Collapsed Variational Inference for Computational Linguistics

Pengyu Wang

St Cross College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Michaelmas 2016

Acknowledgements

First of all, I would like to thank my supervisor, Professor Phil Blunsom for his constant guidance and advice through my studies. I am profoundly grateful for two things. First, Phil accepted my transfer from the category theory group and patiently taught me the knowledge from basic probability theory to Bayesian nonparametrics during the early days when I was new to either computational linguistics or machine learning. Second, Phil provided much research freedom after I obtained required knowledge to conduct research, and offered valuable ideas and insights on the research topics I (seemingly randomly) chose to pursue. I would not have been able to put this thesis together without his help.

I want to thank my former supervisors Professor Samson Abramsky and Professor Bob Coecke, and my departmental advisor Professor Luke Ong, for their agreement with my bold movement to a new research territory. I also thank the computer science department for providing travel funds to conferences.

I would like to thank all my former and current colleagues in the computational linguistics group for many interesting academic discussions; and my friends and various flatmates, for many fun diversions from the rigour of research. In particular, I want to thank an old friend Yi Ding, who although in Stanford, kept me company all the time via Skype; I am grateful for the unfailing friendship.

Finally, I give my most special thanks to my parents and sister for their endless support of my endeavours and their love that gives my strength to finish.

Abstract

Bayesian modelling is a natural fit for tasks in computational linguistics, since it can provide interpretable structures, useful prior controls, and coherent management of uncertainty. However, exact Bayesian inference is intractable for many models of practical interest. Developing both accurate and efficient approximate Bayesian inference algorithms remains a fundamental challenge, especially for the field of computational linguistics where datasets are large and growing and models consist of complex latent structures.

Collapsed variational inference (CVI) is an important milestone that combines the efficiency of variational inference (VI) and the accuracy of Markov chain Monte Carlo (MCMC) (Teh et al., 2006). However, its previous applications were limited to bag-of-words models whose hidden variables are conditionally independent given the parameters, whereas in computational linguistics, the hidden variable dependencies are crucial for modelling the underlying syntactic and semantic relations. To enlarge the application domain of CVI as well as to address the above Bayesian inference challenge, we investigate the applications of collapsed variational inference to computational linguistics.

In this thesis, our contributions are three-fold. First, we solve a number of inference challenges arising from the hidden variable dependencies and derive a set of new CVI algorithms for the two ubiquitous and foundational models in computational linguistics, namely hidden Markov models (HMMs) and probabilistic context free grammars. We also propose CVI for hierarchical Dirichlet process (HDP) HMMs that are Bayesian nonparametric extensions of HMMs.

Second, along the way we propose a set of novel algorithmic techniques, which are generally applicable to a wide variety of probabilistic graphical models in the conjugate exponential family and computational linguistic models using non-conjugate HDP constructions. Therefore, our work represents one step in bridging the gap between increasingly richer Bayesian models in computational linguistics and recent advances in approximate Bayesian inference.

Third, we empirically evaluate our proposed CVI algorithms and their stochastic versions in a range of computational linguistic tasks, such as part-of-speech induction, grammar induction and many others. Experimental results consistently demonstrate that, using our techniques for handling the hidden variable dependencies, the empirical advantages of both VI and MCMC can be combined in a much larger domain of CVI applications.

Contents

List of Figures

Chapter 1

Introduction

Chapter Abstract

In this chapter, we motivate this thesis by discussing the benefits of Bayesian modelling and current limitations of Bayesian inference in the field of computational linguistics. Then we state our contributions and thesis structure.

1.1 Bayesian Modelling

The field of computational linguistics aims to understand and manipulate natural language with computational means. Despite its early history that begins with rules-based systems in 1950s, statistical approaches to computational linguistics have been in ascendancy over the past three decades (Jurafsky and Martin, 2000). This profound shift is not only credited to the availability of large corpora and machine learning techniques, but also driven by the phenomena of natural language. For example, language is riddled with ambiguity, and it is not surprising that statistical approaches based on probability theory (extension of formal theory of logic) have played an important role in modelling this ambiguity.

Statistical approaches to computational linguistics have evolved in several ways. One significant evolution taking place in the early 2000s was the use of fully fledged Bayesian models. Prior to it, most work in statistical computational linguistics had been frequentist, arguably due to the low computational cost as usually thought. Bayesian modelling starts with a prior belief that can represent domain knowledge, in the light of observed data, it is updated to reflect the posterior belief over the model and parameters. Putting aside the

cost of Bayesian inference, Bayesian modelling accommodates for various shortcomings in frequentist modelling and provides a principled framework for encoding prior beliefs, managing uncertainty, sharing statistical strength and comparing models, all of which are essential to capturing the natural language phenomena. In the following, we elaborate on each of these beneficial functionalities in more specific terms in computational linguistics.

1. The ability to incorporate prior knowledge. Building assumptions of linguistic knowledge into models is helpful for improving model performances, sometimes dramatically. For example, given a model with discrete distributions, if we seek to build sparsity into the model, we often set its conjugate Dirichlet priors with hyperparameters less than 1. Sparsity is typical of natural language and applies to many linguistic units including phonemes, words, phrases and others.

2. The ability to integrate over all possible parameter values. Bayesian modelling treats parameters as random variables and provides a coherent approach for managing uncertainty, avoiding the issue of overfitting. In addition, for natural language data that is often too complex to be explained by a single set of parameters, using an ensemble of possible parameters is helpful for making better predictions, especially in the unsupervised setting, where learning is done without labelled data.

3. The ability to share statistical strength. Despite the availability of large corpora, sparse data problems abound in computational linguistics. Statistical sharing, otherwise known as smoothing is used to partially mitigate the curse of dimensionality in sparse data. Hierarchical Bayesian modelling allows the development of sophisticated smoothing strategies that are both effective and justifiable. For example, the effective and widely used KN smoothing strategy (Kneser and Ney, 1995) was later mathematically justified by the hierarchical Pitman-Yor process (Teh, 2006).

4. The ability to select between models. To use a computational linguistic model in a particular dataset, it is often hard to decide the appropriate model size using domain knowledge. Bayesian modelling provides a coherent approach for model selections, by estimating the marginal data likelihoods that automatically encode Ocaam's Razor (Jefferys and Berger, 1992) to penalize over-complex models. Bayesian nonparametrics (BNP) is a more efficient modern approach. For a finite amount of observed data, BNP modelling offers an infinite capacity, only a finite portion of which is used to model the data so that the model size is adjusted adaptively with the data size.

1.2 Bayesian Inference

With the benefits summarized in the previous section, in the past decade there has been an explosion of hierarchical Bayesian models and applications, ranging from topic modelling (Blei et al., 2003), language modelling (Teh, 2006), word segmentation (Goldwater et al., 2009), part-of-speech induction (Blunsom and Cohn, 2011) to grammar induction (Liang et al., 2007). Although Bayesian modelling is well suited for computational linguistic tasks, Bayesian inference is a fundamental challenge. For most models of practical interests, the posterior distributions are intractable. In such situations, we need to approximate the true posteriors by using some approximation techniques.

Variational Inference (VI) and Markov Chain Monte Carlo (MCMC) are two dominant approaches for approximate Bayesian inference. VI casts an inference problem as a constraint optimization problem and uses the solution as a proxy for the true posterior. The common choice of constraints for the models in computational linguistics is assuming the independence between the tightly coupled parameters and hidden variables. This choice allows to develop deterministic and efficient iterative procedures, but it can often lead to poor approximations. On the other hand, MCMC approximates the true posterior by forming an empirical estimate from samples. Although it reaches the exact solution in the limit, in practice MCMC sampling is notoriously slow to converge, especially with distributions over complex structures. Therefore, it remains a challenge to develop algorithms that are both accurate and efficient, in particularly for the field of computational linguistics where datasets are large and growing and models contain complex latent structures.

There has been a constant passion in the Bayesian community for proposing new inference algorithms that are more accurate or more scalable or both than existing ones. Collapsed Variational Inference (CVI), which combines the advantages of both VI and MCMC, is one of the important milestones (Teh et al., 2007). The essential idea of CVI is applying the mean field method on hidden variables after integrating out model parameters such that their tight couplings are preserved. Substantial empirical evidence in topic modelling (Teh et al., 2007; Asuncion et al., 2009) and theoretical justifications (Sato and Nakagawa, 2012) show that CVI with a first order Taylor approximation is significantly more accurate than VI, often comparable with MCMC, and as computationally efficient as VI.

The above advantages suggest that CVI can be a promising venue for developing both accurate and efficient algorithms for models in computational linguistics. However, the

previous success of CVI in topic modelling and other bag-of-words settings (e.g., Kurihara et al. (2007)) relies on the perfect match between the mean field method and the bag-of-words condition. Here we define a model to be i.i.d. (or bag-of-words) if its any two hidden variables are conditionally independent given the model parameters. Given an i.i.d. model, marginalizing out the model parameters induces dependencies that spread over many hidden variables, especially in large datasets; thus the dependency between any two hidden variables is very weak. Therefore, it is reasonable to break the weak dependencies in the collapsed space and to apply the mean field method. We call the induced dependencies *weak* and the original dependencies *strong*, and they are formally defined in Section 2.2.

For the majority of models in the field of computational linguistics, the bag-of-words assumption does not hold. On the contrary, hidden variable dependencies play a significant role in modelling the underlying syntactic or semantic dependencies. Hidden Markov Models (HMMs) with sequential dependencies and Probabilistic Context Free Grammars (PCFGs) with structural dependencies are the two foundational models for learning natural language. For these models, there exist strong dependencies among hidden variables in the original generative process; and after the integration of the model parameters, these strong dependencies will be inherited in the collapsed space. In such contexts, the complete factorization assumption required by the mean field method is not an ideal choice.

As well as developing accurate and efficient CVI algorithms for the two foundational models in computational linguistics, the previous application domain of CVI should be expanded beyond the simple bag-of-words setting. In particular, for probabilistic graphical models that are traditionally learnt by the Expectation Maximization (EM) algorithm (Dempster et al., 1977), with some mild conditions on distribution conjugacy, Beal (2003) proposed an important connection between EM and VI. This connection provides an elegant algorithmic shortcut from EM to VI and has contributed to the widespread applications of VI in graphical models. For CVI, it remains an open question whether and how it can be made generally applicable to probabilistic graphical models and even beyond, so that the advantages of both VI and MCMC can be combined in a much larger application domain.

1.3 Thesis Contributions

In this thesis, we propose a set of new collapsed variational inference algorithms for HMMs, PCFGs and the Bayesian nonparametric extensions of HMMs based on the Hierarchical

Dirichlet Process (HDP). In a number of computational linguistic tasks, including part-of-speech induction, word alignment, text prediction and grammar induction as well as in audio sequence analysis and synthetic datasets, we empirically validate that our proposed CVI algorithms are more accurate (often significantly) than the corresponding EM and VI algorithms with similar training time. Compared with the corresponding MCMC sampling algorithms, our CVI algorithms achieve comparable performances while using significantly less training time (sometimes by an order of magnitude). Along with developing CVI algorithms for the concrete models, we propose a set of novel algorithmic techniques that allow CVI to be applicable to a wide variety of computational linguistic models in the Conjugate Exponential (CE) family as well as those using non-conjugate HDP constructions. In the following, we highlight the specific research contributions in each of the core chapters.

In Chapter 3, we investigate the applications of CVI to HMMs that exhibit sequential dependencies and are ubiquitous in modelling time series data in computational linguistics and many other fields. Given a chain structure in collapsed space where a hidden variable strongly depends on both the previous and next variables, and weakly depends on others, we incrementally derive three CVI algorithms, by exploring different ways of factorizing the collapsed variational distributions. Each algorithm relaxes some of the assumptions or application conditions required by the previous one. Along the way several novel technical tricks are proposed to solve the inference challenges arising from the various factorizations.

1. In the first algorithm, we apply the mean field method in the same way as for i.i.d. models. We propose a new calculation method to address the challenge of calculating the number of state transitions, which arises from breaking the strong dependencies.

