbf{w}_d|\theta_d \sim F(.|\theta_d), \tag{2.2}$$

where F is a given likelihood function parameterized by θ.

Instead of modeling each document $\mathbf{w}_d$ as a single data point, we could model each document as a DP. In this setting,each word w_{dn} is a data point and thus will be associated with a topic sampled from the random measure G_d, where $G_d \sim DP(\alpha, G_0)$. The random measure G_d thus represents the document-specific mixing vector over a potentially infinite number of topics. To share the same set of topics across documents, [116] introduced the Hierarchical Dirichlet Process (HDP). In HDP, see figure 2.1.(c), the document-specific random measures are tied together by modeling the base measure G_0 itself as a random measure sampled from a $DP(\gamma, H)$. The discreteness of the base measure G_0 ensures topic sharing between all the documents.

Integrating out all random measures, we obtain the equivalent Chinese restaurant franchise processes(CRF) [116]. In our document modeling setting, the generative story behind this process proceeds as follows. Each document is refereed to as a restaurant where words inside the document are referred to as customers. The set of documents shares a global menu of topics. The words in each document are divided into groups, each of which shares a table. Each table is associated with a topic (dish in the metaphor), and words sitting on each table are associated with the table's topic. To associate a topic with word w_{di} we proceed as follows. The word can sit on table b_{db} that has n_{db} words with probability $\frac{n_{db}}{i-1+\alpha}$, and shares the topic, ψ_{db} on this table, or picks a new table, b^{new} with probability $\frac{\alpha}{i-1+\alpha}$ and orders a new topic, $\psi_{db^{new}}$ sampled from the global menu. A topic ϕ_k that is used in m_k tables across all documents is ordered from the global menu with

probability $\frac{m_k}{\sum_{l=1}^{K} m_l+\gamma}$, or a new topic, k^{new} not used in any document, is ordered with probability $\frac{\gamma}{\sum_{l=1}^{K} m_l+\gamma}$, $\phi_{k^{new}} \sim H$. If we let θ_{di} denotes the distribution of the topic associated with word w_{di}, then putting everything together we have:

$$\theta_{di}|\theta_{d,1:i-1}, \alpha, \boldsymbol{\psi} \sim \sum_{b=1}^{b=B_d} \frac{n_{db}}{i-1+\alpha}\delta_{\psi_{db}} + \frac{\alpha}{i-1+\alpha}\delta_{\psi_{db^{new}}} \tag{2.3}$$

$$\psi_{db^{new}}|\boldsymbol{\psi}, \gamma \sim \sum_{k=1}^{K} \frac{m_k}{\sum_{l=1}^{K} m_l+\gamma}\delta_{\phi_k} + \frac{\gamma}{\sum_{l=1}^{K} m_l+\gamma} H \tag{2.4}$$

where B_d is the number of tables in document d. This gives a mixed membership model with potentially infinite number of topics and as such can be used to solve the model selection problem in LDA.

Part II

MODELING DYNAMIC CONTENT

3

THE RECURRENT CHINESE RESTAURANT PROCESS

3.1 INTRODUCTION

Dirichlet process mixture models provide a flexible Bayesian framework for estimating a distribution as an infinite mixture of simpler distributions that could identify latent classes in the data [97]. However the full exchangeability assumption they employ makes them an unappealing choice for modeling longitudinal data such as text, audio and video streams that can arrive or accumulate as epochs, where data points inside the same epoch can be assumed to be fully exchangeable, whereas across the epochs both the structure (i.e., the number of mixture components) and the parametrization of the data distributions can evolve and therefore unexchangeable. In this chapter, we present the temporal Dirichlet process mixture model (TDPM) as a framework for modeling these complex longitudinal data, in which the number of mixture components at each time point is unbounded; the components themselves can retain, die out or emerge over time; and the actual parametrization of each component can also evolve over time in a Markovian fashion. In the context of text-stream model, each component can thus be considered as a common themes or latent class that spans consecutive time points. For instance, when modeling the temporal stream of news articles on a, say, weekly basis, moving from week to week, some old themes could fade out (e.g., the mid-term election is now over in US), while new topics could appear over time (e.g., the presidential debate is currently taking place). Moreover, the specific content of the lasting themes could also change over time (e.g, the war in Iraq is developing with some slight shift of focus). The rest of this chapter is organized as follows. First we define the TDPM in Section 3.2 and give three different, and equivalent, constructions for this process: via the *recurrent Chinese restaurant process*, as the infinite limit of a finite dynamic mixture model, and finally via a temporally dependent random measures. In Section 3.3 we give a Gibbs sampling algorithm for posterior inference. Section 3.4 extends the construction to higher order dependencies. In Section 3.5 we use the TDPM to built 1 an infinite dynamic mixture of Gaussian factors (i.e., an infinite mixture Kalman filters of different life-spans) and illustrate it on simulated data. Then in section 3.6, we give a simple non-parametric topic model on top of the TDPM and use it to analyze the NIPS12 collection. In Section 3.7, we discuss relation to related work and in Section 3.8 we conclude and discuss possible future problems.

3.2 THE TEMPORAL DIRICHLET PROCESS MIXTURE MODEL

In many domains, data items are not fully exchangeable, but rather partially exchangeable at best. In the TDPM model[1] to be presented, data are assumed to arrive in T consecutive epochs, and inside the same epoch all objects are fully exchangeable.

1 A preliminary earlier version of the TDPM model first appeared in [19]

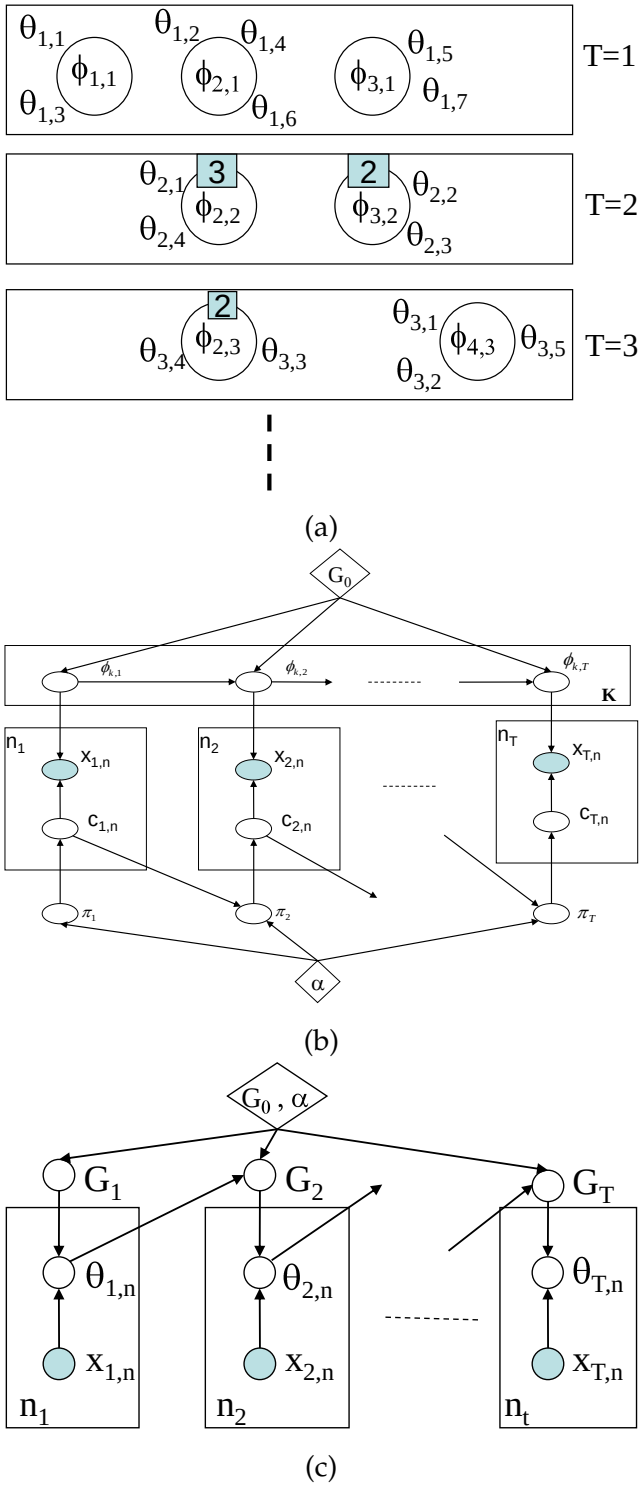


Figure 3.1: Three constructions for the TDPM. a) The recurrent Chinese restaurant process and b) The infinite limit of a finite dynamic mixture model. In b, diamonds represent hyper-parameters, shaded circles are observed variables, and unshaded ones are hidden variables, plates denote replications, where the number of replica is written inside the plate, for instance n_1. (C)Time dependent random measure construction of the TDPM. The direction of the arrows make it explicit that the random measure G_t at time t depends on the atom location on time $t-1$ as well as on the base measure G_0.

Intuitively the TDPM seek to model cluster parameters evolution over time using any time series model, and to capture cluster popularity evolution over time via the rich-gets-richer effect, i.e. the popularity of cluster k at time t is proportionable to how many data points were associated with cluster k at time $t-1$. In the following subsections, we will formalize these notions by giving three equivalent constructions for the TDPM as summarized in Figure 3.1. However, before giving these constructions that parallel those given for the DPM in section 3, we first start by specifying some notations.

3.2.1 *Notations and Conventions:*

We let n_t denotes the number of data points in the t^{th} epoch, and $x_{t,i}$ denotes the i^{th} point in epoch t. The mixture components (clusters) that generate the data can emerge, die out, or evolve its parametrization evolve over time in a Markovian fashion, therefore, we generalize the notion of a mixture into a chain that links the parameters of the mixture component over time. We let ϕ_k denote chain k, $\phi_{k,t}$ denoted the state (parameter value) of chain k at time t, and $n_{k,t}$ denotes the number of data points associated with chain k at time t. Moreover, we use $n_{k,t}^{(i)}$ to denote the same quantity just before the arrival of datum $x_{t,i}$. Note that the chains need not have the same life span; however, once retained over time they keep the same chain index. Moreover, the set of chain indexes available at time t might not be contiguous (because some chains may have died out). Therefore, we define I_t to denote the set of chain indexes available at time t. We sometimes overload notation and use $I_t^{(i)}$ to denote the same quantity just before the arrival of datum $x_{t,i}$. Each data item $x_{t,i}$ is generated from a mixture with parameter $\theta_{t,i}$, if we let $c_{t,i}$ denotes the chain index associated with this datum, then we have $\theta_{t,i} = \phi_{c_{t,i},t}$ — in other words, the set of ϕ's define the unique mixtures/clusters, or put it equivalently, two data items might have equal θ values if they belong to the same cluster. Moreover, we might use the following abbreviations for notational simplicity (z denotes a generic variable):

- $\{z_{t,.}\}$ to denote $\{z_{t,1}, z_{t,2}, \cdots\}$
- $z_{t,1:i}$ to denote $\{z_{t,1}, z_{t,2}, \cdots, z_{t,i}\}$

3.2.2 *The Recurrent Chinese Restaurant Process*

The RCRP depicted in Figure 3.1-a, is a generalization of the CRP introduced in Chapter 2.2. The RCRP operates in epochs, say, days. Customers entered the restaurant in a given day are not allowed to stay beyond the end of this day. At the end of each day, the consumptions of dishes are analyzed by the owner of the restaurant who assumes that popular dishes will remain popular in the next day, and uses this fact to plan the ingredients to be bought, and the seating plan for the next day. To encourage customers in the next day to try out those pre-planed dishes, he records on each table the dish which was served there, as well as the number of customers who shared it. As another incentive, he allows the first customer to set on such a table to order a (flavored) variation of the dish recorded there. In

this metaphor, dishes correspond to chains, and the variation correspond to the dynamic evolution of the chain. The generative process proceeds as follows. At day t, customer i can pick an empty table, k, that was used to serve dish $\phi_{k,t-1}$, with probability equals to $\frac{n_{k,t-1}}{N_{t-1}+i+\alpha-1}$, he then chooses the current flavor of the dish,$\phi_{k,t}$, distributed according to $\phi_{k,t} \sim P(.|\phi_{k,t-1})$. If this retained table k has already $n^{(i)}_{k,t}$customers, then he joins them with probability $\frac{n_{k,t-1}+n^{(i)}_{k,t}}{N_{t-1}+i+\alpha-1}$ and shares the current flavor of the dish there. Alternatively, he can pick a *new empty* table that was not used in the previous day,$t-1$, i.e., not available in I_{t-1}, with probability $\frac{\alpha}{N_{t-1}+i+\alpha-1}$,lets call it K^+, and orders a dish $\phi_{K^+,t} \sim G_0$ — this is the mechanism by which a new chain/cluster emerges. Finally, he can share a *new* table k, with $n^{(i)}_{k,t}$ customers, with probability $\frac{n^{(i)}_{k,t}}{N_{t-1}+i+\alpha-1}$ and shares the newly ordered dish with them. Putting everything together, we have:

$$\theta_{t,i}|\{\theta_{t-1,.}\}, \theta_{t,1:i-1}, G_0, \alpha \sim \frac{1}{N_{t-1}+i+\alpha-1} \times \Big[\sum_{k\in I_{t-1}} \Big(n_{k,t-1}+n^{(i)}_{k,t}\Big)\delta(\phi_{k,t}) + \sum_{k\in I^{(i)}_t - I_{t-1}} n^{(i)}_{k,t}\delta(\phi_{k,t}) + \alpha G_0\Big] \tag{3.1}$$

where in the first summation $\phi_{k,t} \sim P(.|\phi_{k,t-1})$ (i.e. retained from the previous day), and in the second one $\phi_{k,t} \sim G_0$ which is drawn by the j^{th} customer at time t for some $j < i$ (i.e. new chains born at epoch t). If we conveniently define $n_{k,t}$ to be 0 for $k \in I_{t-1} - I^{(i)}_t$ (chains which died out) and similarly $n_{k,t-1}$ be 0 for $k \in I^{(i)}_t - I_{t-1}$ (i.e. newly born chains at time t), then we can compactly write Equation 3.1 as:

$$\theta_{t,i}|\{\theta_{t-1,.}\}, \theta_{t,i:i-1}, G_0, \alpha \sim \frac{1}{N_{t-1}+i+\alpha-1} \times \Big[\sum_{k\in I_{t-1}\cup I^{(i)}_t} \Big(n_{k,t-1}+n^{(i)}_{k,t}\Big)\delta(\phi_{k,t}) + \alpha G_0\Big] \tag{3.2}$$

3.2.3 *The infinite Limit of a finite Dynamic Mixture Model*

In this section we show that the same sampling scheme in Equation (3.2) can be obtained as the infinite limit of the finite mixture model in Figure 3.1-b. We consider the following generative process for a finite dynamic mixture model with K mixtures. For each t do:

1. $\forall k$: Draw $\phi_{k,t} \sim P(.|\phi_{k,t-1})$
2. Draw $\pi_t \sim Dir(n_{1,t-1}+\alpha/K, \cdots, n_{K,t-1}+\alpha/K)$
3. $\forall i \in N_t$ Draw $c_{t,i} \sim Multi(\pi_t)$, $x_{t,i} \sim F(.|\phi_{c_{t,i},t})$

By integrating over the mixing proportion π_t, It is quite easy to write the prior for $c_{t,i}$ as conditional probability of the following form:

$$P(c_{t,i} = k | c_{t-1,1:N_{t-1}}, c_{t,1:i-1}) = \frac{n_{k,t-1} + n_{k,t}^{(i)} + \alpha/K}{N_{t-1} + i + \alpha - 1}. \tag{3.3}$$

If we let $K \to \infty$, we find that the conditional probabilities defining the $c_{t,i}$ reachs the following limit:

$$\begin{aligned} P(c_{t,i} = k | c_{t-1,1:N_{t-1}}, c_{t,1:i-1}) &= \frac{n_{k,t-1} + n_{k,t}^{(i)}}{N_{t-1} + i + \alpha - 1} \\ P(c_{t,i} = \texttt{a new cluter}) &= \frac{\alpha}{N_{t-1} + i + \alpha - 1} \end{aligned} \tag{3.4}$$

Putting Equations (3.3) and (3.4) together, we can arrive at Equation (3.2).

3.2.4 *The Temporarily Dependent Random Measures view of the TDPM*

Here we show that the same process in Section 3.2 can be arrived at if we model each epoch using a DPM and connect the random measures G_t as shown in Figure 5. This appendix is rather technical and is provided only for completeness, however it can be skipped without any loss of continuity.

The derivation here depends on the well known fact that the posterior of a DP is a also a DP [50]. That is, $G|\phi_1, \cdots, \phi_k, G_0, \alpha \sim DP\Big(\alpha + n, \sum_k \frac{n_k}{n+\alpha}\delta(\phi_k) + \frac{\alpha}{n+\alpha}G_0\Big)$, where $\{\phi_k\}$ are the collection of unique values of $\theta_{1:n}$ sampled from G. Now, we consider the following generative process. For each t, do:

1. $\forall k \in I_{t-1}$ draw $\phi_{k,t} \sim P(.|\phi_{k,t-1})$
2. Draw $G_t|\{\phi_{k,t}\}\forall k \in I_{t-1}, G_0, \alpha \sim DP(\alpha + N_{t-1}, G_0^t)$
3. $\forall i \in N_t$, Draw $\theta_{t,i}|G_t \sim G_t \qquad x_{t,n}|\theta_{t,n} \sim F(.|\theta_{t,n})$

where $G_0^t = \sum_{k \in I_{t-1}} \frac{n_{k,t-1}}{N_{t-1}+\alpha}\delta(\phi_{k,t}) + \frac{\alpha}{N_{t-1}+\alpha}G_0$. Now by integrating $G_t \sim DP(N_{t-1} + \alpha, G_0^t)$. We can easily show that:

$$\begin{aligned} \theta_{t,i}|\{\theta_{t-1,.}\}, \theta_{t,1:i-1}, G_0, \alpha \quad \sim \quad & \frac{1}{i + (\alpha + N_{t-1}) - 1} \times \\ & \Big[\sum_{k \in I_t^{(i)}} n_{k,t}^{(i)}\delta(\phi_{k,t}) + (\alpha + N_{t-1})G_0^t\Big] \end{aligned} \tag{3.5}$$

Now substituting G_0^t into the above equation plus some straightforward algebra, we arrive at:

$$\begin{aligned} \theta_{t,i}|\{\theta_{t-1,.}\}, \theta_{t,1:i-1}, G_0, \alpha \quad \sim \quad & \frac{1}{N_{t-1} + i + \alpha - 1} \times \\ & \Big[\sum_{k \in I_t^{(i)}} n_{k,t}^{(i)}\delta(\phi_{k,t}) + \sum_{k \in I_{t-1}} n_{k,t-1}\delta(\phi_{k,t}) + \alpha G_0\Big] \end{aligned} \tag{3.6}$$

which when rearranged is equivalent to Equation 3.2

3.3 GIBBS SAMPLING ALGORITHMS

Given the previous constructions for the TDPM model, we are ready to derive a Gibbs sampling scheme equivalent to algorithm 2 in [97]. The state of the sampler contains both the chain indicator for every data item, $\{c_{t,i}\}$, as well as the value of all the available chains at all time epochs, $\{\phi_{k,t}\}$. We iterate between two steps: given the current state of the chains, we sample a class indicator for every data item, and then given the class indicators for all data item, we update the current state of the chains. We begin by the second step, let $\phi_k^{(x)}$ denote the collection of data points associated with chain k at all time steps, that is $\phi_k^{(x)} = \{\forall t(\forall i \in N_t)\, x_{t,i} | c_{t,i} = k\}$. Note also that conditioning on the class indicators, each chain is conditionally independent from the other chains. Therefore, $P(\phi_k|\{c_{t,i}\}) = P(\{\phi_{k,t}\}|\phi_k^{(x)})$. This calculation depends on both the chain dynamic evolution model $P(.|)$ and the data likelihood $F(.|.)$, therefore, this posterior should be handled in a case by case fashion, for instance, when the dynamic evolution model is a linear state-space model with Gaussian emission (likelihood), this posterior can be calculated exactly via the RTS smoother [72]. Once this posterior is calculated, we can update the current state of the chains by sampling each chain over time as a block from this posterior. Now, we proceed to the first step, for a given data point, $x_{t,i}$, conditioning on the state of the chains and other indicator variables (i.e. how data points other than $x_{t,i}$ are assigned to chains), we sample $c_{t,i}$ as follows:

$$\begin{aligned} &P(c_{t,i}|c_{t-1}, c_{t,-i}, c_{t+1}, x_{t,i}, \{\phi_k\}_{t,t-1}, G_0, \alpha) \propto \\ &P(c_{t,i}|c_{t-1}, c_{t,-i}, x_{t,i}, \{\phi_k\}_{t,t-1}, G_0, \alpha) P(c_{t+1}|c_t), \end{aligned} \tag{3.7}$$

where we introduce the following abbreviations: c_{t-1}, c_{t+1} denotes all indicators at time $t-1$ and t respectively. $c_{t,-i}$ denotes the chain indicators at time t without $c_{t,i}$, and $\{\phi_k\}_{t,t-1}$ denotes all chains alive at either time epoch t or $t-1$, i.e., $\phi_k \forall k \in I_{t-1} \cup I_t$. We also let $n_{k,t}^{(-i)}$ denote $n_{k,t}$ without the contribution of data point $x_{t,i}$. The first factor in Equation (3.7) can be computed using Eq. (3.2) as follows:

$$\begin{aligned} P(c_{t,i} = k \in I_{t-1} \cup I_t|...) &\propto \frac{n_{k,t-1} + n_{k,t}^{(-i)}}{N_{t-1} + N_t + \alpha - 1} F(x_{t,i}|\phi_{k,t}) \\ P(c_{t,i} = K^+|...) &\propto \frac{\alpha}{N_{t-1} + N_t + \alpha - 1} \int F(x_{t,i}|\theta) dG_0(\theta), \end{aligned} \tag{3.8}$$

where K^+ denotes a globally new chain index (i.e., a new chain is born). It should be noted here that in the first part of Equation (3.8), there is a subtlety that we glossed over. When we consider a chain from time $t-1$ that has not been inherited yet at time t (that is $k \in I_{t-1}, n_{k,t}^{(-i)} = 0$), we must treat it exactly as we treat sampling a new chain from G_0 with G_0 replaced by $P(\phi_{k,t}|\phi_{k,t-1})$.

The second factor in Equation (3.7) can be computed with reference to the construction in section 4.3 as follows (note that c_t here includes the current value of $c_{t,i}$ under consideration in Equation (3.7). First, note that computing this part is equivalent to integrating over the mixture weights π_{t+1} which depend on the

counts of the chain indicators at time t. The subtlety here is that in section 4.3 we let $K \to \infty$; however, here we only need to let the two count vectors be of equal size, which is $K_{t,t+1} = |I_{t,t+1}|$, where as defined before $I_{t,t+1} = I_t \cup I_{t+1}$, by padding the counts of the corresponding missing chains with zeros. It is straightforward to show that:

$$P(c_{t+1}|c_t) = \frac{\Gamma\left(\sum_{k\in I_{t,t+1}} n_{k,t} + \alpha/K_{t,t+1}\right)}{\prod_{k\in I_{t,t+1}} \Gamma(n_{k,t} + \alpha/K_{t,t+1})} \times \frac{\prod_{k\in I_{t,t+1}} \Gamma(n_{k,t} + n_{k,t+1} + \alpha/K_{t,t+1})}{\Gamma\left(\sum_{k\in I_{t,t+1}} n_{k,t} + n_{k,t+1} + \alpha/K_{t,t+1}\right)} \tag{3.9}$$

It should be noted that the **cost** of running a full Gibs iteration is $O(n)$ where n is the total number of data points.

3.4 MODELING HIGHER-ORDER DEPENDENCIES

One problem with the above construction of the TDPM is that it forgets too quickly especially when it comes to its ability to model cluster popularity at time $t+1$ based on its usage pattern at time t, while ignoring all previous information before time epoch t. Moreover, once a cluster is dead, i.e. its usage pattern at time t is 0, it can no longer be revived again. Clearly, in some applications one might want to give a slack for a cluster before declaring it dead. For example, when the TDPM is used to model news stories on a daily basis, if a theme that was active in time epoch $t-1$ had no documents associated with it at time t, then the TDPM will consider it dead, however, in practice, this might not be the case.

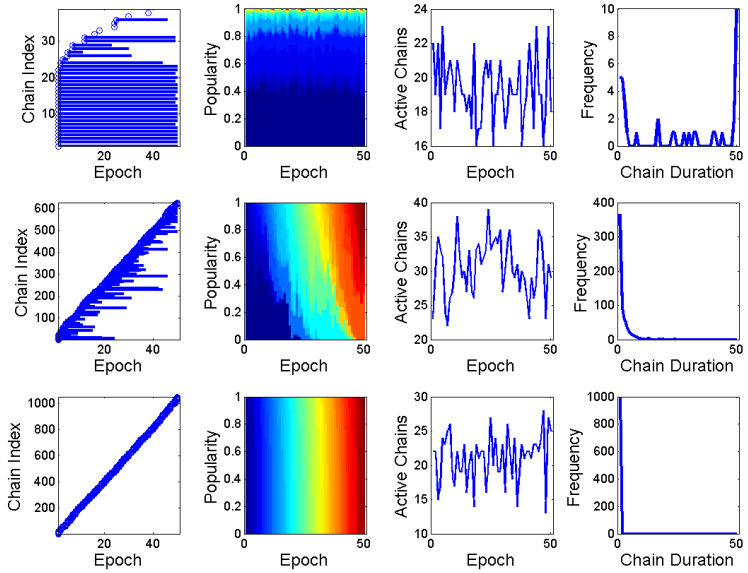


Figure 3.2: Simulating various clustering patterns from a TDPM(α,λ,W). **Top**: DPM, **middle**: a TDPM and **bottom**: a set of independent DPM at each epoch. See section 6 for more details

By analogy to the RCRP equivalent construction, the owner who plans the restaurant ingredients based on a daily usage is less prudent than an owner who considers a larger time frame, perhaps a week. However, one should not treat the usage pattern of cluster k at time t and at time, say, $t-h$, as contributing equally to our prediction of this cluster's popularity at time $t+1$. A possible solution here is to incorporate historic usage patterns by *decaying* their contribution exponentially over time epochs. A similar idea has been proposed in [55], however in [55], each epoch has exactly one data point, and the width of the history window used in [55] is rather infinity — or more precisely at most n. This in fact makes the cost of running a single Gibbs iteration, i.e. sampling all data items once, $O(n^2)$. In the solution we propose here, we define two new hyperparmaters, kernel width, λ, and history size, W. We will describe our approach only using the RCRP for simplicity since as shown before, it is equivalent to the other constructions. To model higher order dependencies, the only difference is that the owner of the restaurant records on each table, not only its usage pattern on day $t-1$, but its weighted cumulative usage pattern over the last W days. Where the weight associated with the count from day $t-h$ is given by $\exp^{\frac{-h}{\lambda}}$, and as such the contribution from epoch $t-h$ decays exponentially over time. A customer $x_{t,n}$ entering the restaurant at time t will behave exactly in the same way as before using the new numbers recorded on the table.

There are two implications to this addition. First, the cost of running one Gibbs iteration is $O(n \times W)$, which is still manageable as W must be smaller than T, the number of epochs, which is in turn much smaller than the total number of data points, n, thus we still maintain a linear time complexity. Second, an active cluster is considered dead if and only if, it is not used for exactly W contiguous echoes, which creates the necessary slack we were looking for. Changing the Gibbs sampling equations in section 5 to accommodate this new addition is very *straightforward* and removed for the light of space.

It is interesting to note that these two new hyper-parameters allow the TDPM to degenerate to either a set of independent DPMs at each epoch when W=0, and to a global DPM, i.e ignoring time, when $W = T$ and $\lambda = \infty$. In between, the values of these two parameters affect the expected life span of a given cluster/chain. The larger the value of W and λ, the longer the expected life span of chains, and vice versa.

To illustrate this phenomenon, we sampled different cluster configurations from the TDPM model by running the RCRP metaphor for T = 50 epochs and seating 300 customers at each epoch. We simulated three hyper-parameter configurations (α, λ, W) as follows. The configuration used at the top of Figure 3.2 is (5,∞,50=T) which reduces the TDPM to a DPM. The configuration at the middle is a TDPM with hyperparmaters (5,.4,4), while the bottom TDPM degenerates to a set of independent DPMs at each epoch by setting the hyper-parameters to (5,.5,0) — in fact the value of λ here is irrelevant. For each row, the first panel depicts the duration of each chain/cluster,the second panel shows the popularity index at each epoch, i.e. each epoch is represented by a bar of length one, and each *active* chain is represented by a color whose length is proportional to its popularity at this epoch. The third panel gives the number of active chains at each epoch and the fourth panel shows the number of chains with a given life-span (duration). This

fourth panel is a frequency curve and in general all TDPMs exhibit a power-law (Zipf's) distribution as the one in the middle, but with different tail lengths, while a DPM and independent DPMs show no such power-law curves. Another interesting observation can be spotted in the second column: note how cluster intensities change smoothly over time in the TDPM case, while it is abrupt in independent DPMs or rarely changing in a global DPM. This shows that TDPM with three tunable variables can capture a wide range of clustering behaviors.

3.5 INFINITE DYNAMIC MIXTURE OF GAUSSIAN FACTORS

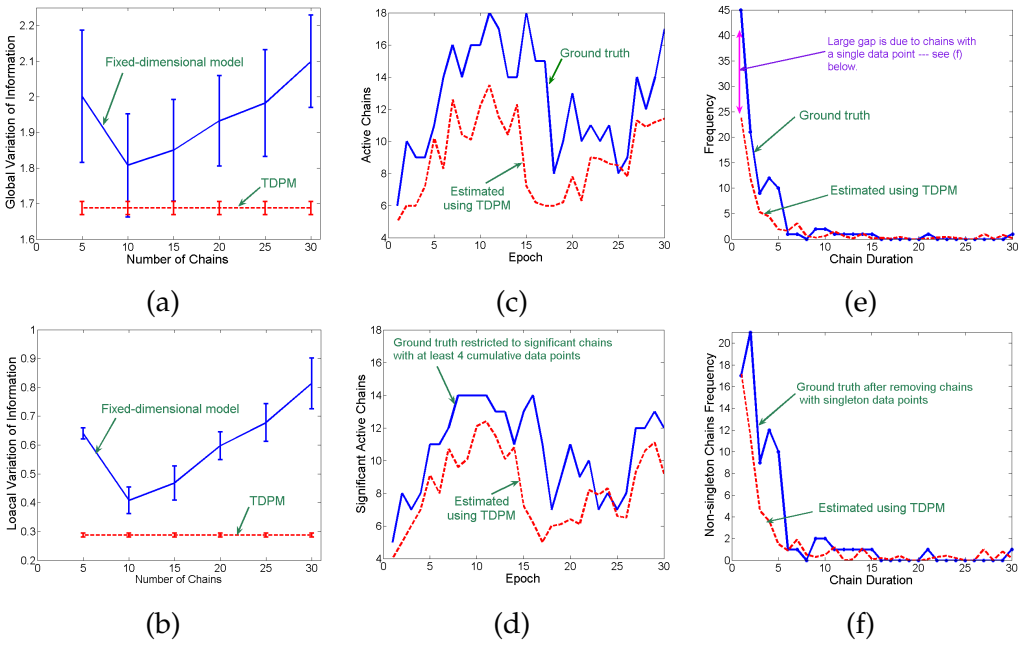


Figure 3.3: Illustrating results on simulated data. Panels (a,b) contrast the accuracy of the recovered clustering, using global and local consistency measures, against that estimated using fixed dimensional models (see text for details). Panels (c-f) illustrate the TDPM ability to vary the number of clusters/chains over time, results from fixed-dimensional models, which is fixed over time, are not shown to avoid cluttering the display. Panel (d) and (f) illustrate that most omissions (errors) are due to insignificant chains. All results are averaged over 10 samples taken 100 iterations apart for the TDPM, and over 10 random initializations for the fixed-dimensional models. Error bars are not shown in panels (c-f) for clarity, however,the maximum standard error is 1.4

.

In this section we show how to use the TDPM model to implement an infinite dynamic mixture of Gaussian factors. We let each chain represent the evolution of the mean parameter of a Gaussian distribution with a fixed covariance Σ. The chain dynamics is taken to be a linear state-space model, and for simplicity, we reduce it to a random walk. More precisely, for a given chain ϕ_k: $\phi_{k,t}|\phi_{k,t-1} \sim N(\phi_{k,t-1}, \rho I)$ and $x_{t,i}|c_{t,i} = k \sim N(\phi_{k,t}, \Sigma)$. The base measure $G_0 = N(0, \sigma I)$. Using the Gibbs sampling algorithm in section 5, computing the chain posterior given its associated data points, $\phi_k^{(x)}$, can be done exactly using the RTS smoother algorithm [73]. We

simulated 30 epochs, each of which has 100 points from the TDMP with the above specification, and with hyperparmaters as follows: $\alpha = 2.5, W = 1, - = .8, ff = 10, J = 0.1$and $^{\circ} = 0.1I$. We ran Gibbs sampling for 1000 iterations and then took 10 samples every 100 iterations for evaluations. The results shown in Figure 3 are averaged over these 10 samples. To measure the success of the TDPM, we compared the clustering produced by the TDPM to the ground truth, and to that produced from a fixed dynamic mixture of Kalman Filters [72] with various number of chains, K =(5,10,15,20,25,30). For each K, we ran 10 trials with different random initializations and averaged the results.

We compared the clustering produced by the two methods, TDPM, and the one with fixed number of *evolving* chains over time, to the ground truth using the variation of information measure in [88]. This measure uses the mutual information between the two clustering under consideration, and their entropy to approximate the distance between them across the lattice of all possible clustering (see [88] for more details). We explored two ways of applying this measure to dynamic clustering, the global variation of information, GVI, and the local variation of information, LVI. In GVI, we ignored time, and considered two data points to belong to the same cluster if they were generated from the same chain at any time point. In LVI, we applied the VI measure at each time epoch separately and averaged the results over epochs. GVI captures global consistency of the produced clustering, while LVI captures local consistency (adaptability to changes in the number of clusters). The results are shown in Figure 3.3-a, 3.3-b (lower values are better)and show that the TDPM is superior to a model in which the number of clusters are fixed over time, moreover, setting K to the maximum number of chains over all time epochs does not help. In addition to these measures, we also examined the ability of the TDPM to track the evolution of the number of chains (Figure 3.3-c)and their duration over time (Figure 3.3-e). These two figures show that in general, the TDPM tracks the correct ground-truth behavior, and in fact most of the errors are due to insignificant chains, i.e. chains/clusters which contain a very small (1-3) data points as shown in Figure 3.3-d and Figure 3-f. It is worth mentioning that the fixed dimension models produce the same number of chains over time, which we omit from Figure 3.3-(c-f) for clarity.

3.6 A SIMPLE NON-PARAMETRIC DYNAMIC TOPIC MODEL

Statistical admixture topic models have recently gained much popularity in managing large document collections. In these models, each document is sampled from a mixture model according to a document's specific mixing vector over the mixture components (*topics*), which are often represented as a multinomial distribution over a given vocabulary. An example of such models is the well-known latent Dirichlet allocation (LDA)[29]. Recent approaches advocate the importance of modeling the dynamics of different aspects of topic models: topic trends [127], topic word distributions [27] and topic correlations [83]. In this section we show how to implement a simple non-parametric dynamic topic model. The model presented here is simpler than mainstream topic models in that each document is generated from a *single* topic rather than from a mixture of topics as in LDA. However, this is not a restriction of our framework, as we will mention in the future work section how

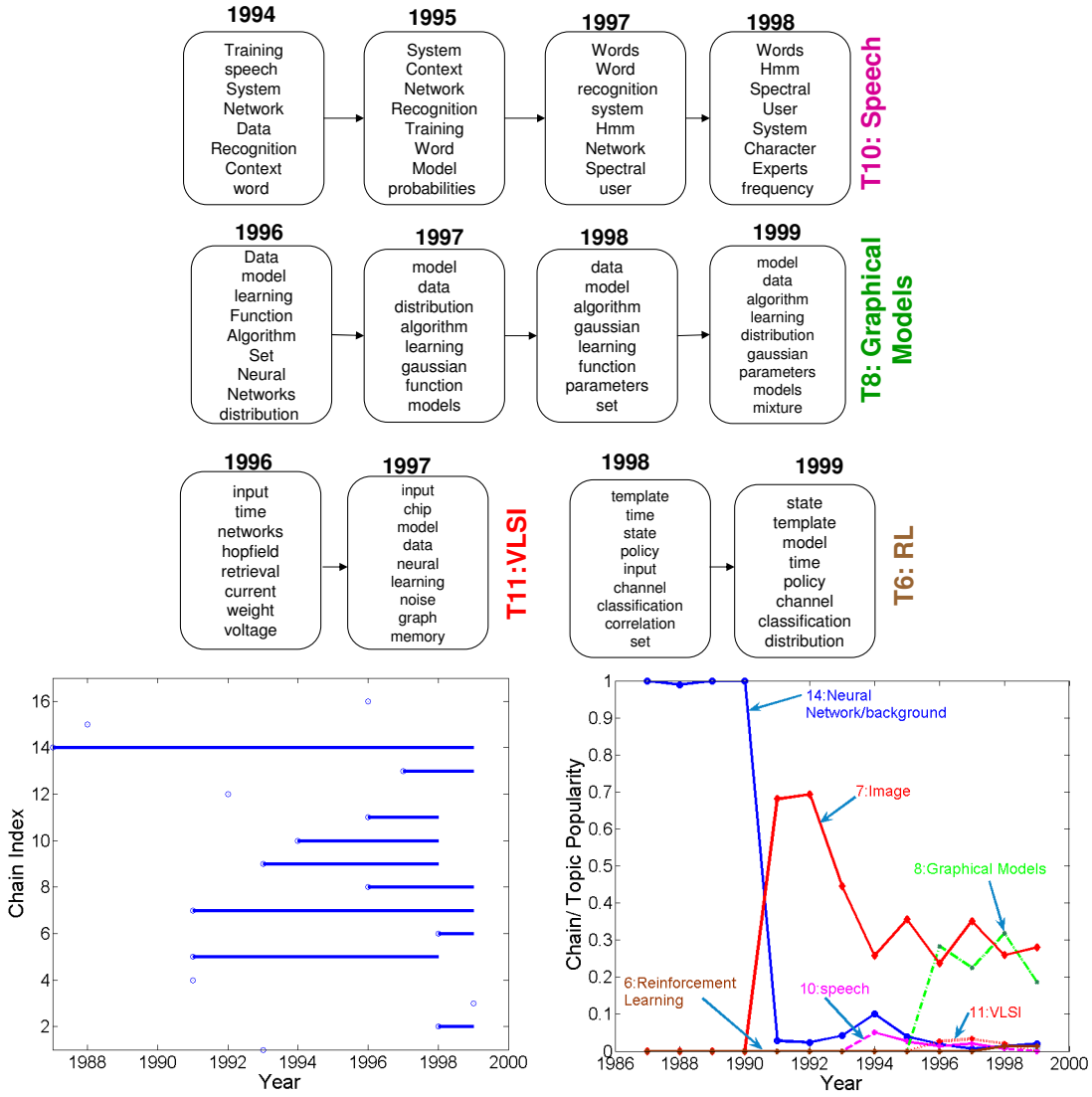


Figure 3.4: Illustrating results on the NIPS12 dataset. **Top**: keywords over time in some topics. **Left-bottom**: chains (topics) death-birth over time. **Right-botoom**: the popularity of some topics over the years, where topics names are hand labeled.

this simple model can be extend to a full-fledged one. The model we present here is only meant as another illustration of the generality of our framework.

To implement this simple non-parametric dynamic topic model, SNDTM for short, let $x_{t,i}$ represent a document composed of word frequency counts. Each chain represents the natural parameter of the multinomial distribution associated with a given topic, similar to the the dynamic LDA model in [27]. Each topic's natural parameter chain, ϕ_k, evolves using a random walk model [27]. To generate a document, first map the natural parameter of its topic $\phi_{k,t}$ to the simplex via the logistic transformation in Equation (8-10), and then generate the document, i.e. $x_{t,i}|c_{t,i} = k \sim \text{Multinomial}(x_{t,i}|\text{Logistic}(\text{ffi}_{k,t}))$.