2. In the second algorithm, we assume the setting of multiple sequences and apply the structured mean field method to update posteriors at the sequence level. We propose and empirically validate a new approximation method to joint posterior predictive distributions in the CE family, to address the challenge that the parameters changing on the fly inside a sentence prohibits efficient dynamic programming.

3. In the third algorithm, we relax the previous condition and factorize a single long sequence into subsequences. As well as meeting the above two challenges, we propose a novel propagation redesign method to address the third challenge that the standard forward-backward algorithm can not be used to infer subsequence posteriors. This

leads to a new sum-product algorithm that allows for message passing across subsequences, effectively mitigating most factorizing loss at the subsequence boundaries. It is the first message passing procedure in collapsed space, whereas previously such a sophisticated procedure was only possible when parameters were provided.

4. We make contributions to collapsed variational inference in the CE family, extending the previous work of Sung et al. (2008). Specifically, we extend the CVI's capability of updating multiple hidden variables and updating sufficient statistics stochastically, yielding Stochastic Collapsed Variational Inference (SCVI) for the CE family, generalizing the SCVI algorithm for latent Dirichlet allocation (Foulds et al., 2013).

5. We discuss the general usage rules of collapsed variational inference for probabilistic graphical models. As with the algorithmic shortcut from EM to VI proposed by Beal (2003), we propose an analogous algorithmic shortcut from EM to CVI, following a systematic approach of reusing/redesigning propagation algorithms of EM. It allows CVI to be generally applicable to a wide variety of graphical models in the CE family.

6. We empirically evaluate our proposed CVI (or SCVI) algorithms in several computational linguistic tasks, settings and evaluation metrics, to confirm that indeed they combine the advantages of both VI (or SVI) and collapsed Gibbs sampling.

The above contributions are partially covered by the papers originally published in the proceedings of AISTATS (Wang and Blunsom, 2013a) and at the NIPS workshop on Time Series Analysis (Wang and Blunsom, 2015b).

In Chapter 4, we extend CVI for HMMs along the Chomsky hierarchy to probabilistic context free grammars that are widely used for representing syntactic structures in computational linguistics, with the following specific contributions.

1. We follow the second CVI algorithm for HMMs and derive CVI for PCFGs. A notable difference between HMMs and PCFGs is that while HMMs can be represented by graphical models, PCFGs can not, since the models structures (i.e., parse trees) are random objects. We propose to define an ordering in grammar rules to simplify the joint posterior predictive distribution before the previously proposed approximation method can be applied. This technique demonstrates that CVI can be applied to statistical models with random model structures.

2. We empirically evaluate our proposed CVI algorithm in three different grammar induction tasks, with standard PCFGs and their two variants, confirming that indeed our algorithm combines the advantages of both VI and Metropolis Hastings sampling.

The above contributions are partially covered by the paper originally published in the proceedings of CONLL (Wang and Blunsom, 2013b).

In Chapter 5, we extend CVI for HMMs to their Bayesian nonparametric counterparts, namely HDP-HMMs (Teh et al., 2006). In practical applications of time series, especially in computational linguistics, it is often hard to decide the number of states a priori using domain knowledge. HDP-HMMs can determine the number of states automatically as well as improve the performances of HMMs by the statistical sharing mechanism. However, the hierarchical structures of HDP, coupled with the sequential dependencies of HMMs, pose significant inference challenges. Specifically, we make the following contributions.

1. We derive a CVI algorithm for HDP-HMMs. For the HMM part, we have presented three algorithms based on different ways of factorizing hidden sequences in Chapter 3. For the HDP part, we make use of the existing algorithmic techniques as in the work of CVI for HDP-LDA by Teh et al. (2008), but propose a slightly different approach in our derivations: instead of taking the normally used functional derivatives that give batch CVI updates only, we take the Euclidean gradients to yield both batch and stochastic CVI updates at the same time.

2. Central to our algorithm is a set of novel algorithmic techniques for computing the probabilities of each state transition's existence, which form important ingredients of the expected table counts (in the metaphor of Chinese Restaurant Franchise). Such probabilities are trivial to compute in HDP-LDA but much harder for HDP-HMMs due to the sequential dependencies. We propose an exact message passing algorithm with high computational complexity, and several approximate techniques, which are empirically evaluated against the exact one. Our techniques are generally applicable to many HDP-based computational linguistic models such as HDP-PCFGs.

3. We empirically evaluate our proposed CVI (or SCVI) algorithm, to show that it can reliably determine the effective number of states in synthetic datasets, it outperforms other benchmarks for HDP-HMMs (two VI algorithms and beam sampling) in large

scale audio sequence analysis and it also improves upon CVI for finite HMMs, providing a new scalable and strong benchmark for time series analysis.

In the above list, the first contribution is partially covered by the paper originally published at the NIPS workshop on Advances in Approximate Bayesian Inference (Wang and Blunsom, 2015a). The second and third contributions are new pieces of research.

Overall this thesis is structured as follows. In Chapter 2, we briefly introduce the background material including probabilistic graphical models, approximate Bayesian inference, the conjugate exponential family and the Dirichlet process. In particular, we argue what we mean by the 'strong' and 'weak' dependencies in posterior distributions using the representation of Markov Random Fields. The core Chapters 3-5 directly build on the background material and employ a number of intermediate results in Chapter 2 such as the conjugacy properties, the propagation rules of the sum-product algorithm and others.

In each of the core Chapters 3-5, our presentation follows the same pattern: we start with a gentle introduction to the background specific to the model, define the model, review certain existing approximate inference algorithms (if they are relevant to our CVI derivations), present our research contributions as we have enumerated above, and evaluate our proposed CVI algorithms. Finally, Chapter 6 concludes this thesis with future work.

Chapter 2

Background

Chapter Abstract

In this chapter, we briefly review the background on which this thesis builds. The background material is organized into four subjects, probabilistic graphical models, approximate Bayesian inference algorithms, the exponential family and conjugacy, and the Dirichlet process. In particular, our representation of Bayesian posterior distributions using the notion of Markov random fields provides a new perspective, which allows us to qualitatively define what we mean by strong and weak dependencies through this thesis.

2.1 Probabilistic Graphical Models

In this section, we first overview the key concepts for directed, undirected and factor graphical models, which we use for defining and explaining models throughout this thesis. For a more thorough treatment, see Bishop (2006, Chapter 8) and Murphy (2012, Chapter 10, Chapter 19). Next, we review the sum product inference algorithm for factor graphs, which we use for designing our own inference procedures.

Starting from a graph comprising a set of nodes that are connected by edges, a probabilistic graphical model associates each node with a random variable and each edge with a possible probabilistic relationship between the nodes. Here and in the rest of the thesis, we denote the set of all variables by $\mathbf{y} = \{\mathbf{x}, \mathbf{z}\}$, where $\mathbf{x}$ is the set of all observed variables and $\mathbf{z}$ is the set of all hidden ones. Alternatively, we can index the variables by the subscript

$t \in \mathcal{T}$, where $\mathcal{T}$ denotes a set of indices. In Section 2.3, we will introduce a superscript i, which represents which data point out of a size I dataset is being referred to. For now, we consider a single data point and omit the superscript. Also we make no formal distinction between the terms 'node', 'variable' and 'variable node' and use them interchangeably.

The edges in a graphical model can be directed or undirected. The directed edges are intuitive for representing casual relationships, while the undirected one are often used for expressing degrees of interaction, agreement or constraints between variables. Nevertheless, it is the absence of some edges that convey useful information: it allows a graphical model to define a family of joint distributions over all the variables, and each joint distribution can be decomposed into a product of factors that only depend on subsets of variables. By the decompositions, one can exploit the conditional independence properties of a graphical model and design efficient inference algorithms.

Bayesian networks (BNs), also called belief networks or probabilistic directed acyclic graphs (DAGs) are graphical models in which all the edges are directed and there are no cycles following any directed edges. We use Bayesian networks for defining hierarchical Bayesian models in this thesis. A Bayesian network expresses a joint distribution over T variables, $\mathbf{y} = \{y_t\}_{t=1}^{T}$ to be a product of T conditional distributions,

$$p(\mathbf{y}|\theta) = \prod_{t=1}^{T} p(y_t|\mathbf{y}_{\text{pa}(t)}, \theta), \tag{2.1}$$

where $\mathbf{y}_{\text{pa}(t)}$ denotes the set of parents of y_t. A node a is called a parent of another node b and conversely b is a child of a, if there is an edge from a to b. In this section, we assume that the parameters of a model, denoted by θ are deterministic and given. In order to keep the notations uncluttered, we omit the implicit dependences on θ.

Markov random fields (MRFs) are another kind of graphical models in which all the edges are undirected. We will use Markov random fields for describing posterior distribution structures in Bayesian inference. A Markov random field expresses a joint distribution over variables $\mathbf{y}$ to be a product of potential functions over its J maximal cliques,

$$p(\mathbf{y}) = \frac{1}{Z} \prod_{j=1}^{J} f_j(D_j(\mathbf{y})), \tag{2.2}$$

where $D_j(\mathbf{y})$ denotes a maximal clique, and f_j denotes the potential function that returns a non-negative value for each setting of variables in $D_j(\mathbf{y})$. A subgraph is said to be a

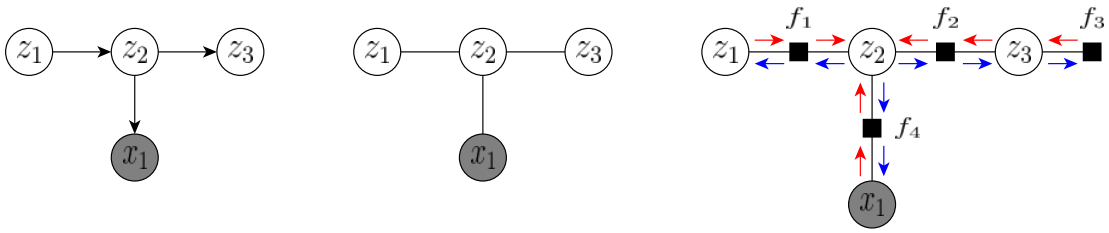

Figure 2.1: Examples of probabilistic graphical models. Left: a Bayesian network. Middle: a Markov random field. Right: a factor graph. We use circles to denote variable nodes with observed ones being shaded, and use small squares to denote factor nodes. In the example of factor graph, we use arrows in two colors to denote messages sent in opposite directions.

clique if it is fully connected (there is an edge between every pair of nodes), and a clique is said to be maximal if the inclusion of any other nodes ceases its full connectivity property. Unlike BNs, the product of potential functions in a MRF can be unnormalized. In (2.2), the normalization constant, denoted by Z ensures that the right hand side sums to one.

We can see that both BNs and MRFs decompose a joint distribution into a product of factors over subsets of variables. Factor graphs make the decompositions more explicit by using additional factor nodes for the factors themselves, as well as variable nodes. Therefore, factor graphs are bipartite graphical models in which there are two types of nodes, all the edge are undirected and they connect nodes of opposite types. A factor graph expresses a joint distribution over variables $\mathbf{y}$ to be a product of J factors,

$$p(\mathbf{y}) = \frac{1}{Z} \prod_{j=1}^{J} f_j(D_j(\mathbf{y})), \tag{2.3}$$

where $D_j(\mathbf{y})$ denotes a subset of $\mathbf{y}$ and the factor f_j is a non-negative function over the subset $D_j(\mathbf{y})$. Since $D_j(\mathbf{y})$ does not have to be a clique as required by MRFs, factor graphs are more flexible about factorizations, and thus subsume MRFs. For example, the normalization constant Z can be considered as a factor defined on the empty set.

Example Figure 2.1 shows three graphical models, one for each type we have defined. The left graph shows a simple Bayesian network, which is a segment of hidden Markov models we study in Chapter 3. It is possible to convert BNs to graphical models of other types. For example, the middle graph shows a Markov random field that is converted from the BN example by dropping the arrows of all edges and equating the potential functions with the corresponding conditional distributions. The general method for converting BNs to MRFs via moralization can be found in Bishop (2006, Chapter 8).