In Equations (3.10), $C(\phi_{k,t})$ is a normalization constant (i.e., the log partition function). We denote this logistic transformation with the function $\text{Logisitc}(.)$. Furthermore, due to the normalizability constrain of the multinomial parameters,

$\vec{\beta}_{k,t}$ only has $M-1$ degree of freedom, where M is the vocabulary length. Thus we only need to represent and evolve the first $M-1$ components of $\phi_{k,t}$ and leave $\phi_{k,t}=0$. For simplicity, we omit this technicality from further consideration.

$$\begin{aligned}\beta_{k,t,m} &= \exp\{\phi_{k,t,m} - C(\phi_{k,t})\}, \quad \forall m = 1,\dots,M \\ \text{where} \qquad & C(\phi_{k,t}) = \log\Big(\sum_{m=1}^{M}\exp\{\phi_{k,t,m}\}\Big).\end{aligned} \tag{3.10}$$

One problem with the above construction is the non-conjugacy between the multinomial distribution and the logistic normal distribution. In essence, we can no longer use vanilla RTS smoother to compute the posterior over each chain as required by the Gibbs sampling algorithm in Section 3.3. In [27], numerical techniques were proposed to solve this problem; here, for simplicity, we use a deterministic Laplace approximation to overcome this non-conjugacy problem. We first put the emission of chain ϕ_k at time t in the exponential family representation. It is quite straightforward to show that:

$$\prod_{x\in\phi^{x}_{k,t}}\prod_{m=1}^{M} p(x_{t,i,m}|\phi_{k,t}) = \exp\{v_{k,t}\phi_{k,t} - |v_{k,t}|\times C(\phi_{k,t})\} \tag{3.11}$$

where $v_{k,t}$ is an M-dimensional (row) vector that represents the histogram of word occurrences from topic k at time step t . And $|.|$ is the L1 norm of a given vector. Equation (3.11) still does not represent a Gaussian emission due to the problematic $C(\phi_{k,t})$. Therefore, we approximate it with a second-order quadratic Taylor approximation around $\hat{\phi}_{k,t}$ — to be specified shortly . This results in a linear and quadratic term of $\phi_{k,t}$. If we let H and g to be the hessian and gradient of such expansion, we can re-arrange equation (3.11) into a gaussian emission with mean $\chi_{\hat{\phi}_{k,t}}$ and covariance $\varphi_{\hat{\eta}_{k,t}}$ given by:

$$\varphi_{\hat{\phi}_{k,t}} = \text{inv}\big(|v_{k,t}|H(\hat{\phi}_{k,t})\big), \tag{3.12}$$

$$\chi_{\hat{\phi}_{k,t}} = \hat{\phi}_{k,t} + \varphi_{\hat{\eta}^{t}_{k}}\big(v_{k,t} - |v_{k,t}|g(\hat{\phi}_{k,t})\big). \tag{3.13}$$

Using this Gaussian approximation to the non-gaussian emission, we can compute the posterior over $\phi_{k,t}|\phi^{(x)}_{k,t}$ using the RTS smoother with observations, and observation noises as given by Equations (3.13) and (3.12) respectively. Due to the high-dimensionality of the associated vectors in this linear state-space model, we approximate the Hessian in the above calculations with its diagonal, which results in an M-independent linear state-space models, one for each word. Moreover, $\hat{\phi}_{k,t}$ is set to inverseLogisitic$\Big(\frac{v_{k,t}}{|v_{k,t}|}\Big)$, which is the inverse logistic transformation of the MLE (maximum likelihood estimation) of the topic's k multinomial distribution at time t.

We used this simple model to analyze the NIPS12 collection that contains the proceedings of the Neural Information Processing Conference from 1987-1999[2].

2 Available from http://www.cs.toronto.edu/ roweis/data.html

Stop words were removed from this collection, we also removed infrequent words and kept only the top most frequent 2000 words. We divided the collection into 13 epochs based on the publication year of the paper. We set the hyperparmaters of the TDPM as in Section 7 with $\alpha = .1$, and we ran Gibbs sampling for 1000 iterations. To speed up convergence, we initialized the sampler from the result of a global non-parametric clustering using the method in [42] which resulted in around 7 clusters, each of which spans the whole 13 years. In Figure 3.4, we display topic durations, which shows that the model indeed captures the death and birth of various topics. In the same figure, we also show the top keywords in some topics (chains) as they evolve over time. As shown in this figure, regardless of the simplicity of the model, it captured meaningful topic evolutions.

3.7 RELATION TO OTHER DYNAMIC NON-PARAMETRIC BAYESIAN APPROACHES

We have purposefully delayed discussing the relationship between the TDPM and other *dependent* DPM models until we lay down the foundation of our model in order to place the discussion in context. In fact, several approaches have been recently proposed to solve the same fundamental problem addressed in this chapter: how to add the notion of time into the DPM. With the exception of [55], most of these approaches use the stick-breaking construction of the DPM [59][86]. In this construction, the DPM is modeled as an infinite mixture model, where each mixture component has a weight associated with it. Coupling the weights and/or (component parameters) of nearby DPMs results in a form of dependency between them. This new process is called ordered-based (Dependent) DPMs. However, we believe that utilizing the CRP directly is easier as we have explained in section 4.2, and more importantly, this approach enables us to model the rich-gets-richer phenomenon, which we believe captures, in a wide range of applications, how a cluster popularity evolves over time. As for the work in [55], we have already explained one difference in Section 3.4. Another difference is that in [55] cluster parameters are fixed and do not evolve over time. Recently, a simple clustering-topic model built on top of [59] was proposed in [113]. This is similar to the experiments we carried in section 8, however, in [113] the cluster (topic) parameters were fixed over time.

3.8 DISCUSSION

In this chapter we presented the temporal Dirichlet process mixture model as a framework for modeling complex longitudinal data. In the TDPM, data is divided into epochs, where the data items within each epoch are partially exchangeable. Moreover, The number of mixture components used to explain the dependency structure in the data is unbounded. Components can retain, die out or emerge over time, and the actual parameterization of each component can also evolve over time in a Markovian fashion. We gave various constructions of the TDPM as well as a Gibbs sampling algorithm for posterior inference. We also showed how to use the TDPM to implement an infinite mixture of Kalman filters as well as a simple non-parametric dynamic topic model.

One question that remains is what marginal distribution is realized by the process introduced in this chapter. Is the current process marginally a DP? In [23], the authors extends our work and gives a new process called distance-dependent CRP that is applicable to any distance function not only time as in our formalization. [23] also shows that these class of processes (which covers the RCRP introduced here) are not marginally DP unless the process degenerates to either a singe DP or a set of independent DPs (see Figure 3.2), and that our process rather defines a distribution of time-varying partitions. What is the implication of these results? It means that if data points are not missing at random in each epoch, then inference is not guaranteed to be consistent. To see this, note that the missing points (say at time t) should also affect the distribution of the data at time $t+1$ according to the process, however, inference for the data at time $t+1$ missed those points. As such, the marginal distribution of the data point at time $t+1$ after the arrival of the new missing data at time t is not the same to the distribution without the missed data (i.e. the process does not satisfy the marginalization constraints [50]). This means that our model is better looked at as a *conditional model* and thus two applications are possible: 1) structure recovery, in which we are interested in $P(c_t|x_t)$, i.e. the clustering configuration (as we are in this Chapter and in Chapters 4, 5 and 8), or 2) future prediction, where we care about computing $P(x_{t+1}|x_{1:t})$ since the process is conditionally DP and thus inference in this case is consistent (as the case in Chapter 5 and 8).

Finally, in [10] we proposed a collapsed variational inference algorithm for the same process yet its efficacy is still to be evaluated. It is interesting to explore search based techniques [42] that showed promising results in the DPM case and achieved up to 200-300 speedup over using Gibbs sampling.

4

TIMELINE: RECOVERING BIRTH/DEATH AND EVOLUTION OF TOPICS IN TEXT STREAM

4.1 INTRODUCTION

With the dramatic increase of digital document collections such as online journal articles, the arxiv, conference proceedings, blogs, to name a few, there is a great demand for developing automatic text analysis models for analyzing these collections and organizing its content. Statistical admixture topic models [29] were proven to be a very useful tool to attain that goal and have recently gained much popularity in managing large collection of documents. Via an admixture model, one can project each document into a low dimensional space where their latent semantic (such as topical aspects) can be captured. This low dimensional representation can then be used for tasks like measuring document-document similarity or merely as a visualization tool that gives a bird's eye view of the collection and guides its exploration in a structured fashion.

An admixture topic model posits that each document is sampled from a fixed-dimensional mixture model according to a document's specific mixing vector over the *topics*. The variabilities in the topic mixing vectors of the documents are usually modeled as a Dirichlet distribution [29], although other alternatives have been explored in the literature [20, 81]. The components of this Dirichlet distribution encode the popularity of each of the topics in the collection. However, document collections often come as temporal streams where documents can be organized into epochs; examples of an epoch include: documents in an issue of a scientific journal or the proceeding of a conference in a given year. Documents inside each epoch are assumed to be exchangeable while the order between documents is maintained across epochs. With this organization, several aspects of the aforementioned static topic models are likely to change over time, specifically: topic *popularity*, topic *word distribution* and the *number* of topics.

Several models exist that could accommodate the evolution of some but not all of the aforementioned aspects. [27] proposed a dynamic topic model in which the topic's word distribution and popularity are linked across epochs using state space models, however, the number of topics are kept fixed. [127] presented the topics over time model that captures topic popularity over time via a beta distribution, however, topic distributions over words and the number of topics were fixed over time, although the authors discussed a non-parametric extension over the number of topics. Moreover, the shapes of permitted topic trends in the TOT model are restricted to those attained by the beta distribution. On the other hand, several models were proposed that could *potentially* evolve all the aforementioned aspect albeit in a simple clustering settings, i.e. each document is assumed to be sampled from a single topic [9, 33, 113]. As we will show in this chapter, accommodating the evolution of the aforementioned aspects in a full-fledged admixture setting is non-trivial and introduces its own hurdles. Moreover, it is widely accepted [29] that

admixture models are superior compared to simple clustering models for modeling text documents, especially for long documents such as research papers.

In this chapter we introduce iDTM: infinite dynamic topic models which can accommodate the evolution of the aforementioned aspects. iDTM allows for unbounded number of topics: topics can be born and die at any epoch, the topics' word distributions evolve according to a first-order state space model, and the topics' popularity evolve using the rich-gets richer scheme via a Δ-order process. iDTM is built on top of the recurrent Chinese restaurant franchise (RCRF) process which introduces dependencies between the atom locations (topics) and weights (popularity) of each epoch. The RCRF process is built on top of the RCRP process introduce in Chapter 3 [9].

To summarize, the contributions of this chapter are:

- A principled formulation of a dynamic topic model that evolves: topic trend, topic distribution, and number of topics over time.
- An efficient sampling algorithm that relies on dynamic maintenance of cached sufficient statistics to speed up the sampler.
- An empirical evaluation and illustration of the proposed model over simulated data and over the NIPS proceedings.
- A study of the sensitivity of the model to the setting of its hyperparameters.

4.2 SETTINGS AND BACKGROUND

In this section, we lay the foundation for the rest of this chapter by first detailing our settings and then reviewing some necessary background to make the chapter self-contained. We are interested in modeling an ordered set of documents $\boldsymbol{w} = (\boldsymbol{w}_1, \cdots, \boldsymbol{w}_T)$, where T denotes the number of epochs and $\boldsymbol{w}_t$ denotes the documents at epoch t. Furthermore, $\boldsymbol{w}_t = (\boldsymbol{w}_{td})_{d=1}^{D_t}$, where D_t is the number of documents at epoch t. Moreover, each document comprises a set of n_{td} words, $\boldsymbol{w}_{td} = (w_{tdi})_{i=1}^{N_{td}}$, where each word $w_{tdi} \in \{1, \cdots, W\}$. Our goal is to discover *potentially* an unbounded number of topics $(\boldsymbol{\phi}_k)_{k=1}^{\infty}$ where each topic $\boldsymbol{\phi}_k = (\phi_{k,t_{k_1}}, \cdots, \phi_{k,t_{k_2}})$ spans a set of epoches where $1 \leqslant t_{k_1} \leqslant t_{k_2} \leqslant T$, and $\phi_{k,t}$ is the topic's word distribution at epoch t.

4.2.1 *Temporal Dirichlet Process Mixture Model: A Recap*

For reader's conveneince, we recap the the temporal Dirichlet process mixture model (TDPM) introduced in 3. TDPM is a framework for modeling complex longitudinal data, in which the number of mixture components at each time point is unbounded; the components themselves can retain, die out or emerge over time; and the actual parameterization of each component can also evolve over time in a Markovian fashion. In TDPM, the random measure G is time-varying, and the process stipulates that:

$$G_t|\phi_{1:k}, G_0, \alpha \sim DP\left(\alpha + \sum_k m'_{kt}, \sum_k \frac{m'_{k,t}}{\sum_l m'_{lt} + \alpha}\delta(\phi_k) + \frac{\alpha}{\sum_l m'_{lt} + \alpha}G_0\right) \tag{4.1}$$

where $\{\phi_{1:k}\}$ are the mixture components available in the previous Δ epochs, in other words, $\{\phi_{1:k}\}$ is the collection of unique values of the parameters $\theta_{t:t-\Delta}$, where θ_{tn} is the parameter associated with data point x_{tn}. If we let m_{kt} denotes the number of parameters in epoch t associated with component k, then m'_{kt}, the prior weight of component k at epoch t is defined as:

$$m'_{kt} = \sum_{\delta=1}^{\Delta} \exp^{\frac{-\delta}{\lambda}} m_{k,t-\delta} \tag{4.2}$$

,where Δ, λ define the width and decay factor of the time-decaying kernel. This defines a $\Delta-$order process where the strength of dependencies between epoch-specific DPs are controlled with Δ, λ. In Chapter 3 and [9] we showed that these two hyper-parameters allow the TDPM to degenerate to either a set of independent DPMs at each epoch when Δ=0, and to a global DPM, i.e ignoring time, when $\Delta = T$ and $\lambda = \infty$. In between, the values of these two parameters affect the expected life span of a given component. The larger the value of Δ and λ, the longer the expected life span of the topic, and vice versa. Finally, the life-span of topics followed a power law distribution [9].

In addition to changing the weight associated with each component, the parameterization ϕ_k of each component changes over time in a markovian fashion, i.e.: $\phi_{kt}|\phi_{k,t-1} \sim P(.|\phi_{k,t-1})$. Integrating out the random measures $G_{1:T}$, the parameters $\theta_{1:t}$ follow a polya-urn distribution with time-decay, or as termed in Chapter 3, the recurrent Chinese restaurant process (RCRP). More formally:

$$\theta_{ti}|\theta_{t-1:t-\Delta}, \theta_{t,1:i-1}, G_0, \alpha \propto \sum_k \left(m'_{kt} + m_{kt}\right)\delta(\phi_{kt}) + \alpha G_0 \tag{4.3}$$

The RCRP allows each document $\mathbf{w}$ to be generated from a single component (topic), thus making it suboptimal in modeling multi-topic documents. On the other hand as we detailed in Section 2.2 the HDP process allows each document to be generated from multiple topics, thus it seems natural to combine the two models to achieve our goal.

4.3 INFINITE DYNAMIC TOPIC MODELS

Now we proceed to introducing our model, iDTM which allows for infinite number of topics with variable durations. The documents in epoch t are modeled using an epoch specific HDP with high-level base measure denoted as G_0^t. These epoch-specific base measures $\{G_0^t\}$ are tied together using the TDPM process of Chapter 3. Integrating all random measure, we get the recurrent Chinese restaurant franchise process (RCRF).

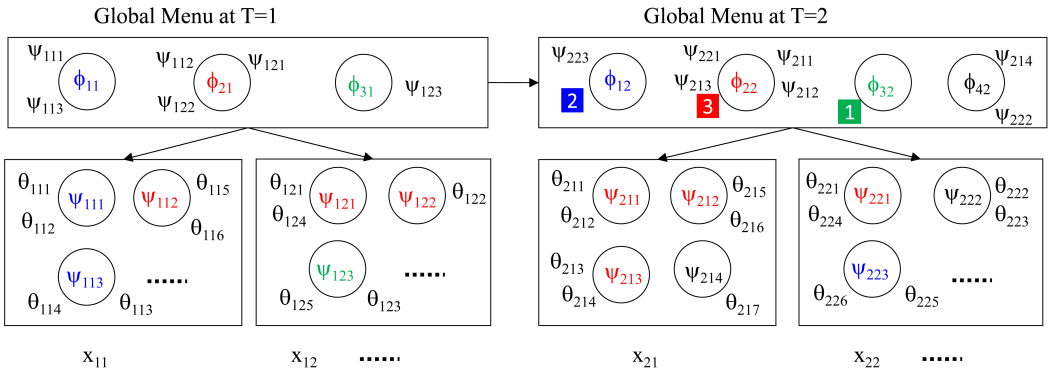


Figure 4.1: The recurrent Chinese restaurant franchise (RCRF) precoces. The figure shows a first-order process with no decay to avoid cluttering the display,however see the text for the description of a general Δ-order process.

Figure 4.1 depicts a RCRF process of order one for clarity, however, in this section we give a description of a general process of order Δ. In the RCRF, the document in each epoch is modeled using a CRFP, and then the global menus of each epoch are tied over time as depicted in Figure 4.1.

As in the RCRP, the popularity of a topic at epoch t depends both on its usage at this epoch, m_{kt} as well as it historic usage at the proceedings Δ epochs, m'_{kt}, where m'_{kt} is given in (4.2). This means that a topic is considered *dead* only when it is unused for a consecutive Δ epochs. For simplicity, we let $m'_{kt} = 0$ for newly-born topics at epoch t, and $m_{kt} = 0$ for topics available to be used(i.e having $m'_{kt} > 0$) but not yet used in any document at epoch t.

The generative process at the first epoch proceeds exactly as in the CRF process. At epoch t, to associate a topic with word w_{tdi} we proceed as follows. Word w_{tdi} can sit on table b that has n_{tdb} customers and has topic ψ_{tdb} with probability $\frac{n_{tdb}}{i-1+\alpha}$. Alternatively, w_{tdi} can choose to start a new table, b_{td}^{new} with probability $\frac{\alpha}{i-1+\alpha}$ and choose a new topic. It can choose an already existing topic from the menu at epoch t with probability $\frac{m_{kt}+m'_{kt}}{\sum_{l=1}^{K_t} m_{lt}+m'_{lt}+\gamma}$, K_t is the number of topics at epoch t. Furthermore, if this topic is inherited but not yet used by any previous word (i.e $m_{kt} = 0$), then w_{tdi} modifies the distribution of this topic: $\phi_{kt} \sim P(.|\phi_{k,t-1})$. Finally, w_{tdi} can choose a brand new topic $\phi_{k_t^{new}} \sim H$, with probability $\frac{\gamma}{\sum_{l=1}^{K_t} m_{lt}+m'_{lt}+\gamma}$ and increment K_t. Putting everything together, we have:

$$\theta_{tdi}|\theta_{td,1:i-1}, \alpha, \boldsymbol{\psi}_{t-\Delta:t} \sim \sum_{b=1}^{b=B_{td}} \frac{n_{tdb}}{i-1+\alpha}\delta_{\psi_{tdb}} + \frac{\alpha}{i-1+\alpha}\delta_{\psi_{tdb^{new}}} \tag{4.4}$$

$$\begin{aligned}\psi_{tdb^{new}}|\boldsymbol{\psi},\gamma \quad &\sim \quad \sum_{k:m_{kt}>0} \frac{m_{kt}+m'_{kt}}{\sum_{l=1}^{K_t} m_{lt}+m'_{lt}+\gamma}\delta_{\phi_{kt}} \\ &+ \quad \sum_{k:m_{kt}=0} \frac{m_{kt}+m'_{kt}}{\sum_{l=1}^{K_t} m_{lt}+m'_{lt}+\gamma}P(.|\phi_{k,t-1}) \\ &+ \quad \frac{\gamma}{\sum_{l=1}^{K_t} m_{lt}+m'_{lt}+\gamma} H \end{aligned} \tag{4.5}$$

If we use the RCRF process as a prior over word assignment to topics in a mixed-membership model, we get the infinite dynamic topic model (iDTM). In iDTM, each word is assigned to a topic as in the RCRF process, and then the word is generated from this topic's distribution. The base measure H is modeled as $H = N(0, \sigma I)$, and the word distribution of the topic k at epoch t, ϕ_{kt} evolves using a random walk kernel as in [27]:$\phi_{k,t}|\phi_{k,t-1} \sim N(\phi_{k,t-1}, \rho I)$. To generate word w_{tdi} from its associated topic, say ϕ_{kt}, we first map the natural parameters of this topic $\phi_{k,t}$ to the simplex via the logistic transformation L, and then generate the word, i.e.: $w_{tdi}|\phi_{kt} \sim \mathcal{M}\big(L(\phi_{kt})\big)$, where $L(\phi_{kt}) = \frac{\exp^{\phi_{kt}}}{\sum_{w=1}^{W}\exp^{\phi_{k,t,w}}}$. This choice introduces non-conjugacy between the base measure and the likelihood function which we have to deal with in Section 4.4.

4.4 A GIBBS SAMPLING ALGORITHM

In this section, we give a Gibbs sampling algorithm for posterior inference in the iDTM. We construct a Markov chain over $(\mathbf{k}, \mathbf{b}, \boldsymbol{\phi})$, where $k_{tdb}, b_{tdi}, \phi_{kt}$ are as given in Section 4.3: the index of the topic on table b in document td, the table index assigned to word w_{tdi}, and the parameterization of topic k at time epoch t, respectively. We use the following notations. Adding a superscript $-i$ to a variable, indicate the same quantity it is added to without the contribution of object i. For example n_{tdb}^{-tdi} is the number of customers sitting on table b in document d in epoch t without the contribution of word w_{tdi}, and $\mathbf{k}_t^{-tdb}$ is $\mathbf{k}_t \backslash k_{tdb}$. We alternate sampling each variable conditioned on its Markov blanket as follows:

Sampling a topic k_{tdb} for table tdb: The conditional distribution for k_{tdb} is given by:

$$\begin{aligned}P(k_{tdb}=k|\mathbf{k}_{t-\Delta:t+\Delta}^{-tdb}, \mathbf{b}_{td}, \boldsymbol{\phi}, \boldsymbol{w}_t) \quad \propto \quad & P(k_{tdb}=k|\mathbf{k}_{t-\Delta:t}^{-tdb}, \boldsymbol{\phi}, \boldsymbol{\nu}_{tdb}) \times \\ & \prod_{\delta=1}^{\Delta} P(\mathbf{k}_{t+\delta}|\mathbf{k}_{t+\delta-\Delta:t+\delta-1}^{-tdb\to k}) \end{aligned} \tag{4.6}$$

where $\boldsymbol{\nu}_{tdb}$ is the frequency count vector (of length W) of the words sitting on table tdb, and the notation $(^{-tdb\to k})$ means the same as $\mathbf{k}_{t-\Delta:t+\Delta}^{-tdb}$ but in addition we set $k_{tdb} = k$. The second factor in (4.6) is the transition probability which measures the likelihood of the table assignments at future epochs if we choose to assign $k_{tdb} = k$. Now we focus on computing one of these probabilities in the second factor in (4.6) at epoch $t + \delta$. With reference to the construction in Section 4.3 and Eq (4.5), and

considering that documents are exchangeable within each epoch, similar to [16], we have [1]:

$$P(\mathbf{k}_{t+\delta}|\mathbf{k}^{-tdb\to k}_{t+\delta-\Delta:t+\delta-1}) = \gamma^{K^{born}_{t+\delta}} \frac{\prod_{s\in K^{born}_{t+\delta}} [1]^{m_{s,t+\delta}} \prod_{s\notin K^{born}_{t+\delta}} [m'^{,-tdb\to k}_{s,t+\delta}]^{m_{s,t+\delta}}}{\prod_{i=1}^{m_{.,t+\delta}} (m'^{,-tdb\to k}_{.,t+\delta} + \gamma + i)} \tag{4.7}$$

where $K^{born}_{t+\delta}$ is the number of topics born at epoch $t+\delta$,$m_{.,t+\delta}$ is the summation of $m_{k,t+\delta}$ over the first dimension (the topic), and $m'_{.,t+\delta}$ is defined similarly. Finally, $[a]^c = a(a+1)\cdots(a+c-1)$.

Now we turn to the first factor in (4.6). Using (4.5), we have:

$$P\left(k_{tdb} = k|\mathbf{k}^{-tdb}_{t-\Delta:t}, \boldsymbol{\phi}, \boldsymbol{\nu}_{tdb}\right) \propto \begin{cases} \left(m^{-tdb}_{kt} + m'_{kt}\right) f(\boldsymbol{\nu}_{tdb}|\phi_{kt}) \\ \qquad\qquad k \text{ is being used} : m^{-tdb}_{kt} > 0 \\ m'_{kt} \int f(\boldsymbol{\nu}_{tdb}|\phi_{kt}) dP(\phi_{kt}|\phi_{k,t-1}) \\ \qquad\qquad k \text{ is available but not used} : m'_{kt} > 0 \\ \gamma \int f(\boldsymbol{\nu}_{tdb}|\phi_{kt}) dH(\phi_{kt}) \\ \qquad\qquad k \text{ is a new topic} \end{cases}$$

Unfortunately, due to the non-conjugacy neither the second nor the third case above can be computed analytically. In Chapter 3 a Laplace approximation was used to fit these integrals. This was possible since the integrals were evaluated over the whole document (a mixture model), however in our setting (mixed-membership model), we need to evaluate these integrals over small groups of words (like words on a given table). We found that the deterministic approximation overestimates the integrals and increases the rate of generating new topics, therefore we resort to algorithm 8 in [97]. In this case, we replace both of these integrals with Q fresh samples from their respective distributions, and equally divide the corresponding probability mass among these new samples. These samples are then treated as if they were already existing topics. We choose to use $Q = 1$ for the transition kernel since in our application, iDTM, this kernel usually has a small variance. Substituting this in Equation (4.8), we get that:

$$P\left(k_{tdb} = k|\mathbf{k}^{-tdb}_{t-\Delta:t}, \boldsymbol{\phi}, \boldsymbol{\nu}_{tdb}\right) \propto \begin{cases} \left(m^{-tdb}_{kt} + m'_{kt}\right) f(\boldsymbol{\nu}_{tdb}|\phi_{kt}) \\ \qquad\qquad k \text{ is used:} m^{-tdb}_{kt} > 0 \\ m'_{kt}\, f(\boldsymbol{\nu}_{tdb}|\phi_{kt}) \\ \qquad\qquad m'_{kt} > 0, m^{-tdb}_{kt} = 0, \phi_{kt} \sim P(.|\phi_{k,t-1}) \\ \frac{\gamma}{Q} f(\boldsymbol{\nu}_{tdb}|\phi^q_{kt}) \\ \qquad\qquad k \text{ is a new topic}, \phi^q_{kt} \sim H, q = 1\cdots Q \end{cases} \tag{4.8}$$

1 In Chapter 3 and in [9], we used a finite dynamic-mixture model, which is equivalent on the limit to RCRP, to compute the same quantity. The formula here is exact and have the same amount of computation as the approximate formula. We also note that our formula gives better results

Sampling a table b_{tdi} for word x_{tdi}: With reference to (4.4), the conditional distribution is :

$$P\left(b_{tdi} = b|\mathbf{b}_{td}^{-tdi}, \mathbf{k}_{t-\Delta:t+\Delta}, \boldsymbol{\phi}, x_{tdi}\right) \propto \begin{cases} n_{tdb}^{-tdi} f(x_{tdi}|\phi_{k_{tjb},t}) \\ \qquad\qquad b \text{ is an existing table} \\ \alpha P\left(k_{tjb^{new}} = k|\mathbf{k}_{t-\Delta:t+\Delta}{}^{-tdi}, \mathbf{b}_{td}^{-tdi}, \boldsymbol{\phi}, x_{tdi},\right) \\ \qquad\qquad b^{new} \text{ is a new table}, k \in \mathbf{k}_t \end{cases} \tag{4.9}$$

Several points are in order to explain (4.9). There are two choices for word w_{tdi}:either to sit on an existing table, or to sit on a new table and choose a new topic. In the second case, we need to sample a topic for this new table which leads to the same equation as in (4.8). Moreover, if w_{tdi} was already sitting on a table by itself, then we need to first remove the contribution of this table from the count vector $\mathbf{m}$. Finally, note that in the second line, P is a proper distribution (i.e. it should be normalized) and thus the total probability mass for sitting on a new table is till α regardless of how many topics are available at epoch t.

Sampling $\boldsymbol{\phi}_k$: $P(\boldsymbol{\phi}_k|\mathbf{b}, \mathbf{k}, \mathbf{x}) = P(\boldsymbol{\phi}_k|\boldsymbol{\nu}_k)$, where $\boldsymbol{\nu}_k = \{\boldsymbol{\nu}_{k,t}\}$, $\boldsymbol{\nu}_{k,t}$ is the frequency count vector of words generated from this topic at epoch t. This a state space model with nonlinear emission, and fortunately there is a large body of literature on how to use Metropolis-Hasting to sample from this distribution [115, 54]. There are two strategy to deal with this posterior: either sample from it as a block, or to sequentially sample each ϕ_{kt}. Both of these options involve an M-H proposal,however, due to the strong correlation between the successive values of ϕ_{kt}, we found that sampling this posterior as a block is superior. Let $q(\boldsymbol{\phi}_k)$ be the proposal distribution, and let $\boldsymbol{\phi}_k^*$ denote a sample from this proposal. The acceptance ratio is $r = \min(1, u)$, where u is as follows (for simplicity assume that the chain starts at $t = t_1$):

$$\frac{H(\phi^*_{k,t_1}) \times \prod_t f(\nu_{kt}|\phi^*_{kt}) P(\phi^*_{kt}|\phi^*_{k,t-1})}{H(\phi_{k,t_1}) \times \prod_t f(\nu_{kt}|\phi_{kt}) P(\phi_{kt}|\phi_{k,t-1})} \times \frac{\prod_t q(\phi_{kt})}{\prod_t q(\phi^*_{kt})} \tag{4.10}$$

With probability r the proposed values are accepted and with probability $1 - r$ the old values are retained. Our proposal is based on a Laplace approximation to the LTR smoother (details of calculating this proposal is given below). A similar Laplace proposal has been used successively in the context of Bayesian inference of the parameters of a Logistic-Normal distribution [63], as well as in the context of non-linear state space models in [54] who also noted that this proposal has high acceptance rate (a fact that we also observed).

Fitting the proposal distribution in (4.10)

Our goal is to sample $P(\boldsymbol{\phi}_k|\boldsymbol{\nu}_k)$, where $\boldsymbol{\nu}_k = \{\boldsymbol{\nu}_{k,t}\}$, $\boldsymbol{\nu}_{k,t}$ is the frequency count vector of words generated from this topic at epoch t. Without loss of generality, and for notational simplicity, we will drop the topic index k, and assume that the topic's lifespan is form 1 to T. Thus we would like to compute $P(\phi_1, \ldots, \phi_T|\nu_1, \ldots, \nu_T)$. This is

a linear state-space model with non-linear emission, thus the RTS smoother [92] will not result in a closed form solution. Therefore, we seek a Laplace-approximation to this posterior and call this approximation, $q(\phi) = \prod_t q(\phi_t)$. To compute q we note that the RTS smoother defines two recurrences: the forward recurrence, and the backward recurrence. Following [92], lets assume that the forward value at epoch $t-1$ is given by $\hat{\alpha}(\phi_{t-1}) \sim \mathcal{N}(\mathbf{u}_{t-1}, \check{}_{t-1})$ where $\check{}$ has a diagonal covariance. The forward equation becomes:

$$\begin{aligned}\hat{\alpha}(\phi_t) &= \mathcal{LN}(v_t|\phi_t) \times \\ &\quad \int_{\phi_t} \mathcal{N}(\phi_t|\phi_{t-1}, \rho I)\mathcal{N}(\phi_{t-1}|\mathbf{u}_{t-1}, \check{}_{t-1}) \\ &= \mathcal{LN}(v_t|\phi_t)\mathcal{N}(\phi_t|\phi_{t-1}, \check{}_{t-1} + \rho I) \end{aligned} \tag{4.11}$$

Equation (4.11) does not result in the desired Gaussian form because of the non-conjugacy between the $\mathcal{LN}$ and the normal distributions. Therefore, we seek a Laplace approximation to (4.11) which puts it back into the desired Gaussian form to continue the recurrence. In this case $\mathbf{u}_t$ is the mode of (4.11) and $\check{}_t$ is the negative inverse Hessian of (4.11) evaluated at the mode. We use a Diagonal-approximation of the Hessian though because of the high-dimensionality of ϕ.

As detailed in [92], the backward recurrence can be defined using the $\hat{\alpha}$'s instead of the data, and can be computed exactly if the dynamic model is linear, which is the case in our model. This backward recurrence computes $q(\phi_t|v_1, \ldots, v_T)$ that we desire.

4.4.1 *Practical Considerations*

A naive implementation of the Gibbs sampling algorithm in 4.4 might be slow. We note here that the difference between the sampler we described in 4.4 and the sampler of a standard CRFP comes in the calculation of the vales of m'_{kt} and for the calculation of (4.7). The case for m' is simple if we note that it needs to be computed only once before sampling variables in epoch t, moreover, because of the form of the exponential kernel used, we can define a recurrence over m'_{kt} as: $m'_{kt} = \left(m'_{k,t-1} + m_{k,t-1}\right)\exp^{\frac{-1}{\lambda}}$ if $t < \Delta$ and $m'_{kt} = \left(m'_{k,t-1} + m_{k,t-1}\right)\exp^{\frac{-1}{\lambda}} - \exp^{\frac{-(\Delta+1)}{\lambda}} m_{k,t-(\Delta+1)}$ otherwise.

On the other hand, a naive implementation of (4.7) costs an $O(K^2\Delta)$ as we need to compute it for Δ epochs and for each k. Here we describe an alternative approach. We divide and multiply (4.7) with $P(\mathbf{k}_{t+\delta}|\mathbf{k}_{t+\delta-\Delta:t+\delta-1})$, which is the likelihood of the table assignments at epoch $t+\delta$ given the current configuration with the old value of $k_{tdb} = k^{old}$. This is legitimate since this value is constant across k. Now we absorb the value we multiplied in the normalization constant, and focus on the following ratio:

$$C^{t+\delta}(k^{old} \to k^{new}) = \frac{P(\mathbf{k}_{t+\delta}|\mathbf{k}^{-tdb\to k}_{t+\delta-\Delta:t+\delta-1})}{P(\mathbf{k}_{t+\delta}|\mathbf{k}_{t+\delta-\Delta:t+\delta-1})} \tag{4.12}$$

where $C^{t+\delta}(k^{old} \to k^{new})$ is the cost contributed by the assignment of the table at epoch $t+\delta$ for moving an occupancy from table k^{old} to k^{new}. In fact (4.12) is all

what we need to compute (4.7) and thus (4.6). The idea here is that all the terms not involving k^{old} and k^{new} will cancel from (4.12), leaving only 2 terms that involve k^{old} and k^{new} to be computed. To see why this is the case, note that $m'^{,-tdb\to k^{new}}_{s,t+\delta}$ reduces to $m'_{s,t+\delta}$ whenever $s \notin \{k^{new}, k^{old}\}$. Furthermore, we can cache this two dimensional array at each epoch and dynamically update it whenever the sampled value in (4.6) for k_{tdb} is different from k^{old}. In this case, we need to update $C^{t+\delta}(k^{old} \to .)$ and $C^{t+\delta}(k^{new} \to .)$. Thus the cost of computing the transition probability reduces from $O(K^2\Delta)$ to at most $O(K\Delta)$. Moreover, especially at later stages of the sampler when tables do not change their topic assignments frequently, the improvement ratio will be more than that.

4.5 EXPERIMENTAL RESULTS

In this section we illustrate iDTM by measuring its ability to recover the death and birth of topics in a simulated dataset and in recovering topic evolution in the NIPS dataset. For all the experiments in this chapter, we place a vague gamma prior (1,1) over the hyperparameters γ, α and sample them separately for each epoch using the method described in [116]. Unless otherwise stated, we use the following values for the hyperparameters: the variance of the base measure H, $\sigma = 10$; the variance of the random walk kernel over the natural parameters of each topic, $\rho = 0.01$; $\Delta = 4$; the number of sample from the base measure $Q = 5$, and finally $\lambda = .5$.

Initialization of the Markov chain is quite important. Our setup proceeds as follows. In the first iteration, we run the Gibbs sampler in the filtering mode (i.e. sampling each epoch conditioned on the Δ preceding epochs only) and used a liberal value for $\alpha = 4$ and $\gamma = 10$ to encourage the generation of large number of topics. An initial large number of topics in desirable since as noted in [117], initializing HDP-like models with large number of topics results in a better mixing than initializing the sampler with a smaller number of topics. In the subsequent iterations, we ran our standard Gibbs sampler that also samples the values of α, γ. Finally, we ran all samplers for 2000 iterations and took 10 samples 200 iterations apart and then used the sample with the highest Likelihood for evaluation and visualization

4.5.1 *Simulation Results*

We generated a simple time-evolving document collection over $T = 20$ epochs. We set the vocabulary size to 16, and hand-crafted the birth-death of 8 topics, as well as their words' distributions as shown in Figure 4.2. Each topic puts its mass on 4 words. At each epoch we add a 5% random noise to each topic's word distribution. We then ran the RCRFP with $\alpha = 1.5$ to generate 100 documents at each epoch each of which having 50 words. γ was set to zero in this generation since the topics layout were fixed by hand. Moreover, Once a topic is alive, say at epoch t, its prior popularity m' is set to the average prior popularity at epoch t. Our goal was to assess the efficacy of iDTM in recovering abrupt death and birth of topics. Finally given the generated data, we ran the sampler described in Section 4.4 to recover the topics and their durations. As depicted in Figure 4.2, iDTM was able to recover the correct distribution for each topic as well as its correct lifespan.

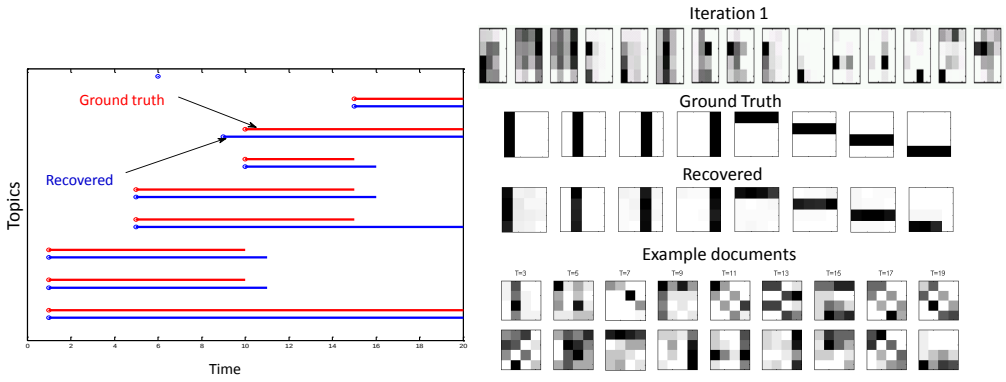


Figure 4.2: Illustrating simulation results. **Left**: topic's death-birth over time (topics numbered from bottom-top). Ground truth is shown in red and recovered in blue. **Right**: from top to bottom, topics' distribution after iteration 1, a posterior sample, ground truth (numbered from left to right), and finally a set of documents at different time epochs from the training data.

4.5.2 *Timeline of the NIPS Conference*

We used iDTM to analyze the proceedings of the NIPS conference from the years 1987-1999. We removed words that appear more than 5000 times or less than 100 times which results in a vocabulary size of 3379 words. The collection contains 1740 documents, where each document contains on average 950 words. Documents were divided into 13 epochs based on the publication year. We ran iDTM to recover the structure of topic evolution in this corpus.

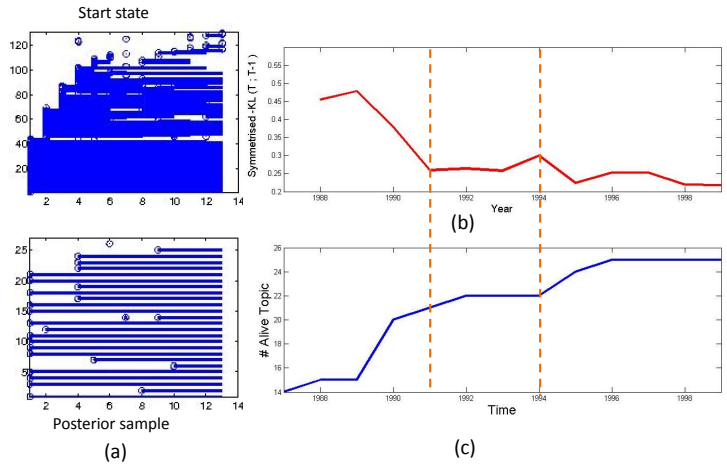


Figure 4.3: **(a)** The sampler initial state and the MAP posterior sample. Each line represents the lifespan of a topic. **(b)** Symmetrized-KL divergence between the unigram distribution of words at epoch $t, t-1$. **(c)** The number of alive topics over years in the NIPS collections.

Figure 4.3.a shows the initial state of the sampler and the MAP posterior sample. Each horizontal line gives the duration of a topic where the x-axis denotes time. Figure 4.3.c shows the number of topics in the collection over time. We also draw the symmetrized KL-divergence between the unigram distribution of words at

epoch t and $t-1$. It can be noticed that whenever there is a sharp change in the KL value, the model responds by changing the number of topics. However, when the KL value is stable (but not zero), the model responds by changing the word distributions of the topics and/or the topics' trends. This is in contrast to DTM which can only change the last two quantities. We would like to add that the trend of always-increasing number of topics is not an artifact of the model, but rather a property of the NIPS conference in this lifespan: none of the topics that we observed die completely during this time period. Moreover, as we illustrated in the simulation study, the model can detect an abrupt death of topics.

In Figure 4.4, we show a *timeline* of the conference pointing out the birth of some of the topics. We also give how their trends change over time and show a few examples of how the top words in each topic change over time. In Figure 4.5, we show a *timeline* of the Kernel topic illustrating, in some years, the top 2 (3 in case of a tie) papers with the highest weights for this topic. Indeed the three papers in 1996 are the papers that started this topic in the NIPS community. We would like to warn here that the papers having the highest weights of a topic need not be the most influential papers about this topic. Perhaps this is true in the year in which the topic was born, but for subsequent years, these papers give an overview of how this topic is being addressed along the years, and it can provide a concise input for summarization systems utilizing topic models as in [60]. Finally it is worth mentioning that iDTM differs from DTM in the way they model topic trends: DTM assumes a smooth evolution of trends, whereas iDTM assumes a non-parametric model and as such can spawn a topic with a large initial trend as in the Kernel topic in 1996.

4.5.2.1 *Quantitative Evaluation*

In Addition to the above qualitative evaluation, we compared iDTM against DTM [27] and against HDP [116]. To compare with DTM, we followed the model in [27], and used a diagonal covariance for the logistic-normal distribution over the topic-mixing vector at each epoch. We linked the means of the logistic-normal distributions at each epoch via a random walk model, and we evolved the distribution of each topic as we did in iDTM. To make a fair comparison, in fitting the variational distribution over the topic's word distribution ϕ in DTM , we used a Laplace variational approximation similar to the one we used in fitting the proposal distribution for iDTM. Moreover, we used importance sampling with the variational distribution as a proposal for calculating the test LL for DTM.

We divided the data into 75% for training and 25% for testing, where the training and test documents were selected uniformly across epochs. As shown in Figure 4.6, iDTM gives better predictive LL over DTM and HDP.

4.5.2.2 *Hyperparameter Sensitivity*

To assess the sensitivity of iDTM to hyperparameters' settings, we conducted a sensitivity analysis in which we hold all hyperparameters fixed at their default values, and vary one of them. As noted earlier, the hyperparameters are: the variance of the base measure σ; the variance of the random walk kernel over the natural parameters of each topic, ρ; and the parameter of the time-decaying kernel,

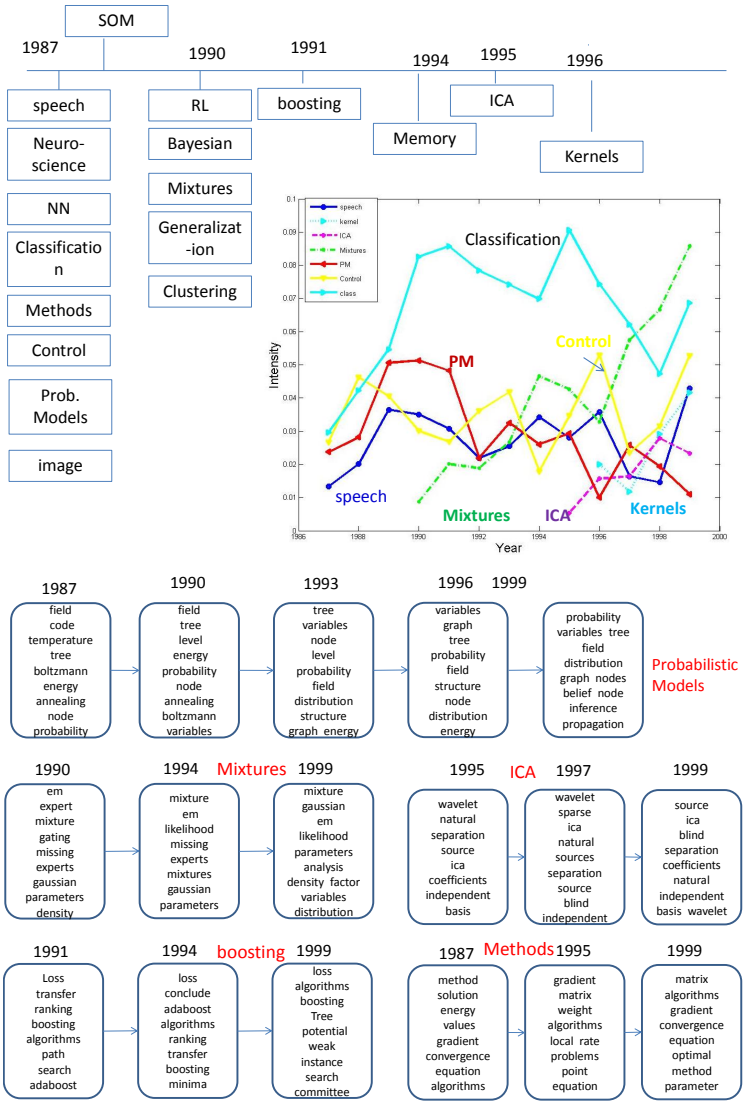


Figure 4.4: Timeline for the NIPS conference. **Top**: birth of a number of topics and their trends over time. **Bottom**: top words in some topics over time.

λ. We should note here that the order of the process Δ can be safely set to T, however, to reduce computation, we can set Δ to cover the support of the time-decaying kernel, i.e, we can choose Δ such that $\exp^{\frac{-\Delta}{\lambda}}$ is smaller than a threshold, say .001. The results are shown in Figure 4.7.

When ρ is set to 1, the performance deteriorates and the topics become incoherent over time. We noticed that in this setting the model recovers only 5 to 7 topics. When ρ is set to .0001, the word distribution of each topic becomes almost fixed over time. In between, the model peaks at $\rho = .01$. It should be clear from the figure that an underestimate of the optimal value of ρ is less detrimental than an overestimate, thus we recommend setting ρ in the range $[.001, .1]$. It should be noted that we could add a prior over ρ and sample it every iteration as well; we leave this for future work.

While varying λ, we fixed $\Delta = T$ to avoid biasing the result. A large value of λ degenerates the process toward HDP, and we noticed that when $\lambda = 6$, some

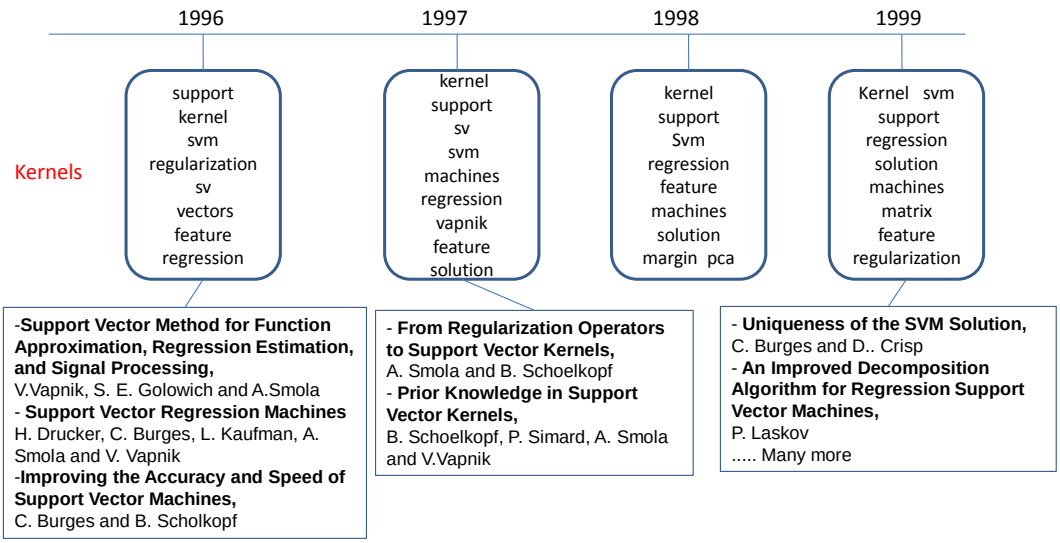


Figure 4.5: Timeline for the Kernel topic. The figure shows the top words in the topic in each year and the top 2 (3 in case of a tie) papers with the highest weights of this topic in some years

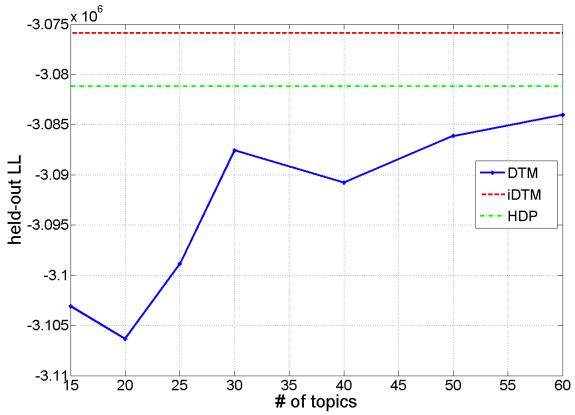


Figure 4.6: Held-out LL comparison between DTM,iDTM and HDP.

topics,like ICA, weren't born and where modeled as a continuation of other related topics. λ depends on the application and the nature of the data. In the future, we plan to place a discrete prior over λ and sample it as well. Finally, the best setting for the variance of the base measure σ is from $[5, 10]$, which results in topics with reasonably sparse word distributions.

4.6 DISCUSSION

In this chapter we addressed the problem of modeling time-varying document collections. We presented a topic model, iDTM, that can adapt the number of topics, the word distributions of topics, and the topics' trend over time. To the best of our knowledge, this is the first model of its kind. We used the model to analyze the NIPS conference proceedings and drew several timelines for the conference:

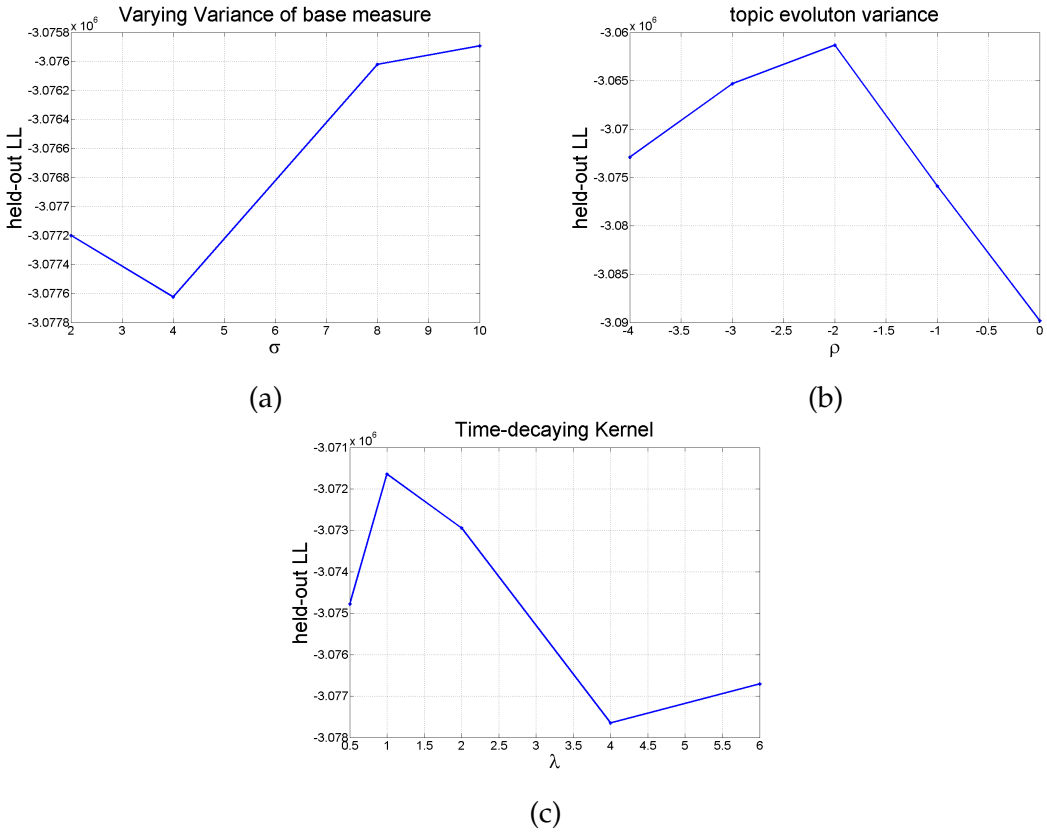


Figure 4.7: Sensitivity of iDTM to hyperparameters. Every panel vary one hyperparameter while keeping all other hyperparameters fixed to their default values. **(a)**: Varying base measure variance σ. **(b)**:Variance of the random walk model over topic parameters ρ (ρ is drawn in log-scale). **(c)**:Parameter of time-decaying kernel λ (Δ is fixed at 13 in this specific experiments)

a timeline of topic birth and evolution as well as a timeline for each topic that shows its trend over time and the papers with the highest weight of this topic in its mixing vector. This information provides a bird's eye view of the collection, and can be used as input to a summarization system for each topic. In the future, we plan to extend our Gibbs sampler to sample all the hyperparameters of the model. We also plan to extend our model to evolve an HDP at various levels, for instance, lower levels might correspond to conferences, and the highest level to time. This framework will enable us to understand topic evolution within and across different conferences or disciplines.

5

INFINITE STORYLINES FROM STREAMING TEXT

5.1 INTRODUCTION

Internet news portals provide an increasingly important service for information dissemination. For good performance they need to provide essential capabilities to the reader:

Clustering: Given the high frequency of news articles — in considerable excess of one article per second even for quality English news sites — it is vital to group similar articles together such that readers can sift through relevant information quickly.

Timelines: Aggregation of articles should not only occur in terms of current articles but it should also account for previous news. This matters considerably for stories that are just about to drop off the radar so that they may be categorized efficiently into the bigger context of related news.

Content analysis: We would like to group content at three levels of organization: high-level topics, individual stories, and entities. For any given story, we would like to be able to identify the most relevant topics, and also the individual entities that distinguish this event from others which are in the same overall topic. For example, while the topic of the story might be the death of a pop star, the identity *Michael Jackson* will help distinguish this story from similar stories.

Online processing: As we continually receive news documents, our understanding of the topics occurring in the event stream should improve. This is not *necessarily* the case for simple clustering models — increasing the amount of data, along time, will simply increase the number of clusters.Yet topic models are unsuitable for direct analysis since they do not reason well at an individual event level.

The above desiderata are often served by *separate* algorithms which cluster, annotate, and classify news. Such an endeavour can be costly in terms of required editorial data and engineering support. Instead, we propose a *unified* statistical model to satisfy all demands simultaneously. We show how this model can be applied to data from a major Internet News portal.

From the view of statistics, LDA and clustering serve two rather incomparable goals, both of which are suitable to address the above problems partially. Yet, each of these tools in isolation is quite unsuitable to address the challenge.

Clustering is one of the first tools one would suggest to address the problem of news aggregation. However, it is deficient in three regards: the number of clusters is a linear function of the number of days (assuming that the expected number of stories per day is constant), yet models such as Dirichlet Process Mixtures [16] only allow for a logarithmic or sublinear growth in clusters. Secondly, clusters have a strong aspect of temporal coherence. While both aspects can be addressed by the Recurrent Chinese Restaurant Process [9], clustering falls short of a third requirement: the model accuracy does not improve in a meaningful way as we obtain more data — doubling the time span covered by the documents simply

doubles the number of clusters. But it contributes nothing to improving our language model.

Topic Models excel at the converse: They provide insight into the *content* of documents by exploiting exchangeability rather than independence when modeling documents [29]. This leads to highly intuitive and human understandable document representations, yet they are not particularly well-suited to clustering and grouping documents. For instance, they would not be capable of distinguishing between the affairs of two *different* athletes, provided that they play related sports, even if the *dramatis personae* were different. We address this challenge by building a *hierarchical* Bayesian model which contains topics at its top level and clusters drawn from a Recurrent Chinese Restaurant Process at its bottom level. In this sense it is related to Pachinko Allocation [81] and the Hierarchical Dirichlet Process [116]. One of the main differences to these models is that we mix *different datatypes*, i.e. distributions and clusters. This allows us to combine the strengths of both methods: as we obtain more documents, topics will allow us to obtain a more accurate representation of the data stream. At the same time, clusters will provide us with an accurate representation of related news articles.

A key aspect to estimation in graphical models is scalability, in particular when one is concerned with news documents arriving at a rate in excess of 1 document per second (considerably higher rates apply for blog posts). There has been previous work on scalable inference, starting with the collapsed sampler representation for LDA [58], efficient sampling algorithms that exploit sparsity [133], distributed implementations [112, 17], and Sequential Monte Carlo (SMC) estimation [32]. The problem of efficient inference is exacerbated in our case since we need to obtain an *online* estimate, that is, we need to be able to generate clusters essentially on the fly as news arrives and to update topics accordingly. We address this by designing an SMC sampler which is executed in parallel by allocating particles to cores. The datastructure is a variant of the tree described by [32]. Our experiments demonstrate both the scalability and accuracy of our approach when compared to editorially curated data of a major Internet news portal.

5.2 STATISTICAL MODEL

In a nutshell, our model emulates the process of news articles. We assume that stories occur with an approximately even probability over time. A story is characterized by a mixture of topics and the names of the key entities involved in it. Any article discussing this story then draws its words from the topic mixture associated with the story, the associated named entities, and any story-specific words that are not well explained by the topic mixture. The latter modification allows us to improve our estimates for a given story once it becomes popular. In summary, we model news story clustering by applying a topic model to the clusters, while simultaneously allowing for cluster generation using the Recurrent Chinese Restaurant Process (RCRP).

Such a model has a number of advantages: estimates in topic models increase with the amount of data available, hence twice as much data will lead to correspondingly improved topics. Modeling a story by its mixture of topics ensures that we have a plausible cluster model right from the start, even after observing

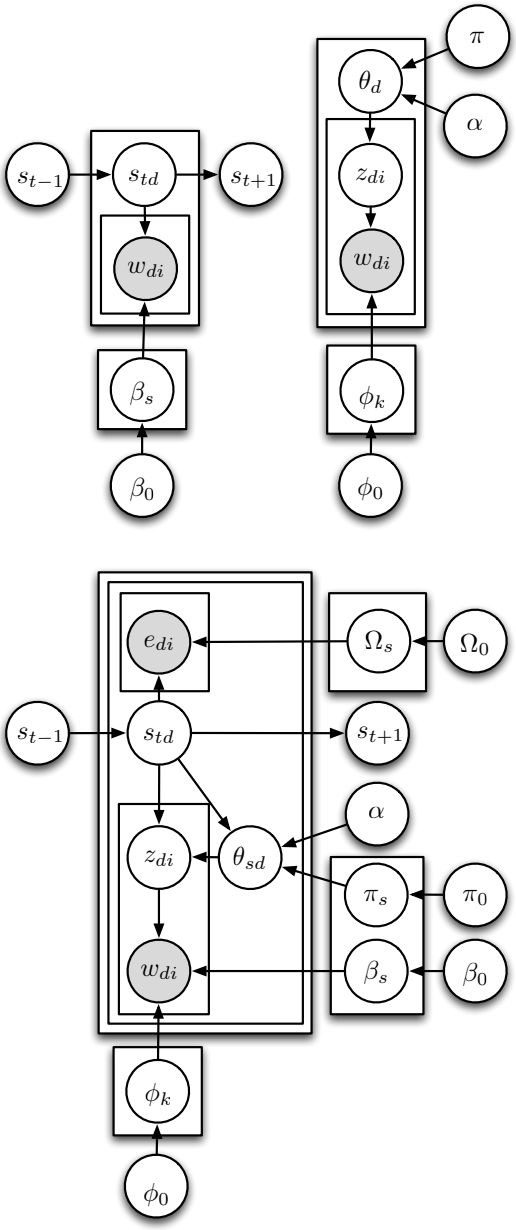


Figure 5.1: Plate diagram of the models. Top left: Recurrent Chinese Restaurant Process clustering; Top right: Latent Dirichlet Allocation; Bottom: Topic-Cluster model.

only one article for a new story. Third, the RCRP ensures a continuous flow of new stories over time. Finally, a distinct named entity model ensures that we capture the characteristic terms rapidly.

Note that while the discussion in this chapter focuses on topic models for dealing with news articles, the applicability of the topic-cluster model is considerably more broad. For instance, in modeling natural images it is reasonable to assume that scenes are composed of basic topical constituents, yet it is equally reasonable to assume that the particular arrangement of these constituents also matters to distinguish images. There we could use the topics to represent constituents and the cluster model to group imagery according to scene arrangement information.

Likewise, our model has applications in bioinformatics, *e.g.* in modeling haplotype distributions.

5.2.1 *Recurrent Chinese Restaurant Process*

A critical feature for disambiguating storylines is time. Stories come and go, and it makes little sense to try to associate a document with a storyline that has not been seen over a long period of time. We turn to the Recurrent Chinese Restaurant Process [9], which generalizes the well-known Chinese Restaurant Process (CRP) [104] to model partially exchangeable data like document streams. The RCRP provides a nonparametric model over storyline strength, and permits sampling-based inference over a potentially unbounded number of stories.

For concreteness, we need to introduce some notation: we denote time (epoch) by t, documents by d, and the position of a word w_{di} in a document d by i. The story associated with document d is denoted by s_d (or s_{dt} if we want to make the dependence on the epoch t explicit). Documents are assumed to be divided into epochs (e.g., one hour or one day); we assume exchangeability only within each epoch. For a new document at epoch t, a probability mass proportional to γ is reserved for generating a new storyline. Each existing storyline may be selected with probability proportional to the sum $m_{st} + m'_{st}$, where m_{st} is the number of documents at epoch t that belong to storyline s, and m'_{st} is the prior weight for storyline s at time t. Finally, we denote by β_s the word distribution for story s and we let β_0 be the prior for word distributions. We compactly write

$$s_{td}|\mathbf{s}_{1:t-1}, \mathbf{s}_{t,1:d-1} \sim \text{RCRP}(\gamma, \lambda, \Delta) \tag{5.1}$$

to indicate the distribution

$$P(s_{td}|\mathbf{s}_{1:t-1}, \mathbf{s}_{t,1:d-1}) \propto \begin{cases} m'_{st} + m_{st}^{-td} & \text{existing story} \\ \gamma & \text{new story} \end{cases} \tag{5.2}$$

As in the original CRP, the count m_{st}^{-td} is the number of documents in storyline s at epoch t, not including d. The temporal aspect of the model is introduced via the prior m'_{st}, which is defined as

$$m'_{st} = \sum_{\delta=1}^{\Delta} e^{-\frac{\delta}{\lambda}} m_{s,t-\delta}. \tag{5.3}$$

This prior defines a time-decaying kernel, parametrized by Δ (width) and λ (decay factor). When $\Delta = 0$ the RCRP degenerates to a set of independent Chinese Restaurant Processes at each epoch; when $\Delta = T$ and $\lambda = \infty$ we obtain a global CRP that ignores time. In between, the values of these two parameters affect the expected life span of a given component, such that the lifespan of each storyline follows a power law distribution [9]. The graphical model is given on the top left in Figure 5.1.

We note that dividing documents into epochs allows for the cluster strength at time t to be efficiently computed, in terms of the components (m, m') in (5.2). Alternatively, one could define a continuous, time-decaying kernel over the time stamps

of the documents. When processing document d at time t' however, computing any story's strength would now require summation over all earlier documents associated with that story, which is non-scalable. In the news domain, taking epochs to be one day long means that the recency of a given story decays only at epoch boundaries, and is captured by m'. A finer epoch resolution and a wider Δ can be used without affecting computational efficiency – it is easy to derive an iterative update $m'_{s,t+1} = \exp^{-1/\lambda}(m_{st} + m'_{st}) - \exp^{-(\Delta+1)/\lambda} m_{s,t-(\Delta+1)}$, which has constant runtime w.r.t. Δ.

5.2.2 *Topic Models*

The second component of the topic-cluster model is given by Latent Dirichlet Allocation [29], as described in the top right of Figure 5.1. Rather than assuming that documents belong to clusters, we assume that there exists a topic distribution θ_d for document d and that each word w_{di} is drawn from the distribution ϕ_t associated with topic z_{di}. Here ϕ_0 denotes the Dirichlet prior over word distributions. Finally, θ_d is drawn from a Dirichlet distribution with mean π and precision α.

1. For all topics t draw
   a) word distribution ϕ_k from word prior ϕ_0
2. For each document d draw
   a) topic distribution θ_d from Dirichlet prior (π, α)
   b) For each position (d, i) in d draw
      i. topic z_{di} from topic distribution θ_d
      ii. word w_{di} from word distribution $\phi_{z_{di}}$

The key difference to the basic clustering model is that our estimate will improve as we receive more data. When using a nonparametric version, i.e. a Dirichlet process, we would end up adding more topics as more data arrives while simultaneously refining the estimates of current topics.

5.2.3 *Time-Dependent Topic-Cluster Model*

We now proceed to defining the contribution of the present chapter — a time-dependent topic-cluster model. While some of the design choices (e.g. the fact that we treat named entities differently) are specific to news, the generic model is more broadly applicable.

We now combine clustering and topic models into our proposed storylines model by imbuing each storyline with a Dirichlet distribution over topic strength vectors with parameters (π, α). For each article in a storyline the topic proportions θ_d are drawn from this Dirichlet distribution.

Words are drawn either from the storyline or one of the topics. This can be modeled by adding an element $K + 1$ to the topic proportions θ_d. If the latent topic indicator $z_n \leqslant K$, then the word is drawn from the topic ϕ_{z_n}; otherwise it is drawn from a distribution linked to the storyline β_s. This story-specific distribution captures the burstiness of the characteristic words in each story.

Topic models usually focus on individual words, but news stories often center around specific people and locations For this reason, we extract named entities e_{di} from text in a preprocessing step, and model their generation directly. Note that we make no effort to resolve names "Barack Obama" and "President Obama" to a single underlying semantic entity, but we do treat these expressions as single tokens in a vocabulary over names.

1. For each topic $k \in 1 \dots K$, draw a distribution over words $\phi_k \sim \text{Dir}(\phi_0)$
2. For each document $d \in \{1, \cdots, D_t\}$:
   a) Draw the storyline indicator $s_{td} | \mathbf{s}_{1:t-1}, \mathbf{s}_{t,1:d-1} \sim \text{RCRP}(\gamma, \lambda, \Delta)$
   b) If s_{td} is a new storyline,
      i. Draw a distribution over words $\beta_{s_{new}} | G_0 \sim \text{Dir}(\beta_0)$
      ii. Draw a distribution over named entities $\Omega_{s_{new}} | G_0 \sim \text{Dir}(\Omega_0)$
      iii. Draw a Dirichlet distribution over topic proportions $\mathbf{1}_{s_{new}} | G_0 \sim \text{Dir}(\pi_0)$
   c) Draw the topic proportions $\theta_{td} | s_{td} \sim \text{Dir}(\alpha \mathbf{1}_{s_{td}})$
   d) Draw the words $\mathbf{w}_{td} | s_{td} \sim \text{LDA}\left(\theta_{s_{td}}, \{\phi_1, \cdots, \phi_K, \beta_{s_{td}}\}\right)$
   e) Draw the named entities $\mathbf{e}_{td} | s_{td} \sim \mathcal{M}\Omega_{s_{td}}$

where $\text{LDA}\left(\theta_{s_{td}}, \{\phi_1, \cdots, \phi_K, \beta_{s_{td}}\}\right)$ indicates a probability distribution over word vectors in the form of a Latent Dirichlet Allocation model [29] with topic proportions $\theta_{s_{td}}$ and topics $\{\phi_1, \cdots, \phi_K, \beta_{s_{td}}\}$. The hyperparameters $\beta_0, \phi_0, \Omega_0, \pi_0$ are all symmetric Dirichlet.

t	time
d	document
(t, d)	document d in epoch t
(d, i)	position i in document d (word or entity)
s_{td}	story associated with document d
s_t	aggregate cluster variables at time t
e_{di}	entity at position i in document d
z_{di}	topic at position i in document d
w_{di}	word at position i in document d
θ_{sd}	topic distribution for document d
Ω_s	Entity distribution for story s
Ω_0	prior for entity distributions
α	Strength of Dirichlet prior over topics
π_0	Mean of Dirichlet prior over topics
π_s	Mean of Dirichlet prior for story s
β_s	word distribution for story specific "topic" for story s
β_0	Dirichlet prior for story-word distribution
ϕ_k	word distributions for topic k.
ϕ_0	Dirichlet prior for topic-word distribution.

5.3 ONLINE SCALABLE INFERENCE

Our goal is to compute *online* the posterior distribution $P(z_{1:T}, \mathbf{s}_{1:T}|x_{1:T})$, where $x_t, z_t, \mathbf{s}_t$ are shorthands for all of the documents at epoch t ($x_{td} = \langle \mathbf{w}_{td}, \mathbf{e}_{td} \rangle$), the topic indicators at epoch t and story indicators at epoch t. Markov Chain Monte Carlo (MCMC) methods which are widely used to compute this posterior are inherently batch methods and do not scale well to the amount of data we consider. Furthermore they are unsuitable for streaming data.

5.3.1 *Sequential Monte Carlo*

Instead, we apply a sequential Monte Carlo (SMC) method known as particle filters [46]. Particle filters approximate the posterior distribution over the latent variables up until document $t, d-1$, i.e. $P(z_{1:t,d-1}, \mathbf{s}_{1:t,d-1}|x_{1:t,d-1})$, where $(1:t, d)$ is a shorthand for all documents up to document d at time t. When a new document td arrives, the posterior is updated yielding $P(z_{1:td}, \mathbf{s}_{1:td}|x_{1:td})$. The posterior approximation is maintained as a set of weighted *particles* that each represent a hypothesis about the hidden variables; the weight of each particle represents how well the hypothesis maintained by the particle explains the data.

The structure is described in Algorithms 1 and 2. The algorithm processes **one document at a time** in the order of arrival. This should not be confused with the

Algorithm 1 A Particle Filter Algorithm

Initialize ω_1^f to $\frac{1}{F}$ for all $f \in \{1, \dots F\}$
for each document d with time stamp t **do**
  for $f \in \{1, \dots F\}$ **do**
    Sample s_{td}^f, z_{td}^f using MCMC
    $\omega^f \leftarrow \omega^f P(x_{td}|z_{td}^f, \mathbf{s}_{td}^f, x_{1:t,d-1})$
  end for
  Normalize particle weights
  if $\|\omega_t\|_2^{-2} <$ threshold **then**
    resample particles
    for $f \in \{1, \dots F\}$ **do**
      MCMC pass over 10 random past documents
    end for
  end if
end for

Algorithm 2 MCMC over document td

$q(s) = P(s|\mathbf{s}_{t-\Delta:t}^{-td})P(\mathbf{e}_{td}|s, \text{rest})$
for iter $= 0$ **to** MAXITER **do**
  for each word w_{tdi} **do**
    Sample z_{tdi} using (5.4)
  end for
  if iter $= 1$ **then**
    Sample s_{td} using (5.5)
  else
    Sample $s*$ using $q(s)$
    $r = \frac{P(z_{td}|s*,\text{rest})P(\mathbf{w}_{td}^{K+1}|s*,\text{rest})}{P(z_{td}|s_{td},\text{rest})P(\mathbf{w}_{td}^{K+1}|s_{td},\text{rest})}$
    Accept $s_{td} \leftarrow s^*$ with probability $\min(r, 1)$
  end if
end for
Return z_{td}, s_{td}

time stamp of the document. For example, we can chose the epoch length to be a full day but still process documents inside the same day as they arrive (although they all have the same timestamp). The main ingredient for designing a particle filter is the proposal distribution $Q(z_{td}, s_{td}|z_{1:t,d-1}, \mathbf{s}_{1:t,d-1}, x_{1:td})$. Usually this proposal is taken to be the prior distribution $P(z_{td}, s_{td}|z_{1:t,d-1}, \mathbf{s}_{1:t,d-1})$ since computing the posterior is hard. We take Q to be the posterior $P(z_{td}, s_{td}|z_{1:t,d-1}, \mathbf{s}_{1:t,d-1}, x_{1:td})$, which minimizes the variance of the resulting particle weights [46]. Unfortunately computing this posterior for a single document is intractable, thus we use MCMC and run a Markov chain over (z_{td}, s_{td}) whose equilibrium distribution is the sought-after posterior. The exact sampling equations af s and z_{td} are given below. This idea was inspired by the work of [68] who used a restricted Gibbs scan over a set of coupled variables to define a proposal distribution, where the proposed value of the variables is taken to be the last sample. Jain and Neal used this idea in

the context of an MCMC sampler, here we use it in the context of a sequential importance sampler (i.e. SMC).

Sampling topic indicators: For the topic of word i in document d and epoch t, we sample from

$$P(z_{tdi} = k | w_{tdi} = w, s_{td} = s, \text{rest}) = \frac{C_{tdk}^{-i} + \alpha \frac{C_{sk}^{-i} + \pi_0}{C_{s\cdot}^{-i} + \pi_0 (K+1)}}{C_{td\cdot}^{-i} + \alpha} \frac{C_{kw}^{-i} + \phi_0}{C_{k\cdot}^{-i} + \phi_0 W} \tag{5.4}$$

where `rest` denotes all other hidden variables, C_{tdk}^{-i} refers to the count of topic k and document d in epoch t, not including the currently sampled index i; C_{sk}^{-i} is the count of topic k with story s, while C_{kw}^{-i} is the count of word w with topic k (which indexes the story if $k = K+1$); traditional dot notation is used to indicate sums over indices (e.g. $C_{td\cdot}^{-i} = \sum_k C_{tdk}^{-i}$). Note that this is just the standard sampling equation for LDA except that the prior over the document's topic vector θ is replaced by it's story mean topic vector.

Sampling story indicators: The sampling equation for the storyline s_{td} decomposes as follows:

$$\begin{aligned} P(s_{td} | \mathbf{s}_{t-\Delta:t}^{-td}, z_{td}, \mathbf{e}_{td}, \mathbf{w}_{td}^{K+1}, \text{rest}) &\propto \underbrace{P(s_{td} | \mathbf{s}_{t-\Delta:t}^{-td})}_{\text{Prior}} \\ \times \underbrace{P(z_{td} | s_{td}, \text{rest}) P(\mathbf{e}_{td} | s_{td}, \text{rest}) P(\mathbf{w}_{td}^{K+1} | s_{td}, \text{rest})}_{\text{Emission}} & \end{aligned} \tag{5.5}$$

where the prior follows from the RCRP (5.2), $\mathbf{w}_{td}^{K+1}$ are the set of words in document d sampled from the story specific language model $\beta_{s_{td}}$, and the emission terms for $\mathbf{w}_{td}^{K+1}, \mathbf{e}_{td}$ are simple ratios of partition functions. For example, the emission term for entities, $P(\mathbf{e}_{td} | s_{td} = s, \text{rest})$ is given by:

$$\frac{\Gamma\left(\sum_{e=1}^{E} [C_{se}^{-td} + \Omega_0]\right)}{\Gamma\left(\sum_{e=1}^{E} [C_{td,e} + C_{se}^{-td} + \Omega_0]\right)} \prod_{e=1}^{E} \frac{\Gamma\left(C_{td,e} + C_{se}^{-td} + \Omega_0\right)}{\Gamma\left(C_{se}^{-td} + \Omega_0\right)} \tag{5.6}$$

Since we integrated out θ, the emission term over z_{td} does not have a closed form solution and is computed using the chain rule as follows:

$$P(z_{td} | s_{td} = s, \text{rest}) = \prod_{i=1}^{n_{td}} P(z_{tdi} | s_{td} = s, z_{td}^{-td,(n \geqslant i)}, \text{rest}) \tag{5.7}$$

where the superscript $-td, (n \geqslant i)$ means excluding all words in document td after, and including, position i. Terms in the product are computed using (5.4).

We alternate between sampling (5.4) and (5.5) for 20 iterations. Unfortunately, even then the chain is too slow for online inference, because of (5.7) which scales linearly with the number of words in the document. In addition we need to compute this term for every active story. To solve this we use a proposal distribution

$$q(s) = P(s_{td} | \mathbf{s}_{t-\Delta:t}^{-td}) P(\mathbf{e}_{td} | s_{td}, \text{rest})$$

whose computation scales linearly with the number of entities in the document. We then sample $s*$ from this proposal and compute the acceptance ratio r which is simply

$$r = \frac{P(z_{td} | s*, \text{rest}) P(\mathbf{w}_{td}^{K+1} | s*, \text{rest})}{P(z_{td} | s_{td}, \text{rest}) P(\mathbf{w}_{td}^{K+1} | s_{td}, \text{rest})}.$$

Thus we need only to compute (5.7) twice per MCMC iteration. Another attractive property of the proposal distribution $q(s)$ is that the proposal is *constant* and does not depend on z_{td}. As made explicit in Algorithm 2 we precompute it once for the entire MCMC sweep. Finally, the unnormalized importance weight for particle f after processing document td, can be shown to be equal to:

$$\omega^f \leftarrow \omega^f P(x_{td}|z_{td}^f, s_{td}^f, x_{1:t,d-1}), \tag{5.8}$$

which has the intuitive explanation that the weight for particle f is updated by multiplying the marginal probability of the new observation x_t, which we compute from the last 10 samples of the MCMC sweep over a given document. Finally, if the effective number of particles $\|\omega_t\|_2^{-2}$ falls below a threshold we stochastically replicate each particle based on its normalized weight. To encourage diversity in those replicated particles, we select a small number of documents (10 in our implementation) from the recent 1000 documents, and do a single MCMC sweep over them, and then finally reset the weight of each particle to uniform.

We note that an *alternative approach* to conducting the particle filter algorithm would sequentially order s_{td} followed by z_{td}. Specifically, we would use $q(s)$ defined above as the proposal distribution over s_{td}, and then sample z_{td} sequentially using Eq (5.4) conditioned on the sampled value of s_{td}. However, this approach requires a huge number of particles to capture the uncertainty introduced by sampling s_{td} before actually seeing the document, since z_{td} and s_{td} are tightly coupled. Moreover, our approach results in less variance over the posterior of (z_{td}, s_{td}) and thus requires fewer particles, as we will demonstrate empirically.

5.3.2 *Speeding up the Sampler*

While the sampler specified by (5.4) and (5.5) and the proposal distribution $q(s)$ are efficient, they scale linearly with the number of topics and stories. [133] noted that samplers that follow (5.4) can be made more efficient by taking advantage of the sparsity structure of the word-topic and document-topic counts: each word is assigned to only a few topics and each document (story) addresses only a few topics. We leverage this insight here and present an extended, efficient data structure in Section 5.3.3 that is suitable for particle filtering.

We first note that (5.4) follows the standard form of a collapsed Gibbs sampler for LDA, albeit with a story-specific prior over θ_{td}. We make the approximation that the document's story-specific prior is constant while we sample the document, i.e. the counts C_{sk}^{-i} are constants. This turns the problem into the same form addressed in [133]. The mass of the sampler in (5.4) can be broken down into three parts: prior mass, document-topic mass and word-topic mass. The first is dense and constant (due to our approximation), while the last two masses are sparse. The document-topic mass tracks the non-zero terms in C_{tdk}^{-i}, and the word-topic mass tracks the non-zero terms in C_{kw}^{-i}.

The sum of each of these masses can be computed once at the beginning of the sampler. The document-topic mass can be updated in $O(1)$ after each word [133], while the word-topic mass is very sparse and can be computed for each word in nearly constant time. Finally the prior mass is only re-computed when

the document's story changes. Thus the cost of sampling a given word is almost *constant* rather than $O(k)$ during the execution of Algorithm 1.

Unfortunately, the same idea can not be applied to sampling s, as each of the components in (5.5) depends on *multiple* terms (see for example (5.6)). Their products do not fold into separate masses as in (5.4). Still, we note that the entity-story counts are sparse ($C_{se}^{-td} = 0$), thus most of the terms in the product component ($e \in E$) of (5.6) reduce to the form $\Gamma(C_{td,e} + \Omega_0)/\Gamma(\Omega_0)$. Hence we simply compute this form once for all stories with $C_{se}^{-td} = 0$; for the *few* stories having $C_{se}^{-td} > 0$, we explicitly compute the product component. We also use the same idea for computing $P(\mathbf{w}_{td}^{K+1}|s_{td}, \text{rest})$. With these choices, the entire MCMC sweep for a given document takes around *50-100ms* when using `MAXITER = 15` and $K = 100$ as opposed to *200-300ms* for a naive implementation.

Hyperparameters:

The hyperparameters for topic, word and entity distributionss, ϕ_0, Ω_0 and β_0 are *optimized* as described in [122] every 200 documents. The topic-prior $\pi_{0,1:K+1}$ is modeled as asymmetric Dirichlet prior and is also *optimized* as in [122] every 200 documents. For the RCRP, the hyperparameter γ_t is epoch-specific with a Gamma(1,1) prior; we *sample* its value after every batch of 20 documents [49]. The kernel parameters are set to $\Delta = 3$ and $\lambda = 0.5$ — results were robust across a range of settings. Finally we set $\alpha = 1$

5.3.3 *Implementation and Storage*

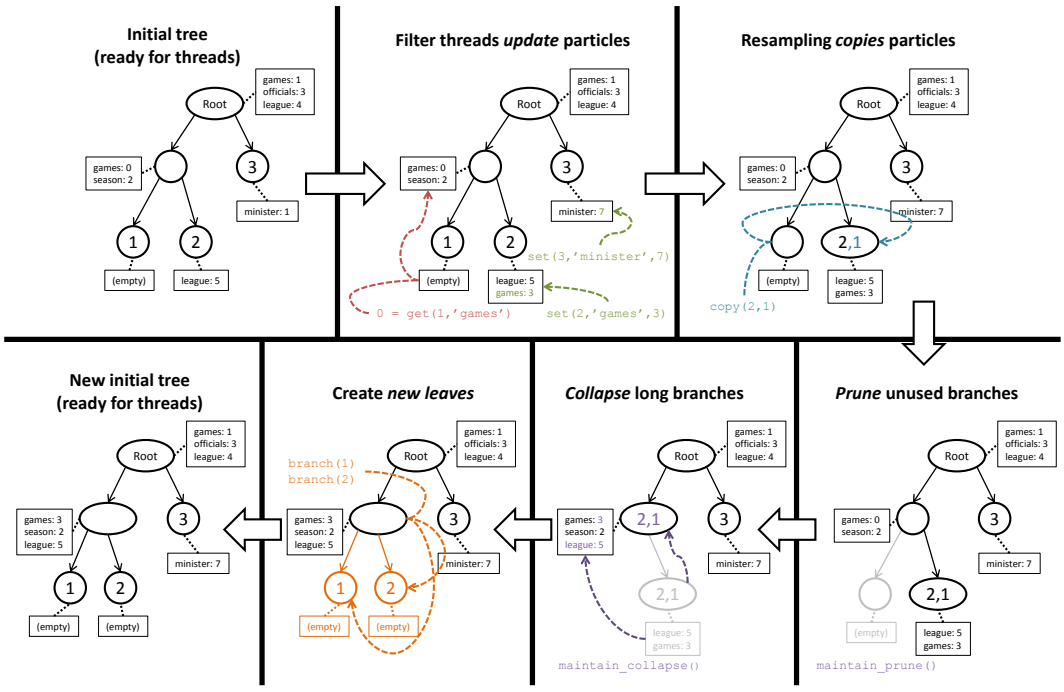


Figure 5.2: Inheritance tree operations in the context of our SMC algorithm. Numbers within a vertex represent associated particles. Each vertex's hash map is represented by a table, connected by a dotted line.