In the rest of this section, we focus on factor graphs, since they are more flexible and specific about exact forms of factorizations and we use such representations for deriving our algorithms. The right graph of Figure 2.1 shows a factor graph, which is also converted from the BN example by creating two factor nodes f_2 and f_3 for the conditional distribution $p(z_3|z_2)$, as well as creating f_1 for $p(z_2|z_1)$ and f_4 for $p(x_1|z_2)$. In particular, the factor node f_3 is useful when there is probabilistic information about z_3 only, which is the case we will see in Section 3.6, when we break a long HMM sequence into subsequences. By the structures of this factor graph, the joint distribution factorizes as follows,

$$p(x_1, z_1, z_2, z_3) = \frac{1}{Z} f_1(z_1, z_2) f_2(z_2, z_3) f_3(z_3) f_4(x_1, z_2). \tag{2.4}$$

Graphical models allow us to visualize conditional independence relationships between variables, which are important for designing efficient inference algorithms without enumerating all settings of all variables. Two sets of variables A and B are conditionally independent given a third set of variables C, denoted by $A \perp\!\!\!\perp B \,|\, C$, if $p(A|B, C) = p(A|C)$ or equivalently, $p(A, B|C) = p(A|C)p(B|C)$. In our factor graph example shown in Figure 2.1 (right), we can by hand verify that $p(z_3|z_1, z_2) = p(z_3|z_2)$, and thus $z_3 \perp\!\!\!\perp z_1 \,|\, z_2$. Alternatively, we can perform a simple graphical inspection: for factor graphs, $A \perp\!\!\!\perp B \,|\, C$ if all paths connecting any variable in A and any variable in B pass through at least one variable node in C. For analogous arguments for BNs and MRFs, see Bishop (2006, Chapter 8).

2.1.1 Exact Inference

The problem of inference in graphical models is computing marginals over one or more variables, optionally conditioned on observations. In this subsection, we review exact inference for factor graphs, namely the sum product algorithm (Wiberg, 1996; Kschischang et al., 2001). The exact inference for BNs, namely the belief propagation algorithm (Pearl, 1988) and the one for MRFs, namely the junction tree algorithm (Lauritzen and Spiegelhalter, 1988; Jordan, 1999) are presented in details in Jensen (1996).

Given a factor graph, the sum product algorithm starts with passing messages around the underlying graph, and then queries about arbitrary marginals can be easily computed using these messages. We denote the neighbours of a variable y_t, a set of factors, by $ne(y_t)$, and similarly we denote the neighbours of a factor f_j, a set of variables, by $ne(f_j)$. We use $\setminus$ to denote the exclusion. For example, $ne(f_j) \setminus y_t$ denotes the set of all neighbours of f_j excluding y_t. The message passing scheme is summarized by the following two rules:

1. The message sent from a factor node f_j to a variable node y_t is,

$$m_{f_j \to y_t}(y_t) = \sum_{ne(f_j) \setminus y_t} f_j(ne(f_j)) \prod_{y_{t'} \in ne(f_j) \setminus y_t} m_{y_{t'} \to f_j}(y_{t'}). \tag{2.5}$$

2. The message sent from a variable node y_t to a factor node f_j is,

$$m_{y_t \to f_j}(y_t) = \prod_{f_{j'} \in ne(y_t) \setminus f_j} m_{f_{j'} \to y_t}(y_t). \tag{2.6}$$

As we can see, a message sent from a node to its neighbour requires the node to receive messages from all of its other neighbours first. For example, the first rule in (2.5) states that the message from a factor f_j to a variable y_t is by multiplying the factor f_j by the product of incoming messages from all its neighbours excluding y_t, and summing over all the neighbours excluding y_t. Thus, a message passing ordering needs to be defined. For singly connected factor graphs (i.e., trees) that we deal with in this thesis, we can randomly define a node in a graph to be the root. Then passing messages away from the root (or in reverse order) guarantees that each node will receive enough messages to send out one.

After we pass messages from the leaves to the root of a factor graph as well as from the root to the leaves using the above two rules, the marginal over an arbitrary variable is,

$$p(y_t) \propto \prod_{f_j \in ne(y_t)} m_{f_j \to y_t}(y_t), \tag{2.7}$$

and the marginal over an arbitrary subset of variables associated with the factor f_j is,

$$p(D_j(\mathbf{y})) \propto f_j(D_j(\mathbf{y})) \prod_{y_t \in ne(f_j)} m_{y_t \to f_j}(y_t), \tag{2.8}$$

where their normalization constants can be found by summing over any right hand side.

We have implicitly assumed that all the variables are hidden in the above presentation of the sum product algorithm. Sometimes we wish to compute marginals conditioned on some observations. In this case, we can eliminate the observed variables by absorbing their observed and constant values into the corresponding factors. In other words, we work with the joint distribution $p(\mathbf{z}, \mathbf{x} = \hat{\mathbf{x}})$, where $\hat{\mathbf{x}}$ denotes the observed values. By running the sum product algorithm, we can obtain the required conditionals, for example, $p(z_t|\mathbf{x})$, up to a normalization constant, whose value again can be found by a local summation.

Example In the factor graph example shown in Figure 2.2 (right), we pick the node z_2 to be its root. Then we perform message passing in both directions as indicated by the red and blue arrows, respectively. Now we are ready to efficiently obtain any marginals conditioned on the observation $x_1 = \hat{x}_1$, where $\hat{x}_1$ is the observed value. For example,

$$p(z_2|x_1 = \hat{x}_1) = \frac{p(x_1 = \hat{x}_1, z_2)}{\sum_{z_2} p(x_1 = \hat{x}_1, z_2)}, \tag{2.9}$$

in which

$$p(x_1 = \hat{x}_1, z_2) \propto m_{f_1 \to z_2}(z_2) m_{f_2 \to z_2}(z_2) m_{f_4 \to z_4}(z_2). \tag{2.10}$$

2.2 Approximate Bayesian Inference

In the previous section, we have considered model parameters to be a set of fixed values. In this section, we treat model parameters as random variables, and review some approximate Bayesian inference techniques on which our algorithms will build.

Given a Bayesian model, we denote the set of all model parameters by θ; and as before, we denote the sets of all observed and hidden variables by $\mathbf{x}$ and $\mathbf{z}$, respectively. Since θ are unknown and random, we place a prior distribution over θ, denoted by $p(\theta)$ to reflect our prior belief about the parameters' likelihoods. In this setting, the problem of Bayesian inference is computing the posterior distribution over both $\mathbf{z}$ and θ given $\mathbf{x}$,

$$p(\mathbf{z}, \theta|\mathbf{x}) = \frac{p(\mathbf{x}, \mathbf{z}|\theta)p(\theta)}{p(\mathbf{x})}, \tag{2.11}$$

in which the denominator, known as the marginal likelihood is,

$$p(\mathbf{x}) = \int p(\mathbf{x}, \mathbf{z}|\theta)p(\theta)d\mathbf{z}d\theta. \tag{2.12}$$

In (2.11), we can see that Bayes' rule provides a mean of updating the posterior distribution in light of observed data. Then the posterior distribution can be used to reveal hidden structures and to make predictions. Unlike the maximum likelihood estimation, the Bayesian approach does not try to fit parameters to observations. Therefore, such approach does not suffer from overfitting. In (2.12), the marginal likelihood has been shown to automatically encode Occam's razor, penalizing over-complex models (Jefferys and Berger, 1992). Thus, this quantity is often used to select between models.

The central challenge of Bayesian posterior inference is the integration in (2.12). For most models in practice, these integrals are not only in high dimensional space but also analytically intractable. In such situations, we need to approximate the exact answers by using some approximation techniques; and these range from the Laplace approximation, variational inference, loopy belief propagation to a wide variety of Monte Carlo methods. In this section, we review Markov Chain Monte Carlo, variational inference, and collapsed variational inference, since they are closely related to our later developments.

It might be worth pausing to clarify a terminology. In a Bayesian model, if the parameters and hidden variables are viewed on an equal footing, and their union is denoted by $\mathbf{h}$, then the Bayesian inference problem in (2.11) is in fact an inference problem, $p(\mathbf{h}|\mathbf{x})$, as we have seen in the previous section. For this reason, we choose to use the shorter terminology 'inference' instead of 'Bayesian inference' in this thesis. Nevertheless, we still employ two distinct sets of notations for parameters and hidden variables to emphasise the different roles they play in Bayesian models: models parameters are global variables, and they govern the generation of local hidden and observed variables.

2.2.1 Variational Inference

Variational inference (VI) (Jordan, 1999; Wainwright and Jordan, 2008) casts an inference problem as the solution to a constraint optimization problem. Then the solution is used as an approximation to the true posterior. Specifically, in the VI framework, we first define a variational distribution $q(\mathbf{h})$ that is restricted in a tractable family, denoted by $\mathcal{Q}$. Recall that $\mathbf{h}$ includes both the hidden variables $\mathbf{z}$ and model parameters θ. Next, we seek to find the optimal member in $\mathcal{Q}$, which has the minimum Kullback-Leibler (KL) distance to the true posterior, by solving the following constraint optimization problem,

$$q^\star(\mathbf{h}) = \underset{q(\mathbf{h})\in\mathcal{Q}}{\operatorname{argmin}} \operatorname{KL}(q(\mathbf{h})||p(\mathbf{h}|\mathbf{x})), \tag{2.13}$$

where a KL distance from one distribution to another is defined as follows,

$$\operatorname{KL}(q(x)||p(x)) = \mathbb{E}_q[\log q(x) - \log p(x)]. \tag{2.14}$$

Since the marginal likelihood in (2.12) is constant, the above minimization problem is equivalent to the following maximization problem,

$$q^\star(\mathbf{h}) = \underset{q(\mathbf{h})\in\mathcal{Q}}{\operatorname{argmax}} \mathcal{L}^{\mathrm{VI}}(q), \tag{2.15}$$

where $\mathcal{L}^{\mathrm{VI}}(q)$, known as the model evidence lower bound (ELBO) is defined by,

$$\begin{aligned}\mathcal{L}^{\mathrm{VI}}(q) &\triangleq \mathbb{E}_q[\log p(\mathbf{x}, \mathbf{h})] - \mathbb{E}_q[\log q(\mathbf{h})] \\ &= \log p(\mathbf{x}) - \mathrm{KL}(q(\mathbf{h})||p(\mathbf{h}|\mathbf{x})) \\ &\leq \log p(\mathbf{x}). \end{aligned} \tag{2.16}$$

Broadly speaking, there are many ways to restrict $q(\mathbf{h})$, for example by assuming its parametric form. The most commonly used methodology is partitioning $\mathbf{h}$ into subsets of variables and assuming that $q(\mathbf{h})$ factorizes with respect to these subsets. If the partition is simply a collection of singletons (i.e., complete factorization), the resulting VI algorithm is often called mean field; if the partition is a collection of subsets of variables, the resulting VI algorithm is often called structured mean field[1]. In choosing partitions, a rule of thumb is to preserve as many important dependencies in true posteriors as possible, as long as the efficiency of the resulting VI algorithm is at an acceptable level.

Now we review a particular VI algorithm, which is also known as variational (Bayesian) expectation maximization (VBEM) (Beal, 2003), and serves as a benchmark in comparison with our collapsed VI algorithms. In VBEM, the variational distribution $q(\mathbf{h})$ is factorized by assuming independence between parameters and hidden variables,

$$q(\mathbf{h}) \approx q(\mathbf{z})q(\theta). \tag{2.17}$$

To maximize $\mathcal{L}^{\mathrm{VI}}(q)$, a coordinate ascent method can be applied, i.e., updating each q factor while fixing others. By taking functional derivatives of $\mathcal{L}^{\mathrm{VI}}(q)$ and setting them to zeros (for an introduction to the calculus of variation, see Bishop (2006, Appendix D)), the update equations of VBEM is given by,

$$\text{VBE step: } q(\mathbf{z}) \propto \exp\{\mathbb{E}_{q(\theta)}[\log p(\mathbf{x}, \mathbf{z}, \theta)]\}, \tag{2.18}$$

$$\text{VBM step: } q(\theta) \propto \exp\{\mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}, \mathbf{z}, \theta)]\}, \tag{2.19}$$

and the two steps are iterated until convergence.