Implementing our parallel SMC algorithm for large datasets poses runtime and memory challenges. While our algorithm mostly works on single particles by spawning one computational thread per particle filter, it also has to copy entire particles during particle re-sampling (done via a master thread). Hence we require a *thread-safe* data structure that supports fast updates of individual particles' data, *and* fast copying of particles.

It should be obvious that the naive implementation, where each particle has its own set of arrays for storage, is very inefficient when it comes to particle resampling — in the worst case, we would have to duplicate all particle arrays element-by-element. Worse, our memory requirements would grow linearly in the number of particles, making large data streams impractical even for modest numbers of particles.

INHERITANCE TREES Instead, we employ an idea from Canini *et al.* [32], in which particles maintain a memory-efficient representation called an "inheritance tree". In this representation, each particle is associated with a tree vertex, which stores the actual data. The key idea is that child vertices inherit their ancestors' data, so they need only store changes relative to their ancestors, in the form of a dictionary or hash map. To save memory, data elements with value 0 are not explictly represented unless necessary (e.g. when a parent has nonzero value). New vertices are created only when writing data, and only under two circumstances: first, when the particle to be changed shares a vertex with other particles, and second, when the particle to be changed is associated with an interior vertex. In both cases, a new leaf vertex is created for the particle in question.

This representation dramatically reduces memory usage for large numbers of particles, and also makes particle replication a constant runtime operation. The tradeoff however, is that data retrieval becomes linear time in the depth of the tree, although writing data remains (amortized) constant time. This disadvantage can be mitigated via tree maintenance operations, in which we *prune* branches without particles and then *collapse* unnecessary long branches — refer to Figure 5.2 for an example. With tree maintenance, data retrieval becomes a practically constant time operation.

THREAD SAFETY Thus far, we have only described Canini *et al.*'s version of the inheritance tree, which is *not* thread-safe. To see why, consider what happens when particle 1 is associated with the parent vertex of particle 2. If a thread writes to particle 1 while another is reading from particle 2, it may happen that the second thread needs to read from the parent vertex. This creates a race condition, which is unacceptable for our parallel algorithm.

To ensure thread safety, we augment the inheritance tree by requiring every particle to have its own *leaf* in the tree. This makes particle writes thread-safe, because no particle is ever an ancestor of another, and writes only go to the particle itself, never to ancestors. Furthermore, every particle is associated with only one computational thread, so there will never be simultaneous writes to the *same* particle. On the other hand, data reads, even to ancestors, are inherently thread-safe and present no issue. To maintain this requirement, observe that particle-vertex associations can only change during particle resampling, which is handled by a a

master thread. Immediately after resampling, we *branch* off a new leaf for every particle at an interior node. Once this is done, the individual filter threads may be run safely in parallel.

In summary, the inheritance tree has four operations, detailed in Figure 5.2:

1. `branch(f)`: creates a new leaf for particle `f`.
2. `get(f,i)`, `set(f,i,value)`: retrieve/write data elements `i` for particle `f`.
3. `copy(f,g)`: replicate particle `f` to `g`.
4. `maintain()`: prune particle-less branches and collapse unnecessary long branches.

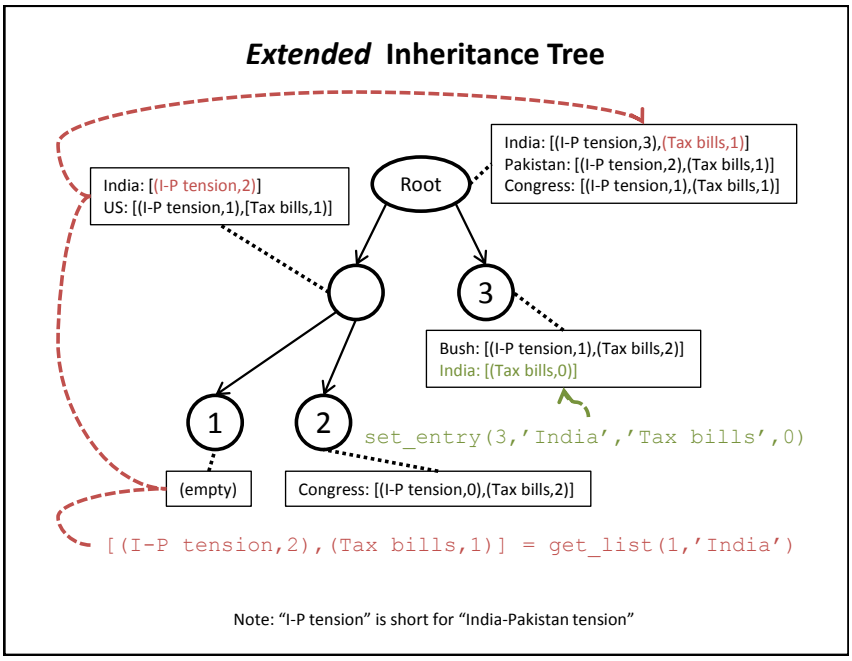


Figure 5.3: Operations on an *extended inheritance tree*, which stores sets of objects in particles, shown as lists in tables connected to particle-numbered tree nodes. Our algorithm requires particles to store some data as sets of objects instead of arrays — in this example, for every named entity, e.g. "Congress", we need to store a set of (story,association-count) pairs, e.g. ("Tax bills",2). The extended inheritance tree allows (a) the particles to be replicated in constant-time, and (b) the object sets to be retrieved in amortized linear time. Notice that every particle is associated with a leaf, which ensures thread safety during write operations. Internal vertices store entries common to leaf vertices.

EXTENDED INHERITANCE TREES Our thread-safe inheritance tree supports most of our data storage needs. However, parts of our algorithm require storage of *sets of objects*, rather than integer values. For example, our story sampling equation (5.5) needs the set of stories associated with each named entity, as well as the number of times each story-to-entity association occurs.

To solve this problem, we extend the basic inheritance tree by making its hash maps store *other hash maps* as values. These second-level hash maps then store objects as key-value pairs; note that individual objects *can be shared with parent*

vertices — see Figure 5.3. Using the story sampling equation (5.5) as an example, the first-level hash map uses named entities as keys, and the second-level hash map uses stories as keys and association counts as values (Figure 5.3 shows an example with stories taken from Figure 5.4).

Observe that the count for a particular story-entity association can be retrieved or updated in amortized constant time. Retrieving all associations for a given entity is usually linear in the number of associations. Finally note that the list associated with each key (NE or word) is not sorted as in [133] as this will prevent sharing across particles. Nevertheless, our implementation balances storage and execution time.

5.4 EXPERIMENTS

We examine our model on *three* English news samples of varying sizes extracted from Yahoo! News over a two-month period. Details of the three news samples are listed in Table 5.1. We use the named entity recognizer in [138], and we remove common stop-words and tokens which are neither verbs, nor nouns, nor adjectives. We divide each of the samples into a set of 12-hour epochs (corresponding to AM and PM time of the day) according to the article publication date and time. For all experiments, we use 8-particles running on an 8-core machine, and unless otherwise stated, we set **MAXITER**=15.

5.4.1 *Structured Browsing*

In Figure 5.4 we present a qualitative illustration of the utility of our model for structure browsing. The storylines include the UEFA soccer championships, a tax bill under consideration in the United States, and tension between India and Pakistan. Our model identifies connections between these storylines and relevant high-level topics: the UEFA story relates to a more general topic about sports; both the tax bill and the India-Pakistan stories relate to the politics topics, but only the latter story relates to the topic about civil unrest. Note that each storyline contains a plot of strength over time; the UEFA storyline is strongly multimodal, peaking near the dates of matches. This demonstrates the importance of a flexible nonparametric model for time, rather than using a unimodal distribution.

End-users can take advantage of the organization obtained by our model by browsing the collection of high-level topics and then descend to specific stories indexed under each topic — like opening a physical newspaper and going directly to the sports page. However, our model provides a number of other affordances for structured browsing which were not possible under previous approaches. Figure 5.5 shows two examples. First, users can request similar stories by topic: starting from a story about India-Pakistan tension, the system returns a story about the Middle-east conflict, which also features topics such as diplomacy, politics, and unrest. In addition, users can focus their query with specific keywords or entities: the right side of Figure 5.5 shows the result of querying for similar stories but requiring the keyword *nuclear* to have high salience in the term probability vector of any story returned by the query. Similarly, users might require a specific entity — for example, accepting only stories that are a topical match and include the entity

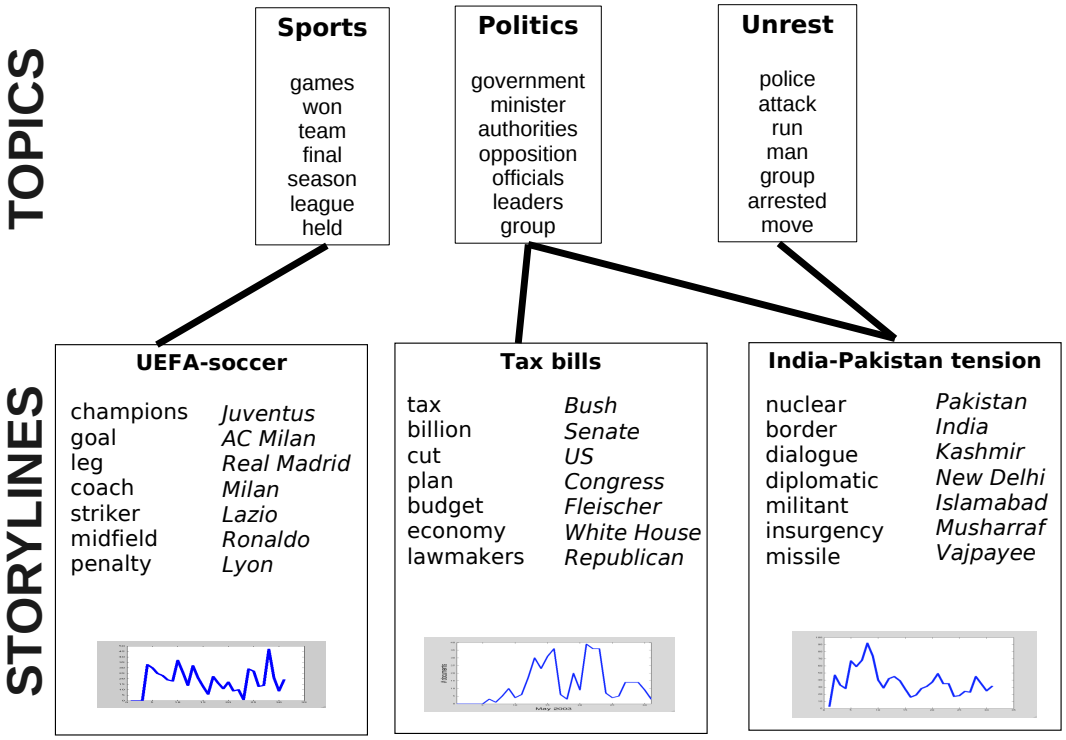


Figure 5.4: Some example storylines and topics extracted by our system. For each storyline we list the top words in the left column, and the top named entities at the right; the plot at the bottom shows the storyline strength over time. For topics we show the top words. The lines between storylines and topics indicate that at least 10% of terms in a storyline are generated from the linked topic.

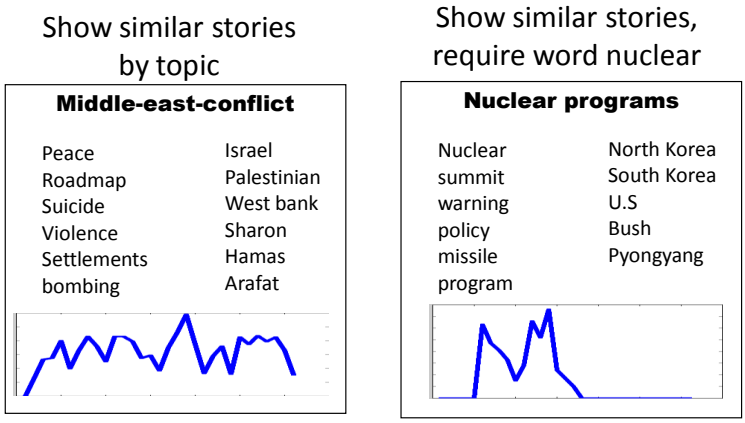


Figure 5.5: An example of structure browsing of documents related to the India-Pakistan tensions (see text for details).

Vajpayee. This combination of topic-level analysis with surface-level matching on terms or entities is a unique contribution of our model, and was not possible with previous technology.

5.4.2 *Evaluating Clustering Accuracy*

We evaluate the clustering *accuracy* of our model over the Yahoo! news datasets. Each dataset contains 2525 editorially judged "must-link" (45%) and "cannot-link" (55%) article pairs. Must-link pairs refer to articles in the same story, whereas cannot-link pairs are not related.

For the sake of evaluating clustering, we compare against a variant of a strong 1-NN (single-link clustering) baseline [40]. This simple baseline is the best performing system on TDT2004 task and was shown to be competitive with Bayesian models [137]. This method finds the closest 1-NN for an incoming document among all documents seen thus far. If the distance to this 1-NN is above a threshold, the document starts a new story, otherwise it is linked to its 1-NN. Since this method examines all previously seen documents, it is not scalable to large-datasets. In [103], the authors showed that using locality sensitive hashing (LSH), one can restrict the subset of documents examined with little effect of the final accuracy. Here, we use a similar idea, but we even allow the baseline to be fit *offline*. First, we compute the similarities between articles via LSH [62, 56], then construct a pairwise similarity graph on which a single-link clustering algorithm is applied to form larger clusters. The single-link algorithm is stopped when no two clusters to be merged have similarity score larger than a threshold *tuned* on a separate validation set (our algorithm has *no access* to this validation set). In the remainder of this paper, we simply refer to this baseline as LSHC.

From Table 5.1, we see that our *online, single-pass* method is competitive with the *off-line* and tuned baseline on all the samples and that the difference in performance is larger for small sample sizes. We believe this happens as our model can isolate story-specific words and entities from background topics and thus can link documents in the same story even when there are few documents in each story.

5.4.3 *Hyperparameter Sensitivity*

We conduct five experiments to study the effect of various model hyperparameters and tuning parameters. First, we study the effect of the number of topics. Table 5.2 shows how performance changes with the number of topics K. It is evident that $K = 50 - 100$ is sufficient. Moreover, since we optimize π_0, the effect of the number of topics is negligible [122] For the rest of the experiments in this section, we use sample-1 with $K = 100$.

Second, we study the number of Gibbs sampling iterations used to process a single document, `MAXITER`. In Figure 5.6, we show how the time to process each document grows with the number of processed documents, for different values of `MAXITER`. As expected, doubling `MAXITER` increases the time needed to process a document, however performance only increases marginally.

Third, we study the effectiveness of optimizing the hyperparameters ϕ_0, β_0 and Ω_0. In this experiment, we turn off hyperparameter optimization altogether, set $\phi_0 = .01$ (which is a common value in topic models), and vary β_0 and Ω_0. The results are shown in Table 5.3. Moreover, when we enable hyperparameter optimization, we obtain $(\phi_0, \beta_0, \Omega_0) = (0.0204, 0.0038, 0.0025)$ with accuracy 0.8289, which demonstrates its effectiveness.

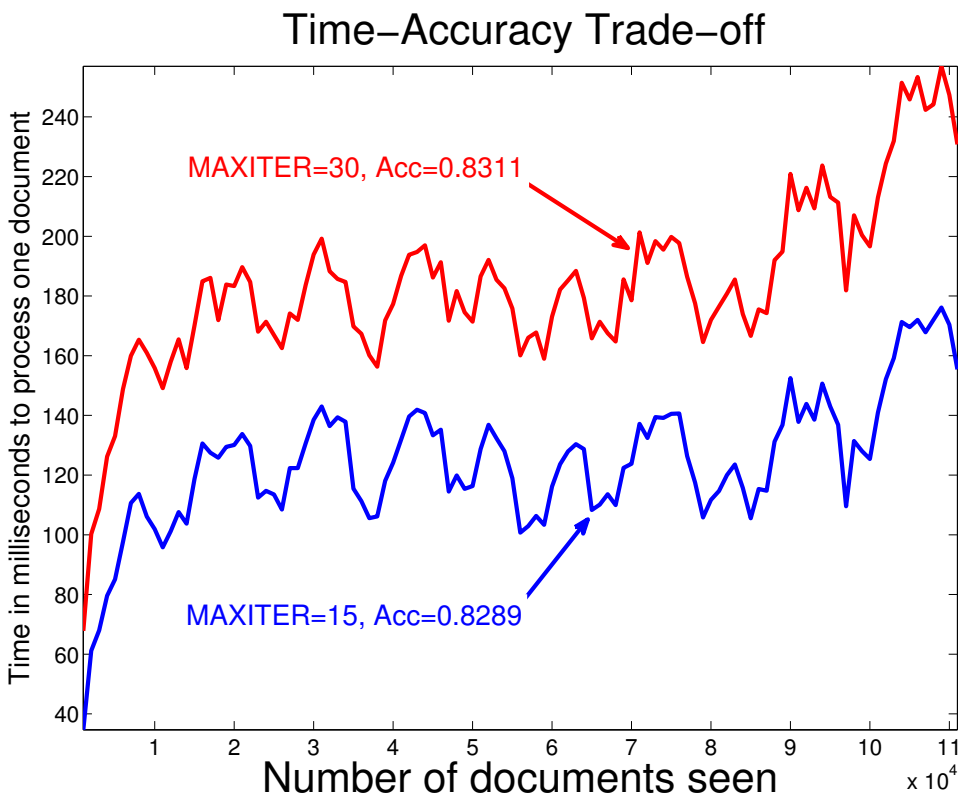


Figure 5.6: Effect of MAXITER, sample-1, K = 100

Fourth, we tested the contribution of each feature of our model (Table 5.4). As evident, each aspect of the model improves performance. We note here that removing time not only makes performance suboptimal, but also causes stories to persist throughout the corpus, eventually increasing running time to an astounding 2 seconds per document.

Finally, we show the effect of the number of particles in Table 5.5. This validates our earlier hypothesis that the restricted Gibbs scan over (z_{td}, s_{td}) results in a posterior with small variance, thus only a few particles are sufficient to get good performance.

5.5 RELATED WORK

Our approach is non-parametric over stories, allowing the number of stories to be determined by the data. In similar fashion [137] describe an online clustering approach using the Dirichlet Process. This work equates storylines with clusters, and does not model high-level topics. Also, non-parametric clustering has been previously combined with topic models, with the cluster defining a distribution over topics [136, 121]. We differ from these approaches in several respects: we incorporate temporal information and named entities, and we permit both the storylines and topics to emit words.

Recent work on topic models has focused on improving scalability; we focus on sampling-based methods, which are most relevant to our approach. Our approach is most influenced by the particle filter of [32], but we differ in that the high-order

Table 5.1: Details of Yahoo! News dataset and corresponding clustering accuracies of the baseline (LSHC) and our method (Story), $K = 100$.

Sample No.	Sample size	Num Words	Num Entities	Story Acc.	LSHC Acc.
1	111,732	19,218	12,475	**0.8289**	0.738
2	274,969	29,604	21,797	**0.8388**	0.791
3	547,057	40,576	32,637	**0.8395**	0.800

Table 5.2: Clustering accuracies vs. number of topics.

sample-No.	K=50	K=100	K=200	K=300
1	0.8261	**0.8289**	0.8186	0.8122
2	0.8293	**0.8388**	0.8344	0.8301
3	**0.8401**	0.8395	0.8373	0.8275

Table 5.3: The effect of hyperparameters on sample-1, with $K = 100, \phi_0 = .01$, and no hyperparameter optimization.

	$\beta_0 = .1$	$\beta_0 = .01$	$\beta_0 = .001$
$\Omega_0 = .1$	0.7196	0.7140	0.7057
$\Omega_0 = .01$	0.7770	0.7936	0.7845
$\Omega_0 = .001$	0.8178	**0.8209**	**0.8313**

Table 5.4: Component Contribution, sample-1, $K = 100$.

Removed Feature	Time	Names entites	Story words	Topics *(equiv. RCRP)*
Accuracy	0.8225	.6937	0.8114	0.7321

Table 5.5: Number of particles, sample-1, $K = 100$.

#Particles	4	8	16	32	50
Accuracy	0.8101	08289	0.8299	0.8308	0.8358

dependencies of our model require special handling, as well as an adaptation of the sparse sampler of [133].

5.6 DISCUSSION

We present a scalable probabilistic model for extracting storylines in news and blogs. The key aspects of our model are (1) a principled distinction between topics and storylines, (2) a non-parametric model of storyline strength over time, and (3) an online efficient inference algorithm over a non-trivial dynamic non-parametric model. We contribute a very efficient data structure for fast-parallel sampling and demonstrated the efficacy of our approach on hundreds of thousands of articles from a major news portal.

Part III

MODELING MULTI-FACETED CONTENT

6

STRUCTURED CORRESPONDENCE MODELING OF SCIENTIFIC FIGURES

6.1 INTRODUCTION

The rapid accumulation of literatures on a wide array of biological phenomena in diverse model systems and with rich experimental approaches has generated a vast body of online information that must be properly managed, circulated, processed and curated in a systematic and easily browsable and summarizable way. Among such information, of particular interest due to its rich and concentrated information content, but presenting unsolved technical challenges for information processing and retrieval due to its complex structures and heterogeneous semantics, are the diverse types of figures present in almost all scholarly articles. Although there exist a number of successful text-based data mining systems for processing on-line biological literatures, the unavailability of a reliable, scalable, and accurate figure processing systems still prevents information from biological figures, which often comprise the most crucial and informative part of the message conveyed by an scholarly article, from being fully explored in an automatic, systematic, and high-throughout way.

Compared to figures in other scientific disciplines, biological figures are quite a stand-alone source of information that summarizes the findings of the research being reported in the articles. A random sampling of such figures in the publicly available PubMed Central database would reveal that in some, if not most of the cases, a biological figure can provide as much information as a normal abstract. This high-throughput, information-rich, but highly complicated knowledge source calls for automated systems that would help biologists to find their information needs quickly and satisfactorily. These systems should provide biologists with a structured way of browsing the otherwise unstructured knowledge source in a way that would inspire them to ask questions that they never thought of before, or reach a piece of information that they would have never considered pertinent to start with.

The problem of automated knowledge extraction from biological literature figures is reminiscent of the actively studied field of multimedia information management and retrieval. Several approaches have been proposed to model associated text and images for various tasks like annotation [100], retrieval [70, 131] and visualization [22]. However, the *structurally-annotated* biological figures pose a set of new challenges to mainstream multimedia information management systems that can be summarized as follows:

- **Structured Annotation:** as shown in Figure 6.1, biological figures are divided into a set of sub-figures called *panels*. This hierarchical organization results in a local and global annotation scheme in which portions of the caption are associated with a given panel via the panel pointer (like "(a)" in Figure 6.1), while other portions of the caption are shared across all the panels and

provide contextual information. We call the former *scoped* caption, while we call the later *global* caption. How can this *annotation* scheme be modeled effectively?

- **Free-Form Text:** unlike most associated text-image datasets, the text annotation associated with each figure is a free-form text as opposed to high-quality, specific terms that are highly pertinent to the information content of the figure. How can the relevant words in the caption be discovered automatically?
- **Multimodal Annotation:** although text is the main source of modality associated with biological figures, the figure's caption contains other entities like protein names, GO-term locations and other gene products. How can these entities be extracted and modeled effectively?

We address the problem of modeling structurally-annotated biological figures by extending a successful probabilistic graphical model known as the *correspondence latent Dirichlet allocation* [22] (cLDA) model, which was successfully employed for modeling annotated images. We present the struct-cLDA (structured, correspondence LDA) model that addresses the aforementioned challenges in biological literature figures. The rest of this chapter is organized as follows. In Section 6.2, we give an overview of our approach and basic preprocessing of the data. Then in Section 6.3, we detail our model in a series of simples steps. Section 6.4 outlines a collapsed Gibbs sampling algorithm for inference and learning. In Section 6.5 we provide a comprehensive evaluation of our approach using qualitative and quantitative measures. Finally in Section 6, we provide a simple transfer learning mechanism from non-visual data and illustrate its utility. The model presented in this chapter has been integrated into the publicly available *Structured Literature Image Finder* (SLIF) system , first described in [94]. Our system has recently participated in the Elsevier Grand Challenge on Knowledge Enhancement in the Life Science, which is an international contest created to improve the way scientific information is communicated and used, and was selected as one of the 4 finalists among the 70 participating teams [1].

6.2 FIGURE PRE-PROCESSING

In this section we briefly give an overall picture of the SLIF system (Structured Literature Image Finder). SLIF consists of a pipeline for extracting structured information from papers and a web application for accessing that information. The SLIF pipeline is broken into three main sections: caption processing, image processing (which are refereed to as figure preprocessing in Figure 6.1) and topic modeling, as illustrated as Figure 6.1.

The pipeline begins by finding all figure-captions pairs. Each caption is then processed to identify biological entities (e.g., names of proteins)[75]. The second step in caption processing is to identify pointers from the caption that refer to a specific panel in the figure , and the caption is broken into "scopes" so that terms can be linked to specific parts of the figure [39]. The image processing section begins by splitting each figure into its constituent panels, followed by describing each

1 http://www.elseviergrandchallenge.com/finalists.html

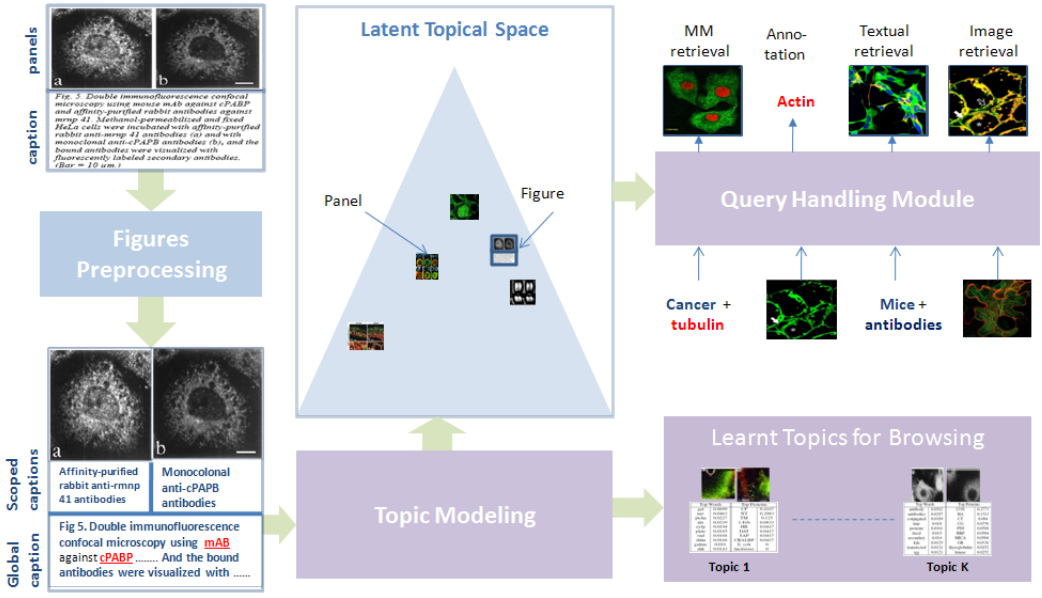


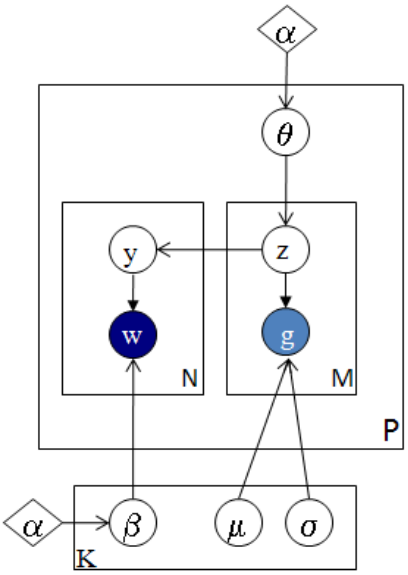
Figure 6.1: Overview of our approach, please refer to Section 2 for more details. (Best viewed in *color*)

panel using a set of biologically relevant image features.In our implementation, we used a set of high-quality 26 image features that span morphological and texture features [93].

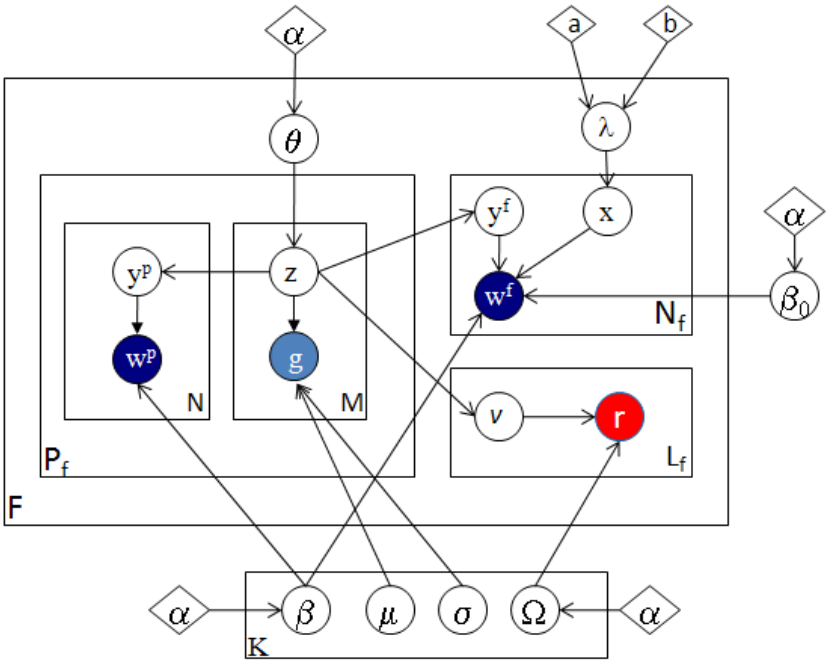
The first two steps result in panel-segmented, structurally and multi-modally annotated figures as shown in the bottom-left of Figure 6.1 (Discovered protein entities are underlined and highlighted in red). The last step in the pipeline, which is the main focus in this chapter, is to discover a set of latent themes that are present in the collection of papers. These themes are called topics and serve as the basis for visualization and semantic representation. Each topic consists of a triplet of distributions over words, image features and proteins. Each figure in turn is represented as a distribution over these topics, and this distribution reflects the themes addressed in the figure. Moreover, each panel in the figure is also represented as a distribution over these topics as shown in Figure 6.1; this feature is useful in capturing the variability of the topics addressed in figures with a wide coverage and allows retrieval at either the panel or figure level. This representation serves as the basis for various tasks like image-based retrieval, text-based retrieval and multimodal-based retrieval. Moreover, these discovered topics provide an overview of the information content of the collection, and structurally guide its exploration.

6.3 STRUCTURED CORRESPONDENCE TOPIC MODELS

In this section we introduce the struct-cLDA model (structured correspondence LDA model) that addresses all the challenges introduced by structurally-annotated biological figures. As the name implies, struct-cLDA builds on top of and extends cLDA which was designed for modeling associated text and image data. We begin



(a) cLDA: Correspondence LDA



(b)struct-cLDA: structured cLDA

Figure 6.2: The cLDA and struct-cLDA Models. Shaded circles represent observed variables and their colors denote modality (blue for words and red for protein entities), unshaded circles represent hidden variables, diamonds represent model parameters, and plates represent replications. Some super/subscripts are removed for clarity — see text for explanation.

by introducing the cLDA; then in a series of steps we show how we extended the cLDA to address the new challenges introduced by the structurally-annotated biological figures. In Figure 6.2, we depict side-by-size the graphical representations of the original cLDA model and our struct-cLDA model, to make explicit the new

innovations. Following conventions in the machine learning literature, we use bold face letters to denote vectors and normal face letters for scalars. For example, $\mathbf{w}^p$ is the vector containing all words that appear in panel p. That is, $\mathbf{w}^p = (w_1^p, \cdots, w_{N_p}^p)$, where N_p is the number of words in panel p.

6.3.1 *The Correspondence LDA*

The cLDA model is a generative model for annotated data – data with multiple types where the instance of one type (such as a caption) serves as a description of the other type (such as an image). cLDA employs a semantic entity known as the *topic* to drive the generation of the annotated data in question. Each topic is represented by two content-specific distributions: a topic-specific word distribution, and a topic-specific distribution over image features. For example, imagine a topic on microarray analysis, the word distribution may have high frequency on words like genes, arrays, normalization, etc., and the image-feature distribution may be biased toward red and green colors. Whereas for a topic on Drosophila development, the word distribution should now have high frequency on *even-skipped*, embryo, wing, etc., and the image-feature distribution would now bias toward certain texture and color features typical in microscopic images of the developing Drosophila embryos. The topic-specific word distribution is modeled as a multinomial distribution over words, denoted by $\text{Multi}(\beta)$; and image features are real-valued and thus follows a Gaussian distribution, denoted by $N(\mu, \sigma)$. As mentioned in Section 2, in our study, each panel is described using $M = 26$ image features, thus each topic has 26 Gaussian distributions: one for each image feature.

The generative process of a figure f under cLDA is given as follows:

1. Draw $\theta_f \sim \text{Dir}(\alpha_1)$
2. For every image feature g_m
   a) Draw topic $z_m \sim \text{Multi}(\theta_f)$
   b) Draw $g_m | z_m = k \sim N(\mu_{k,m}, \sigma^2_{k,m})$
3. For every word w_n in caption
   a) Draw topic $y_n \sim \text{Unif}(z_1, ..., z_m)$
   b) Draw $w_n | y_n = k \sim \text{Multi}(\beta_k)$

In step 1 each figure f samples a topic-mixing vector, θ_f from a Dirichlet prior. The component $\theta_{f,k}$ of this vector defines how likely topic k will appear in figure f. For each image features in the figure, g_m, a topic indicator,z_m , is sampled from θ_f, and then the image feature itself is sampled from a topic-specific image-feature distribution specified by this indicator. The correspondence property of cLDA is manifested in the way the words in the caption of the figure are generated in step 3. Since each word should, in principle, describes a portion of the image, the topic indicator of each word, y, should *correspond* to one of those topic indicators used in generating the image features. Specifically, the topic indicator of the n-th word, y_n, is sampled uniformly from those indicators associated with the image features of the figure as in step 3.(a). Finally, the word, w_n, is sampled from the selected

topic's distribution over words. This generative model explicitly define a likelihood function of the observed data (i.e. annotated figures), thus it offers a principled way of fitting the model parameters such as α_1, μ and **ff**, based on a maximum likelihood principled [22]. Given the model and the data, one can predict the topical content of the figure by computing the posterior estimation of the topic vector θ_f associated with the figure. Since the image feature and word distributions of each topic are highly correlated, the correspondence between image features and words in the caption is enforced.

6.3.2 *Structured Correspondence LDA*

In this section we detail the struct-cLDA model that addressed the new challenges introduced by biological figures. Figure 6.2 depicts a graphical representation of the model. In a struct-cLDA, each topic, k, now has a triplet representation: a multinomial distribution of words β_k, a multinomial distribution over protein entities Ω_k, and a set of M normal distributions, one for each image feature. The full generative scheme of a multi-panel biological figure, f under this model is outlined below:

1. Draw $\theta_f \sim \text{Dir}(\alpha_1)$
2. Draw $\lambda_f \sim \text{Beta}(a, b)$
3. For every panel p in P_f:
   a) For every image feature g_m^p in panel p:
      i. Draw topic $z_m^p \sim \text{Multi}(\theta_f)$
      ii. Draw $g_m^p | z_m^p = k \sim N(\mu_{k,m}, \sigma_{k,m}^2)$
   b) For every word w_n^p in scoped caption of panel p
      i. Draw topic $y_n^p \sim \text{Unif}(z_1^p, ..., z_m^p)$
      ii. Draw $w_n^p | y_n^p = k \sim \text{Multi}(\beta_k)$
4. For every word w_n^f in global caption:
   a) Draw coin $x_n \sim \text{Bernoulli}(\lambda_f)$
   b) If($x_n == 1$)
      i. Draw topic $y_n^f \sim \text{Unif}(\mathbf{z^1}, \cdots, \mathbf{z^{P_f}})$
      ii. Draw $w_n^f | y_n^f = k \sim \text{Multi}(\beta_k)$
   c) If($x_n == 0$)
      i. Draw $w_n^f \sim \text{Multi}(\beta_0)$
5. For every protein entity r_l in global caption:
   a) Draw topic $v_l \sim \text{Unif}(\mathbf{z^1}, \cdots, \mathbf{z^{P_f}})$
   b) Draw $r_l | v_l = k \sim \text{Multi}(\Omega_k)$

In the following subsections we break the above generative steps into parts each of which addresses a specific challenge introduced by biological figures.

6.3.2.1 *Modeling Scoped Caption*

In this subsection, we describe how we approached the problem of modeling scoped and global captions. As shown in Figure 6.1, the input to the topic modeling module is a partially-segmented figure where some of the words in the caption are associated directly with a given panel, say p, and the remaining words serve as a global caption which is shared across all the P_f panels in figure f, and provides contextual information. There are two obvious approaches to deal with this problem that would enable the use of the *flat* cLDA model described in Section 6.3.1:

- *Scoped-only annotation*: in this scheme the input to the cLDA model is the panels with their associated scoped captions. Clearly this results in an *under-representation* problem as contextual information at the figure level is not included.
- *Caption replication*: in this scheme the whole figure caption is replicated with each panel, and this constitutes the input to the cLDA model. Clearly this results in an *over-representation* problem and a bias in the discovered topics towards over-modeling figures with large number of panels due to the replication effect.

In addition to the above problems, resorting to a panel-level abstraction is rather suboptimal because of the lack of modeling the interaction between panels at the figure level which precludes processing retrieval queries at the whole figure level.

We introduce scoping to model this phenomenon. As shown in Figure 6.2.(b), the topic-mixing vector θ_f of the figure is shared across all the panels (step 1). However, each panel's set of words, $\mathbf{w}^p = (w_1^p, w_2^p \cdots, w_{N_p}^p)$, correspond only to this panel's image features. Moreover, words in the figure's global caption, $\mathbf{w}^f$, correspond to all the image features in all the panels of this figure. This suggests a two-layer cLDA generative process:the scoped caption is generated in 3.(b) which with 3.(a) represents exactly the same generative process of cLDA over image features of a panel and words in the scoped caption of this panel. In the next subsection we will detail the generation of words in the global caption.

6.3.2.2 *Modeling Global Caption*

As we noted earlier, the global caption is shared across all panels, and represents contextual information or a description that is shared across all panels. This suggests that words in the global caption of figure f can be generated by corresponding them to the collective set of topic indicators used in generating the image features in all the panels – this is in fact equivalent to a flat cLDA model between words in the global caption and the image features in all the panels. If we took this approach, we found that corpus-level stopwords in the form of non content-bearing words(like: bar, cell, red and green) appear at the top of all the discovered topics due to their high frequencies. This problem is well known in the topic modeling community and is solved by *pre-processing* the collection and removing these *stop-words* using a *fixed* vocabulary list. However, we would like our model to be able to discover these corpus-level stopwords (which we will refer to as *background* words) *automatically* rather than manually *specifying* them for each corpus. In fact, inherent

in the modeling assumption of cLDA is the fact that annotations are specific to the figure and of high quality: that is, every word in the caption describes a part in the image. However, captions in biological figures are free-form text and therefore this assumption is violated. To solve this problem, we use factoring which is similar to background subtraction in [35]. Specifically, we introduce a background topic, β_0, that is used to generate the corpus-level stopwords. This process is equivalent to *factoring* these stopwords from content-bearing topics — we call this process *factoring*.

The generative process for the global caption now proceeds as follows. With each figure, f, we associate a coin whose bias is given by λ_f (step 2). This bias models the ratio of content-bearing to background words in the figure, and is sampled individually for each figure from a *beta* distribution. As shown in step 4, to generate a word in the global caption, w_n^f, we first flip the coin and name the outcome, x_n. If x_n is head, we pick a topic indicator for this word,y_n^f,uniformly from the topic indicators used to generate the image features of the panels in this Figure (step 4.(b)). Then, we generate the word from this topic's distribution over words, $\beta_{y_n^f}$. On the other hand, if x_n is tail, we sample this word from the background topic β_0 (step 4.(c)).[2]

6.3.2.3 *Modeling Multimodal Annotation*

The final step to reach our goal is to model multimodal annotations. For simplicity, we restrict our attention here to protein annotations, although other forms of gene products like GO-terms could be added similarly. For simplicity, we only allow protein annotations to appear in the global caption although it is very straightforward to model scoped multimodal annotation in the same way we modeled scoped word captions. Generating a protein entity is very similar to generating a word in the global caption. To generate a protein entity, r_l, in step 5.(a) we pick a topic indicator for this protein entity,v_l,uniformly from the topic indicators used to generate the image features of the panels in this Figure. Then, we generate the protein entity from this topic's distribution over protein entities Ω_{v_l}.