As well as deriving the above results, Beal (2003) points out that the Expectation Maximization (EM) algorithm (Dempster et al., 1977) is a special case of VBEM, with a further restriction for $q(\theta)$: $q(\theta)$ degenerates, $q(\theta) = \delta(\theta = \theta^\star)$. The delta function $\delta(\cdot)$ returns one

[1] In some literatures (e.g., Wiegerinck (2000); Xing et al. (2003)), the term 'generalized mean field' refers to the case in which the partition is a collection of *disjoint* subsets, whereas the term 'structured mean field' refers to a more complicated case in which the subsets are allowed to *overlap*.

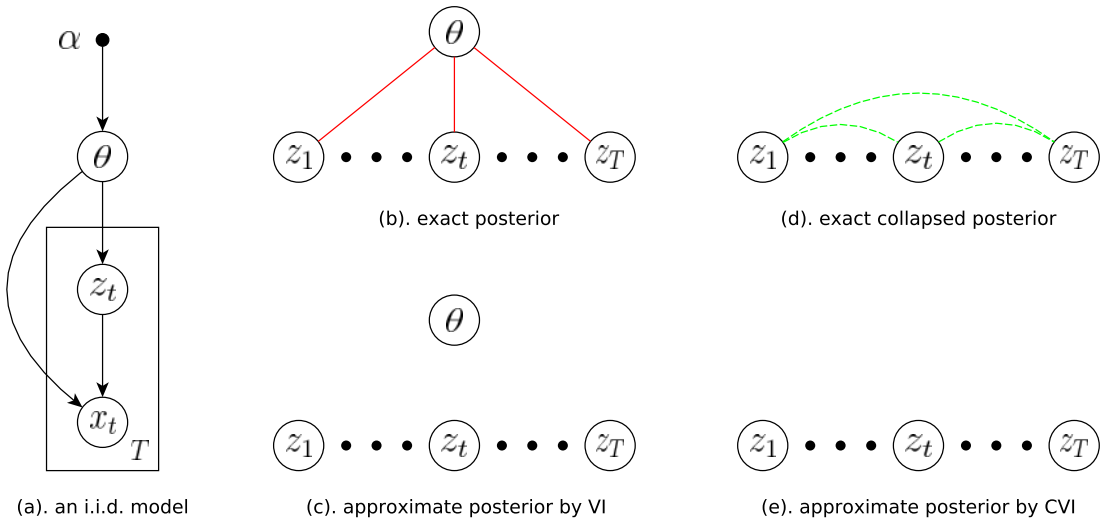

Figure 2.2: Graphical model representations of an i.i.d. model and the various posteriors. (a): the original generative model. (b)-(e): the posteriors assumed by Gibbs sampling (b), variational inference (c), collapsed Gibbs sampling (d), and collapsed variational inference (e). We use red lines to denote dependencies that are inherited from original models, and dashed green lines to denote dependencies that are induced by collapsing parameters.

if its argument is true and zero otherwise. In other words, EM approximates $q(\theta)$ by point estimation. More importantly, Beal (2003) proposed an algorithmic shortcut from EM to VBEM: any propagation algorithm (assuming its existence) in the expectation step of EM can be directly adopted in the VBE step of VBEM by inverting expected natural parameters (defined in the next section). This proposal has contributed to the widespread applications of variational inference in graphical models such as those we study in this thesis.

Although the above variational approach is well studied and widely used, its main drawback is the independence assumption between θ and $\mathbf{z}$. Neglecting how fluctuations in θ induce fluctuations in $\mathbf{z}$ (and vice-versa) allows for analytic updates, but this can potentially lead to very inaccurate local optimal solutions, since θ and $\mathbf{z}$ can be strongly dependent in true posteriors. In the following example, we use the notion of Markov random fields to explain dependencies in true and approximate posteriors.

Example We consider a simple i.i.d. model with T pairs of hidden and observed variables shown in Figure 2.2 (a). It is trivial to write down its posterior,

$$p(\mathbf{z}, \theta|\mathbf{x}) = \frac{\prod_{t=1}^{T} p(x_t, z_t|\theta)p(\theta)}{p(\mathbf{x})}. \tag{2.20}$$

To see why it is a Markov random field shown in Figure 2.2 (b), we identify $p(\theta)$ and $p(x_t, z_t|\theta)$ for $t=\{1, ..., T\}$ as the $T+1$ potential functions over cliques, as well as identify $p(\mathbf{x})$ as the normalization constant. Figure 2.2 (c) shows the posterior distribution assumed by variational inference, where $\mathbf{z}$ and θ are disconnected by dropping their dependencies. Since (2.20) suggests that $\{z_t\}$ directly depend on θ in the terms $p(x_t, z_t|\theta)$, we call such dependencies *strong*. We may wish not to break such strong dependencies in order to get more accurate solutions. This motivates collapsed variational inference (Teh et al., 2007; Sung et al., 2008).

2.2.2 Collapsed Variational Inference

The idea of collapsed variational inference (CVI) is applying variational inference after model parameters are marginalized out (thus the strong dependencies between model parameters and hidden variables are preserved). Hence, this collapsed variational approach has potential to produce more accurate solutions than variational inference.

Formally, in the CVI framework, we first marginalize out the model parameters, yielding the following quantity, known as the complete data marginal likelihood,

$$p(\mathbf{x}, \mathbf{z}) = \int p(\mathbf{x}, \mathbf{z}|\theta)p(\theta)d\theta. \tag{2.21}$$

Next, we introduce a variational distribution $q(\mathbf{z})$ that is restricted in a tractable family, denoted by $\mathcal{Q}$, and seek to solve the following constraint optimization problem,

$$q^\star(\mathbf{z}) = \underset{q(\mathbf{z})\in\mathcal{Q}}{\operatorname{argmax}}\, \mathcal{L}^{\text{CVI}}(q), \tag{2.22}$$

in which $\mathcal{L}^{\text{CVI}}(q)$, another lower bound on $\log p(\mathbf{x})$ is defined by,

$$\begin{aligned}\mathcal{L}^{\text{CVI}}(q) &\triangleq \mathbb{E}_q[\log p(\mathbf{x}, \mathbf{z})] - \mathbb{E}_q[\log q(\mathbf{z})] \\ &\leq \log p(\mathbf{x}).\end{aligned} \tag{2.23}$$

Finally, the optimal $q^\star(\mathbf{z})$ is used as an approximate solution to $p(\mathbf{z}|\mathbf{x})$. If the posterior over model parameters is ever required, we can take a single VBM step of VBEM.

Alternatively, CVI can be viewed as a special case of VI (Teh et al., 2007), where the dependencies between model parameters and hidden variables are modelled in an exact fashion. That is we write $q(\mathbf{h}) = q(\theta|\mathbf{z})q(\mathbf{z})$. Then maximizing $\mathcal{L}^{\text{VI}}(q)$ w.r.t. $q(\theta|\mathbf{z})$ results in updating $q(\theta|\mathbf{z})$ by the true posterior $p(\theta|\mathbf{z}, \mathbf{x})$. Plugging the result into $\mathcal{L}^{\text{VI}}(q)$ recovers

$\mathcal{L}^{\text{CVI}}(q)$, which is tighter than $\mathcal{L}^{\text{VI}}(q)$ with any other choices of $q(\theta|\mathbf{z})$. Hence, to model the exact dependencies between two sets of variables, it is sufficient to integrate out either set.

For restricting $q(\mathbf{z})$, in theory we should be able to adopt the wide variety of methodologies that have been proposed for restricting $q(\mathbf{h})$ in VI. Nevertheless, to date, the only method proposed for restricting $q(\mathbf{z})$ is the complete factorization,

$$q(\mathbf{z}) \approx \prod_{t=1}^{T} q(z_t). \tag{2.24}$$

By this assumption, we maximize $\mathcal{L}^{\text{CVI}}(q)$ w.r.t. each q factor, yielding a set of iterative update equations as follows,

$$q(z_t) \propto \exp\{\mathbb{E}_{q(\mathbf{z}^{\setminus t})}[\log p(\mathbf{x}, \mathbf{z})]\} \tag{2.25}$$

where $\mathbf{z}^{\setminus t}$ denotes the set of all hidden variables with z_t removed. When the RHS of (2.25) is too expensive to compute in practice, we can apply Taylor approximations to some of its components. This often results in simple expressions that only require computing expected sufficient statistics of $\{\mathbf{x}, \mathbf{z}\}$.

CVI with the mean field assumption has been successful for a number of i.i.d. models (Teh et al., 2007; Kurihara et al., 2007; Sung et al., 2008; Teh et al., 2008). A key reason for such success lies in the perfect match between the mean field assumption and the i.i.d. condition. Given an i.i.d. model, the hidden variables are conditionally independent given the model parameters. Marginalizing out the parameters induces dependencies that spread over many hidden variables; and thus the dependency between any two hidden variables is very weak. Hence, we can reasonably assume that the hidden variables are independent with each other in collapsed space and apply a mean field method.

We use our i.i.d. example shown in Figure 2.2 (a) to give more insights about the induced dependencies in the true posterior and explain why it is reasonable to break them in the approximate posterior. We first write down the true posterior in collapsed space,

$$p(\mathbf{z}|\mathbf{x}) = \frac{p(\mathbf{x}, \mathbf{z})}{p(\mathbf{x})}. \tag{2.26}$$

If our example model is in the conjugate exponential family (which is defined in the next section and this condition is assumed to be true until Chapter 5), then $p(\mathbf{x}, \mathbf{z})$ can be tractably computed and the result is a function of sufficient statistics in $\{\mathbf{x}, \mathbf{z}\}$. Since this function can not be factorized further, there must exist dependencies among all the hidden

variables. We call such dependencies induced, since they do not exist in the original model. With these induced dependencies, the true posterior can be viewed as an MRF shown in Figure 2.2 (d) by identifying $p(\mathbf{x}, \mathbf{z})$ as the only potential function over the maximal clique that contains all the variables, and identifying $p(\mathbf{x})$ as the normalization constant.

Figure 2.2 (e) shows the approximate posterior assumed by CVI, in which the induced dependencies are dropped. Since all $\{z_t\}$ decide the sufficient statistics that in turn decide the potential function $p(\mathbf{x}, \mathbf{z})$; given a large T, any z_t is negligible to the statistics. In other words, changing the value of one variable barely affects the value of another variable. Hence, we consider the induced dependencies to be *weak*. Breaking these dependencies fits ideally with the mean field theory: one particular variable relies on the field (i.e. sufficient statistics), through which it interacts with other variables. Therefore, we expect the approximate posterior assumed by CVI to be accurate in the i.i.d. setting.

However, collapsed variational inference has not been applied to more complex models that exhibit for example sequential dependencies (e.g. in hidden Markov models) and structural dependencies (e.g. in probabilistic context free grammars). For models that do not satisfy the i.i.d. condition, there exist strong dependencies among hidden variables in the original generative models, and these dependencies will be inherited in the collapsed space. In these contexts, the complete factorization assumption, required by the mean field method is not an optimal choice. In Chapter 3, we extend the current application domain of CVI to the same application domain of VBEM. We address the challenge of using the complete factorization, explore other factorization strategies and propose an algorithmic shortcut from EM to CVI, analgous to the shortcut from EM to VBEM by Beal (2003).

We conclude this subsection by briefly reviewing another research line of CVI, which focuses on the theoretical analysis of CVI. In particular, Sato and Nakagawa (2012) interpreted CVI with the first order Taylor approximation to (2.25) as an alpha divergence minimization with $\alpha = 1$, whereas VI and exact CVI are alpha divergence minimizations with $\alpha = 0$. By this interpretation, they answered the question raised by Asuncion et al. (2009), who empirically observed that CVI with the first order Taylor approximation often performs better than CVI. This is because inference with $\alpha = 1$ is much less affected by the zero-forcing effect that induces the mode-seeking property than $\alpha = 0$ (Minka, 2005). This theoretical result is orthogonal to our algorithmic contributions in this thesis.