It should be noted that while protein entities are words, modeling them separately from other caption words has several advantages. First, protein mentions have high variance, *i.e.*, the same protein entity can be referred to in the caption using many textual forms. Mapping these forms, also known as *protein mentions*, to protein entities is a standard process known as *normalization* [78]; our implementation followed our earlier work in [75]. Second, protein entities are rare words and have different frequency spectrums than normal caption words, thus modeling them separately has the advantage of discovering the relationship between words and protein entities despite this frequency mismatch: when protein entities are modeled as normal words, they do not always appear at the top of each topic's distribution over words due to their low frequencies compared to other caption words' frequencies, and thus can be missed. Moreover, endowing each topic with two distributions over words and protein entities results in more interpretable

2 Beta Distribution is the conjugate prior to the Bernoulli distribution which makes posterior inference simpler.

topics (see Section 6.5.1) and enables more advanced query types (see Section 6.5.3 and 6.5.4).

6.3.2.4 *A Note about Hyperparmaters*

All multinomial distributions in the models in Figure 6.2 are endowed with a dirichlet prior to avoid overfitting as discussed in [22]. Perhaps the only part that warrants a description is our choice for the prior over the mean and variance of each image feature's distribution. Each image feature is modeled as a normal distribution whose mean and variance are topic-specific. The standard practice is to embellish the parameters of a normal distribution with an inverse Wishart prior, however, here we took a simpler approach. We place a non-informative prior over the values of the mean parameters of these image features, that is all values are equally likely, that is $\mu_{k,m} \sim$ Unif. Our intuition stems from the fact that different features have different ranges. However, we place an inverse prior over the variance to penalize large variances: $\sigma^2_{k,m} \propto 1/\sigma^2_{k,m}$ (see [53] chapter 3.2). The reason for this choice stems from the noise introduced during calculation of the image features. Without this prior, a maximum likelihood (ML) estimation of σ^2 in a given topic is not robust to outliers. This is because the model is free to increase the variance arbitrarily in order to accommodate the data assigned to this topic. However, with our choice of this form of the prior, it can be shown (see [53] chapter 3.2) that the predictive (posterior) distribution over future image features assigned to this topic follows a Student-t distribution, which is known to be a robust version of the normal distribution that would have resulted if we had used an MLE estimate of the variance parameter instead.

6.4 A COLLAPSED GIBBS SAMPLING ALGORITHM

The main inference tasks can be summarized as follows:

- **Learning**: Given a collection of figures, find a point estimate for the model parameters (i.e. each topic's distribution over words, protein entities and image features).
- **Inference**: Given a new figure, and a point estimate of the model parameters, find the latent representation of this figure over topics (θ_f).

Under a hierarchical Bayesian setting, both of these tasks can be handled via posterior inference. Given the generative process and hyperparmaters choices, outlined in section 6.3.2 we seek to compute:

$$P(\mathbf{f}_{1:F}, \beta_{1:K}, \mu_{1:K}, \sigma^2_{1:K}, \Omega_{1:K}, \beta_0 | \alpha_{1:4}, a, b, \mathbf{w}, \mathbf{g}, \mathbf{r}),$$

where $\mathbf{f}$ is shorthand for the hidden variables $(\theta_f, \lambda_f, \mathbf{y}, \mathbf{z}, \mathbf{x}, \mathbf{v})$ in figure f. The above posterior probability can be easily written down from the generative model in section 6.3.2, however, we omit it for the lack for space. The above posterior is intractable,and we approximate it via a collapsed Gibbs sampling procedure [58] by integrating out, i.e. collapsing, the following hidden variables: the topic-mixing

vectors of each figure, θ_f, the coin bias λ_f for each figure, as well as the topic distributions over all modalities ($\beta_k, \Omega_k, \mu_{k,m}, \sigma^2_{k,m}$, and β_0).

Therefore, the state of the sampler at each iteration contains only the following topic indicators for all figures: topic indicators for words in the global caption ($\mathbf{y}^f$), topic indicators for words in the scoped captions for all panels ($\mathbf{y}^1, \cdots, \mathbf{y}^{P_f}$), topic indicators for all image features ($\mathbf{z}^1, \cdots, \mathbf{z}^{P_f}$), and topic indicators for all protein entities in the global caption ($\mathbf{v}$). We alternate sampling each of these variables conditioned on its Markov blanket until convergence. At convergence, we can calculate expected values for all the parameters that were integrated out, especially for the topic distributions over all modalities, and for each figure's latent representation (mixing-vector). To ease the calculation of the Gibbs sampling update equations we keep a set of sufficient statistics (SS) in the form of co-occurrence counts and sum matrices of the form C^{EQ}_{eq} to denote the number of times instance e appeared with instance q. For example, C^{WK}_{wk} gives the number of times word w was sampled from topic k. Moreover, we follow the standard practice of using the subscript $-i$ to denote the same quantity it is added to without the contribution of item i. For example,$C^{WK}_{wk,-i}$ is the same as C^{WK}_{wk} without the contribution of word w_i. For simplicity, we might drop dependencies on the panel or figure whenever the meaning is implicit form the context.

Sampling a topic (y^p_n) for a given panel word (w^p_n):

$$P(y^p_n = k | w^p_n = w, \mathbf{y}^{\mathbf{p}}_{-\mathbf{n}}, \mathbf{w}^{\mathbf{p}}_{-\mathbf{n}}, \mathbf{z}^{\mathbf{p}}) \propto \frac{C^{KP}_{kp}}{\sum_{k'} C^{KP}_{k'p}} \frac{C^{WK}_{wk,-n} + \alpha_2}{\sum_{w'} C^{WK}_{w'k,-n} + W\alpha_2} \quad (6.1)$$

Sampling a topic (v_l) for a given protein entity (r_l):

$$P(v_l = k | r_l = r, \mathbf{v}_{-l}, \mathbf{r}_{-l}, \mathbf{z}) \propto \frac{C^{KF}_{kf}}{\sum_{k'} C^{KF}_{k'f}} \frac{C^{RK}_{rk,-l} + \alpha_4}{\sum_{r'} C^{RK}_{r'k,-l} + R\alpha_4} \quad (6.2)$$

The above two local distributions have the same form which consists of two terms. The first term measures how likely it is to assign topic k to this word (protein entity) based on the topic indicators of the *corresponding* image features (at the panel level in Equation (6.1), and at the figure level in Equation (6.2) — i.e. the set of all image features' indicators in all panels). The second term measures the probability of generating this word (protein entity) from topic k's distribution over words (protein entities) .

Sampling a coin and a topic (x_n, y^f_n) for a given global caption word (w^f_n):

For a given word in the shared global caption, it is easier to sample (x_n, y^f_n) as a block — a similar approach was used in [35].

$$P(x_n = 0 | x_{-n}, w^f_n = w, \mathbf{w}_{-\mathbf{n}}) \propto \frac{C^{XF}_{0f,-n} + b}{\sum_{x'} C^{XF}_{x'f,-n} + a + b} \frac{C^{W0}_{w,-n} + \alpha_3}{\sum_{w'} C^{W0}_{w',-n} + W\alpha_3} \quad (6.3)$$

Where, C^{W0}_w is the word count matrix for the background topic, and C^{XF}_{xf} counts, in figure f, how many words were assigned to the background topic ($x = 0$) and

how many words were assigned to a panel's image feature and were thus sampled from a latent topic ($x = 1$). Similarly,

$$P(x_n = 1, z^f_n = k | x_{-n}, w^f_n = w, \mathbf{w_{-n}}, \mathbf{z}) \propto \frac{C^{KF}_{kf}}{\sum_{k'} C^{KF}_{k'f}} \frac{C^{XF}_{1f,-n} + a}{\sum_{x'} C^{XF}_{x'f,-n} + a + b} \frac{C^{WK}_{wk,-n} + \alpha_2}{\sum_{w'} C^{WK}_{w'k,-n} + W\alpha_2} \tag{6.4}$$

The above two equations can be normalized to form a $K+1$ multinomial distribution — K information-bearing topics when $x_n = 1$, in addition to the background topic when $x = 0$.

Sampling a topic (z^p_m) for the m^{th} image feature (g^p_m) in panel p:

Perhaps this is the most involved equation as all other topic indicators in the figure/panels are influenced by the topic indicators of the image features. For simplicity, the $(|\cdots)$ in the equation below is a shorthand for all these topic indicators which are: topic indicators of words in the global caption ($\mathbf{y}^f$), topic indicators of words in the scoped caption of panel p ($\mathbf{y}^p$), topic indicators of all other image features in all panels ($\mathbf{z}^1, \cdots, \mathbf{z}^{P_f}$), and topic indicators for protein entities in the global caption ($\mathbf{v}$).

$$\begin{aligned} P(z^p_m = k | g^p_m = g, \cdots) \quad \propto & \\ & \frac{C^{KF}_{kf,-m} + \alpha_1}{\sum_{k'} C^{KF}_{k'f,-m} + K\alpha_1} \; t(g; \hat{\mu}_{k,m}, \hat{\sigma}^2_{k,m}, C^{MK}_{mk,-m} - 1) \times \\ & \text{Unif}(\mathbf{y}^f | z^p_m = k) \times \text{Unif}(\mathbf{v} | z^p_m = k) \times \text{Unif}(\mathbf{y}^p | z^p_m = k) \end{aligned} \tag{6.5}$$

where $t(g; \mu, \sigma^2, n)$ is a student t-distribution with mean μ, variance σ^2, and n degree of freedom (see [53] chapter 3.2). $\hat{\mu}_{k,m}$ is the sample mean of the values of image feature m that are assigned to topic k, and $\hat{\sigma}^2_{k,m}$ is defined similarly. C^{MK}_{mk} is the number of times image feature m was sampled from topic k. The first two parts in Equation (6.5) are similar to the previous sampling equations: they measure the comparability of joining a topic given the observed feature and the topics assigned to neighboring image features. However, since every other annotation in the figure is generated based on the topic indicator of the image feature, three extra terms are needed. These terms measure how likely is the current assignment of the topic indicators of other annotations — panel words, figure words, and protein entities — given the new assignment to this image feature's topic indicator. Notice that this **uniform** probabilities are exactly the same probabilities that appeared in the generative process, and also the same factors that appeared as the first fraction in Eqs. (6.4,6.2,6.1) respectively — however after updating the corresponding C matrix with the new value of z^p_m under consideration. For example, $\text{Unif}(\mathbf{y}^p)$ is computed as follows — please recall that $\mathbf{y}^p$ is a vector of topic indicators for words in panel p, that is $\mathbf{y}^p = (y^p_1, y^p_2, \cdots, y^p_{N_p})$:

$$\text{Unif}(\mathbf{y}^p | z^p_m = k) = \prod_{n=1}^{N_p} \frac{C^{KP}_{y^p_n p,-m} + \mathbf{1}(y^p_n = k)}{\sum_{k'} C^{KP}_{k'p,-m} + \mathbf{1}(y^p_n = k)}$$

Intuitively, for each word n in panel p, the above fraction measures how likely is its topic y^p_n, if we sample the image feature under consideration from topic k. The $\mathbf{1}$ is the indicator function which is 1 if and only if the expression inside it is evaluated to be true. Therefore, adding this function to the sufficient statistics C^{KP} has the effect of updating C^{KP} with the current topic assignment under consideration (k) for image feature m. This fraction is exactly the same as the first fraction in Equation (6.1). $\text{Unif}(\mathbf{y^f}|z^p_m = k)$ and $\text{Unif}(\mathbf{v}|z^p_m = k)$ are defined similarly with relation to the first fraction in Equation (6.4) and Equation (6.2) respectively.

Eqs. (6.1-6.5) are iterated until convergence, then expected values for the collapsed variables can be obtained via simple normalization operations over their respective count matrices using posterior samples. Moreover, the expected topics' distributions over each modality can be calculated similarly.

For instance, the figure's latent representation (θ_f) and the figure's foreground/background bias (λ_f) can be calculated as follows[3]:

$$\begin{aligned} E[\theta_{f,k}] &= \frac{C^{KF}_{kf} + \alpha_1}{\sum_{k'} C^{KF}_{k'f} + K\alpha_1} \\ E[\lambda_f] &= \frac{C^{XF}_{1f} + a}{C^{XF}_{1f} + C^{XF}_{0f} + a + b} \end{aligned} \tag{6.6}$$

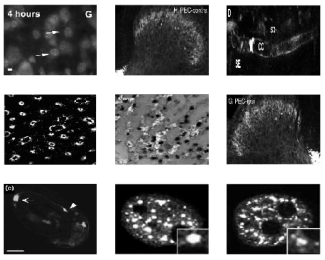
Moreover, the topics' distributions over each modality can be calculated as follows[4]:

$$\begin{aligned} E[\beta_{w,k}] &= \frac{C^{WK}_{wk} + \alpha_2}{\sum_{w'} C^{WK}_{w'k} + W\alpha_2} \\ E[\Omega_{r,k}] &= \frac{C^{RK}_{rk} + \alpha_4}{\sum_{r'} C^{RK}_{r'k} + R\alpha_4} \end{aligned} \tag{6.7}$$

The mean ($\hat{\mu}_{k,m}$) and variance ($\hat{\sigma}^2_{k,m}$) for topic k's distribution over image feature m are calculated as the empirical mean and variance of the values of image features of type m that are assigned to this topic, i.e., the image features whose topic indicators = k.

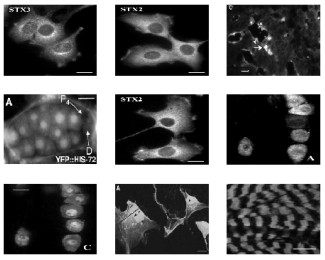
At *test time*, to obtain the latent representation of a *test figure*, we hold the topic count matrices *fixed*, iterate Eqs. (6.1-6.5) until convergence, and then calculate the expected latent representation of each figure from posterior samples after convergence.

EBA Method



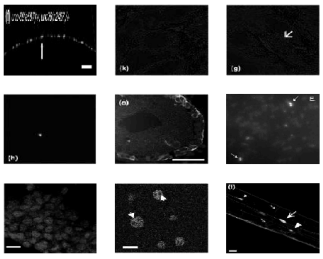
Top Words		Top Proteins	
mei	0.0231	Actin	0.1875
sex	0.0191	VP26	0.1562
filaments	0.0181	Some	0.0938
vessels	0.0171	RT	0.0625
interphase	0.0171	SCP3	0.0625
eba	0.0131	E-H	0.0312
oocytes	0.0121	2a	0.0312
injury	0.0121	PA	0.0312
focal	0.0121	L4	0.0312
representation	0.0111	L1	0.0312

Cell structure



Top Words		Top Proteins	
embryo	0.0265	HRP	0.3571
muscle	0.0251	ZO-1	0.1429
mitochondrial	0.0223	Map	0.0714
structure	0.0181	PC	0.0714
filaments	0.0181	SSW	0.0714
yfp	0.0153	P4	0.0714
epithelial	0.0153	Df	0.0714
stromule	0.0153	GFP-MAP4	0.0714
nestin	0.0153	Septin2	0.0714
plastid	0.0139	SE	0.001

Tumors



Top Words		Top Proteins	
unc	0.0304	Cx43	0.15
dcx	0.0291	DNA	0.1
hbl	0.0291	L1	0.1
actin	0.0198	RNA	0.05
cbp	0.0198	caspase-3	0.05
optical	0.0146	E15	0.05
tumor	0.0146	AM	0.05
murine	0.0146	Ki-67	0.05
signals	0.0146	DAPI	0.05
positive	0.0132	1 protein	0.05

Figure 6.3: Illustrative three topics from a 20-topics run of the struct-cLDA model. For each topic we show: the top words with their probabilities, the top protein entities with their probabilities, and the top ranked panels under this topic. The topics were labeled manually.

6.5 EXPERIMENTAL RESULTS

We evaluated our models on a set of articles that were downloaded from the publicly available PubMed[5] database. We applied the preprocessing steps described on section 6.2 to extract the figures, segment the captions and extract protein entities. The resulting dataset that we used for the experiments in this chapter consists of 5556 panels divided into 750 figures. Each figure has on average 8 panels, however some figures have up to 25 panels. The number of word types is 2698, and the number of protein entity types discovered is 227. The average number of words per caption is 150 words. We divided this dataset into 85% for training and 15% for testing. For all the experiments reported below, we set all the α hyperparmaters to .01 (except $\alpha_1 = .1$), and (a, b) to 3 and 1 respectively. We found that the final results are not largely sensitive to these assignments. We ran Gibbs sampling to learn the topics until the in-sample likelihood converges which took a few hundred iterations for all models.

For comparison, we used cLDA and LSI as baselines. To apply cLDA to this dataset, we *duplicated* the whole figure caption and its associated protein entities with each panel to obtain a *flat* representation of the *structured* figures. Therefore, we will refer to this model as cLDA-d. For LSI, we followed the same strategy and then concatenated the word vector, image features and protein entities to form

3 Technically, the expressions in Equation (6.6) are due to only one sample from the posterior after convergence. The standard practice is to collect multiple samples from the posterior and average the result of Equation (6.6) over these samples. We omitted this technicality for simplicity.

4 As we noted above, the expressions in Equation (6.7) are due to only one sample from the posterior after convergence. The standard practice is to collect multiple samples from the posterior and choose the topics' distributions associated with the sample with the highest marginal data loglikelihood (as averaging the topic distributions across samples is not well-defined). We also omitted this technicality for simplicity.

5 http://www.pubmedcentral.nih.gov

a single vector. Moreover, to understand the contribution of each feature in our model (scoping vs factoring), we removed factoring from the struct-cLDA model to obtain a model that only employs scoping, and we call the resulting model struct-cLDA^{-f}.

In the following subsections we provide a quantitative as well as a qualitative evaluation of our model and compare it against the LSI and cLDA baselines over various visualization and retrieval tasks. Clearly, our goal from these comparisons is just to show that a straightforward application of simpler flat models can not address our problem adequately. In this chapter, we extended cLDA to cope with the structure of the figures under consideration, however, adapting LSI, and other related techniques, to cope with this structure is left as an open problem. Moreover, our choice of comparing against LSI for annotation and retrievals tasks stems from the fact that in these tasks our own proposed model serves merely as a dimensionality reduction technique.

6.5.1 *Visualization and Structured Browsing*

In this section, we examine a few topics discovered by the struct-cLDA model when applied to the aforementioned dataset. In Figure 6.3, we depict three topics from a run of the struct-cLDA with K=20. For each topic we display its top words, protein entities, and the top 9 panels under this topic (i.e. the panels with the highest component for this topic in their latent representation θ_f). It is interesting to note that all these topics are biologically meaningful. For instance, the first topic represents the theme related to the EBA (The enucleation before activation) method which is a conventional method of producing an embryo and comprises enucleating oocytes, transferring donor cells into oocytes, fusing the oocytes and the donor cells, and activating the fused reconstruction cells. Clearly the occurrence of the word "oocytes" in this topic is relevant. Moreover, protein "VP26" has been shown in various studies to interact with protein "actin" during these procedures. The second topics is about various cell structure types. For example,"stromule" is a microscopic structure found in plant cells and extends from the surface of all "plastid" types which are major organelles found in plants and algae. The third topic is about tumor-related experiments. Examining its top words and proteins we found "cbp" and "hbl" which are known tumor suppressors. Moreover, "actin" has been shown to be an important protein for tumor developments due to its rule in cell division. Also, "Cx43" is a genetic sequence that codes for a protein that has tumor suppressing effect, moreover, protein "Caspases-3" is a member of the Caspases family which plays essential roles in apoptosis (programmed cell death). Interestingly, "UNC" appears in this topic due to the wide usage of the University of North Carolina tumor dataset.

These topics enable biologists to have an overview of the themes that are available in the collection, and provide them with a structured way of browsing the otherwise unstructured collection. For instance, the user might choose to expand the tumor topic and retrieve figures in which this topic is highly represented.Moreover, given a figure f, as shown in Figure 6.4, the system can visualize its topic decomposition, i.e. what are the topics represented in this figure along with its weights, either at the whole figure level, θ_f, or at the panel level.

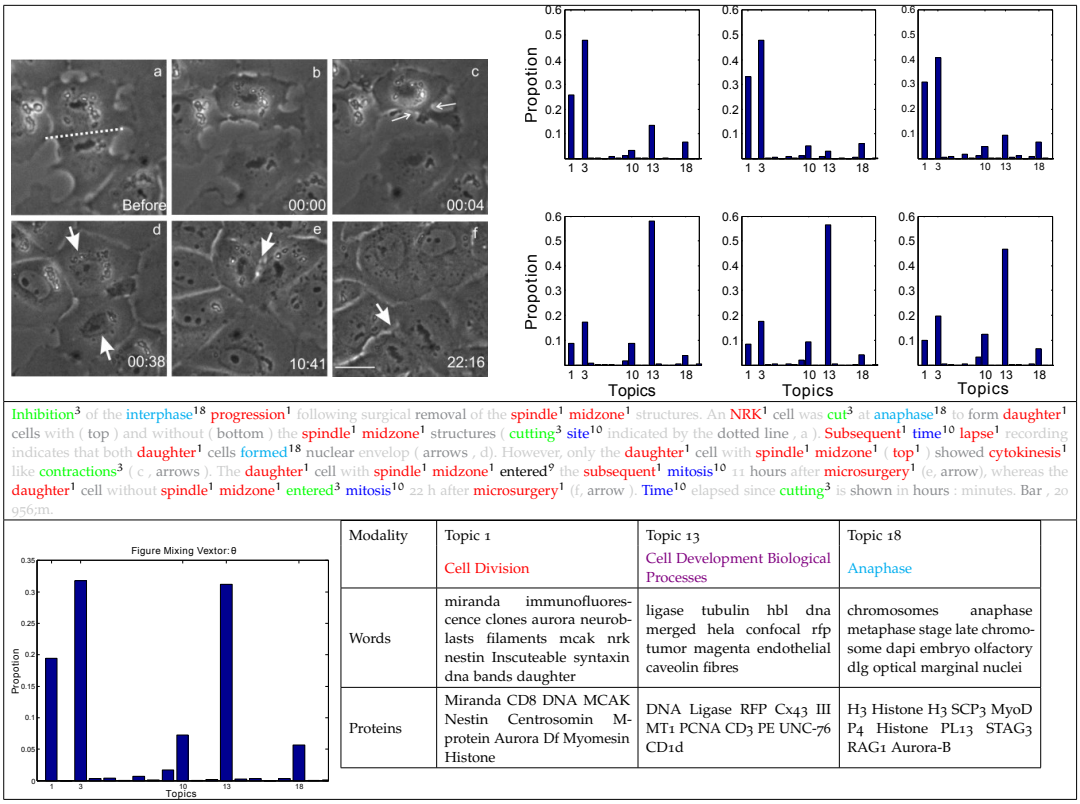


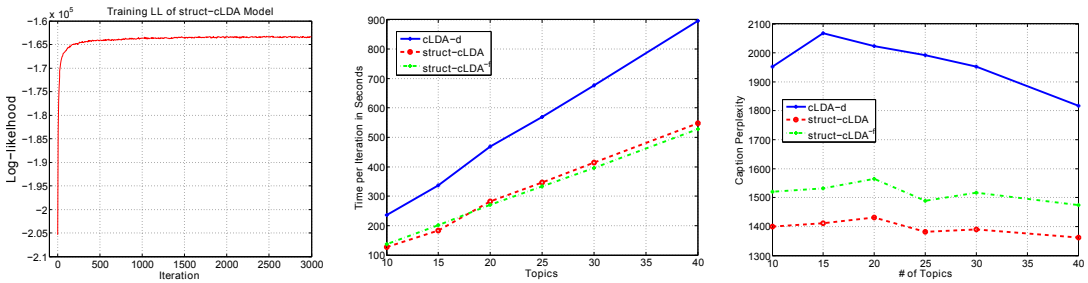
Figure 6.4: Illustrating topic decomposition and structured browsing. A biological figure tagged with its topic decomposition at different granularities: each panel (top-right), caption words (second row), and the whole figure (bottom-left). In tagging the caption, light grey colors are used for words that were removed during pre-processing stages, and dark grey colors are used for background words. Some topics are illustrated at the bottom row. (best viewed in color)

The figure in Figure 6.4 is composed of 6 panels, and thus our model gives a topic decomposition of each panel, a topic decomposition of the whole figure (bottom-left), and a topic assignment for each word in the caption. This, in fact, is a key feature of our model as it breaks the figure into its themes and allows biologist to explore each of these themes separately. For instance, this figure addresses several phases of cell division, under a controlled condition, that starts from the *Anaphase* stage (panels (a-c)) and progresses towards post *mitosis* stages (panels (d-f)). Indeed, our model was able to discern these stages correctly via the latent representation it assigns to each panel. Please note that this figure represents a case in which the scoped-caption module was not able to segment the caption due to the unorthodox referencing style used in this figure, however, the model was able to produce a reasonable latent representation. In the bottom-right of Figure 6.4, we show three important topics addressed in this figures. It is quite interesting that Topic 13, which corresponds to various biological processes important to cell division, was associated with this figure mainly due to its image content, not its word content. Moreover, while the figure does not mention any protein entities, the associated protein entities with each topic play key roles during all stages of cell division addressed in the figure: for instance, *dna ligase* is an important protein for DNA replication. Therefore, the biologist might decide to retrieve similar figures

(based on the latent representation of the whole figure) that address cell division under the same conditions, retrieve figures that address a given stage per see (based on the latent representation of some panels), or further explore a given topic by retrieving its representing figures as we discussed earlier. These features glue the figures in the whole collection via a web of interactions enabled by the similarity between the latent representation of each figure at multiple granularities. Moreover, this unified latent representation enables comparing figures with largely different number of panels.

Table 6.1: The effect of the background topic

Factored Model		Non-factored Model: struct-cLDA^{-f}			
Background Topic		Normal Topic 1		Normal Topic 2	
cells	0.0559	**red**	0.0458	**cells**	0.09
cell	0.0289	**green**	0.0385	**bar**	0.0435
bar	0.0265	**cells**	0.0351	**cell**	0.0386
gfp	0.0243	infected	0.0346	antibody	0.0318
scale	0.024	actin	0.0244	protein	0.0282
red	0.0197	transfected	0.0222	staining	0.0202
green	0.0188	**images**	0.0218	visualized	0.0171
images	0.0188	membrane	0.0167	expressed	0.0141
arrows	0.0157	fluorescent	0.0167	section	0.0129
shown	0.0151	fixed	0.0163	tissue	0.0129



(a) (b) (c)

Figure 6.5: Understating model's features contributions: (a) Convergence (b) Time per iteration and (c) Perplexity

.

Finally, in Table 6.1 we examine the effect of introducing the factored background topic on the quality of the discovered topics. Table 6.1 shows the background topic from the struct-cLDA model which clearly consists of corpus-level stopwords that carry no information. Examining a few topics discovered using a non-factored model (i.e. by removing the factoring component from struct-cLDA), it is clear that many of these stopwords (underlined in Table 6.1) found its way to the top list in seemingly information-bearing topics, and thus obscure their clarity and clutter the representation.

6.5.2 *Timing Analysis and Convergence*

Figure 6.5.b compares the time, in seconds, consumed by each model in performing a full Gibbs iteration over the training set. All the models were coded in Matlab. It is clear from this figure that replicating the caption with each panel in order to be able to use a standard cLDA model increases the running time considerably. In addition both of the two models converges after roughly the same number of iteration (a few hundred for this dataset as depicted for the struct-cLDA model in Figure 6.5.a). This result shows that while struct-LDA is seemingly more *sophisticated* than its ancestor cLDA, this sophistication does not incur a large added penalty on the running time, on the contrary it runs even faster, and also enhances the performance (as shown qualitatively in Table 6.1 and quantitatively using perplexity analysis in Figure 6.5.c to be detailed in Section 6.5.3.1) and enables sophisticated retrieval tasks (as will be shown using the struct-cLDA model in Sections 6.5.3 and 6.5.4).

6.5.3 *Annotation Task*

Since the main goal of the models presented in this chapter is discovering the correspondence between mixed modalities, therefore, a good model once observed parts of the figure, should be able to annotate the figure with the missing modalities. In this section, we examine the ability of the struct-cLDA model to predict the textual caption of the figure based on observing the image features, and predict protein entity annotations of a given figure based on observing its image features and textual caption.

6.5.3.1 *Caption Perplexity*

For the first task, we computed the perplexity of the figures's caption based on observing its image features. Perplexity, which is used in the language modeling community, is equivalent algebraically to the inverse of the geometric mean per-word likelihood, that is:

$$\texttt{Perplexity} \;=\; \exp\left[\frac{-\sum_f \log p(\mathbf{w}^f, \{\mathbf{w}^p\} | \{\mathbf{g}^p\})}{\sum_f \left(N_f + \sum_{p=1}^{P_f} N_p\right)}\right]$$

The above conditional probability can be computed by running the Gibbs sampling algorithm of Section 4 by iterating Equation (6.5) only until convergence (with no words or protein entities used). A number of posterior samples can then be generated from this posterior distribution by resuming the Gibbs Sampling on Eqs. (6.1,6.3 and 6.4) while holding the image features topic indicators fixed. These samples are then used to compute the average likelihood of the caption conditioned on the image features. Figure 6.5.(c) compares caption perplexity using cLDA-d, struct-cLDA, and struct-cLDA^{-f}. This experiment shows that modeling the figure as a whole via the struct-cLDA^{-f} model is better than *duplicating* the caption across panels, as this duplicating results in over representation and less accurate predictions. Moreover, factoring out background words, as in the struct-cLDA model, further improves the performance because, as was shown in Table 6.1, it

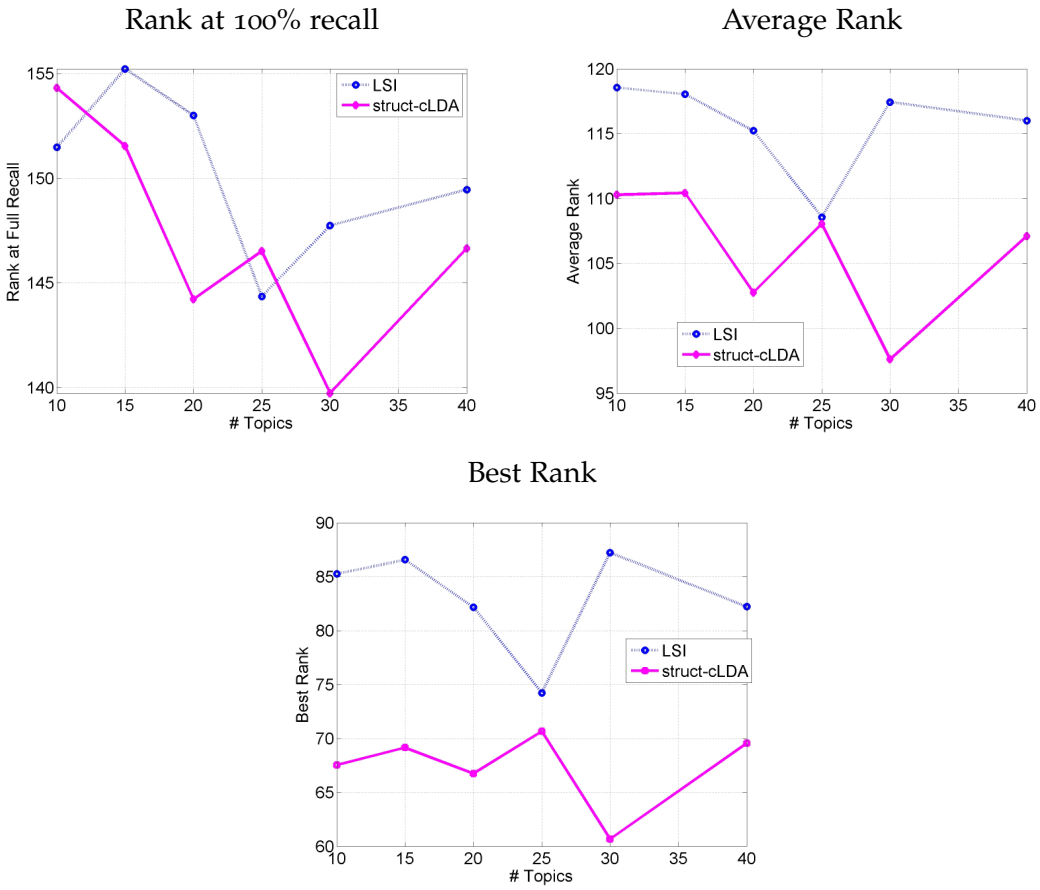


Figure 6.6: Evaluating protein annotation quality based on observing text and image features using various measures. Figures show comparison between the struct-cLDA model and LSI.(*Lower better*)

.

excludes non content-bearing words from being associated with image features and thus misleading the predictions.

6.5.3.2 *Protein Annotation*

To annotate a test figure based on observing its image features and caption words, we first project the figure to the latent topic space using the observed parts of the figure by first iterating Eqs. (6.1,6.3,6.4,6.5 until convergence, and then collecting posterior samples for θ_f. Moreover, from the training phase, we can compute each topic's distribution over the protein vocabulary (Ω_k). Finally, the probability that figure f is annotated with protein r, can be computed as follows:

$$P(r|f) = \sum_k P(k|f)P(r|k) = \sum_k \theta_{kf}\Omega_{rk} \tag{6.8}$$

It is interesting to note that the above measure is equivalent to a dot product in the latent topic space between the figure representation θ_f and the latent representation of the protein entity r — as we can consider Ω_{rk} as the projection of the protein

entity over the k^{th} topical dimension. Protein entities can then be ranked based on this measure. We compare the ranking produced by the struct-cLDA with that produced by LSI. Applying LSI to the training dataset results in a representation of each term (image feature, protein entity, and text word) over the LSI semantic space. These terms are then used to project a new figure in the testset onto this space using "folding" as discussed in [43]. Afterwards cosine similarity is used as the distance measure for ranking. We evaluated each ranking using three measures: the highest (low in value) rank, average rank and lowest rank (Rank at 100% recall) of the actual annotations as it appear in the recovered rankings. Figure 6.6 shows the result across various number of topics (factors for LSI).

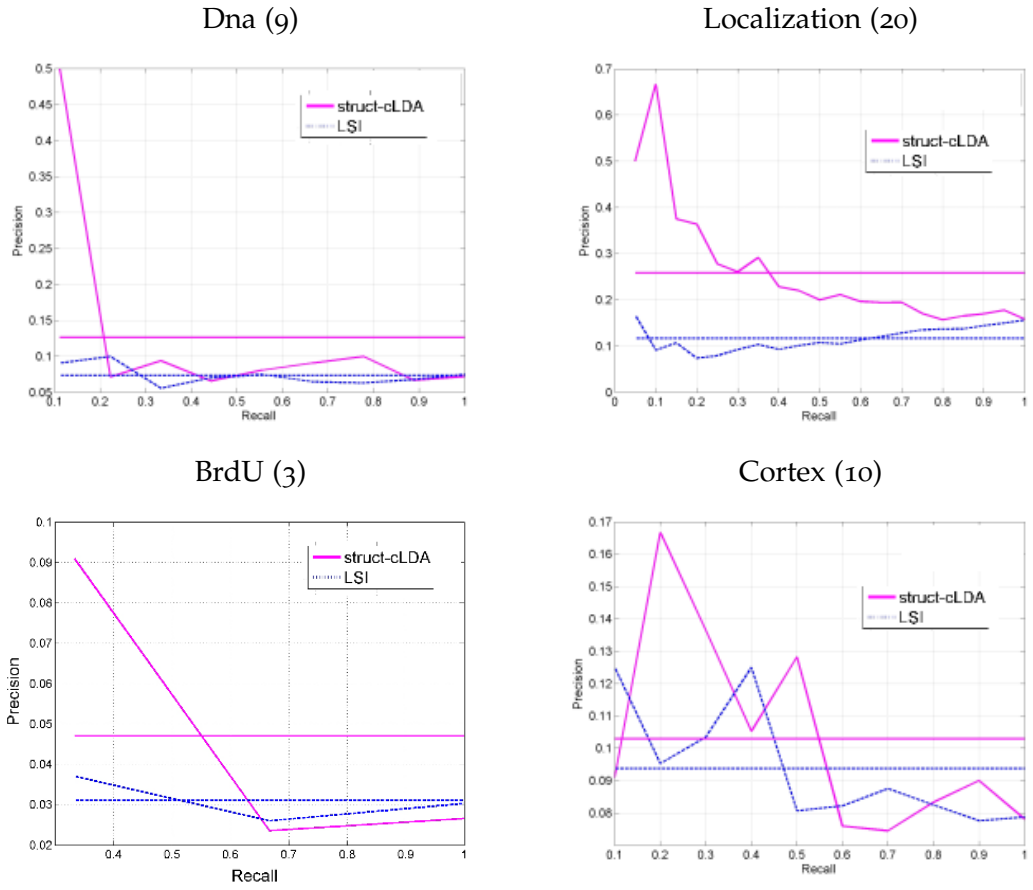


Figure 6.7: Illustrating figure retrieval performance. Each column depicts the result for a give query written on its top with the number of true positives written in parenthesis (the size of the test set is 131 figures). The figure shows comparisons between struct-cLDA and LSI. The *horizontal lines* are the average precision for each model. (Better viewed in *color*)

.

6.5.4 *Multi-Modal Figure Retrieval*

Perhaps the most challenging task in multimedia retrieval is to retrieve a set of images based on a multimodal query. Given a query composed of a set of text words and protein entities, $q = (w_1, \cdots, w_n, r_1, \cdots, r_m)$, we can use the query

langauge model [105] as a measure to evaluate the likelihood of the query give a test document as follows:

$$\begin{aligned} P(q|f) &= \prod_{w \in q} P(w|f) \prod_{r \in q} P(r|f) \\ &= \prod_{w \in q} \left[\sum_k \theta_{kf} \beta_{wk} \right] \prod_{r \in q} \left[\sum_k \theta_{kf} \Omega_{rk} \right] \end{aligned} \tag{6.9}$$

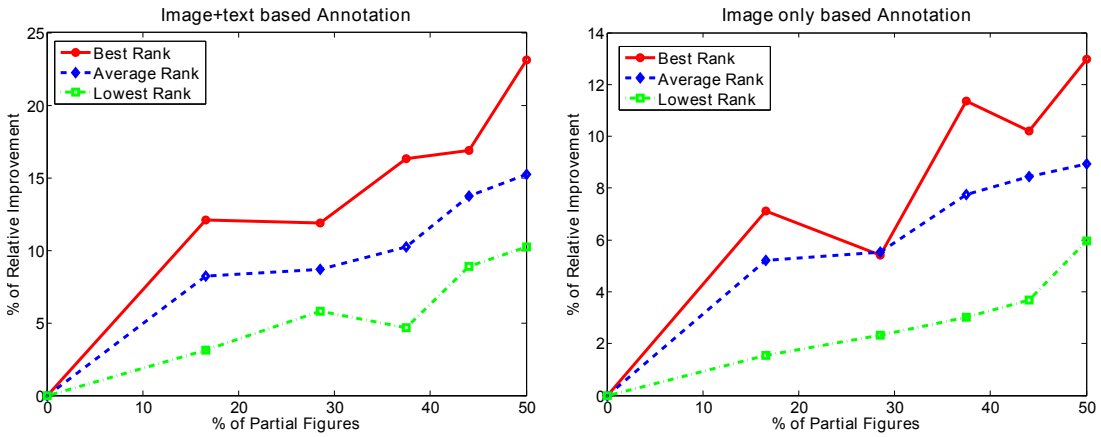
As we noted in Equation (6.8), $p(w|f)$ is a simple dot product operation between the latent representations of the word w and the latent representation of figure f in the induced topical space. The above measure can then be used to rank figures in the testset for evaluation. We compared the performance of struct-cLDA to LSI. Each of the two models has access to *only* the image features of figures in the testset. Query computations in LSI are handled using cosine similarity after folding both the test figures and the query onto the LSI space [43]. Figure 6.7 shows the precision-recall curves over 4 queries. For a given query, an image is considered relevant if the query words appear in its caption (which is hidden from both LSI and struct-cLDA, and is only used for evaluation). As shown in Figure 6.7, struct-cLDA compares favorably to LSI across a range of factors (we only show the result for $K = 15$ for space limitations but the same behavior was observed as we vary the number of factors).

6.6 TRANSFER LEARNING FROM PARTIAL FIGURES

In this section, we explore the utility of using non-visual data to enhance the performance of our model. We restrict our attention on textual date accompanied with protein entities. This data can be in the form of biological abstracts tagged with protein entities, or other biological figures that lack visual data (which we refer to as partial figures). Partial figures occur frequently in our pipeline due to the absence of the resolution of the figure which is necessary for normalization of the image features. We focus here on this case, although the former case can be handled accordingly. A partial figure f comprises a set of global words and protein entities can be generated as follows:

1. Draw $\theta_f \sim \text{Dir}(\alpha_1)$
2. Draw $\lambda_f \sim \text{Beta}(a, b)$
3. For every word w_n^f in global caption:
   a) Draw coin $x_n \sim \text{Bernoulli}(\lambda_f)$
   b) If($x_n == 1$)
      i. Draw topic $y_n^f \sim \text{Mult}(\theta_f)$
      ii. Draw $w_n^f | y_n^f = k \sim \text{Multi}(\beta_k)$
   c) If($x_n == 0$)
      i. Draw $w_n^f \sim \text{Multi}(\beta_0)$

4. For every protein entity r_l in global caption:
   a) Draw topic $v_l \sim \text{Unif}(y^f_1, \cdots, y^f_{N_f})$
   b) Draw $r_l | v_l = k \sim \text{Multi}(\Omega_k)$



(a) (b)

Figure 6.8: Illustrating the utility of using partial figures as a function of its ratio in the training set. The task is protein annotation based on (a) Figure's image and text and (b) Image content of the figure only

.