2.2.3 Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) (Neal, 1993; Gilks, 1996) is an alternative inference technique based on stochastic approximations. It estimates a probability distribution using random samples that are generated by a Markov chain mechanism. Formally, a Markov chain is specified by a transition matrix $\mathcal{P}$, where $\mathcal{P}(\mathbf{h}, \mathbf{h}')$ denotes the probability of transitioning into the state $\mathbf{h}'$ from the state $\mathbf{h}$. By defining $\mathcal{P}$ appropriately (the detailed conditions for $\mathcal{P}$ can be found in Bishop (2006, Chapter 11)), the Markov chain can have the true posterior $p(\mathbf{h}|\mathbf{x})$ that we desire as its equilibrium (or stationary) distribution. That is,

$$p(\mathbf{h}|\mathbf{x}) = \int_{\mathbf{h}'} \mathcal{P}(\mathbf{h}', \mathbf{h}) p(\mathbf{h}|\mathbf{x}). \tag{2.27}$$

Once the Markov chain has reached its equilibrium distribution (i.e., converged), we simply collect a series of samples $\mathbf{h}^s \sim \mathcal{P}(\mathbf{h}^{s-1}, \mathbf{h}^s)$ to form a stochastic approximation to the true posterior,

$$p(\mathbf{h}|\mathbf{x}) \approx \frac{1}{S} \sum_{s=1}^{S} \mathbf{h}^s. \tag{2.28}$$

Gibbs sampling (GS) is the simplest MCMC algorithm that satisfies the conditions for $\mathcal{P}$. Its basic idea is instead of sampling the whole set of variables, $\mathbf{h}$ at once; we sample each of the variables in turn, conditioned on all remaining variables. Let $\mathbf{h} = \{\mathbf{h}_t\}_{t=1}^T$ and $\mathbf{h}^{\setminus t} = \mathbf{h} \setminus \{h_t\}$, the transition matrix for Gibbs sampling is given by,

$$\mathcal{P}(\mathbf{h}, \mathbf{h}') = p(h'_t|\mathbf{h}^{\setminus t}, \mathbf{x}) \prod_{j \neq t} \delta(h_j = h'_j), \tag{2.29}$$

The conditional distributions in (2.29) are the core computations for a Gibbs sampler. From the perspective of graphical models, due to the conditional independence properties, the conditionings on $\mathbf{h}^{\setminus t}$ can be further reduced to the Markov blankets[2] of each h_t. In our i.i.d. model example shown in Figure 2.2 (a), for each z_t we have $p(z_t|\mathbf{z}^{\setminus t}, \theta, \mathbf{x}) = p(z_t|\theta, \mathbf{x})$. Such reduced conditionings make the derivations of required conditional distributions much easier and allow Gibbs sampling to be widely used in graphical models.

In this thesis, we use two variants of Gibbs sampling as benchmarks in comparison with our CVI algorithms. Collapsed Gibbs sampling (CGS) improves upon GS by marginalizing out model parameters. In our i.i.d. model example shown in Figure 2.2 (a), CGS draws

[2]The Markov blanket of a variable is defined to be the minimal set of variables conditioned on which the variables is independent of all remaining variables.

z_t from $p(z_t|\mathbf{z}^{\setminus t}, \mathbf{x})$ in the collapsed space. Again if our model is in the conjugate exponential family, $p(z_t|\mathbf{z}^{\setminus t}, \mathbf{x})$ can be tractably computed up to a normalization constant. In the bag-of-words setting, CGS has been empirically observed to converge more quickly than GS (Griffiths and Steyvers, 2004). Another useful strategy for improving GS is sampling blocks of variables. The blocked Gibbs sampling is particularly useful when some variables are strongly dependent, sampling them altogether increases the mobility of the sampler (moving to a very different state) to avoid being trapped in a local optimal state.

Although Gibbs sampling, or MCMC more generally, in the limit can produce samples from the true posteriors that we desire, in practice, it is notoriously difficult to assess the convergence of a sampling method, and normally a sufficiently large number of samples may be required to produce accurate results. Therefore, it is fair to say that in most cases variational inference is a more efficient alternative. As we will see in our experiments, our collapsed variational inference algorithms can match the performances of Gibbs sampling with significantly less training time.

2.3 The Conjugate Exponential Family

In this section, we introduce the exponential family and conjugate priors. We give some of the conjugacy properties that are used in this thesis. Although we deal with categorical data in computational linguistics, our algorithms are generally applicable to distributions in the exponential family. We use the Discrete-Dirichlet distribution pair as a running example.

2.3.1 Definitions

The exponential family is a broad class of probability distributions including the Bernoulli, Beta, Chi-Squared, Dirichlet, Discrete, Gamma, Gaussian, Multinomial, Poisson, Wishart and many others. A probability distribution over $\mathbf{x}$ given parameters θ is in the exponential family if it is of the form,

$$p(\mathbf{x}|\theta) = h_l(\mathbf{x}) \exp\{\eta(\theta)^T s(\mathbf{x}) - a_l(\eta(\theta))\}. \tag{2.30}$$

The natural parameters $\eta(\theta)$ and sufficient statistics $s(\mathbf{x})$ are vector functions. We can also write the distribution as $p(\mathbf{x}|\eta)$, in which case the distribution is directly parametrized

in terms of η. The base measure $h(\mathbf{x})$ and log normalizer $a(\eta)$ are scalar functions. The log normalizer ensures that the distribution is normalized,

$$a_l(\eta) = \int h_l(\mathbf{x}) \exp\{\eta^T s(\mathbf{x})\} d\mathbf{x}. \tag{2.31}$$

Since we often draw many local data points from the same distribution in (2.30), we have used the subscript l to denote 'local'. This is in contrast with the subscript g that denotes 'global' in the prior in (2.33). The subscripts provide a straightforward distinction between data points that are local variables and parameters that are global variables.

Example The discrete distribution (or multinomial distribution with a single observation) over an integer $x \in [1, K]$ given a K dimensional real vector ϕ (with constraints: $\phi_k > 0$ and $\sum_{k=1}^{K} \phi_k = 1$) can be written in the exponential form,

$$p(x|\phi) \triangleq \phi_x = \exp\{\sum_{k=1}^{K} \log \phi_k \delta(x = k)\} = \exp\{(\log \phi)^T \delta_x\}, \tag{2.32}$$

where $\delta(\cdot)$ is the standard indicator function that returns the value 1 if its argument is true and 0 otherwise. δ_x is an indicator vector and the value in its k^{th} entry is $\delta(x = k)$. The natural parameters $\eta(\phi) = \log \phi$ and the sufficient statistics $s(x) = \delta_x$. Note that there are other exponential representations for the discrete distribution. Besides (2.32), common alternatives include using a one-hot binary vector $\mathbf{x} = (x_1, ..., x_K)$ for an observation and using a $K-1$ parameter vector ϕ ($\phi_k > 0$ and $\sum_{k=1}^{K-1} \phi_k < 1$) for parameters.

For any distribution in the exponential family, there exists a conjugate prior. The conjugacy means that the posterior has the same functional form as the prior. A conjugate prior over η given natural parameters λ takes the form,

$$p(\eta|\lambda) = h_g(\eta) \exp\{\lambda_1^T \eta + \lambda_2^T (-a_l(\eta)) - a_g(\lambda)\}. \tag{2.33}$$

The natural parameters $\lambda = (\lambda_1, \lambda_2)$ where λ_1 has the same dimension as η and λ_2 has the dimension of one. The sufficient statistics are $(\eta, -a_l(\eta))$.

Example The Dirichlet distribution over a K dimensional real vector ϕ (with constraints: $\phi_k > 0$ and $\sum_{k=1}^{K} \phi_k = 1$) given a K dimensional real vector β (with constraints: $\beta_k > 0$)

can also be written in the exponential form,

$$p(\phi|\beta) \triangleq \frac{\Gamma(\sum_{k=1}^{K}\beta_k)}{\prod_{k=1}^{K}\Gamma(\beta_k)}\prod_{k=1}^{K}\phi_k^{\beta_k-1}$$
$$= \exp\{(\beta-1)^T(\log\phi) - (\sum_{k=1}^{K}\log\Gamma(\beta_k) - \log\Gamma(\sum_{k=1}^{K}\beta_k)\}, \tag{2.34}$$

where $\Gamma(x) = \int_{x=0}^{\infty} x^{t-1}e^{-x}dx$ is the generalised factorial function. The natural parameters $\lambda(\beta) = (\beta_1 - 1, \beta_2 - 1, ..., \beta_K - 1, \sum_{k=1}^{K}\beta_k - K)$ and the sufficient statistics are $(\log\phi, 0)$.

2.3.2 Properties

Members of the exponential family have shared properties. This allows us to derive algorithms without loss of generality. In this subsection, we give four properties that relate to conjugacy and we use them either directly or as building blocks in this thesis.

Suppose there are $I + M$ data points, $\{\mathbf{x}^i\}_{i=1}^{I+M}$. Each data point $\mathbf{x}^i$ is drawn from the same distribution $p(\cdot|\eta)$ in the exponential family in (2.30). The parameters η are drawn from the conjugate prior $p(\cdot|\lambda)$ in (2.33). The conjugacy properties about the posterior, marginal, predictive, and joint predictive distributions are given as follows.

Property 1. The posterior $p(\eta|\{\mathbf{x}\}_{i=1}^{I}, \lambda) = p(\eta|\lambda^{[1,I]})$ has the same form as the prior $p(\eta|\lambda)$ with updated hyperparameters, $\lambda_1^{[1,I]} = \lambda_1 + \sum_{i=1}^{I} s(\mathbf{x}^i)$ and $\lambda_2^{[1,I]} = \lambda_2 + I$.

It is easy to show by some algebraic manipulation,

$$p(\eta|\mathbf{x}, \lambda) \propto p(\mathbf{x}|\eta)p(\eta|\lambda)$$
$$= \prod_{i=1}^{I} h_l(\mathbf{x}^i)\exp\{\eta^T t(\mathbf{x}^i) - a_l(\eta)\}h_g(\eta)\exp\{\lambda_1^T\eta + \lambda_2^T(-a_l(\eta)) - a_g(\lambda)\}$$
$$\propto h_g(\eta)\exp\{\underbrace{(\lambda_1 + \sum_{i=1}^{I} s(\mathbf{x}^i))}_{\lambda_1^{[1,I]}}{}^T\eta + \underbrace{(\lambda_2 + I)}_{\lambda_2^{[1,I]}}{}^T(-a_l(\eta))\}. \tag{2.35}$$

In this thesis, we use $[a, b]$ to denote the closed integer interval, or more precisely a set of integers $\{a, a+1, a+2, ..., b\}$. In (2.35), we have used the interval $[1, I]$ as superscripts of λ to denote the updated hyperparameters after observing the data points $\{\mathbf{x}^i\}_{i=1}^{I}$. For the rest of this section, we use the following notations,

$$\lambda_1^A = \lambda_1 + \sum_{j\in A} s(\mathbf{x}^j), \qquad \lambda_2^A = \lambda_2 + |A|, \tag{2.36}$$

where A is a set of integers representing indices and $|A|$ is the cardinality of A.

Example In the Discrete-Dirichlet case, suppose there are $I + M$ discrete observations, $\{x^i\}_{i=1}^{I+M}$. Each $x^i \sim$ Discrete(ϕ) defined in (2.32) for $i \in [1, I + M]$, and $\phi \sim$ Dir(β) defined in (2.34). The posterior $p(\phi|\{x^i\}_{i=1}^I, \beta) =$ Dir$(\beta^{[1,I]})$, where $\beta^{[1,I]} = \beta + C^{[1,I]}$ and $C^{[1,I]}$ is a counting function that takes $\{x^i\}_{i=1}^I$ as its argument and returns a vector whose k^{th} entry being $\sum_{i=1}^I \delta(x^i = k)$. As with λ, we use the same superscript conventions both for β and C. Given a set of integers A,

$$\beta_k^A = \beta + \sum_{j \in A} \delta(x^j = k), \qquad C_k^A = \sum_{j \in A} \delta(x^j = k). \tag{2.37}$$

Property 2. The marginal distribution $p(\{\mathbf{x}^i\}_{i=1}^I|\lambda)$ after marginalizing out η composes of the analytical known log normalizers in the following form,

$$\begin{aligned}
p(\{\mathbf{x}^i\}_{i=1}^I|\lambda) &= \int p(\{\mathbf{x}^i\}_{i=1}^I|\eta)p(\eta|\lambda)d\eta \\
&= \int \prod_i h_l(\mathbf{x}^i) \exp\{\eta^T s(\mathbf{x}^i) - a_l(\eta)\} h_g(\eta) \exp\{\lambda_1^T \eta + \lambda_2^T(-a_l(\eta)) - a_g(\lambda)\} d\eta \\
&= \big(\prod_i h_l(\mathbf{x}^i)\big) \int h_g(\eta) \exp\{(\lambda_1^{[1,I]})^T \eta + (\lambda_2^{[1,I]})^T(-a_l(\eta))\} d\eta \exp\{-a_g(\lambda)\} \\
&= \big(\prod_i h_l(\mathbf{x}^i)\big) \exp\{a_g(\lambda^{[1,I]}) - a_g(\lambda)\}
\end{aligned} \tag{2.38}$$

Example The marginal distribution in the Discrete-Dirichlet case is,

$$p(\{x^i\}_{i=1}^I|\beta) = \frac{\Gamma(\sum_k \beta_k)}{\Gamma(I + \sum_k \beta_k)} \prod_k \frac{\Gamma(\beta_k + C_k^{[1,I]})}{\Gamma(\beta_k)} \tag{2.39}$$

Property 3. The posterior predictive distribution $p(\mathbf{x}^{I+1}|\{\mathbf{x}^i\}_{i=1}^I, \lambda)$ composes of the analytical known log normalizers in the following form,

$$\begin{aligned}
p(\mathbf{x}^{I+1}|\{\mathbf{x}^i\}_{i=1}^I, \lambda) &= \frac{p(\{\mathbf{x}^i\}_{i=1}^{I+1}|\lambda)}{p(\{\mathbf{x}^i\}_{i=1}^I|\lambda)} \\
&= h_l(\mathbf{x}^{I+1}) \frac{\exp\{a_g(\lambda^{[1,I+1]}) - a_g(\lambda)\}}{\exp\{a_g(\lambda^{[1,I]}) - a_g(\lambda)\}} \\
&= h_l(\mathbf{x}^{I+1}) \exp\{a_g(\lambda^{[1,I+1]}) - a_g(\lambda^{[1,I]})\},
\end{aligned} \tag{2.40}$$

Example The predictive posterior in the Discrete-Dirichlet case is,

$$\begin{aligned} p(x^{I+1}|\{x^i\}_{i=1}^I,\beta) =& \frac{\Gamma(I+\sum_k \beta_k)}{\Gamma(I+1+\sum_k \beta_k)} \prod_k \frac{\Gamma(\beta_k + C_k^{[1,I+1]})}{\Gamma(\beta_k + C_k^{[1,I]})} \\ =& \frac{C_{x^{I+1}}^{[1,I]} + \beta_{x^{I+1}}}{I+\sum_k \beta_k}, \end{aligned} \tag{2.41}$$

where the last line uses the property of the Gamma function, $\Gamma(x+1) = x\Gamma(x)$.

Property 4. The joint posterior predictive distribution $p(\{\mathbf{x}^m\}_{m=I+1}^{I+M}|\{\mathbf{x}^i\}_{i=1}^I,\lambda)$ composes of the analytical known log normalizers in the following form,

$$\begin{aligned} & p(\{\mathbf{x}^m\}_{m=I+1}^{I+M}|\{\mathbf{x}^i\}_{i=1}^I,\lambda) \\ =& \big(\prod_{m=I+1}^{I+M} h_l(\mathbf{x}^m)\big) \frac{\exp\{a_g(\lambda^{[1,I+M]}) - a_g(\lambda)\}}{\exp\{a_g(\lambda^{[1,I]}) - a_g(\lambda)\}} \\ =& \big(\prod_{m=I+1}^{I+M} h_l(\mathbf{x}^m)\big) \exp\{a_g(\lambda^{[1,I+M]}) - a_g(\lambda^{[1,I]})\} \quad \text{(standard formalism)} \\ =& \big(\prod_{m=I+1}^{I+M} h_l(\mathbf{x}^m)\big) \exp\{\textstyle\sum_{m'=I+1}^{I+M} a_g(\lambda^{[1,m']}) - a_g(\lambda^{[1,m'-1]})\} \\ =& \prod_{m=I+1}^{I+M} \underbrace{h_l(\mathbf{x}^m) \exp\{a_g(\lambda^{[1,m]}) - a_g(\lambda^{[1,m-1]})\}}_{\text{predictive over } \mathbf{x}^m \text{ given } \{\mathbf{x}^j\}_{j=1}^{m-1}} \quad \text{(our formalism)} \end{aligned} \tag{2.42}$$

Example The joint predictive posterior in the Discrete-Dirichlet case is,

$$\begin{aligned} & p(\{x^m\}_{m=I+1}^{I+M}|\{x^i\}_{i=1}^I,\beta) \\ =& \frac{\Gamma(I+\sum_k \beta_k)}{\Gamma(I+M+\sum_k \beta_k)} \prod_k \frac{\Gamma(\beta_k + C_k^{[1,I+M]})}{\Gamma(\beta_k + C_k^{[1,I]})} \quad \text{(standard formalism)} \\ =& \prod_{m=I+1}^{I+M} \frac{C_{x^m}^{[1,I]} + \sum_{j=I+1}^{m-1} \delta(x^j = x^m) + \beta_{x^m}}{m-1+\sum_k \beta_k} \quad \text{(our formalism)} \end{aligned} \tag{2.43}$$

2.4 Bayesian Nonparametrics

The Dirichlet process (DP) (Ferguson, 1973) has been an important cornerstone of Bayesian nonparametrics. It is widely used to construct models in which model sizes are unbounded a priori (Escobar and West, 1994; Rasmussen, 2000; Blei and Jordan, 2005). In this section,

we review the basic definitions of the Dirichlet process, and in particular its stick breaking construction that we use for defining models in Chapter 5.

The Dirichlet process is often viewed as the infinite extension of the Dirichlet distribution we have reviewed in Section 2.3. Its formal definition was introduced by Ferguson (1973). Let $(\Omega, \mathcal{F}, H)$ be a probability space, where Ω is a sample space, $\mathcal{F}$ is a σ-algebra[3] , and $H\colon \mathcal{F} \to [0, 1]$ is a probability measure. A random probability measure G is distributed according to the Dirichlet process parametrized by H and a concentration parameter α, denoted by $G \sim \text{DP}(\alpha, H)$, if for any finite partition $\{A_j \subseteq \Omega\colon j \in [1, J]\}$ of Ω,

$$(G(A_1), G(A_2), ..., G(A_J)) \sim \text{Dir}(\alpha H(A_1), \alpha H(A_2), ..., \alpha H(A_J)). \tag{2.44}$$

This definition is based on the consistent finite marginals, and it is convenient for deriving the DP's posterior and other properties using the properties of the finite Dirichlet distribution. Nevertheless, this definition is not constructive, leaving the questions such as how to draw samples from the DP.

Alternatively, Sethuraman (1994) proposed an explicit definition based on the stick breaking process. An infinite random vector $\pi = (\pi_1, \pi_2, ...)$ is distributed according to the stick breaking process with the concentration parameter α, denoted by $\pi \sim \text{GEM}(\alpha)$ (GEM standard for Griffiths, Engen and McCloskey) if

$$\pi_k = \tilde{\pi}_k \prod_{l<k} (1 - \tilde{\pi}_l), \qquad \tilde{\pi}_k \sim \text{Beta}(1, \alpha), \qquad \text{for } k = 1, 2, ... \tag{2.45}$$

Let δ_ϕ be an atomic measure such that $\delta_\phi(A) = 1$ if $\phi \in A$, otherwise it is 0. Then summing over a set of weighted atomic measures is also a measure, for example, $\sum_{k=1}^{\infty} \pi_k \delta_{\phi_k}$. Provided that π is a random draw from $\text{GEM}(\alpha)$, and for $k = 1, 2, ...$, ϕ_k are i.i.d. random draws from the base measure H; Sethuraman (1994) shows that $G = \sum_{k=1}^{\infty} \pi_k \delta_{\phi_k}$ is a random probability measure distributed according to the $\text{DP}(\alpha, H)$.

Since G is a random probability measure over the measurable space $(\Omega, \mathcal{F})$ (due to $\sum_{k=1}^{\infty} \pi_k = 1$ with probability one), we can treat it as a distribution over Ω. This distribution is discrete and has a countably infinite number of supports. For this reason, the DP is known as the infinite extension of the Dirichlet distribution.

[3] A σ-algebra $\mathcal{F}$ is a collection of subsets of Ω that satisfy the following conditions: (1) $\mathcal{F}$ is not empty; (2) if $A \in \mathcal{F}$, then $\Omega \setminus A \in \mathcal{F}$; and (3) if $A_j \in \Omega$ for $j \in [1, J]$, then $\cup_{j=1}^{J} A_j \in \mathcal{F}$.

Chapter 3

Collapsed Variational Inference for HMMs

Chapter Abstract

In this chapter, we investigate the application of collapsed variational inference to hidden Markov models. Incrementally, we propose three algorithms: each relaxes some of the assumptions or application conditions required by the previous one (Wang and Blunsom, 2013a, 2015b). The novel techniques proposed along with the algorithms are generally applicable to a wide variety of probabilistic graphical models in the conjugate exponential family. We validate our collapsed variational algorithms in the Part-of-Speech tagging, word alignment and textual prediction tasks, showing that our algorithms are as scalable as, but significantly more accurate than variational inference. Our algorithms perform comparably with collapsed Gibbs sampling with much less training time.

3.1 Introduction

Hidden Markov Models (HMMs) are widely used for representing sequential data in various fields including computational linguistics, speech recognition, information retrieval, computer vision, bioinformatics and finance. The core theory of HMMs, together with the celebrated forward-backward (or Baum-Welch) algorithm was developed by Baum and his

colleagues (Baum and Petrie, 1966; Baum et al., 1970). As a simple but effective statistical tool, the popularity of HMMs soared in the following decades, yielding a variety of applications and elaborations, reviewed by Juang and Rabiner (1991). Smyth et al. (1997) expressed HMMs as Bayesian networks, which inspired the development of more complex and hierarchical networks such as factorial HMMs (Ghahramani and Jordan, 1997) and tree decision HMMs (Jordan et al., 1997). To exploit the advantages of the Bayesian paradigm (no overfit, coherent treatment of uncertainty), MacKay (1997) proposed to place priors on HMMs and derived an ensemble learning algorithm. Since then, there has been an explosion of new approximate inference algorithms for Bayesian HMMs (Scott, 2002; Beal, 2003; Goldwater and Griffiths, 2007; Wang and Blunsom, 2013a; Foti et al., 2014; Wang and Blunsom, 2015b), all of which aim to improve the accuracy or the scalability or both.

Variational inference (VI) for HMMs (MacKay, 1997; Beal, 2003) seeks to minimise the divergence between the true posterior and an approximation in which the parameters and hidden sequences are assumed independent. This strong assumption allows for an efficient iterative solution, but it can lead to poor approximations. Alternatively, collapsed Gibbs sampling (CGS) for HMMs (Goldwater and Griffiths, 2007) integrates out the parameters, and draws samples for hidden variables in turn from the true posterior. In theory, CGS reaches the true posterior after convergence. In practice it is notoriously difficult to assess the convergence of samplers, and mixing is slow for distributions with tightly coupled latent variables like the HMM. It remains a challenge to develop algorithms that are accurate, efficient and deterministic, especially for large scale problems in our application domain of computation linguistics.