In essence, the captions words are moved to the highest level in the correspondence hierarchy, and a factored, flat cLDA model is used that generates protein entities from the topic indicators used in generating the figure's words. As we made explicit in the above generative process, the partial figures share the same set of topic parameters over words and proteins ($\beta_{1:K}$, $\Omega_{1:K}$). Extending the collapsed Gibbs sampling algorithm from Section 6.4 for this case is quite straightforward and omitted.

To balance the partial and full figures, we first extract the word and protein vocabulary using only the full figures, then project the partial figures over this vocabulary, and finally keep partial figures that retain at least one protein annotation and train the model over this larger set. To evaluate the utility of partial figures, we used the protein annotation task of Section 6.5.3. In this task, a test figure is annotated based on its text and image features. As shown in Figure 6.8.(a), the performance increases as the ratio of partial figures in the training set increases . This behavior should be expected because the annotation is based on both the text and image features of the test figure. However, interestingly, we found that the annotation quality also increases if we annotate the test figures after observing only its image features as shown in Figure 6.8.(b). This shows that, during training, the model was able to *transfer* text-protein correlations form the partial figures to image-protein correlations via the triplet topic representations (a mechanism which is illustrated in Figure .

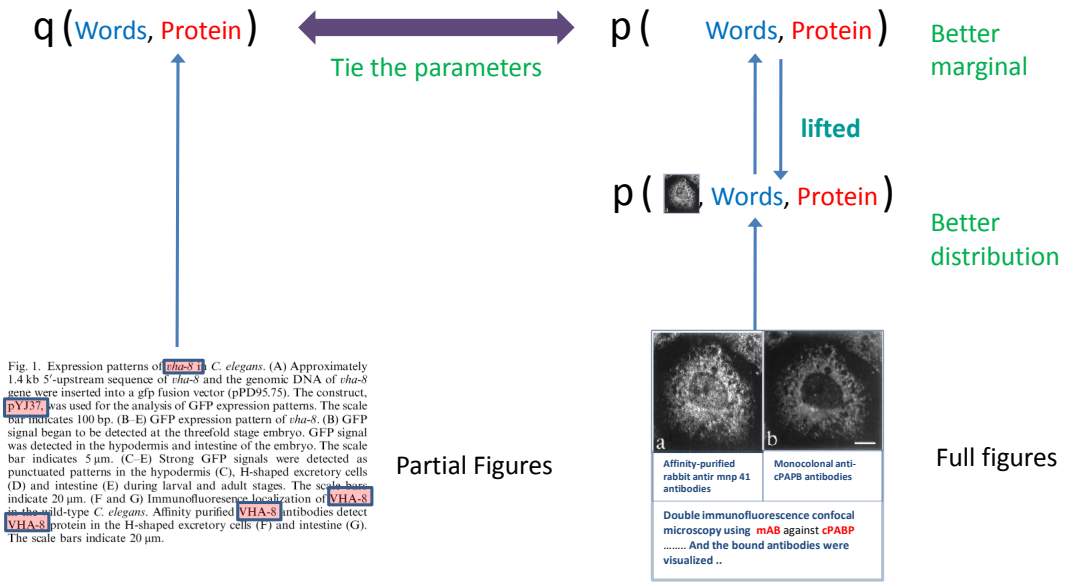


Figure 6.9: Illustration the transfer learning mechanism from partial figures.

.

6.7 DISCUSSION

In this chapter we addressed the problem of modeling structurally and multi-modally annotated biological figures for visualization and retrieval tasks. We presented the structured correspondence LDA model that addresses all the challenges posed by these figures. We illustrated the usefulness of our models using various visualization and retrieval tasks. Recent extensions to LDA and cLDA bear resemblances to some features in the models presented in this paper, such as [99] in its ability to model entities, and [18, 69] in their abilities to model many-many annotations. However, our goal in this paper was mainly focused on modeling biological figures with an eye towards building a model that can be useful in various domains where modeling uncertain *hierarchical*, *scoped* associations is required. In the future, we plan to extend our model to incorporate other sources of hierarchal correspondences like modeling the association between figures and the text of their containing papers.

7

SUPERVISED AND SEMI-SUPERVISED MULTI-VIEW ANALYSIS OF IDEOLOGICAL PERSPECTIVE

7.1 INTRODUCTION

With the avalanche of user-generated articles over the web, it is quite important to develop models that can recognize the ideological bias behind a given document, summarize where this bias is manifested on a topical level, and provide the user with alternate views that would help him/her staying informed about different perspectives. In this chapter, we follow the notion of ideology as defines by Van Dijk in [45] as "a set of general abstract beliefs commonly shared by a group of people." In other words, an ideology is a set of ideas that directs one's goals, expectations, and actions. For instance, *freedom of choice* is a general aim that directs the actions of "liberals", whereas *conservation of values* is the parallel for "conservatives".

We can attribute the lexical variations of the word content of a document to three factors:

- **Writer Ideological Belief.** A liberal writer might use words like freedom and choice regardless of the topical content of the document. These words define the abstract notion of belief held by the writer and its frequency in the document largely depends on the writer's style.
- **Topical Content.** This constitutes the main source of the lexical variations in a given document. For instance, a document about abortion is more likely to have facts related to abortion, health, marriage and relationships.
- **Topic-Ideology Interaction.** When a liberal thinker writes about abortion, his/her abstract beliefs are materialized into a set of concrete opinions and stances, therefore, we might find words like: pro-choice and feminism. On the contrary, a conservative writer might stress issues like pro-life, God and faith.

Given a collection of ideologically-labeled documents, our goal is to develop a computer model that *factors* the document collection into a representation that reflects the aforementioned three sources of lexical variations. This representation can then be used for:

- **Visualization.** By visualizing the abstract notion of belief in each ideology, and the way each ideology approaches and views mainstream topics, the user can view and contrast each ideology side-by-side and build the right mental landscape that acts as the basis for his/her future decision making.
- **Classification or Ideology Identification.** Given a document, we would like to tell the user from which side it was written, and explain the ideological bias in the document at a topical level.

- **Staying Informed: Getting alternative views[1].** Given a document written from perspective A, we would like the model to provide the user with other documents that represent alternative views about the same topic addressed in the original document.

In this chapter, we approach this problem using Topic Models [29]. We introduce a factored topic model that we call multi-view Latent Dirichlet Allocation or mview-LDA for short. Our model views the word content of each document as the result of the interaction between the document's idealogical and topical dimensions. The rest of this chapter is organized as follows. First, in Section 7.2, we review related work, and then present our model in Section 7.3. Then in Section 7.4, we detail a collapsed Gibbs sampling algorithm for posterior inference. Sections 7.5 and 7.6 give details about the dataset used in the evaluation and illustrate the capabilities of our model using both qualitative and quantitative measures. Section 7.7 describes and evaluates the efficacy of a semi-supervised extension, and finally in Section 7.8 we conclude and list several directions for future research.

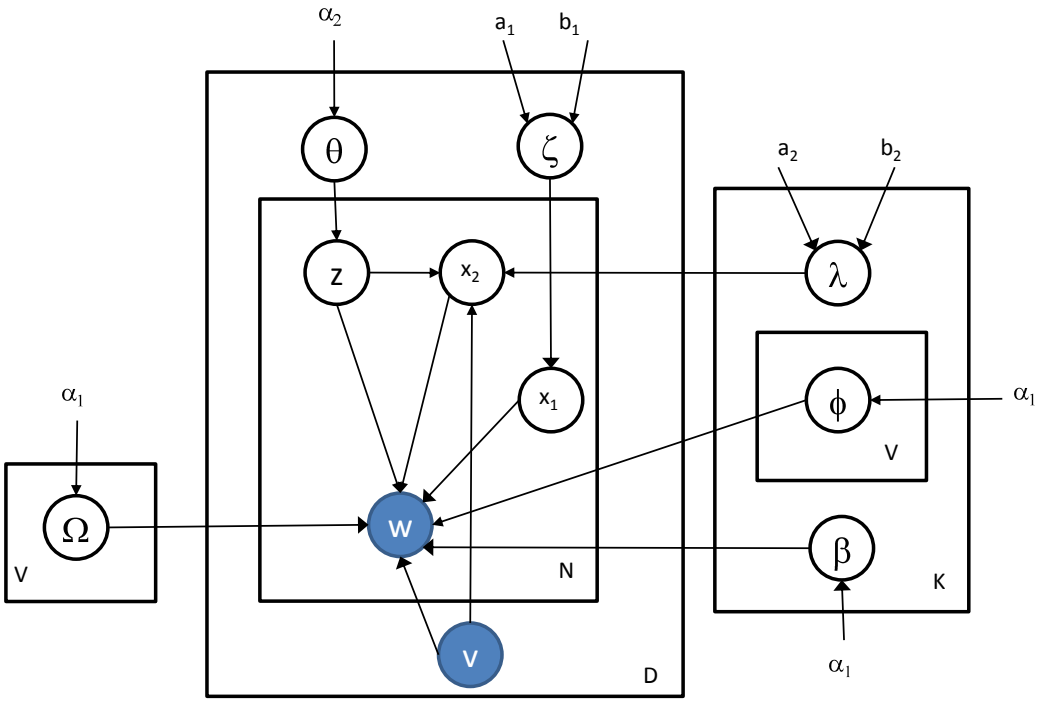
7.2 RELATED WORK

Ideological text is inherently subjective, thus our work is related to the growing area of subjectivity analysis [130, 110]. The goal of this area of research is to learn to discriminate between subjective and objective text. In contrast,in modeling ideology, we aim toward contrasting two or more ideological perspectives each of which is subjective in nature. Further more, subjective text can be classified into sentiments which gave rise to a surge of work in automatic opinion mining [130, 135, 134, 120, 106] as well as sentiment analysis and product review mining [96, 66, 101, 31, 119, 118, 87, 85]. The research goal of sentiment analysis and classification is to identify language used to convey positive and negative opinions, which differs from contrasting two ideological perspectives. While ideology can be expressed in the form of a sentiment toward a given topic,like abortion, ideological perspectives are reflected in many ways other than sentiments as we will illustrate later in the chapter. Perhaps more related to this chapter is the work of [51, 84] whose goal is to detect bias in news articles via discriminative and generative approaches, respectively. However, this work still addresses ideology at an abstract level as opposed to our approach of modeling ideology at a topical level. Finally, independently, [102] gives a construction similar to ours however for a different task.

7.3 MULTI-VIEW TOPIC MODELS

In this section we introduce multi-view topic models, or mview-LDA for short. Our model, mview-LDA, views each document as the result of the interaction between its topical and idealogical dimensions. The model seeks to explain lexical variabilities in the document by attributing this variabilities to one of those dimensions or to their interactions. Topic models, like LDA, define a generative process

1 In this chapter, we use the words ideology, view, perspective interchangeably to denote the same concept



Variable	Meaning
w	word
v	document's ideology
z	topic
x_1, x_2	word switches, one per word (see text)
θ	document-specific distribution over topics
ξ	document's expected usage of the ideology's background topic
Ω	ideology's background-topic
β	ideology-independent topic distribution
ϕ	ideology-specific topic distribution
λ	topic bias across ideology

Figure 7.1: A plate diagram of the graphical model.

for a document collection based on a set of parameters. LDA employs a semantic entity known as *topic* to drive the generation of the document in question. Each topic is represented by a topic-specific word distribution which is modeled as a multinomial distribution over words, denoted by Multi(β). The generative process of LDA proceeds as follows:

1. Draw topic proportions $\theta_d|\alpha \sim \text{Dir}(\alpha)$.
2. For each word

(a) Draw a topic $z_n|\theta_d \sim \text{Mult}(\theta_d)$.

(b) Draw a word $w_n|z_n, \beta \sim \text{Multi}(\beta_{z_n})$.

In step 1 each document d samples a topic-mixing vector θ_d from a Dirichlet prior. The component $\theta_{d,k}$ of this vector defines how likely topic k will appear in document d. For each word in the document w_n, a topic indicator z_n is sampled from θ_d, and then the word itself is sampled from a topic-specific word distribution specified by this indicator. Thus LDA can capture and represent lexical variabilities via the components of θ_d which represents the topical content of the document. In the next section we will explain how our new model mview-LDA can capture other sources of lexical variabilities beyond topical content.

7.3.1 *Multi-View LDA*

As we noted earlier, LDA captures lexical variabilities due to topical content via θ_d and the set of topics $\beta_{1:K}$. In mview-LDA each document d is tagged with the ideological view it represents via the observed variable v_d which takes values in the discrete range: $\{1, 2, \cdots, V\}$ as shown in Fig. 7.1. For simplicity, lets first assume that $V = 2$. The topics $\beta_{1:K}$ retain the same meaning: a set of K multinomial distributions each of which represents a given theme or factual topic. In addition, we utilize an ideology-specific topic Ω_v which is again a multinomial distribution over the same vocabulary. Ω_v models the abstract belief shared by all the documents written from view v. In other words, if v denotes the liberal perspective, then Ω_v gives high probability to words like progressive, choice, etc. Moreover, we defined a set of $K \times V$ topics that we refer to as ideology-specific topics. For example, topic $\phi_{v,k}$ represents how ideology v addresses topic k. The generative process of a document d with ideological view v_d proceeds as follows:

1. Draw $\xi_d \sim \text{Beta}(a_1, b_1)$
2. Draw topic proportions $\theta_d|\alpha \sim \text{Dir}(\alpha_2)$.
3. For each word w_n
   a) Draw $x_{n,1} \sim \text{Bernoulli}(\xi_d)$
   b) If$(x_{n,1} = 1)$
      i. Draw $w_n|x_{n,1} = 1 \sim \text{Multi}(\Omega_{v_d})$
   c) If$(x_{n,1} = 0)$
      i. Draw $z_n|\theta_d \sim \text{Mult}(\theta_d)$.
      ii. Draw $x_{n,2}|v_d, z_n \sim \text{Bernoulli}(\lambda_{z_n})$
      iii. If$(x_{n,2} = 1)$
         A. Draw $w_n|z_n, \beta \sim \text{Multi}(\beta_{z_n})$.
      iv. If$(x_{n,2} = 0)$
         A. Draw $w_n|v_d, z_n \sim \text{Multi}(\phi_{v_d,z_n})$.

In step 1, we draw a document-specific biased coin,ξ_d. The bias of this coin determines the proportions of words in the document that are generated from its ideology background topic Ω_{v_d}. As in LDA, we draw the document-specific topic

proportion θ_d from a Dirichlet prior. θ_d thus controls the lexical variabilities due to topical content inside the document.

To generate a word w_n, we first generate a coin flip $x_{n,1}$ from the coin ξ_d. If it turns head, then we proceed to generate this word from the ideology-specific topic associated with the document's ideological view v_d. In this case, the word is drawn independently of the topical content of the document, and thus accounts for the lexical variation due to the ideology associated with the document. The proportion of such words is document-specific by design and depends on the writer's style to a large degree. If $x_{n,1}$ turns to be tail,we proceed to the next step and draw a topic-indicator z_n. Now, we have two choices: either to generate this word directly from the ideology-independent portion of the topic β_{z_n}, or to draw the word from the ideology-specific portion ϕ_{v_d,z_n}. The choice here is **not** document specific, but rather depends on the interaction between the ideology and the specific topic in question. If the ideology associated with the document holds a strong opinion or view with regard to this topic, then we expect that most of the time we will take the second choice, and generate w_n from ϕ_{v_d,z_n}; and vice versa. This decision is controlled by the Bernoulli variable λ_{z_n}. Therefore, in step c.ii, we first generate a coin flip $x_{n,2}$ from λ_{z_n}. Based on $x_{n,2}$ we either generate the word from the ideology-independent portion of the topic β_{z_n}, and this constitutes how the model accounts for lexical variation due to the topical content of the document, or generate the word from the ideology-specific portion of the topic ϕ_{v_d,z_n}, and this specifies how the model accounts for lexical variation due to the interaction between the topical and ideological dimensions of the document.

Finally, it is worth mentioning that the decision to model λ_{z_n}[2] at the topic-ideology level rather than at the document level, as we have done with ξ_d, stems from our goal to capture ideology-specific behavior on a corpus level rather than capturing document-specific writing style. However, it is worth mentioning that if one truly seeks to measure the degree of bias associated with a given document,then one can compute the frequency of the event $x_{n,2} = 0$ from posterior samples. In this case, λ_{z_n} acts as the prior bias only. Moreover, computing the frequency of the event $x_{n,2} = 0$ and $z_n = k$ gives the document's bias toward topic k per se.

Finally, it is worth mentioning that all multinomial topics in the model: β, Ω, ϕ are generated once for the whole collection from a symmetric Dirichlet prior, similarly, all bias variables, $\lambda_{1:K}$ are sampled from a Beta distribution also once at the beginning of the generative process.

7.4 POSTERIOR INFERENCE VIA COLLAPSED GIBBS SAMPLING

The main tasks can be summarized as follows:

- **Learning**: Given a collection of documents, find a point estimate of the model parameters (i.e. $\beta, \Omega, \phi, \lambda$,etc.).
- **Inference**: Given a new document, and a point estimate of the model parameters, find the posterior distribution of the latent variables associated with

2 In an earlier version of the work we modeled λ on a per-ideology basis, however, we found that using a single shared λ results in more robust results

the document at hand:
$(\theta_d, \{x_{n,1}\}, \{z_n\}, \{x_{n,2}\})$.

Under a hierarchical Bayesian setting, both of these tasks can be handled via posterior inference. Under the generative process, and hyperparmaters choices, outlined in section 7.3, we seek to compute:

$$P(\mathbf{d}_{1:D}, \beta_{1:K}, \mathbf{\Omega}_{1:V}, \boldsymbol{\phi}_{\mathbf{1:V,1:K}}, \boldsymbol{\lambda}_{1:K} | \alpha, a, b, \mathbf{w}, \mathbf{v}),$$

where $\mathbf{d}$ is a shorthand for the hidden variables $(\theta_d, \xi_d, \mathbf{z}, \mathbf{x_1}, \mathbf{x_2})$ in document d. The above posterior probability is unfortunately intractable,and we approximate it via a collapsed Gibbs sampling procedure [58, 53] by integrating out, i.e. collapsing, the following hidden variables: the topic-mixing vectors θ_d and the ideology bias ξ_d for each document, as well as all the multinomial topic distributions: (β, Ω and ϕ) in addition to the ideology-topic biases given by the set of – random variables.

Therefore, the state of the sampler at each iteration contains only the following topic indicators and coin flips for each document:$(\mathbf{z}, \mathbf{x_1}, \mathbf{x_2})$. We alternate sampling each of these variables conditioned on its Markov blanket until convergence. At convergence, we can calculate expected values for all the parameters that were integrated out, especially for the topic distributions, for each document's latent representation (mixing-vector) and for all coin biases. To ease the calculation of the Gibbs sampling update equations we keep a set of sufficient statistics (SS) in the form of co-occurrence counts and sum matrices of the form C^{EQ}_{eq} to denote the number of times instance e appeared with instance q. For example, C^{WK}_{wk} gives the number of times word w was sampled from the ideology-independent portion of topic k. Moreover, we follow the standard practice of using the subscript $-i$ to denote the same quantity it is added to without the contribution of item i. For example,$C^{WK}_{wk,-i}$ is the same as C^{WK}_{wk} without the contribution of word w_i. For simplicity, we might drop dependencies on the document whenever the meaning is implicit form the context.

For word w_n in document d, instead of sampling $z_n, x_{n,1}, x_{n,2}$ independently, we sample them as a block as follows:

$$P(x_{n,1} = 1 | w_n = w, v_d = v) \propto (C^{DX_1}_{d1,-n} + a_1) \times \frac{C^{VW}_{vw,-n} + \alpha_1}{\sum_{w'}(C^{VW}_{vw',-n} + \alpha_1)}$$

$$\begin{aligned} P(x_{n,1} = 0, x_{2,n} = 1, z_n = k | w_n = w, v_d = v) \\ \propto (C^{DX_1}_{d0,-n} + b_1) \times \frac{C^{KX_2}_{k1,-n} + a_2}{C^{KX_2}_{k1,-n} + C^{KX_2}_{k0,-n} + a_2 + b_2} \\ \times \frac{C^{KW}_{kw,-n} + \alpha_1}{\sum_{w'}(C^{KW}_{kw',-n} + \alpha_1)} \times \frac{C^{DK}_{dk,-n} + \alpha_2}{\sum_{k'}(C^{DK}_{dk',-n} + \alpha_2)} \end{aligned}$$

$$
\begin{aligned}
&P(x_{n,1} = 0, x_{2,n} = 0, z_n = k | w_n = w, v_d = v) \\
&\quad \propto (C^{DX_1}_{d0,-n} + b_1) \times \frac{C^{KX_2}_{k0,-n} + b_2}{C^{KX_2}_{k1,-n} + C^{KX_2}_{k0,-n} + a_2 + b_2} \\
&\quad \times \frac{C^{VKW}_{vkw,-n} + \alpha_1}{\sum_{w'}(C^{VKW}_{vkw',-n} + \alpha_1)} \times \frac{C^{DK}_{dk,-n} + \alpha_2}{\sum_{k'}(C^{DK}_{dk',-n} + \alpha_2)}
\end{aligned}
$$

The above three equations can be normalized to form a $2 * K + 1$ multinomial distribution: one component for generating a word from the ideology topic, K components for generating the word from the ideology-independent portion of topic $k = 1, \cdots, K$, and finally K components for generating the word from the ideology-specific portion of topic $k = 1, \cdots, K$. Each of these $2 * K + 1$ cases corresponds to a unique assignment of the variables $z_n, x_{n,1}, x_{n,2}$. Therefore, our Gibbs sampler just repeatedly draws sample from this $2 * K + 1$-components multinomial distribution until convergence. Upon convergence, we compute point estimates for all the collapsed variables by a simple marginalization of the appropriate count matrices. During **inference**, we hold the corpus-level count matrices fixed, and keep sampling from the above $2 * K + 1$-component multinomial while only changing the document-level count matrices: C^{DK}, C^{DX_1} until convergence. Upon convergence, we compute estimates for ξ_d and θ_d by normalizing C^{DK} and C^{DX_1} (or possibly averaging this quantity across posterior samples). As we mentioned in Section 7.3, to compute the ideology-bias in addressing a given topic say k in a given document, say d, we can simply compute the expected value of the event $x_{n,2} = 0$ and $z_n = k$ across posterior samples.

7.5 DATA SETS

We evaluated our model over three datasets: the bitterlemons croups and a two political blog-data set. Below we give details of each dataset.

7.5.1 *The Bitterlemons dataset*

The bitterlemons corpus consists of the articles published on the website `http://bitterlemons.org/`. The website is set up to contribute to mutual understanding between Palestinians and Israelis through the open exchange of ideas. Every week, an issue about the Israeli-Palestinian conflict is selected for discussion, and a Palestinian editor and an Israeli editor contribute one article each addressing the issue. In addition, the Israeli and Palestinian editors invite one Israeli and one Palestinian to express their views on the issue. The data was collected and pre-proceed as describes in [84]. Overall, the dataset contains 297 documents written from the Israeli's point of view, and 297 documents written from the Palestinian's point of view. On average each document contains around 740 words. After trimming words appearing less than 5 times, we ended up with a vocabulary size of 4100 words. We split the dataset randomly and used 80% of the documents for training and the rest for testing.

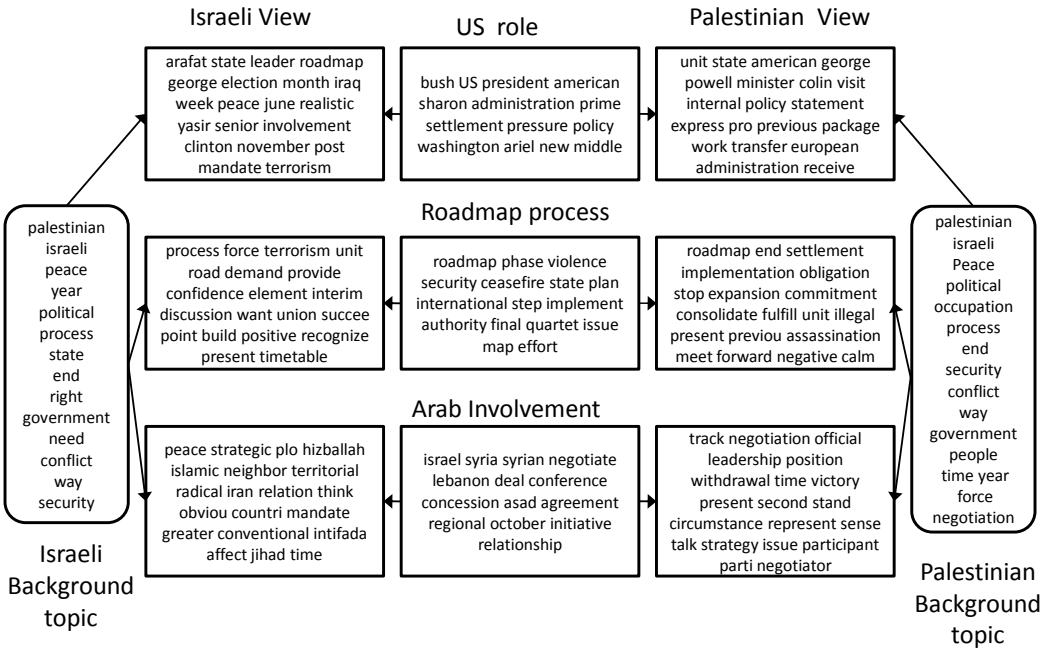


Figure 7.2: Illustrating the big picture overview over the bitterlemons dataset using few topics. Each box lists the top words in the corresponding multinomial topic distribution. See text for more details

7.5.2 *The Political Blog Datasets*

The first dataset refereed to as *Blog-1* is a subset of the data collected and processed in [132]. The authors in [132] collected blog posts from blog sites focusing on American politics during the period November 2007 to October 2008. We selected three blog sites from this dataset: the Right Wing News (right-ideology) ; the Carpetbagger, and Daily Kos as representatives of the liberal view (left-ideology). After trimming short posts of less than 20 words, we ended up with 2040 posts distributed as 1400 from the left-wing and the rest from the right-wing. On average, each post contains around 100 words and the total size of the vocabulary is 14276 words. For this dataset, we followed the train-test split in [132]. In this split each blog is represented in both training and test sets. Thus this dataset does not measure the model's ability to generalize to a totally different writing style.

The second dataset refereed to as *Blog-2* is similar to Blog-1 in its topical content and time frame but larger in its blog coverage [47]. Blog-2 spans 6 blogs: three from the left-wing and three from the right-wing. The dataset contains 13246 posts. After removing words that appear less then 20 times, the total vocabulary becomes 13236 with an average of 200 words per post. We used 4 blogs (2 from each view) for training and held two blogs (one from each view) for testing. Thus this dataset measures the model's ability to generalize to a totally new blog.

7.6 EXPERIMENTAL RESULTS

In this section we gave various qualitative and quantitative evaluations of our model over the datasets listed in Section 7.5. For all experiments, we set $\alpha_1 =$

$.01, \alpha_2 = .1, a = 1$ and $b = 1$. We run Gibbs sampling during training for 1000 iterations. During inference, we ran Gibbs sampling for 300 iterations, and took 10 samples, with 50-iterations lag, for evaluations.

7.6.1 *Visualization and Browsing*

One advantage of our approach is its ability to create a "big-picture" overview of the interaction between ideology and topics. In figure 7.2 we show a portion of that diagram over the bitterlemons dataset. First note how the ideology-specific topics in both ideology share the top-three words, which highlights that the two ideologies seek peace even though they still both disagree on other issues. The figure gives example of three topics: the US role, the Roadmap peace process, and the Arab involvement in the conflict (the name of these topics were hand-crafted). For each topic, we display the top words in the ideology-independent part of the topic (β), along with top words in each ideology's view of the topic (ϕ).

For example, when discussing the roadmap process, the Palestinian view brings the following issues: [the Israeli side should] implement the obligatory points in this agreement, stop expansion of settlements, and move forward to the commitments brought by this process. On the other hand, the Israeli side brings the following points: [Israelis] need to build confidence [with Palestinian], address the role of terrorism on the implementation of the process, and ask for a positive recognition of Israel from the different Palestinian political parties. As we can see, the ideology-specific portion of the topic **needn't** always represent a **sentiment** shared by its members toward a given topic, but it might rather includes extra important dimensions that need to be taken into consideration when addressing this topic.

Another interesting topic addresses the involvement of the neighboring Arab countries in the conflict. From the Israeli point of view, Israel is worried about the existence of hizballah [in lebanon] and its relationship with radical Iran, and how this might affect the Palestinian-uprising (Intifada) and Jihad. From the other side, the Palestinians think that the Arab neighbors need to be involved in the peace process and negotiations as some of these countries like Syria and Lebanon are involved in the conflict.

The user can use the above chart as an entry point to retrieve various documents pertinent to a given topic or to a given view over a specific topic. For instance, if the user asks for a representative sample of the Israeli(Palestinian) view with regard to the roadmap process, the system can first retrieve documents tagged with the Israeli(Palestinian) view and having a high topical value in their latent representation θ over this topic. Finally, the system then sorts these documents by how much bias they show over this topic. As we discussed in Section 7.4, this can be done by computing the expected value of the event $x_{n,2} = 0$ and $z_n = k$ where k is the topic under consideration.

7.6.2 *Classification*

We have also performed a classification task over all the datasets. The Scenario proceeded as follows. We train a model over the training data with various number

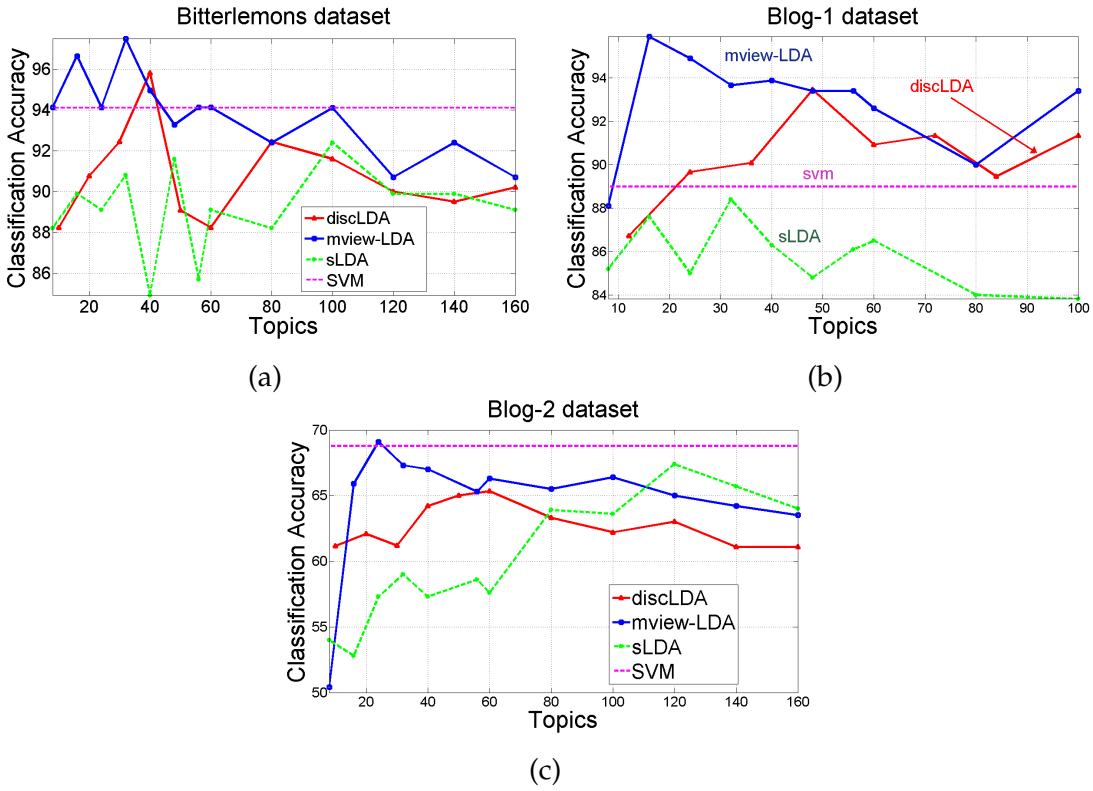


Figure 7.3: Classification accuracy over the Bitterlemons dataset in (a) and over the two blog datasets in (b) and (c). For SVM we give the best result obtained across a wide range of the SVM's regularization parameter(not the cross-validation result).

of topics. Then given a test document, we predict its ideology using the following equation:

$$v_d = \text{argmax}_{v \in V} P(\mathbf{w_d}|v); \tag{7.1}$$

We use three baselines. The first baseline is an SVM classifier trained on the normalized word frequency of each document. We trained SVM using a regularization parameter in the range $\{1, 10, 20, \cdots, 100\}$ and report the best result (i.e. no cross-validation was performed). The other two are supervised LDA models: supervised LDA (sLDA) [37, 28] and discLDA [77]. discLDA is a conditional model that divides the available number of topics into class-specific topics and shared-topics. Since the code is not publicly available, we followed the same strategy in the original paper and share 0.1K topics across ideologies and then divide the rest of the topics between ideologies[3]. However, unlike our model, there are no internal relationships between these two sets of topics. The decision rule employed by discLDA is very similar to the one we used for mview-LDA in Eq (7.1). For sLDA, we used the publicly available code by the authors.

3 [77] gave an optimization algorithm for learning the topic structure (transformation matrix), however since the code is not available, we resorted to one of the fixed splitting strategies mentioned in the paper. We tried other splits but this one gives the best results

As shown in Figure 7.3, our model performs better than the baselines over the three datasets. We should note from this figure that mview-LDA peaks at a small number of topics, however, each topic is represented by three multinomials. Moreover, it is evident from the figure that the experiment over the blog-2 dataset which measures each model's ability to generalizes to a totally unseen new blog is a harder task than generalizing to unseen posts form the same blog. However, our model still performs competitively with the SVM baseline. We believe that separating each topic into an ideology-independent part and ideology-specific part is the key behind this performance, as it is expected that the new blogs would still share much of the ideology-independent parts of the topics and hopefully would use similar (but no necessarily all) words from the ideology-specific parts of each topic when addressing this topic.

Finally, it should be noted that the bitterlemons dataset is a multi-author dataset and thus the models were tested on some authors that were not seen during training, however, two factors contributed to the good performance by all models over this dataset. The first being the larger size of each document (740 words per document as compared to 200 words per post in blog-2) and the second being the more formal writing style in the bitterlemons dataset.

7.6.3 *An Ablation Study*

To understand the contribution of each component of our model, we conducted an ablation study over the bitterlemons dataset. In this experiment we turned-off one feature of our model at a time and measured the classification performance. The results are shown in Figure 7.4. Full, refers to the full model; No-Ω refers to a model in which the ideology-specific background topic Ω is turned-off; and No-ϕ refers to a model in which the ideology-specific portions of the topics are turned-off. As evident from the figure, ϕ is more important to the model than Ω and the difference in performance between the full model and the No-ϕ model is rather significant. In fact without ϕ the model has little power to discriminate between ideologies beyond the ideology-specific background topic Ω.

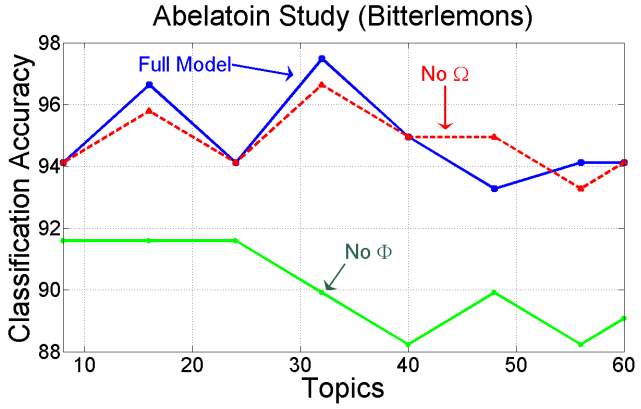


Figure 7.4: An Ablation study over the bitterlemons dataset.

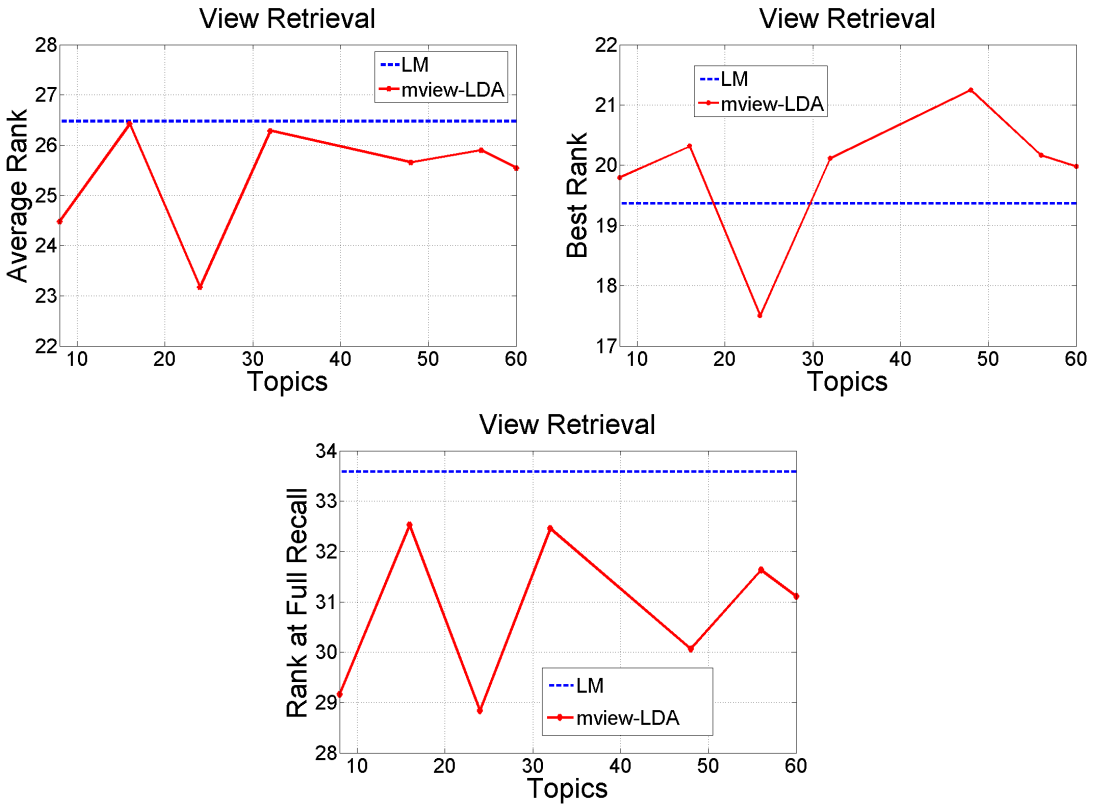


Figure 7.5: Evaluating the performance of the view-Retrieval task. Figure compare performance between mview-LD vs. an SVM+a smoothed language model approach using three measures: average rank, best rank and rank at full recall. (*Lower better*)

.

7.6.4 *Retrieval: Getting the Other Views*

To evaluate the ability of our model in finding alternative views toward a given topic, we conducted the following experiment over the Bitterlemons corpus. In this corpus each document is associated with a meta-topic that highlights the issues addressed in this document like: “A possible Jordanian role", “Demography and the conflict",etc. There are a total of 148 meta-topics. These topics were not used in fitting our model but we use them in the evaluation as follows. We divided the dataset into 60% for training and 40% for testing. We trained mview-LDA over the training set, and then used the learned model to infer the latent representation of the test documents as well as their ideologies. We then used each document in the training set as a query to retrieve documents from the test set that address the same meta-topic in the query document but from the other-side’s perspective. Note that we have access to the view of the query document but not the view of the test document. Moreover, the value of the meta-topic is only used to construct the ground-truth result of each query over the test set. In addition to mview-LDA, we also implemented a strong baseline using SVM+Dirichlet smoothing that we will refer to as LM. In this baseline, we build an SVM classifier over the training set, and use Dirichlet-smoothing to represent each document (in test and training

set) as a multinomial-distribution over the vocabulary. Given a query document d, we rank documents in the test set by each model as follows:

- mview-LDA: we computed the cosine-distance between $\theta_d^{mv-LDA-shared}$ and $\theta_{d'}^{mv-LDA-shared}$ weighted by the probability that d' is written from a different view than v_d. The latter quantity can be computed by normalizing $P(v|d')$. Moreover, $\theta_{d,k}^{mv-LDA-shared} \propto \sum_n I\big[(x_{n,1} = 0) \text{ and } (x_{n,2} = 1) \text{ and } (z_n = k)\big]$, and n ranges over words in document d. Intuitively, we would like $\theta_d^{mv-LDA-shared}$ to reflect variation due to the topical content, but not ideological view of the document.
- LM: For a document d', we apply the SVM classifier to get $P(v|d')$, then we measure similarity by computing the cosine-distance between the smoothed multinomial-distribution of d and d'. We combine these two components as in mview-LDA.

Finally we rank documents in the test set in a descending-order and evaluate the resulting ranking using three measures: the rank at full recall (lowest rank), average rank, and best rank of the ground-truth documents as they appear in the predicted ranking. Figure 7.5 shows the results across a number of topics. From this figure, it is clear that our model outperforms this baseline over all measures. It should be noted that this is a very hard task since the meta-topics are very fine-grained like: Settlements revisited, The status of the settlements, Is the roadmap still relevant?,The ceasefire and the roadmap: a progress report,etc. We did not attempt to cluster these meta-topics since our goal is just to compare our model against the baseline.

7.7 A MH-BASED SEMI-SUPERVISED ALGORITHM

In this section we present and assess the efficacy of a semi-supervised extension of mview-LDA. In this setting, the model is given a set of ideologically-labeled documents and a set of unlabeled documents. One of the key advantages of using a probabilistic graphical model is the ability to deal with hidden variables in a principled way. Thus the only change needed in this case is adding a single step in the sampling algorithm to sample the ideology v of an unlabeled document as follows:

$$P(v_d = v|rest) \quad \propto \quad P(\mathbf{w_d}|v_d = v, \mathbf{z_d}, \mathbf{x_{1,d}}, \mathbf{x_{2,d}})$$

Note that the probability of the indicators $(\mathbf{x_{1,d}}, \mathbf{x_{2,d}}, \mathbf{z_d})$ do not depend on the view of the document and thus got absorbed in the normalization constant, and thus one only needs to measures the likelihood of generating the words in the document under the view v. We divide the words into three groups: $A_d = \{w_n|x_{n,1} = 1\}$ is the set of words generated from the view-background topic, $B_{d,k} = \{w_n|z_n = k, x_{n,1} = 0, x_{n,2} = 1\}$ is the set of words generated from β_k, and $C_{d,k} = \{w_n|z_n = k, x_{n,1} = 0, x_{n,2} = 0\}$ is the set of words generated from $\phi_{k,v}$. The probability of $B_{d,k}$ does not depend on the value of v and thus can be absorbed into the normalization factor. Therefore, we only need to compute the following probability:

R	mview-LDA	ss-mview-LDA
80	65.60	66.41
60	62.31	65.43
20	60.87	63.25

Table 7.1: Classification performance of the semi-supervised model. R is the ratio of labeled documents.

$$\begin{aligned} P(A_d, C_{d,1:K}|v_d = v, \text{rest}) &= \prod_k \int_{\phi_{k,v}} P(C_{d,k}|\phi_{k,v}, \text{rest}) p(\phi_{k,v}|\text{rest}) d\phi_{k,v} \\ &\times \int_\Omega P(A_d|\Omega, \text{rest}) p(\Omega|\text{rest}) d\Omega \end{aligned} \tag{7.2}$$

All the integrals in (7.2) reduce to the ratio of two log partition functions. For example, the product of integrals containing $C_{d,k}$ reduce to:

$$\begin{aligned} &\prod_k \int_{\phi_{k,v}} P(C_{d,k}|\phi_{k,v}, \text{rest}) p(\phi_{k,v}|\text{rest}) d\phi_{k,v} \\ &= \prod_k \frac{\prod_w \Gamma\left(C^{DKW,X_2=0}_{dkw} + C^{VKW}_{vkw,-d} + \alpha_1\right)}{\Gamma\left(\sum_w \left[C^{DKW,X_2=0}_{dkw} + C^{VKW}_{vkw,-d} + \alpha_1\right]\right)} \times \frac{\Gamma\left(\sum_w \left[C^{VKW}_{vkw} + \alpha_1\right]\right)}{\prod_w \Gamma\left(C^{VKW}_{vkw,-d} + \alpha_1\right)} \end{aligned} \tag{7.3}$$

Unfortunately, the above scheme does not mix well because the value of the integrals in (7.2) are very low for any view other than the view of the document in the current state of the sampler. This happens because of the tight coupling between v_d and the indicators $(\mathbf{x_1}, \mathbf{x_2}, \mathbf{z})$. To remedy this problem we used a Metropolis-Hasting step to sample $(v_d, \mathbf{x_1}, \mathbf{x_2}, \mathbf{z})$ jointly. We construct a set of V proposals each of which is indexed by a possible view: $q_v(\mathbf{x_1}, \mathbf{x_2}, \mathbf{z}) = P(\mathbf{x_1}, \mathbf{x_2}, \mathbf{z}|v_d = v, \mathbf{w_d})$. Since we have a collection of proposal distributions, we select one of them at random at each step. To generate a sample from $q_{v*}()$, we run a few iterations of a restricted Gibbs scan over the document d conditioned on fixing $v_d = v*$ and then take the last sample jointly with $v*$ as our proposed new state. With probability min(r,1), the new state $(v*, \mathbf{x_1}*, \mathbf{x_2}*, \mathbf{z}*)$ is accepted, otherwise the old state is retained. The acceptance ratio,r, is computed as: $r = \frac{p(\mathbf{w_d}|v*, \mathbf{x_1}*, \mathbf{x_2}*, \mathbf{z}*)}{p(\mathbf{w_d}|v, \mathbf{x_1}, \mathbf{x_2}, \mathbf{z})}$, where the non-* variables represent the current state of the sampler. It is interesting to note that the above acceptance ratio is equivalent to a likelihood ratio test. We compute the marginal probability $P(\mathbf{w_d}|..)$ using the estimated-theta method [124].

We evaluated the semi-supervised extension using the blog-2 dataset as follows. We reveal R% of the labels in the training set; then we train mview-LDA only over the labeled portion and train the semi-supervised version (ss-mview-LDA) on both the labeled and unlabeled documents. Finally we evaluate the classification performance on the test set. We used $R = \{20, 40, 80\}$. The results are given in Table 7.1 which shows a decent improvement over the supervised mview-LDA.

7.8 DISCUSSION

In this chapter, we addressed the problem of modeling ideological perspective at a topical level. We developed a factored topic model that we called multiView-LDA or mview-LDA for short. mview-LDA factors a document collection into three set of topics: ideology-specific, topic-specific, and ideology-topic ones. We showed that the resulting representation can be used to give a bird-eyes' view to where each ideology stands with regard to mainstream topics. Moreover, we illustrated how the latent structure induced by the model can by used to perform bias-detection at the document and topic level, and retrieve documents that represent alternative views.

It is important to mention that our model induces a hierarchical structure over the topics, and thus it is interesting to contrast it with hierarchical topic models like hLDA [25] and PAM [82, 90]. First, these models are unsupervised in nature, while our model is supervised. Second, the semantic of the hierarchical structure in our model is different than the one induced by those models since documents in our model are constrained to use a specific portion of the topic structure while in those models documents can freely sample words from any topic. Finally,in the future we plan to extend our model to perform joint modeling and summarization of idealogical discourse.

Part IV

MODELING USERS

8

LERANING UERS DYNAMIC ACTIVITY PROFILES

8.1 INTRODUCTION

Computational advertising, content targeting, personalized recommendation, and web search, all benefit from a detailed knowledge of the interests of the user in order to personalize the results and improve relevance. For this purpose, user activity is tracked by publishers and third parties through browser cookies that uniquely identify the user visits to web sites. A variety of different actions are associated with each user as web page visits, search queries, etc. These actions are distilled into a compact description of the *user profile*.

In this paper we focus on generation of compact and effective user profiles from the history of user actions. One of the key challenges in this task is that the user history is a mixture of user interest over a period of time. In order to reason about the user, the personalization layer needs to be able to separate between the distinct interests to avoid degrading the user experience.

The framework presented in this paper is based on topic models which is able to capture the *user interests* in an *unsupervised* fashion. We demonstrate the effectiveness of this framework for the *audience selection* task in display advertising. Audience selection is one of the core activities in display advertising where the goal is to select a good target for a particular campaign.

There are several challenges with topical analysis of the user action streams. First, for production strength system, the analysis has to scale to hundreds of millions of users with tens, if not hundreds of actions daily. Most of the topical analysis models are computationally expensive and cannot perform at this scale. The second issue is that of time dependence: Users' interests change over time and it is this change that proves to be commercially very valuable since it may indicate purchase intents. For example, if a user suddenly increases the number of queries for the Caribian and cruises, this may indicate interest in a cruise vacation. The appearance of this new topic of interest in the user's stream of actions can be predictive of his interest. Furthermore, there are external effects that may govern user behavior. For instance, an underwater oil spill does not convert users into deep sea exploration aficionados yet it affects search behavior and thus needs to be filtered out.

Finally, generating good features that *describe* user behavior does not necessarily translate into good features that are *predictive* of commercially relevant decisions. In other words, we ultimately want to obtain discriminatively useful features. Hence we would like to obtain a user profiling algorithm which has at least *the potential* of being adapted to the profiling task at hand.

We propose a coherent approach using Bayesian statistics to achieve all those goals and we show that it excels at the task of predicting user responses for display advertising targeting.

In summary the contributions of this paper are as follow:

- We develop a Bayesian approach for online modeling of user interests.

- Our model incrementally adapts both the user-topic associations and the topic composition based on the stream of user actions.
- We show how topic models can be used for improved targeting for display advertising, an application that requires analysis of *tens of millions* of users in real time.

8.2 FRAMEWORK AND BACKGROUND

We begin by spelling out the intuition behind our model in qualitative terms and describe the concepts behind the statistical model which is capable of addressing core profile generation issues, which we will address below and in Section 8.3:

- unsupervised topic acquisition
- dynamic topic descriptions
- dynamic user topical description
- filtering-out of global events

When categorizing user behavior it is tempting to group users into categories. That is, one might aim to group users together based on their (similar) behavior. While this has been used successfully in the analysis of user data [1] it tends to be rather limited when it comes to large numbers of users and large amounts of behavioral data: With millions of users it makes sense to use more than 1000 clusters in order to describe the user behavior. However, an inflation of clusters decreases the interpretability of the clusters considerably. Secondly, we would like to exploit sharing of statistical strength between clusters, e.g. if a number of clusters contain users interested in Latin music one would like to transfer modeling knowledge between them.

This can be addressed by topic models such as Latent Dirichlet Allocation (LDA) [21]. There objects (users) are characterized by a mixture of topics (interests). It is intuitively clear that such a topical mixture may carry considerably more information. For instance, even if we are allowed to choose only two out of 1000 topics we can already model 500,000 possible combinations. In other words, we can model the same variety of observations in a considerably more compressed fashion than what would be needed if we wanted to describe things in terms of clusters. The other key advantage of such models is that they are fully *unsupervised*. That is, they require *no* editorial data. Instead, they build the entire model by attempting to describe the data available. A key benefit in this setting is that it allows us to cover the entire range of interests rather than only those that an editor might be aware of. As a result this method is not language and culture specific. In the following we will be using the words topic and interest interchangeably.

Our strategy is thus to describe users as a mixture of topics and to assume that each of their actions is motivated by choosing a topic of interest first and subsequently a word to describe that action from the catalog of words consistent with that particular interest. For the purpose of this paper the **user actions** are either to issue a query or view an object (page, search result, or a video). We represent each user as a *bag of words* extracted from those actions and we use the term *user action* to denote generating a **word** from this bag. For instance, when issuing a query, each word in the query is an action. Similarly, when viewing a

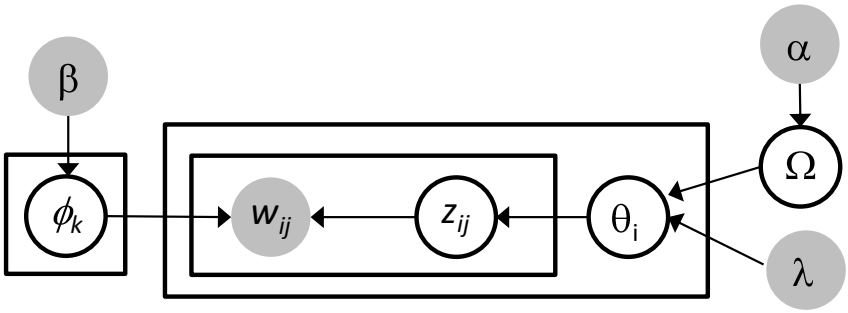


Figure 8.1: Latent Dirichlet Allocation.

video, each textual tag of the video represents an actions, and when viewing a page, each word in the title of the page is an action.

In the following subsections we first review the Latent Dirichelt Allocation model (LDA) and then give an equivalent Polya-Urn scheme representation which will serve as the basis for the time-varying model described later in the paper.

8.2.1 *Latent Dirichlet Allocation*

LDA was introduced by [21] for the purpose of document modeling. The associated graphical model is given in Figure 8.1. In it, the variables w_{ij} correspond to the j-th action of user i that we observe. All remaining variables are considered latent, i.e. they are unobserved.
The associated graphical model assumes that the order in which the actions w_{ij} occur is irrelevant, both in terms of their absolute position relative to other users and in terms of their local position. θ_i represents user i's specific distribution over topics (interests), and ϕ_k represents the topic distribution over words. Typically one chooses for $p(\theta_i|\lambda\Omega)$ and for $p(\phi_k|\beta)$ Dirichlet distributions, hence the name Dirichlet Allocation. The full generative process is specified as follow:

1. Draw once $\Omega|\alpha \sim \text{Dir}(\alpha/K)$.
2. Draw each topic, $\phi_k|\beta \sim \text{Dir}(\beta)$.
3. For each user i:
   a) Draw topic proportions $\theta_i|\lambda, \Omega \sim \text{Dir}(\lambda\Omega)$.
   b) For each word
      (a) Draw a topic $z_{ij}|\theta_d \sim \text{Mult}(\theta_i)$.
      (b) Draw a word $w_{ij}|z_{ij}, \phi \sim \text{Multi}(\phi_{z_{ij}})$.

In the above generative process, we decided to factor out the global distribution over topics into two parts: Ω and λ. Ω is a normalized vector, i.e., a probability distribution and represents the prior distribution over topics (interests) across users, and is sampled from a symmetric Dirichlet prior parametrized by α/K. λ on the other hand, is a scalar, and controls how each user's distribution over interests might vary from the prior distribution. This factored representation is critical for the rest of this paper.

8.2.2 *A Polya-Urn Representation of LDA*

In order to make explicit how the above LDA generative process captures local and global users' interests, we present an equivalent representation obtained by integrating out the global topic distribution Ω and the user-specific topic distribution θ. Let us assume that user i has generated $j-1$ actions and considering generating action j. Among those $j-1$ actions, let n_{ik} represent the number of actions expressing interest k. To generate action j the user might choose to either repeat an old interest with probability proportional to n_{ik} (and increment n_{ik}) or to consider a totally new interest with probability proportional to λ. In the former case, the user action is controlled by the user's mindset, and in the latter case the new interest is decided by considering the global frequency of interests across all users. We let m_k represent the global frequency by which interest k is expressed across users. Thus, user i can select to express interest k with probability proportional to $m_k + \alpha/K$ and increment m_k (as will as n_{ik}). This allows the model to capture the fact that not every action expressed by the user represents a genuine user's interest. For example, a user might search for articles about the oil spill just because of the large buzz created about this incident. Putting everything together, we have:

$$P(z_{ij} = k | w_{ij} = w, rest) \propto \left(n_{ik} + \lambda \frac{m_k + \frac{\alpha}{K}}{\sum_{k'} m_{k'} + \alpha} \right) P(w_{ij} | \phi_k) \tag{8.1}$$

In the literature, this model is known as the hierarchical Polya-Urn model [19] in which previously expressed interests have a *reinforcing* effect of being re-expressed either at the user-level or across users. Moreover, this model is also equivalent to a *fixed-dimensional* version of the hierarchical Dirichlet process (HDP) [116]. In Figure 8.2 we graphically show this representation. The static model is equivalent to a single vertical slice (with no prior over m nor n). This figure makes explicit that every visit to the global process by user i creates a new table which is denoted by a big circle. Thus m_k represents the total number of tables associated with topic k across all users. Note that in the standard HDP metaphor, to generate an action j, one first selects a table proportional to its popularity (or a new table with probability $\propto \lambda$), and then generates the action from the topic associated with the selected table. The process described above is strictly equivalent since the probability of choosing topic k under this standard HDP metaphor is thus equal to the number of words assigned to all tables serving topic k (which we denoted by n_{ik}). In Section 8.3, we will describe the time-varying version of this process.

8.3 TIME-VARYING USER MODEL: TVUM

We now introduce our model : time-varying user model (TVUM). In Section 8.2.1, we assumed that user actions are fully exchangeable, and that user's interests are fixed over time. It is reasonable though to assume that a user's interests are not fixed over time, instead, we may assume that the proportions change and that new interests may arise. For instance, after the birth of a baby users will naturally be interested in parenting issues and their preferences in automobiles

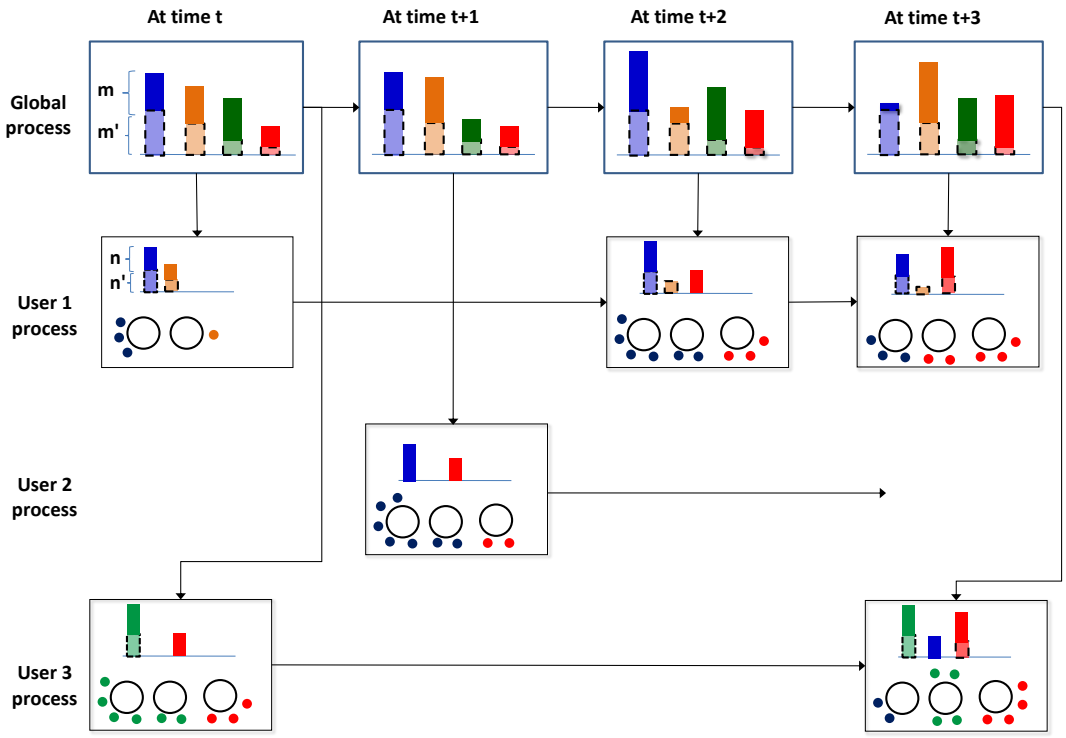


Figure 8.2: A time-varying hierarchical Polya-urn representation of the TVUM process (can also be regarded as a fixed-dimensional,**fully-evolving** recurrent Chinese restaurant franchise process [12]). Here (unlink in [12]) all levels are evolving. The top level represents the global process that captures the global topic trends. Each row gives the process for a given user. Each bar represents a topic's popularity (either global popularity m, or user-specific popularity n). The dotted (bottom) part of each bar represents the prior ($\tilde{m}$ for global topic prior, and $\tilde{n}$ for user-specific topic prior). To avoid clutter, we only show first-order dependencies, however, the global process evolves according to an exponential kernel and the user-specific processes evolve according to a multi-scale kernel as shown in Figure 8.3. Note here also that user interactions are sparse and users need not appear at all days nor enter in the system in the same day. Finally, small circles in user processes represent words and are colored with their corresponding topics. (best viewed in color)

may change. Furthermore, specific projects, such as planning a holiday, purchasing a car, obtaining a mortgage, etc. will lead to a marked change in user activity. In the classical document context of LDA this is the equivalent of allowing the topics contained in a document to drift slowly as the document progresses. In the context of users this means that we assume that the daily interest distribution per user is allowed to drift slowly over time.

We divide user actions into epochs based on the time stamp of the action. The epoch length depends on the nature of the application and can range from a few minutes to a full day as in our motivating application. Figure 8.2 depicts the generative process of TVUM. Users' actions inside each epoch are modeled using an epoch-specific , fixed-dimensional hierarchical Polya-Urn model as described in Section 8.2.2. As time goes by, three aspects of the static model change: the global distribution over interests, the user-specific distribution over interests and the topic

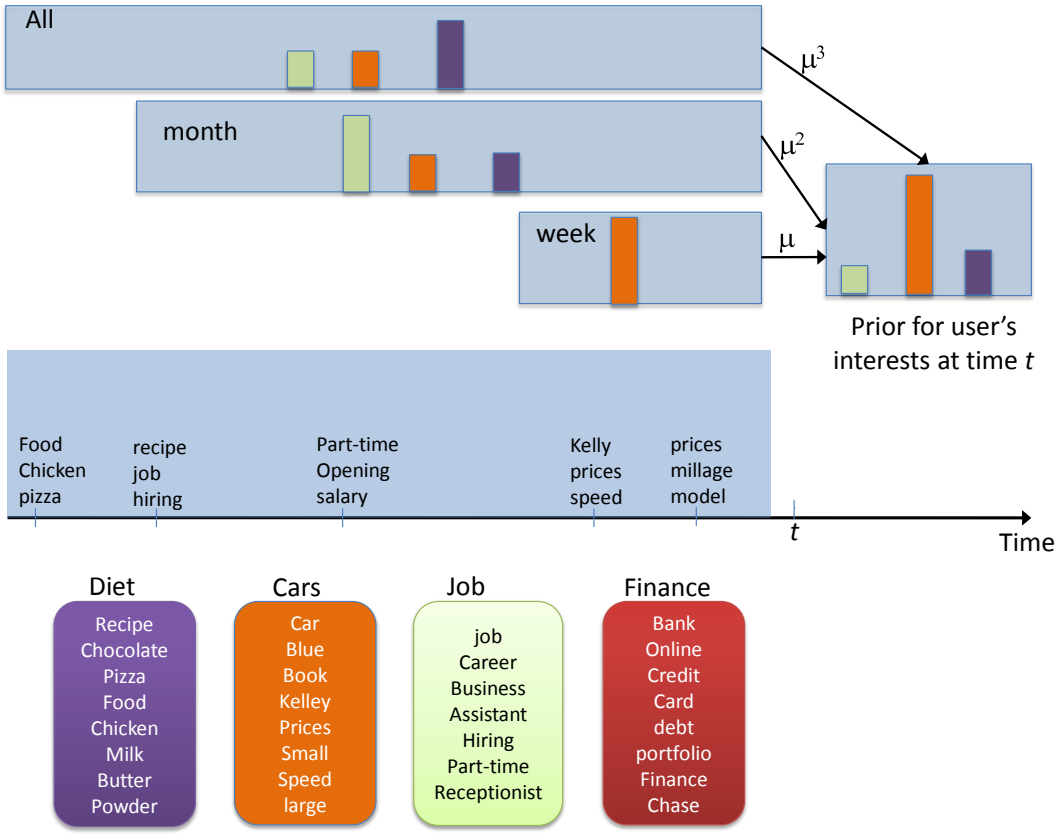


Figure 8.3: An illustrative example describing how TVUM captures the user's long and short-term interests. (best viewed in color)

distribution over words, ϕ. We address each of which in the following subsections and then summarize the full process in Section 8.3.4

8.3.1 *Dynamic Global Interests*

The global trend of interests change over time. For example, the release of the iPhone 4 results in an increase in the global interest of the 'mobile gadgets' topic (intent) for a few days or weeks to follow. To capture that, we use a similar idea to that introduced in [12] and stipulate that the popularity of an interest k at time t depends both on the topic usages at time t, m_k^t, and on its historic usages at previous epochs, where the contribution of the previous epochs is summarized as $\tilde{m}_k^t$. We use exponential decay with kernel parameter κ defined as follows:

$$\tilde{m}_k^t = \sum_{h=1}^{t-1} \exp^{\frac{h-t}{\kappa}} m_k^h. \tag{8.2}$$

8.3.2 *Dynamic User's Interests*

Now we turn to model the dynamics in the user-specific interests. The topic trends of a given user n_{ik} is now made dependent on time via n_{ik}^t. This is after all, the

very variable that we wish to estimate. We could use the same exponential decay idea that we used in modeling the change in the global trends of interests, however, the change in the user's interests over time is rather more complicated. Consider the set of actions observed from user i up to time $t-1$ as depicted in Figure 8.3. For simplicity, assume that all of these actions are words from queries. Each day, we observe a set of queries issued by the user. In the figure we also list a set of 4 interests (topics). For each topic, we give the top words in its distribution as recorded in ϕ. What could be the expected interests of the user at time t? To answer this question, we observe that we can factor out the interests of the user into short-term interests and long-term (or persistent) interests. For example, this user has a long-term interest in Diet (Food) related materials. From the user history we can also discern two transient short-term interests localized on time: one over finding a job and the other over buying a car. To discover the long-term interests of the user, we could count the frequency of expressing interest k across the whole user history. This gives the long-term interests of the user. As depicted in Figure 8.3, this shows that the user has a long-term interest over Diet. Similarly we could compute the same quantity for the recent week and month to get the short-term interests of the user. As shown in Figure 8.3, it is clear that in the recent month, the user was mainly looking for a job and then started to look for a car in the recent week. Thus to get the *expected* user-specific's popularity over interests $\tilde{\mathbf{n}}_i^t$ for user i at time t , we combine these three levels of abstractions using a weighted sum as follows:

$$\tilde{n}_{ik}^t = \mu_{\text{week}}\tilde{n}_{ik}^{t,\text{week}} + \mu_{\text{month}}\tilde{n}_{ik}^{t,\text{month}} + \mu_{\text{all}}\tilde{n}_{ik}^{t,\text{all}} \tag{8.3}$$

Here $\tilde{n}_{ik}^t$ is our estimate for user i's interest over topic k at time t, and $\tilde{n}_{ik}^{t,\text{week}} = \sum_{h=t}^{t-7} n_{ik}^h$, n_{ik}^h is the frequency of expressing interest k by user i at time h, and $\tilde{n}^{t,\text{week}}$ is just the sum of these frequencies over the past week. The other two quantities $\tilde{n}^{t,\text{all}}$ and $\tilde{n}^{t,\text{month}}$ are defined accordingly over their respective time ranges. The set of weights μ gives the contribution of every abstraction to the final estimate. In Figure 8.3, we show the final estimate at time t and it shows that we expect that the user will continue to look for cars, might still look for Diet related content, or with some probability considers a new job. There are a few observations here. First, if we had used an exponential decay as we employed for modeling the global trends, the model would have forgotten quickly about the user's long-term interest in Diet. This happens because the user history is sparse and the time between the user's activities might range from days to weeks. However, for modeling the change in the global trends of topics, it is enough to consider the preceding days to get a good estimate about the future: if an interest starts to fade, it is likely that it will die soon unless the data at future days tells us something else. The second observation is that $\tilde{\mathbf{n}}_i^t$ is not enough to capture what the user will look for at time t, as the user might still consider a new interest. For example, the user might develop an interest in obtaining a loan to finance his new car purchase. To model this effect, we combine the aforementioned prior with the global distribution over interests, as we will detail in Section 8.3.4, to generate the user actions at time t.

Finally, it is hard to fit or tune three different weights as in (8.3), thus we use $\mu_{\text{week}} = \mu, \mu_{\text{month}} = \mu^2$ and $\mu_{\text{all}} = \mu^3$, where $\mu \in [0, 1]$. For values of μ close to

0, more weight is given to the short-term interests and for values of μ close to 1 a uniform weight is given to all levels which implies more weight given to the long-term interests as they are aggregates over a longer time period. It is quite straightforward to optimize μ on a per user basis using gradient decent or set it to a default value for all users.

8.3.3 *Dynamic Topics*

The final ingredient in our model is rendering the topics themselves dynamic. Indeed as we observe more user interactions, we expect the topic distribution over words to change in response to world events like the release of a new car. This feature is related to earlier work in dynamic topic models [27, 12] which uses non-conjugate logistic normal priors (which are challenging during inference) and [67] which uses multi-scale priors similar to Figure 8.3. Our approach here resembles [67] but with a simpler form for the prior. We achieve this dynamic effect by making the topic distribution over words ϕ_k time dependent, i.e. we have ϕ_k^t which now depends on the smoothing prior $\tilde{\iota}_k^t$ in addition to the static prior β. The component $\tilde{\beta}_{kw}^t$ of $\tilde{\iota}_k^t$ depends on a decayed version of the frequencies of observing word w from topic k at times $1 : t-1$. This is similar to the way we modeled the dynamics of the global distribution over interests, and we use an exponential decay kernel parametrized by κ_0:

$$\tilde{\beta}_{kw}^t = \sum_{h=1}^{t-1} \exp^{\frac{h-t}{\kappa_0}} n_{kw}^h, \tag{8.4}$$

Here n_{kw}^h is the frequency of observing word w from topic k at day h. Finally, we have $\phi_k^t \sim \mathrm{Dir}\left(\tilde{\iota}_k^t + \beta\right)$

Note that the decision to *add* β to the above prior enforces that the components of the prior corresponding to new words (i.e. words that have $\tilde{\beta}_{kw}^t = 0$)is non-zero and as such new words can appear as time goes by.

8.3.4 *Summary of TVUM*

We now put all the pieces together and give the full generative process of TVUM. Consider generating action j for user i at time t. User i can choose an interest k with probability proportional to $n_{ik}^t + \tilde{n}_{ik}^t$ and then increments n_{ik}^t. Alternatively, the user can choose a new interest with probability proportional to λ. To select this new interest, he considers the global trend of interests across users: he select interest k with probability proportional to $m_k^t + \tilde{m}_k^t + \alpha/K$ and increment m_k^t as well as n_{ik}^t. Finally the user generates the word w_{ij}^t from topic k's distribution at time t, ϕ_k^t. Putting everything together, we have:

$$(z_{ij}^t = k | w_{ij}^t = w, \mathrm{rest}) \propto \left(n_{ik}^t + \tilde{n}_{ik}^t + \lambda \frac{m_k^t + \tilde{m}_k^t + \frac{\alpha}{K}}{\sum_{k'} m_{k'}^t + \tilde{m}_{k'}^t + \alpha}\right) P(w_{ij}^t | \phi_k^t) \tag{8.5}$$

In fact, taking the limit of this model as $K \to \infty$ we get the recurrent Chinese restaurant franchise process as in [12] albeit with all levels being evolved and with

different level-specific time-kernels. In [12] only the top level evolves since a given document does not persist across time epochs. Moreover, our fixed-dimensional approximation is more amenable for distributed inference and is still not highly affected by the number of topics as we will demonstrate in the experimental section.

8.3.5 *Inference*

The problem of inferring interests for millions of users over several months is formidable and it considerably exceeds the scale of published work on scalable topic models including [112]. As a result exact inference, even by sampling, is computationally infeasible: for instance, in a collapsed sampler along the lines of [58, 133] the temporal interdependence of topic assignments would require inordinate amounts of computation even when resampling single topic assignments for user actions: it affects all actions within a range of one month if we use a correspondingly long dependence model to describe the topic and interest evolution.

A possible solution is to use Sequential Monte Carlo (SMC) methods such as those proposed by [32] for online inference of topic models. The problem is that due to the long-range dependence the particles in the SMC estimator quickly become too heavy, so we need to rebalance and resample fairly frequently. This problem is exacerbated by the sheer amount of data — we need to distribute the inference algorithm over several computers. This means that whenever we resample particles we need to transfer and update the state of many computers. Such an approach very quickly becomes infeasible and we need to resort to further approximations.

We make the design choice of essentially following an inference procedure. That is, we only perform forward inference through the set of dependent variables for each time step and we attempt to infer $\mathbf{z}^t$ only given estimates and observations for $t' \leqslant t$. This allows us to obtain improving results as time progresses, alas at the expense of rather suboptimal estimates at the beginning of the inference chain.

A second reason for the design choice to perform forward sampling only is a practical one: data keeps on arriving at the estimator and we obviously would like to update the user profiles as we go along. This is only feasible if we need not revisit old data constantly while new data keeps on arriving. Our setting allows effectively for an online sampling procedure where we track the current interest distribution and the nature of interests relative to incoming data. Thus to summarize our inference procedure incrementally runs a *fast* batch MCMC over the data at epoch t given the state of the sampler at earlier epochs.

8.3.5.1 *Sampling Details*

Our sampler resembles in spirit the collapsed, direct-assignment sampler with augmented representation as described in [116]. We collapse the topic multinomials (ϕ^t) and compute the posterior over $\mathbf{z}^t$ given assignments to hidden variables at previous epochs. As noted in Equation (8.5), sampling z would couple topic indicators across all users via $\mathbf{m}^t$ (recall from Figure 8.2 that m_k^t is the number of tables across all users serving topic k). This fact is undesirable especially when users are distributed across machines. To remedy this, we augment the representation by

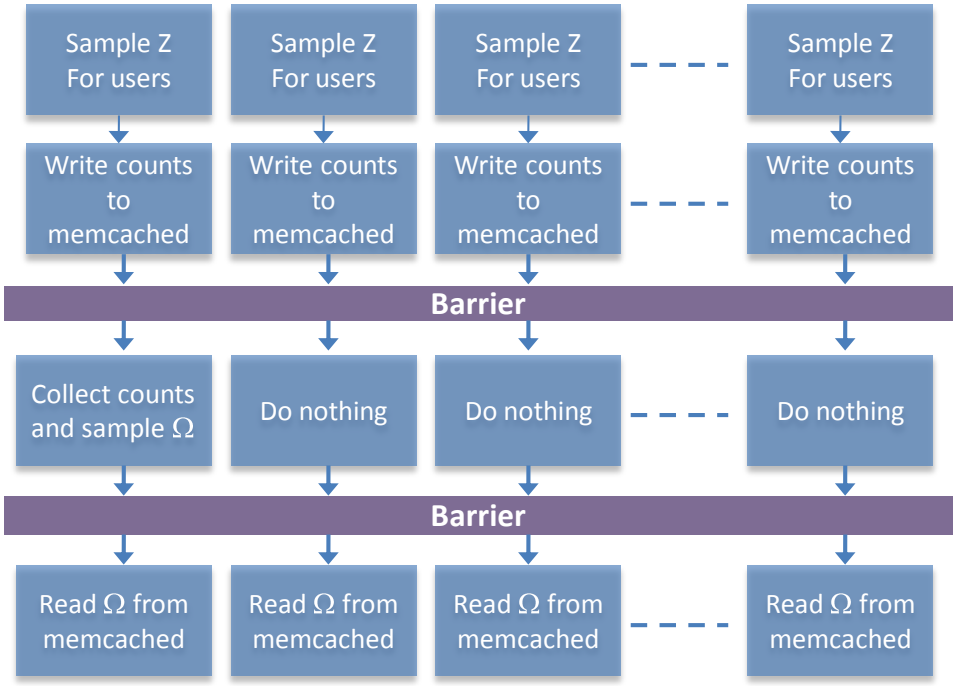


Figure 8.4: Synchronization during a sampling cycle.

instantiating and sampling the global topic distribution at time time, i.e. Ω^t. Thus the sampler alternate between sampling $\mathbf{z}^t, \mathbf{m}^t$, and Ω_t. It should be noted that collapsing ϕ^t also introduces coupling across users, however we deal with that using the same architecture in [112].

Sampling z_{ij}^t: The conditional probability for z_{ij}^t has the same structure as in LDA albeit with the previously defined compound prior. We have:

$$P(z_{ij}^t = k | w_{ij}^t = w, \Omega^t, \tilde{\mathbf{n}}_i^t) \propto \left(n_{ik}^{t,-j} + \tilde{n}_{ik}^t + \Lambda\Omega^t\right) \frac{n_{kw}^{t,-j} + \tilde{\beta}_{kw}^t + \beta}{\sum_l n_{kl}^{t,-j} + \tilde{\beta}_{kl}^t + \beta} \tag{8.6}$$

where n_{kw}^t is the number of times a word w was sampled form topic k (this is known as the topic-word matrix). The notation $-j$ means excluding the contribution of word j. The structure of (8.6)allows us to utilize the efficient sparse sampler described in [133] which in fact was another reason to our choice of using the augmented representation (i.e. instantiating Ω^t).

Sampling m_k^t: Since our sampler does not explicitly maintain the word-table assignments, we need to sample $m_k^t = \sum_i m_{ik}^t$. Where m_{ik}^t is the number of tables in user i's process serving topic k. This last quantity m_{ik}^t follows what is called the Antoniak distribution [116]. A sample from this distribution can be obtained as follows. First, set $m_{ik}^t = 0$, then for $j = 1 \cdots n_{ik}^t$, flip a coin with bias $\frac{\lambda\Omega_k^t}{j-1+\lambda\Omega_k^t}$, and increment m_{ik}^t if the coin turns head. The final value of m_{ik}^t is a sample from the Antoniak distribution. Recall that n_{ik}^t is the number of words expressing interest k from user i at time t.

Sampling Ω^t: By examining the structure of Equations (8.1,8.5), and the equivalence between the construction in Section 8.2.2 and LDA, it is straighforward to see that $P(\Omega^t | \mathbf{m}^t, \tilde{\mathbf{m}}^t) \sim \text{Dir}(\tilde{\mathbf{m}}^t + \mathbf{m}^t + \alpha/K)$.

This constitutes a **single** Gibbs cycle. For each time epoch (day) we iterate this cycle **100 times** over all active users in that day. 100 iterations were enough due to 1) the small size of user observations at each day, and 2) the informative prior from earlier days.

8.3.5.2 *System Aspects*

To be able to apply the model to tens of millions of users, we use a distributed implementation following the architecture in [112]. The state of the sampler comprises the topic-word counts matrix, and the user-topic counts matrix. The former is shared across users and is maintained in a distributed hash table using `memcached` [112]. The later is user-specific an can be maintained locally in each client. We distribute the users at time t across N clients. Each client executes a foreground thread to implement (8.6) and a background thread that synchronizes its local copy of the topic-word counts matrix with the global copy in memcached. As shown in Figure 8.4, after this step, Ω^t and m_k^t should be sampled. However, sampling Ω^t requires a *reduction* step across clients to collect and sum m_{ik}^t. First, each client writes to memcached its contribution for m_k^t (which is the sum of the values of m_{ik}^t for users i assigned to this client), then the clients reach a *barrier* where they wait for each other to proceed to the next stage of the cycle. We implemented a sense-reversing barrier algorithm which arranges the nodes in a tree and thus has a latency that scales logarithmically with N [89]. After this barrier, all counts are in memcached and as such one client sums these counts to obtain $\mathbf{m}^t$, uses it to sample Ω^t, and finally writes the sampled value of Ω^t back to the memcached. All other clients wait for this event (signaled via a barrier), and then read the new value of Ω^t and this finalizes a single cycle.

8.4 EXPERIMENTS

In our experiments we demonstrate three aspects which are crucial for practical use of the TVUM for the purpose of user profiling:

- We show that the discovered user interests are effective when used as features for behavioural targeting. This holds both in the sense of generating *interpretable* features and in the sense of generating *predictive* models.
- Secondly we demonstrate the scalability of our inference algorithm across both the number of machines and number of users. In particular we demonstrate that our algorithm is capable of processing tens of millions of users quite efficiently. This is over 10 times larger than the largest dataset analyzed by [112] (they report the largest scale results we are aware of) who only discuss plain LDA.
- Third, we show that our model and inference algorithm are robust to a wide range of parameter settings. This means that it is applicable to a wide range of problems without the need to excessively hand-tune its parameters.

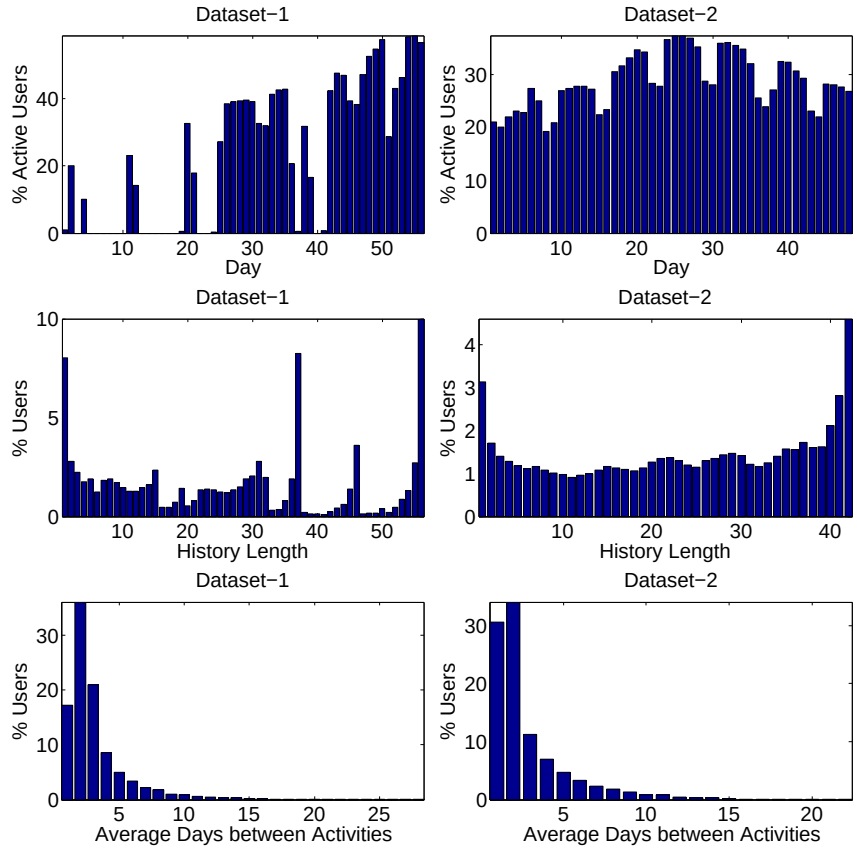


Figure 8.5: Characteristics of the data sets.

dataset	# days	# users	Voc	# campaigns	size
1	56	13.34M	100K	241	242GB
2	44	33.5M	100K	216	435GB

Table 8.1: Basic statistics of the data used.