To address the above challenge, in this chapter we develop collapsed variational inference (CVI) algorithms for HMMs. As we have reviewed in Section 2.2, it is ideal to apply collapsed variational inference (Teh et al., 2007) to models whose hidden variables are conditional independent given parameters. In such settings, hidden variables are only weakly dependent on each other after parameters are integrated out. Thus, applying the mean field method in collapsed space is reasonable and the empirical result of CVI is often superior to VI. However, in the time dependent setting of HMMs, the mean field method is not optimal since there are strong first order dependencies among hidden variables even in the collapsed space. Whether and how CVI can be applied and its empirical performance in a time dependent setting are unknown.

We investigate CVI for HMMs incrementally and propose three algorithms (Sections 3.4-3.6). Each algorithm relaxes some of the assumptions or application conditions required by the previous one. First, we apply the vanilla mean field method to update posteriors at the variable level in the same way as for i.i.d. models. A challenge of calculating transition counts arises from breaking sequential dependencies and thus we propose a new calculation method. Second, we consider the common scenario of many independent time series and apply the structured mean field method to update posteriors at the cluster level. A challenge in this case is that inside a cluster surrogate parameters change on the fly, and thus we propose a simple approximation method, which is reasonable by intuition and accurate by empirical validation. Subsequently, the variational posteriors exhibit the desired first order Markov dependencies. Third, we relax the previous condition and break a single long time series into subsequences. As well as the above two challenges, the third challenge is that the standard forward-backward algorithm is not directly applicable to infer subsequences. To address it, we propose a novel propagation redesign method.

One recent trend in approximate Bayesian inference is to develop stochastic optimization procedures for batch algorithms. An important milestone was made by Hoffman et al. (2013), who proposed to compute cheap gradients based on minibatches of data and derived stochastic variational inference (SVI) for the CE family. Following this achievement, numerous SVI algorithms have been developed and one relevant example to this chapter is SVI for HMMs (Foti et al., 2014). Combining the advantages of CVI and SVI, Foulds et al. (2013) proposed a stochastic collapsed variational inference (SCVI) algorithm for latent Dirichlet allocation, outperforming SVI. In Section 3.7, we consider HMMs whose emission distributions are in the exponential family and derive SCVI for the CE family, building on the CVI for the CE family framework by Sung et al. (2008).

In Section 3.8, we validate our (stochastic) CVI algorithms for HMMs in the language processing tasks, Part-of-Speech (PoS) induction and word alignment, as well as the standard prediction performance. For PoS induction, we use a number of evaluation metrics in both the semi-supervised and unsupervised settings (Christodoulopoulos et al., 2010). By all of the evaluation metrics, our (stochastic) CVI algorithms consistently and often significantly outperform the (stochastic) VI algorithms. CGS has slight performance advantages over CVI when datasets are very small, while CVI can be one order of magnitude faster.

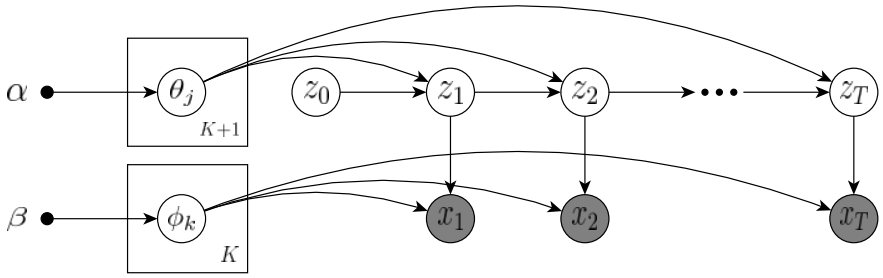

Figure 3.1: Graphical model representation of a Bayesian HMM.

3.2 Definitions

A hidden Markov model consists of an observed sequence $\mathbf{x} = \{x_t\}_{t=1}^T$ and a corresponding hidden state sequence $\mathbf{z} = \{z_t\}_{t=0}^T$. Let there be K hidden states. For convenience, we set the start state 0 and $z_0 = 0$. Let $\boldsymbol{\theta}$ be the transition matrix where $\theta_{j,k} = p(z_t = k|z_{t-1} = j)$, and θ_0 be the initial state distribution where $\theta_{0,k} = p(z_1 = k)$. Here we focus on discrete HMMs and consider the general exponential family case in Section 3.7. In the discrete case, let ϕ_k parametrize the Discrete distribution at each state k with W possible observations and $\phi_{k,w} = p(x_t = w|z_t = k)$. In a Bayesian setting, we place symmetric Dirichlet priors with hyperparameters α and β on each θ_j and ϕ_k, respectively. Formally, the generative process of a Bayesian HMM is,

$$\text{for } j = 0, ..., K, \quad \theta_j \sim \text{Dir}(\alpha), \tag{3.1}$$

$$\text{for } t = 1, ..., T, \quad z_t|z_{t-1}, \theta \sim \text{Mult}(\theta_{z_{t-1}}), \tag{3.2}$$

$$\text{for } k = 1, ..., K, \quad \phi_k \sim \text{Dir}(\beta), \tag{3.3}$$

$$\text{for } t = 1, ..., T, \quad x_t|z_t, \phi \sim \text{Mult}(\phi_{z_t}). \tag{3.4}$$

The graphical model representation is given in Figure 3.1. Having defined the model, the joint probability of the parameters and variables is,

$$p(\mathbf{x}, \mathbf{z}, \theta, \phi|\alpha, \beta) = p(\theta|\alpha)p(\phi|\beta) \prod_{t=1}^{T} p(z_t|z_{t-1}, \theta)p(x_t|z_t, \phi). \tag{3.5}$$

As we have reviewed in Section 2.3, the sufficient statistics for Discrete distributions are the delta functions; with multiple observations, we have also introduced the counting

functions. For discrete HMMs, we define the following counting functions,

$$C_{j,k}^{[1,T]}(\mathbf{z}) \triangleq \sum_{t=1}^{T} \delta(z_{t-1} = j, z_t = k), \tag{3.6}$$

$$C_{k,w}^{[1,T]}(\mathbf{x}, \mathbf{z}) \triangleq \sum_{t=1}^{T} \delta(z_t = k, x_t = w), \tag{3.7}$$

and they represent the number of transitions from state j to state k and number of observations w at state k, respectively. As before we omit the arguments of the counting functions. The superscripts are by default $[1, T]$ and thus can be omitted too. In other words, $C_{j,k}$ and $C_{k,w}$ are the shorthand notations for $C_{j,k}^{[1,T]}(\mathbf{z})$ and $C_{k,w}^{[1,T]}(\mathbf{x}, \mathbf{z})$, respectively.

We often use the dot to denote the summed out column, e.g.,

$$C_{j,\cdot} \triangleq \sum_{k=1}^{K} C_{j,k} \qquad\qquad C_{\cdot,k} \triangleq \sum_{j=0}^{K} C_{j,k} \tag{3.8}$$

It is standard to use the superscript $\backslash t$ to denote exclusion of the t^{th} variable. For example, $\mathbf{z}^{\backslash t} = \{z_{t'}\}_{t'=1 \wedge t' \neq t}^{T}$. Here we also use $\backslash t$ to denote the exclusion of the contributions from the t^{th} variables. Formally, we define $C_{j,k}^{\backslash t}$ and $C_{k,w}^{\backslash t}$ as follows,

$$\begin{aligned} C_{j,k}^{\backslash t} &\triangleq \sum_{\substack{t'=1 \\ t' \neq t \wedge t' \neq t+1}}^{T} \delta(z_{t'-1} = j, z_{t'} = k) \\ &= C_{j,k} - \delta(z_{t-1} = j, z_t = k) - \delta(z_t = j, z_{t+1} = k), \end{aligned} \tag{3.9}$$

$$C_{k,w}^{\backslash t} \triangleq \sum_{\substack{t'=1 \\ t' \neq t}}^{T} \delta(z_{t'} = k, x_{t'} = w) = C_{k,w} - \delta(z_t = k, x_t = w). \tag{3.10}$$

Note that for transition, $C_{j,k}^{\backslash t}$ is defined to exclude both transitions from $t-1$ to t and t to $t+1$. As a result, the argument of the counting function $C_{j,k}^{\backslash t}$ is $\mathbf{z}^{\backslash t}$. Since their superscripts are consistent, then the argument can be safely omitted.

3.3 Approximate Inference for HMMs

In this section, we review the VI and CGS algorithms for HMMs as they are widely used for learning HMMs, which serve as benchmarks in our experiments. Some parts of VI and CGS are used in the derivations of our CVI algorithms.

3.3.1 Variational Inference

The standard Baum-Welch training algorithm (Baum and Petrie, 1966) for HMMs is a special case of a general class of algorithms, namely expectation-maximization (EM) (Dempster et al., 1977), which can be further considered as a subclass of variational inference (Beal, 2003). Variational inference introduces an evidence lower bound (ELBO) on the log marginal likelihood $\log p(\mathbf{x})$, denoted by $\mathcal{L}^{\text{VI}}$,

$$\log p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z},\theta,\phi)}[\log p(\mathbf{x},\mathbf{z},\theta,\phi)] - \mathbb{E}_{q(\mathbf{z},\theta,\phi)}[\log q(\mathbf{z},\theta,\phi)] \triangleq \mathcal{L}^{\text{VI}}, \tag{3.11}$$

with $q(\mathbf{z},\theta,\phi)$ a variational posterior, and it is factorized by assuming independence between parameters and latent variables,

$$q(\mathbf{z},\theta,\phi) = q(\mathbf{z})q(\theta,\phi). \tag{3.12}$$

Maximizing the ELBO requires an coordinate ascent update, alternating between $q(\mathbf{z})$ and $q(\theta,\phi)$. For the Baum-Welch algorithm based on the maximum likelihood estimation, $q(\theta)$ degenerates, that is $q(\theta) = \delta(\theta = \theta^\star)$. Hence, the Baum-Welch algorithm consists of the Expectation and Maximization steps,

$$\text{E step: } q(\mathbf{z}) = p(\mathbf{z}|\mathbf{x},\theta^\star,\phi^\star), \tag{3.13}$$

$$\text{M step: } \theta^\star,\phi^\star = \underset{\theta,\phi}{\operatorname{argmax}}\, \mathbb{E}_{q(\mathbf{z})}[p(\mathbf{x},\mathbf{z}|\theta,\phi)]. \tag{3.14}$$

Solving the E step yields,

$$\begin{aligned} q(\mathbf{z}) &\propto p(\mathbf{x},\mathbf{z}|\theta^\star,\phi^\star) \\ &= \prod_{t=1}^{T} p(z_t|z_{t-1},\theta^\star)p(x_t|z_t,\phi^\star) \\ &= \prod_{t=1}^{T} \theta^\star_{z_{t-1},z_t}\phi^\star_{z_t,x_t}. \end{aligned} \tag{3.15}$$

The distribution over the whole sequence $q(\mathbf{z})$ is intractable, but the forward backward dynamic programming algorithm can be used to calculate the marginals $q(z_t)$ and $q(z_{t-1},z_t)$ that are required in the M step,

$$q(z_t) \propto \alpha_t(z_t)\beta_t(z_t), \tag{3.16}$$

$$q(z_t,z_{t+1}) \propto \alpha_t(z_t)\theta^\star_{z_t,z_{t+1}}\phi^\star_{z_{t+1},x_{t+1}}\beta_{t+1}(z_{t+1}), \tag{3.17}$$

where $\alpha_t(k) = p(z_t = k, \mathbf{x}_{1:t}|\theta^\star, \phi^\star)$ is the forward probability of being in state k at time step t together with the observations before and include t; $\beta_t(k) = p(\mathbf{x}_{t+1:T}|z_t = k, \theta^\star, \phi^\star)$ is the backward probability of seeing observations after t, given that the state is k at time step t. Both are computed recursively. Details can be found in Bishop (2006).