8.4.1 *Datasets and Tasks*

We used two datasets each of which spans approximately two months. The exact dates of each dataset are hidden for privacy reasons. In each case we collected a set of ad-campaigns and sampled a set of users who were shown ads in this campaign during the covered time period. For each user, we represented the users' interaction history as a sequence of tokens. More precisely, the user *is represented* at day t as *a bag of words* comprising the queries they issued at this day and the textual *content* of the pages they view on this day. This constitutes the input to all the models discussed in the following subsections. Thus all datasets are Yahoo! privacy policy compliant, since we don't retain the precise page the user viewed on Yahoo! Sports (for instance) but rather retain only the description of the page at a higher level in terms of words extracted from the content.

For evaluation purposes, we also collect the user's response to individual ads: the ads they converted on (positive response); and the ads they ignored either by not clicking or not coverting post-click (negative response). We used the users'

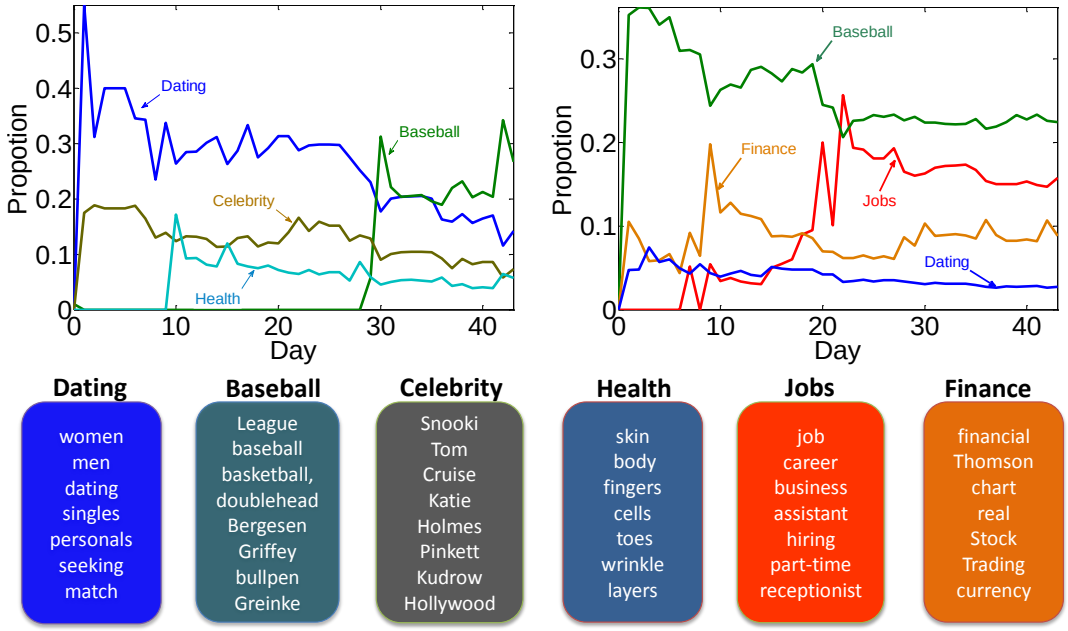


Figure 8.6: Dynamic interests of two users.

responses to ads in the last week as a test set and the remaining responses as a training set. The characteristics of each dataset are shown in Figure 8.5 and Table 8.1. As evident from Figure 8.5, the first dataset is sparser than the second dataset which presents a challenge for any model.

Our task is to build a model based on users' history that can predict whether or not a user will convert given an ads. We break this problem in two parts. First, we translate the user's history into a feature-vector profile using the TVUM and other baselines. Second, the profiles are used to build an SVM model that is evaluated over the test set using ROC area under the curve as metrics.

8.4.2 *Interpretability*

Figure 8.6 qualitatively illustrates our model. For each user we plot the interest distribution over 6 salient topics. For each topic we print the top words as recorded in ϕ. The top part of the figure shows the dynamic interest of two users (User A — left; User B — right) as a function of time. Our model discovers that A has a *long-term* interest in dating, health, and celebrity albeit to a varying degree. In the last 10 days, A developed a *short-term* interest in baseball, quite possibly due to the global rise in interest in baseball (within our dataset). On the other hand B has a *long-term* interest in sports and finance; B was interested in dating in the first week, and was mainly looking for jobs in the last month. These discovered time-varying features for each user are useful for timely targeting of users with products. The following experiments buttress this intuition.

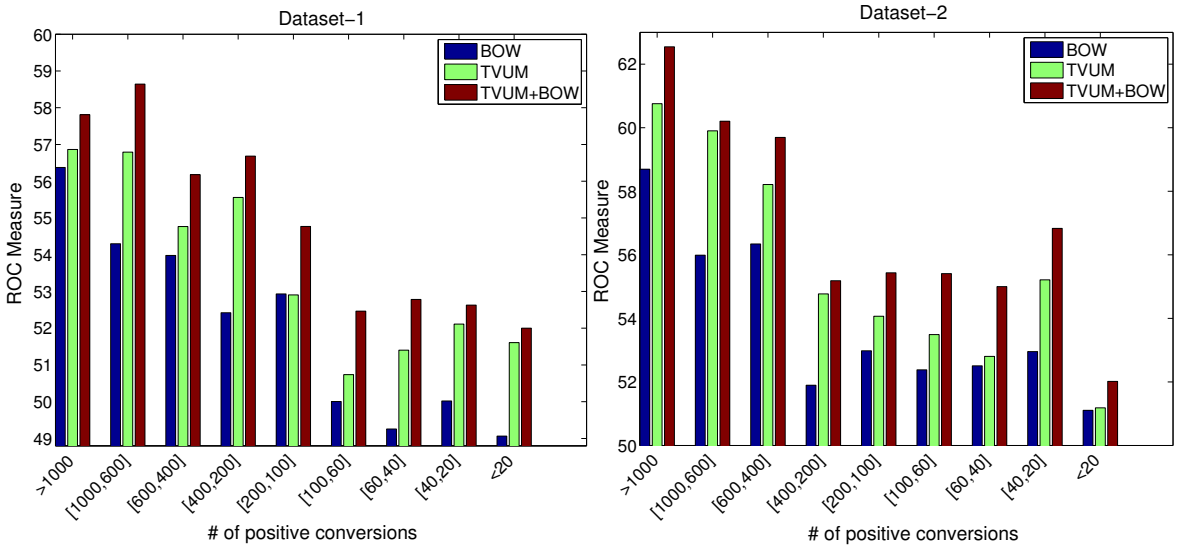


Figure 8.7: Performance vs. number of conversions.

8.4.3 *Performance*

To show that the proposed **TVUM** is effective for extracting user interest we compare it to two baselines: a bag of words model **(BOW)** which does *not* incorporate any topical or temporal dependencies, and a **static LDA** model which incorporates topical information but *no* time dependence. The bag of words (BOW) model just outputs the concatenation of the user history up until time t as its representation of the user's interest at time t. This is in fact a very strong baseline. One advantage of the TVUM is that it is capable of processing user data in an online fashion, that is, we may estimate the user interest distribution one day at a time, sweeping through the dataset.

The training data for the SVM consists of the user's distribution over intent at time t (or a bag of words alternatively) and the user's response to the ad. For the TVUM, we used the following default parameters:[1] we set the number of topics $K = 200, \beta = 0.01, \mu = e^{-1} \approx .6, \kappa = \kappa_0 = \lambda = 1$.

In Figure 8.7, we show a comparison between this BOW baseline and the TVUM as a function of the number of positive conversions in each campaign. In this figure, we also show a third model that combines the BOW features with the TVUM features. As evident, our model improves over the baseline in both datasets across all the campaigns. We observe that, in general, the number of positive examples in ad campaigns is scarce. Therefore, models that feature a compact representation of the user's history overall perform better and can leverage the BOW features to further improve its performance. We can also notice from this figure that the gain over the BOW baseline is larger in campaigns with medium to small number of conversions. We also experimented with a variation of the BOW baseline featuring **BOW+time decay**. In this variation of BOW, we use an *exponential decay* of the form $a^{t'-t}$ over the *word counts* in the BOW baseline such that distant history will

1 It is possible to learn μ for users with long history (greater than two weeks). The likelihood of μ is Drichlet-multinomial compound and μ can be learnt using gradient descent. We leave this as a future work.

Table 8.2: Average ROC measure across models (TVUM: time-varying user model, BOW: bag of words baseline).

	BOW	TVUM	TVUM+BOW	LDA+BOW
dataset 1	54.40	55.78	**56.94**	55.80
dataset 2	57.03	58.70	**60.38**	58.54

Table 8.3: Time (in minutes) per Gibbs iteration, i.e. time to process a single day for all users.

topics		50	100	200
dataset 1	50 machines	0.7	0.9	1.2
dataset 2	100 machines	0.9	1.1	1.5

Table 8.4: Effect of the number of topics on predictive accuracy.

	topics	TVUM	TVUM + BOW
dataset 1	50	55.32	56.01
	100	55.5	56.56
	200	**55.8**	**56.94**
dataset 2	50	59.10	60.40
	100	**59.14**	**60.60**
	200	58.7	60.38

Table 8.5: The effect of the topic-decay parameter on the performance of TVUM+BOW.

Decay(κ_0)		none	1	2	3
dataset 1	200 topics	56.2	**56.94**	56.51	56.51
dataset 2	100 topics	**60.68**	60.60	60.40	60.40

be given less weight at time t, however, the best result in this case was obtained when $a = 1$ (i.e. ignoring time), thus our model was able to leverage temporal information in a better way.

To summarize the performance of each model over all campaigns, we give in Table 8.2 a weighted average of the ROC measure across campaigns, where the weight of each campaign is proportional to the number of positive examples in the campaign.

Finally, Table 8.2 also shows a comparison with a static LDA model+BOW features. Due to the large scale of this dataset, we can not apply batch LDA directory over it. Instead, we sampled 2M users from the training set and built a static LDA model over them. Then, we used variational inference to infer our estimate of the user's interest at time t, where t can be in the training or test set. Note here that LDA does not have the notion of a user in the model in a sense that when predicting the user's history at time t_1 and later at t_2, the user history up until time t_1 and time t_2 are presented as separate documents to the model. As evident from Table 8.2, our model beats static LDA on both datasets due to its ability to model time, and to maintain a coherent representation of the user as

a time-evolving entity (note that in this application of predicting conversion, an increase of 1-2 points of the ROC measure is considered significant)

8.4.4 *Speed and Efficiency*

One key advantage of our model is its ability to process data in an online and distributed fashion. This allows us to process data at a scale quite unlike any system that we are aware of. Rather than processing a small number of millions of documents (i.e. users) [129, 98] while requiring a large cluster of computers, and rather than being able to process tens of millions of documents without time dependence [112], we are able to process tens to hundreds of millions of users in a dynamic fashion at a speed of less than **two hours per day** of data being processed. This is quite a significant improvement since it makes topic models suitable for Yahoo's user base.

Table 8.3 provides the time for a single Gibbs-sampling iteration for different number of topics and using different number of machines. Since dataset 2 is almost double the size of dataset 1 in terms of the number of users, we doubled the number of machines when processing dataset 2. As shown in this table, the time per iteration is almost the same across the two datasets. The slight increase in time for dataset 2 arises because of the cost of synchronization and inter-client communications which was kept low thanks to the logarithmic complexity of the tree-based barrier algorithm.

8.4.5 *Sensitivity Analysis*

We evaluate the sensitivity of our results to the model's parameters. Bayesian models depend on the specification of the parameters which encode our prior belief. However, no matter what prior is used, as long as the prior does not exclude the correct answer, the model can reach the true conclusion as the size of the dataset increases (a fact which is true in our application). We first study the effect of the number of topics and then the effect of the decay parameter κ_0 for the topic distribution over words. The effect of the decay parameter for the global topic distribution κ was negligible.

Number of Topics: We varied the number of topics in the range 50, 100, 200. Results are presented in Table 8.4. From these tables, we first observe that the effect on the final performance is not quite large in both datasets. This is largely due to the fact that our inference algorithm optimizes over the topic's global distribution Ω which allows it to adapt the *effective* number of topics for each dataset [122]. For dataset 1, the performance increases as we add more topics. This is due to the sparsity and non-homogeneity of this dataset. Thus, the arrival of new users at each day requires more topics to model their intents properly. Whereas dataset 2 was more homogeneous, and as such, 100 topics were enough. We recommend setting the number of topics in the low hundreds for behavioural targeting applications.

Topic-Distribution Decay: Decaying the topic's word distribution has two important roles. First, it allows the model to recover from a poor initial estimate, and second, it enables the model to capture drifts in the topic's focus. As shown in

Table 8.5 this feature helps in the first dataset due to the initial poor estimate of the topics in the first few days because of the low number of available users. However for the second homogeneous dataset, the performance *slightly* increases if we turn this feature off. We recommend setting κ_0 to a value between 1 and 3. It is possible to learn this parameter, however, this requires the storage of all the past topic distributions over time which is prohibitive in terms of storage requirements. In our implementation, because of the form of the decay function, we only need to discount the topic-word counts at the end of each day and use this as the initial estimate for the following day.

8.5 RELATED WORK

The emergence of the web has allowed for collection and processing of user data magnitudes larger than previously possible. Thus has resulted in spike of interest in user data analysis and profile generation as reported in [36, 52, 74, 79]. Profile generation has been reported for a few different applications. In [114] the authors describe profiles for search personalization. Here, the authors build profiles based on the query stream of the user and users similar to it. The authors also report an alternative that is based on the relevance feedback approaches in information retrieval over the documents that the user have perceived as relevant. Both techniques are orthogonal to the work presented in this paper and could be used to produce a potentially richer set of of features that will serve as an input to the topical analysis.

User profile generation is also studied in other online settings and also for content recommendation (e.g., [61, 76, 80]). Most of these focus on detecting the user's short term vs long term interest and using these in the proposed application. In our case, we blend the short term and long term interests into a single profile. A survey of user profile generation can be found in [52].

In the area of audience selection, Provost et al. [108] have recently shown that user profiles can be built based on co-visitation patterns of social network pages. These profiles are used to predict the performance of brad display advertisements over 15 campaigns. In [36] the authors discuss prediction of clicks and impressions of events (queries, page views) within a set of predefined categories. Supervised linear poisson regression is used to model these counts based on a labeled set of user-class pairs of data instances.

While there are many profile generating algorithms satisfying a partial set of requirements outlined in this paper we are unaware of methods covering the entire range of the desiderata. In the topical analysis area, there are many predictive algorithms which try modeling the observed data via static generative model. The examples include singular value decomposition of the (user, action) matrix thus yielding a technique also known as Latent Semantic Indexing [44] or more advanced techniques such as Probabilistic Latent Semantic Indexing [64] or Latent Dirichlet Allocation [21]. However, while reducing the dimensionality of the space (the number for unique features), LSI and PLSI yield a dense vectorial representation. All of these approaches are static in terms of the user and to be able to apply new data we need to recompute the topical model from scratch. Finally, several models exist in the literature that could accommodate the evolution of topic global trends ,

topic distribution over words [27, 67] and the number of topics [12]. However, we are not aware of *any* attempts to model the intra-document drift of topics which corresponds to a user in our application – which is the *main contribution* of this paper.

8.6 CONCLUSION

In this paper, we addressed the problem of user profile generation for behavioural targeting. We presented a time-varying hierarchical user model that captures both the user's long term and short term interests. While models of this nature were not known to scale for web-scale applications, we showed a streaming distributed inference algorithm that both scales to tens of millions of users and adapts the inferred user's interest as it gets to know more about the user. Our learnt user representation was experimentally proven to be effective in computational advertising.

There are several directions for future research. First, we focused in this paper on the dynamic aspect of user profiles, and integrated all available user actions into a single vocabulary. While this approach seems promising, it would be beneficial to model each facet of the user action independently perhaps using a different language model for each facet. It is also possible to build a supervised model as in [28] that uses the users' responses to historical ads in inferring the user's interests.

Part V

CONCLUSIONS

9 CONCLUSIONS AND FUTURE WORK

Online content has become the mean for information dissemination in many domains. It is reported that there are 50 million scientific journal articles published thus far [71], 126 million blogs [1], an average of one news story published per second [2], and around 27 million tweets per day. With this diverse and dynamic content, it is easy for users to miss the big picture in a sea of irrelevant content. In this thesis we focused on building a structured representation of content and users and derived efficient online and scalable inference algorithms to learn this representation from data. We have the following conclusions:

- *Mixed membership models and non-parametric Bayesian models* are very *flexible and modular* framework for modeling both users and content. We have demonstrated this flexibility by considering two different problems (temporal dynamics and multi-faceted content) and addressing them in two different domains (social media and scientific publications).

- *Non-parametric Bayesian models are not slow*. It is true that exact posterior inference in graphical models in general is an intractable problem, however, the posterior distribution is mostly peaked in real world applications, especially when there are large data, and as such the posterior distribution can be efficiently approximated. Moreover, we have demonstrated in this thesis that such models can be made to work on large scale (Chapter 8) and on real time (Chapter 5).

- *Unlabeled data helps*. While this fact is not surprising, we have demonstrated this fact on this thesis using the two domains we considered (as well as in a related work in the context of deep architectures [14]).

- *Tightly-coupled variables* are key to discover structure in the data. For instance the tight relationship between the story indicator (or ideology) of the document and its topic distribution (Chapter 5 and 7) were key to the success of these models. However, *tightly-coupled variables make inference challenging* especially when all the tightly-coupled variables are hidden as is the case in Chapter 5 and in the semi-supervised model in Chapter 7. Usually this problem is solved by using a blocked sampler that samples the coupled variables as a block, however this is intractable in our models since this requires sampling from a joint distribution whose size is exponential in the number of words in the document. In this thesis we proposed to *carry a few iterations of a restricted Gibbs scan* over those coupled variables and use a posterior sample from this scan as a proposal. This idea was inspired by the split-merge sampler proposed by Neal [68]. We have shown that this proposal

1 http://www.blogpulse.com/
2 Personal communication with the Yahoo! news team

works well inside a Metropolis-Hasting sampler in Chapter 7 and inside a sequential monto Carlo algorithm in Chapter 5.

We identify three directions for future work:

- *Dynamic multi-faceted models:* while we addressed the problems of modeling dynamic and multi-faceted content separately in this thesis, the modularity of our approaches makes it easy to combine our models. For instance, with such models one can examine how different ideologies converge and diverge over time with regard to their views on several issues.
- *Joint models:* in this thesis we addressed the problem of modeling scientific figures separately from modeling research papers. However, a joint whole-paper model would enable building better information retrieval systems.
- *Supervised models:* it is also interesting to close the loop in the user model proposed in Chapter 8 and use the user's response to the recommend content in the model using supervised topic models [28, 139]. Moreover, integrating user-user relationships form social network is another fruitful direction for future work perhaps using elements from our earlier work in network modeling[8, 6, 95]

BIBLIOGRAPHY

[1] D. Agarwal and S. Merugu. Predictive discrete latent factor models for large scale dyadic data. In P. Berkhin, R. Caruana, and X. Wu, editors, *KDD*, pages 26–35. ACM, 2007.

[2] Amr Ahmed, Andrew O. Arnold, Luis Pedro Coelho, Joshua Kangas, Abdul-Saboor Sheikk, Eric P. Xing, William W. Cohen, and Robert F. Murphy. Structured literature image finder. In *The Annual Meeting of The ISMB BioLINK Special Interest Group, held in association with ISMB09*, 2009.

[3] Amr Ahmed, Andrew O. Arnold, Luis Pedro Coelho, Joshua Kangas, Abdul-Saboor Sheikk, Eric P. Xing, William W. Cohen, and Robert F. Murphy. Parsing text and figures in biomedical literature. *J. Web Semantics*, 2010.

[4] Amr Ahmed, Qirong Ho, Choon hui, Jacob Eisenstein, Alex Somla, and Eric P. Xing. Online inference for the infinte topic-cluster model: Storylines from text stream. In *AISTATS*, 2011.

[5] Amr Ahmed, Yucheng Low, Mohamed Aly, Vanja Josifovski, and Alex Somla. Dynamic activity profiles for display advertising targeting. In *KDD*, 2011.

[6] Amr Ahmed, Le Song, and Eric Xing. Time-varying networks: Recovering temporally rewiring genetic networks during the life cycle of drosophila melanogaster. *RECOMB System Biology Satellite Meeting, Boston*, 2008.

[7] Amr Ahmed and Eric Xing. On tight approximate inference of the logistic normal topic admixture model. In *AISTATS*, 2007.

[8] Amr Ahmed and Eric P. Xing. Tesla: Recovering time-varying networks of dependencies in social and biological studies. *Proceeding of The National Academy of Science*, (29):11878–11883.

[9] Amr Ahmed and Eric P. Xing. Dynamic non-parametric mixture models and the recurrent chinese restaurant process: with applications to evolutionary clustering. In *SDM*, pages 219–230. SIAM, 2008.

[10] Amr Ahmed and Eric P. Xing. Collapsed variational inference for time-varying dirichlet process mixture models. In *Nonparametric Bayes Workshop, NIPS WS*, 2009.

[11] Amr Ahmed and Eric P. Xing. Staying informed: Supervised and semi-supervised multi-view topical analysis of ideological perspective. In *EMNLP*, 2010.

[12] Amr Ahmed and Eric P. Xing. Timeline: A dynamic hierarchical dirichlet process model for recovering birth/death and evolution of topics in text stream. In *UAI*, 2010.

[13] Amr Ahmed, Eric P. Xing, William W. Cohen, and Robert F. Murphy. Structured correspondence topic models for mining captioned figures in biological literature. In *KDD*, pages 39–48. ACM, 2009.

[14] Amr Ahmed, Kai Yu, Wei Xu, Yihong Gong, and Eric Xing. Training hierarchical feed-forward visual recognition models using transfer learning from pseudo-tasks. In *Proceeding of the 10th European Conference of Computer Vision*, 2008.

[15] Edoardo M. Airoldi, David M. Blei, Stephen E. Fienberg, and Eric P. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9:1981–2014, 2008.

[16] C. E. Antoniak. Mixtures of dirichlet processes with applications to bayesian nonparametric problems. *The Annals of Statistics*, 2(6):1152–1174, 1974.

[17] A. Asuncion, P. Smyth, and M. Welling. Asynchronous distributed learning of topic models. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *NIPS*, pages 81–88. MIT Press, 2008.

[18] Kobus Barnard, Pinar Duygulu, David Forsyth, Nando de Freitas, David M. Blei, and Michael I. Jordan. Matching words and pictures. *Journal of Machine Learning Research*, 3:1107–1135, 2003.

[19] D. Blackwell and J. MacQueen. Ferguson distributions via polya urn schemes. *The Annals of Statistics*, 1(2):353–355, 1973.

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

[21] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, January 2003.

[22] D. M Blei and Michael I. Jordan. Modeling annotated data. In *SIGIR*, 2003.

[23] David M. Blei and Peter Frazier. Distance dependent chinese restaurant processes. In *ICML*, pages 87–94, 2010.

[24] David M. Blei, Thomas L. Griffiths, and Michael I. Jordan. The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. *J. ACM*, 57(2), 2010.

[25] David M. Blei, Thomas L. Griffiths, Michael I. Jordan, and Joshua B. Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. In *NIPS*. MIT Press, 2003.

[26] David M. Blei and John D. Lafferty. Correlated topic models. In *NIPS*, 2005.

[27] David M. Blei and John D. Lafferty. Dynamic topic models. In *ICML*, volume 148, pages 113–120. ACM, 2006.

[28] David M. Blei and Jon D. McAuliffe. Supervised topic models. In *NIPS*. MIT Press, 2007.

[29] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[30] Jordan L. Boyd-Graber and David M. Blei. Syntactic topic models. In *NIPS*, pages 185–192. MIT Press, 2008.

[31] S. R. K. Branavan, Harr Chen, Jacob Eisenstein, and Regina Barzilay. Learning document-level semantic properties from free-text annotations. In *ACL*, pages 263–271. The Association for Computer Linguistics, 2008.

[32] K. R. Canini, L. Shi, and T. L. Griffiths. Online inference of topics with latent dirichlet allocation. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2009.

[33] F. Caron, M. Davy, and A. Doucet. Generalized polya urn for time-varying dirichlet processes. In *UAI*, 2007.

[34] J. Chang and D. Blei. Relational topic models for document networks. In *AISTATS*, 2009.

[35] Chaitanya Chemudugunta, Padhraic Smyth, and Mark Steyvers. Modeling general and specific aspects of documents with a probabilistic topic model. In *NIPS*, pages 241–248, 2006.

[36] Y. Chen, D. Pavlov, and J.F. Canny. Large-scale behavioral targeting. In J.F. Elder, F. Fogelman-Soulié, P.A. Flach, and M. J. Zaki, editors, *Proceedings of the 15th SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 209–218. ACM, 2009.

[37] W. Chong, D. Blei, and L. Fei Fei. Simultaneous image classification and annotation. In *CVPR*, pages 1903–1910, 2009.

[38] Luis Pedro Coelho, Amr Ahmed, Andrew Arnold, Joshua Kangas, Abdul-Saboor Sheikk, Eric P. Xing, William W. Cohen, and Robert F. Murphy. Structured literature image finder: Extracting information from text and images in biomedical literature. *Lecture Notes in Bioinformatics*, 2010.

[39] William W. Cohen, Richard Wang, and Robert F. Murphy. Understanding captions in biomedical publications. In *Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-03)*, pages 499–504, 2003.

[40] Margaret Connell, Ao Feng, Giridhar Kumaran, Hema Raghavan, Chirag Shah, and James Allan. Umass at tdt 2004. In *TDT 2004 Workshop Proceedings*, 2004.

[41] J. Lafferty D. Blei. Visualizing topics with multi-word expressions. *Arxiv*, page arXiv:0907.1013v1, July 2009.

[42] H. Daume. Fast search for dirichlet process mixture models. 2007.

[43] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407, 1990.

[44] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41, June 1990.

[45] T. A. Van Dijk. Ideology: A multidisciplinary approach. 1998.

[46] Arnaud Doucet, Nando de Freitas, and Neil Gordon. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, 2001.

[47] J. Eisenstein and E.P. Xing. The cmu-2008 political blog corpus. *CMU-ML-10-101 Technical Report*, 2010.

[48] E. Erosheva, S. Fienberg, and J. Lafferty. Mixed membership models of scientific publications. *PNAS*, 101(1), 2004.

[49] M. Escobar and M. West. Exchangeable and partially exchangeable random partitions. *Journal of the American Statistical Association*, 90:577–588, 1995.

[50] T. S. Ferguson. A bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230, 1973.

[51] Blaz Fortuna, Carolina Galleguillos, and Nello Cristianini. *Detecting the bias in media with statistical learning methods*. Taylor and Francis Publisher, 2008.

[52] S. Gauch, M. Speretta, A. Chandramouli, and A. Micarelli. User profiles for personalized information access. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The Adaptive Web, Methods and Strategies of Web Personalization*, volume 4321 of *Lecture Notes in Computer Science*, pages 54–89. Springer, 2007.

[53] A. Gelman, J. Carlin, Hal Stern, and Donald Rubin. *Bayesian Data Analysis*. Chapman-Hall, 2 edition, 2003.

[54] J. Geweke and H. Tanizaki. Bayesian estimation of state-space model using the metropolis-hastings algorithm within gibbs sampling. *Computational Statistics and Data Analysis*, 37(2):151–170, 2001.

[55] X. Zhu Z. Ghahramani and J. Lafferty. Time-sensitive dirichlet process mixture models. In *Technical Report CMU-CALD-05-104*, 2005.

[56] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In M. P. Atkinson, M. E. Orlowska, P. Valduriez, S. B. Zdonik, and M. L. Brodie, editors, *Proceedings of the 25th VLDB Conference*, pages 518–529, Edinburgh, Scotland, 1999. Morgan Kaufmann.

[57] Thomas L. Griffiths, Mark Steyvers, David M. Blei, and Joshua B. Tenenbaum. Integrating topics and syntax. In *NIPS*, 2004.

[58] T.L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101:5228–5235, 2004.

[59] J.E. Griffn and M.F.J. Steel. Order-based dependent dirichlet processes. *Journal of the American Statistical Association*, 101(473):1566–1581, 2006.

[60] A. Haghighi and L. Vanderwende. Exploring content models for multi-document summarization. In *HLT-NAACL*, 2009.

[61] Ahmed Hassan, Rosie Jones, and Kristina Lisa Klinkner. Beyond dcg: User behaviour as a predictor of a successful search. In *WSDM 2010*, pages 221–230, 2010.

[62] T. Haveliwala, A. Gionis, and P.Indyk. Scalable techniques for clustering the web. In *WebDB*, 2000.

[63] P. Hoff. Nonparametric modeling of hierarchically exchangeable data. In *UW Statistics Department Technical Report no. 421*, 2003.

[64] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42(1/2):177–196, 2001.

[65] Thomas Hofmann. Probabilistic latent semantic indexing. In *SIGIR 1999*, 1999.

[66] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *KDD*, pages 168–177. ACM, 2004.

[67] T. Iwata, T. Yamada, Y. Sakurai, and N. Ueda. Online multiscale dynamic topic models. In *KDD*, 2010.

[68] Sonia Jain and Radford M. Neal. A split-merge Markov chain Monte Carlo procedure for the Dirichlet process mixture model. *Journal of Computational and Graphical Statistics*, 13(1):158–182, 2004.

[69] V. Jain, E. Learned Miller, and A. McCallum. People-LDA: Anchoring topics to people using face recognition. In *ICCV*, pages 1–8, 2007.

[70] Jeon, J., Lavrenko, V., and Manmatha, R. Automatic image annotation and retrieval using cross-media relevance models. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 119–126, 2003.

[71] Arif E. Jinha. Article 50 million: an estimate of the number of scholarly articles in existence. *Learned Publishing*, 23(3):258–263, 2010.

[72] R. Kalman. A new approach to linear filtering and prediction problems. 82:35–45, 1960.

[73] R. E. Kalman. A new approach to linear filtering and prediction problems. *J. Basic Eng.*, 82:35–45, 1960.

[74] H. R. Kim and P. K. Chan. Learning implicit user interest hierarchy for context in personalization. In W. L. Johnson, E. André, and J. Domingue, editors, *Proceedings of the 2003 International Conference on Intelligent User Interfaces (IUI-03)*, pages 101–108, New York, 2003. ACM Press.

[75] Zhenzhen Kou, William W. Cohen, and Robert F. Murphy. High-recall protein entity recognition using a dictionary. In *ISMB (Supplement of Bioinformatics)*, pages 266–273, 2005.

[76] Ravi Kumar and Andrew Tomkins. A characterization of online search behavior. In *19th International World Wide Web Conference (WWW2010)*, pages 561–570, 2010.

[77] Simon Lacoste-Julien, Fei Sha, and Michael I. Jordan. DiscLDA: Discriminative learning for dimensionality reduction and classification. In *NIPS*, pages 897–904. MIT Press, 2008.

[78] F. Leitner and A. Valencia. A text-mining perspective on the requirements for electronically annotated abstracts. (582):1178–1181, 2008.

[79] L. Li, Z. Yang, B. Wang, and M. Kitsuregawa. Dynamic adaptation strategies for long-term and short-term user profile to personalize search. In G. Dong, X. Lin, W. Wang, Y. Yang, and J. Xu-Yu, editors, *Advances in Data and Web Management*, volume 4505 of *Lecture Notes in Computer Science*, pages 228–240. Springer, 2007.

[80] Lihong Li, Wei Chu, John Langford, and Robert Schapire. A contextual bandit approach to personalized news article recommendation. In *19th International World Wide Web Conference (WWW2010)*, pages 661–670, 2010.

[81] W. Li and A. McCallum. Pachinko allocation: Dag-structured mixture models of topic correlations. *ICML*, 2006.

[82] Wei Li and Andrew McCallum. Pachinko allocation: DAG-structured mixture models of topic correlations. In *ICML*, volume 148, pages 577–584. ACM, 2006.

[83] Wei Li, Xuerui Wang, and Andrew McCallum. A continuous-time model of topic co-occurrence trends. In *AAAI Workshop on Event Detection*, 2006.

[84] Wei-Hao Lin, Eric P. Xing, and Alexander G. Hauptmann. A joint topic and perspective model for ideological discourse. In *ECML/PKDD (2)*, pages 17–32, 2008.

[85] Xu Ling, Qiaozhu Mei, ChengXiang Zhai, and Bruce R. Schatz. Mining multi-faceted overviews of arbitrary topics in a text collection. In *KDD*, pages 497–505. ACM, 2008.

[86] S. MacEachern. Dependent dirichlet processes. *Technical report, Dept. of Statistics, Ohio State university*, 2000.

[87] Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. Topic sentiment mixture: modeling facets and opinions in weblogs. In *WWW*, pages 171–180. ACM, 2007.

[88] Meila. Comparing clusterings – an axiomatic view. In *ICML: Machine Learning: Proceedings of the Seventh International Conference, 1990*, 2005.

[89] J. Mellor-Crummey and M. L. Scott. Algorithms for scalable synchronization on shared-memory multiprocessors. *ACM Transactions on Computer Systems*, 9(1):21–65, February 1991.

[90] David M. Mimno, Wei Li, and Andrew McCallum. Mixtures of hierarchical topics with pachinko allocation. In *ICML*, volume 227, pages 633–640. ACM, 2007.

[91] David M. Mimno and Andrew McCallum. Topic models conditioned on arbitrary features with dirichlet-multinomial regression. In *UAI*, pages 411–418. AUAI Press, 2008.

[92] T. Minka. From hidden markov models to linear dynamical systems. *TR 53*, 1998.

[93] Robert F. Murphy, Meel Velliste, and Gregory Porreca. Robust numerical features for description and classification of subcellular location patterns in fluorescence microscope images. *VLSI Signal Processing*, 35(3):311–321, 2003.

[94] Robert F. Murphy, Meel Velliste, Jie Yao, and Gregory Porreca. Searching online journals for fluorescence microscope images depicting protein subcellular location patterns. In *BIBE*, pages 119–128, 2001.

[95] Ramesh Nallapati, Amr Ahmed, Eric P. Xing, and William W. Cohen. Joint latent topic models for text and citations. In *KDD*, pages 542–550. ACM, 2008.

[96] Tetsuya Nasukawa and Jeonghee Yi. Sentiment analysis: capturing favorability using natural language processing. In *K-CAP'03*, pages 70–77.

[97] R. M. Neal. Markov chain sampling methods for dirichlet process mixture models. *Technical Report 9815, University of Toronto, Department of Statistics and Department of Computer Science*, 1998.

[98] D. Newman, A. Asuncion, P. Smyth, and M. Welling. Distributed algorithms for topic models. *Journal of Machine Learning Research*, 10:1801–1828, 2009.

[99] David Newman, Chaitanya Chemudugunta, and Padhraic Smyth. Statistical entity-topic models. In *KDD*, pages 680–686. ACM, 2006.

[100] J. Y. Pan, H. J. Yang, C. Faloutsos, and P. Duygulu. GCap: Graph-based automatic image captioning. In *Multimedia Data and Document Engineering*, 2004.

[101] Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2007.

[102] Michael Paul and Roxana Girju. Cross-cultural analysis of blogs and forums with mixed-collection topic models. In *EMNLP*, pages 1408–1417. ACL, 2009.

[103] S. Petrovic, M. Osborne, and V. Lavrenko. Streaming first story detection with application to twitter. In *NAACL*, 2010.

[104] Jim Pitman. Exchangeable and partially exchangeable random partitions. *Probability Theory and Related Fields*, 102(2):145–158, 1995.

[105] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 275–281, 1998.

[106] Ana-Maria Popescu and Oren Etzioni. Extracting product features and opinions from reviews. In *HLT/EMNLP*. The Association for Computational Linguistics, 2005.

[107] J. Pritchard, M. Stephens, and P. Donnelly. Inference of population structure using multilocus genotype data. *Genetics*, 155:945–959, 2004.

[108] Foster J. Provost, Brian Dalessandro, Rod Hook, Xiaohan Zhang, and Alan Murray. Audience selection for on-line brand advertising: privacy-friendly social network targeting. In *KDD*, pages 707–716, 2009.

[109] Duangmanee Putthividhya, Hagai Thomas Attias, and Srikantan S. Nagarajan. Independent factor topic models. In *ICML*, volume 382, page 105. ACM, 2009.

[110] Ellen M. Riloff and Janyce Wiebe. Learning extraction patterns for subjective expressions. In *In Proceedings of CoNLL-2003*. Association for Computational Linguistics, 2003.

[111] Michal Rosen-Zvi, Chaitanya Chemudugunta, Thomas Griffiths, Padhraic Smyth, and Mark Steyvers. Learning author-topic models from text corpora. *ACM Transactions on Information Systems*, 28(1):401–420, January 2010.

[112] A.J. Smola and S. Narayanamurthy. An architecture for parallel topic models. In *Very Large Databases (VLDB)*, 2010.

[113] N. Srebro and S. Roweis. Time-varying topic models using dependent dirichlet processes. *Technical report, Department of Computer Science, University of Toronto*, 2005.

[114] Kazunari Sugiyama, Kenji Hatano, and Masatoshi Yoshikawa. Adaptive web search based on user profile constructed without any effort from users. In *WWW*, pages 675–684, 2004.

[115] H. Tanizaki. Nonlinear and non-gaussian state-space modeling with monte carlo techniques: A survey and comparative study. In *Handbook of Statistics Vol.21, Stochastic Processes: Modeling and Simulation, Chap.22, pp.871-929 (C.R. Rao and D.N. Shanbhag, Eds.)*, 2003.

[116] Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(576):1566–1581, 2006.

[117] Y.W. Teh, K. Kurihara, and M. Welling. Collapsed variational inference for hdp. In *NIPS*, 2007.

[118] Ivan Titov and Ryan T. McDonald. A joint model of text and aspect ratings for sentiment summarization. In *ACL*, pages 308–316. The Association for Computer Linguistics, 2008.

[119] Ivan Titov and Ryan T. McDonald. Modeling online reviews with multi-grain topic models. In *WWW*, pages 111–120. ACM, 2008.

[120] Peter D. Turney and Michael L. Littman. Measuring praise and criticism: Inference of semantic orientation from association. *ACM TOIS*, 4(21), 2003.

[121] H. Wallach. Structured topic models for language. Technical report, PhD. Cambridge, 2008.

[122] H. Wallach, D. Mimno, and A. McCallum. Rethinking lda: Why priors matter. In *In Advances in Neural Information Processing Systems 22*, 2009.

[123] Hanna M. Wallach. Topic modeling: beyond bag-of-words. In *ICML*, volume 148, pages 977–984. ACM, 2006.

[124] Hanna M. Wallach, Iain Murray, Ruslan Salakhutdinov, and David M. Mimno. Evaluation methods for topic models. In *ICML*, 2009.

[125] C. Wang, B. Thiesson, C. Meek, and D. Blei. Relational topic models for document networks. In *AISTATS*, 2009.

[126] Chong Wang, David M. Blei, and David Heckerman. Continuous time dynamic topic models. In *UAI*, pages 579–586. AUAI Press, 2008.

[127] X. Wang and A. McCallum. Topics over time: A non-markov continuous-time model of topical trends. In *KDD*, 2006.

[128] Xuerui Wang, Andrew McCallum, and Xing Wei. Topical N-grams: Phrase and topic discovery, with an application to information retrieval. In *ICDM*, pages 697–702. IEEE Computer Society, 2007.

[129] Y. Wang, H. Bai, M. Stanton, W. Chen, and E. Chang. Plda: Parallel latent dirichlet allocation for large-scale applications. In *In Proc. of 5th International Conference on Algorithmic Aspects in Information and Management*, 2009.

[130] Janyce Wiebe, Theresa Wilson, Rebecca Bruce, Matthew Bell, and Melanie Martin. Learning subjective language. *Computational Linguistics*, 30(3):277–308, 2004.

[131] Jun Yang, Yan Liu, Eric P. Xing, and Alexander G. Hauptmann. Harmonium models for semantic video representation and classification. In *SDM*. SIAM, 2007.

[132] Tae Yano, William W. Cohen, and Noah A. Smith. Predicting response to political blog posts with topic models. In *HLT-NAACL*, pages 477–485. The Association for Computational Linguistics, 2009.

[133] L. Yao, D. Mimno, and A. McCallum. Efficient methods for topic model inference on streaming document collections. In *KDD'09*, 2009.

[134] H. Yu and V. Hatzivassiloglou. Thumbs up? sentiment classification using machine learning techniques. In *EMNLP*, 2002.

[135] H. Yu and V. Hatzivassiloglou. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *EMNLP*, 2003.

[136] K. Yu, S. Yu, , and V. Tresp. Dirichlet enhanced latent semantic analysis. In *AISTATS*, 2005.

[137] Jian Zhang, Yiming Yang, and Zoubin Ghahramani. A probabilistic model for online document clustering with application to novelty detection. In *Neural Information Processing Systems*, 2004.

[138] Y. Zhou, L. Nie, O. Rouhani-Kalleh, F. Vasile, and S. Gaffney. Resolving surface forms to wikipedia topics. In *Proceedings of the 23rd International Conference on Computational Linguistics COLING*, pages 1335–1343, August 2010.

[139] Jun Zhu, Amr Ahmed, and Eric P. Xing. MedLDA: maximum margin supervised topic models for regression and classification. In *ICML*, volume 382, 2009.