Solving the M step yields,

$$\theta^\star_{j,k} = \frac{\sum_t q(z_{t-1} = j, z_t = k)}{\sum_t q(z_{t-1} = j)} = \frac{\mathbb{E}_{q(\mathbf{z})}[C_{j,k}]}{\mathbb{E}_{q(\mathbf{z})}[C_{j,\cdot}]}, \tag{3.18}$$

$$\phi^\star_{k,w} = \frac{\sum_t q(z_t = k)\delta(x_t = w)}{\sum_t q(z_t = k)} = \frac{\mathbb{E}_{q(\mathbf{z})}[C_{k,w}]}{\mathbb{E}_{q(\mathbf{z})}[C_{\cdot,k}]}. \tag{3.19}$$

Variational inference generalizes EM by putting no restrictions on the parametric form of $q(\theta, \phi)$. Thus, the update in the M step becomes,

$$q(\theta, \phi) \propto \exp\{\mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}, \mathbf{z}|\theta, \phi) + \log p(\theta, \phi)]\} \tag{3.20}$$

Solving the above equation results in Dirichlet distributions with updated hyperparameters by the conjugacy property 1 in Section 2.3. Based on the theorem relating EM and VI (Beal, 2003, p.74), we can equivalently make a minor change to the M step and reuse the forward-backward implementation in the E step based on the inverted expected natural parameters, which are defined as follows,

$$\tilde{\theta}_{j,k} = \frac{\exp(\psi(\mathbb{E}_{q(\mathbf{z})}[C_{j,k}] + \alpha))}{\exp(\psi(\mathbb{E}_{q(\mathbf{z})}[C_{j,\cdot}] + K\alpha))}, \tag{3.21}$$

$$\tilde{\phi}_{k,w} = \frac{\exp(\psi(\mathbb{E}_{q(\mathbf{z})}[C_{k,w}] + \beta))}{\exp(\psi(\mathbb{E}_{q(\mathbf{z})}[C_{\cdot,k}] + W\beta))}, \tag{3.22}$$

where $\psi(x) = \frac{\partial \Gamma(x)}{\partial x}$ is the digamma function.

Ignoring how fluctuations in (θ, ϕ) induce fluctuations in $\mathbf{z}$ (and vice-versa) allows for analytic iterations, and both the EM and VI algorithms are efficient and easy to implement. Nevertheless, the independence assumption may potentially lead to very inaccurate estimates. The parameters and latent variables are strongly dependent in the true posterior $p(\mathbf{z}, \theta, \phi|\mathbf{x}, \alpha, \beta)$, which is proportional to the joint distribution given by (3.5). As we see in the following, CGS and CVI model the strong dependencies in an exact fashion.

3.3.2 Collapsed Gibbs Sampling

The collapsed Gibbs sampler for HMMs (Goldwater and Griffiths, 2007) marginalizes out parameters, thus dealing them exactly. It leads to the complete-data marginal likelihood,

$$p(\mathbf{x}, \mathbf{z}|\alpha, \beta) = \int p(\mathbf{z}, \mathbf{x}|\theta, \phi)p(\theta, \phi|\alpha, \beta)d\theta d\phi. \tag{3.23}$$

Although the above integration is in high dimensions, it is in fact easy to compute due to the conjugacy between Dirichlet and Discrete distributions. Specifically, we can apply the conjugacy property 2 in Section 2.3 and obtain,

$$\begin{aligned} p(\mathbf{x}, \mathbf{z}|\alpha, \beta) =& \prod_{j=0}^{K} \frac{\Gamma(K\alpha)}{\Gamma(K\alpha + C_{j,\cdot})} \Big(\prod_{k=1}^{K} \frac{\Gamma(\alpha + C_{j,k})}{\Gamma(\alpha)} \Big) \\ & \times \prod_{k=1}^{K} \frac{\Gamma(W\beta)}{\Gamma(W\beta + C_{\cdot,k})} \Big(\prod_{w=1}^{W} \frac{\Gamma(\beta + C_{k,w})}{\Gamma(\beta)} \Big). \end{aligned} \tag{3.24}$$

The CGS then draws hidden sequence samples by drawing each z_t conditioned on other variables in turn. Let $\mathbf{z}^{\setminus t}$ denote $\mathbf{z}$ excluding z_t, the posterior predictive $p(z_t|\mathbf{x}, \mathbf{z}^{\setminus t}, \alpha, \beta)$ can be derived by using the conjugacy property 3 in Section 2.3,

$$\begin{aligned} p(z_t|\mathbf{x}, \mathbf{z}^{\setminus t}, \alpha, \beta) &\propto \frac{p(\mathbf{x}, \mathbf{z}|\alpha, \beta)}{p(\mathbf{x}^{\setminus t}, \mathbf{z}^{\setminus t}|\alpha, \beta)} \\ &\propto (C^{\setminus t}_{z_{t-1}, z_t} + \alpha) \times \frac{C^{\setminus t}_{z_t, z_{t+1}} + \alpha + \delta(z_{t-1}{=}z_t{=}z_{t+1})}{C^{\setminus t}_{z_t, \cdot} + K\alpha + \delta(z_{t-1}{=}z_t)} \times \frac{C^{\setminus t}_{z_t, x_t} + \beta}{C^{\setminus t}_{\cdot, z_t} + W\beta}, \end{aligned} \tag{3.25}$$

where $C^{\setminus t}_{k,w}$ and $C^{\setminus t}_{j,k}$ have been defined formally in Section 3.2. The existence of the delta functions is due to sampling two transitions (from $t-1$ to t and from t to $t+1$) at the same time.

The CGS does not make any independence assumptions, and draws samples from the true posterior $p(\mathbf{z}|\mathbf{x}, \alpha, \beta)$. However, as with other MCMC samplers, it is often hard to assess convergence, and one needs to set the number of samples and burn-in period based on experience. In practice, one often draws as many samples as possible (within the limited time frame) to reduce sampling variance, and thus it is much less efficient than VI.

Griffiths and Steyvers (2004) observed that CGS for LDA converged relatively quickly. In LDA, the conditional distribution for the currently updating variable depends on other variables only through the counts, i.e. the dependency on any particular other variable is very small. Hence quick convergence is to be expected. For HMMs the conditional distribution for z_t in (3.25) depends on the states of the previous hidden variable (z_{t-1}) and the

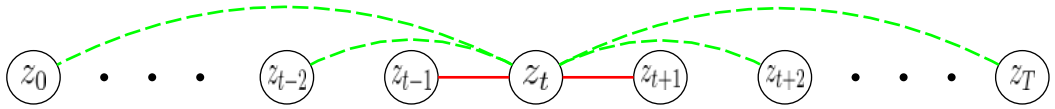

Figure 3.2: Graph depicting the true posterior $p(\mathbf{z}|\mathbf{x}, \alpha, \beta)$ in a collapsed HMM (only the dependencies relating to z_t are drawn). Strictly speaking, it is a fully connected Markov random field. For describing our algorithms, we make the following distinctions: the solid red lines depict the strong dependencies inherited from the original model, and the dashed green lines depict the weak dependencies induced by marginalizing out parameters.

next hidden variable (z_{t+1}), as well as the global counts. Such strong dependencies make CGS for HMMs much slower to converge (Gao and Johnson, 2008).

3.4 CVI Algorithm 1 for HMMs

The goal of collapsed variational inference is to approximate the true posterior $p(\mathbf{z}|\mathbf{x}, \alpha, \beta)$ that is intractable to compute. Figure 3.2 gives a graphical depiction of the true posterior, which essentially can be considered as a fully connected Markov random field. In other words, there is dependency between every pair of variables. Hence, any marginals are intractable. As we have discussed in Section 2.2, we consider the dependencies inherited from the original HMM to be strong and those induced by integrating out parameters to be weak. Unlike an i.i.d. latent variable model shown in Figure 2.2 (d), the true posterior of a collapsed HMM consists of both types of dependencies.

The distinctions between inherited and induced dependencies have also been reflected in the update equation of CGS. In the conditional distribution (3.25), the dependencies of z_t on z_{t-1} and z_{t+1} are clearly different from its dependencies on other variables. Take z_{t-1} for example, in (3.25), the term $C^{\backslash t}_{z_{t-1},z_t}$ is given by,

$$C^{\backslash t}_{z_{t-1},z_t} \triangleq \sum_{\substack{t'=1 \\ t'\neq t \wedge t'\neq t+1}}^{T} \delta(z_{t'-1} = z_{t-1}, z_{t'} = z_t), \tag{3.26}$$

and we can see that $C^{\backslash t}_{z_{t-1},z_t}$ strongly depends the value of z_{t-1}, but the value of any another hidden variable merely affects this counting term, especially when T is large.

Collapsed variational inference integrates out parameters and introduces an evidence lower bound on the log marginal likelihood $\log p(\mathbf{x})$, denoted by $\mathcal{L}^{\text{CVI}}$,

$$\log p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}, \mathbf{z}|\alpha, \beta)] - \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] \triangleq \mathcal{L}^{\text{CVI}}, \tag{3.27}$$

with $q(\mathbf{z})$ a variational posterior. In Section 2.2, we have shown that $\mathcal{L}^{\text{CVI}}$ is a tighter bound than $\mathcal{L}^{\text{VI}}$, and integrating out parameters is equivalent to modelling the dependencies of parameters on hidden variables in an exact fashion.

In our first CVI algorithm, we assume that the hidden variables are mutually independent in the collapsed space (shown in Figure 3.3),

$$q(\mathbf{z}) = \prod_{t=1}^{T} q(z_t). \tag{3.28}$$

It is a strong assumption for HMMs, since both the strong and weak dependencies have been broken. However, when we examine the assumption of VI in (3.12), the strong dependencies between parameters and hidden variables are also broken. It is not immediately apparent which assumption is weaker, and thus this CVI algorithm has the potential to lead to better approximations than VI. Furthermore, our assumption is reminiscent of the naive mean field algorithm for Ising models, which breaks the pairwise dependencies and provides a first step in investigating more structured mean field algorithms in the uncollapsed setting (where model parameters are provided) (Xing et al., 2003).

The standard mean field update for $q(z_t)$ is given by,

$$q(z_t) \propto \exp\{\mathbb{E}_{q(\mathbf{z}^{\setminus t})}[\log p(\mathbf{x}, \mathbf{z}|\alpha, \beta)]\} \tag{3.29}$$

$$\propto \exp\{\mathbb{E}_{q(\mathbf{z}^{\setminus t})}[\log p(z_t|\mathbf{x}, \mathbf{z}^{\setminus t}, \alpha, \beta)]\} \tag{3.30}$$

Although in previous work (Teh et al., 2007; Sung et al., 2008), the standard approach for deriving a CVI algorithm is plugging the complete-data marginal likelihood into (3.29), a simpler and more explicit approach is to plug the unnormalized posterior predictive distribution into (3.30). We can easily verify that the proportional relation still holds.

We plug (3.25) into (3.30), obtaining the first CVI update for HMMs,

$$q(z_t) \propto \exp\{\mathbb{E}[\log(C^{\setminus t}_{z_{t-1},z_t} + \alpha)]\} \times \frac{\exp\{\mathbb{E}[\log(C^{\setminus t}_{z_t,z_{t+1}} + \alpha + \delta(z_{t-1}{=}z_t{=}z_{t+1}))]\}}{\exp\{\mathbb{E}[\log(C^{\setminus t}_{z_t,\cdot} + K\alpha + \delta(z_{t-1}{=}z_t))]\}}$$
$$\times \frac{\exp\{\mathbb{E}[\log(C^{\setminus t}_{z_t,x_t} + \beta)]}{\exp\{\mathbb{E}[\log(C^{\setminus t}_{\cdot,z_t} + W\beta)]\}}. \tag{3.31}$$

The exact computation of the expected log functions in the above update is intractable. We follow Teh et al. (2007) and use a Taylor approximation. Originally, Teh et al. (2007) proposed the second order Taylor approximation which was shown to under perform the

Figure 3.3: Graph depicting the variational posterior $q(\mathbf{z})$ assumed by the CVI algorithm 1. All the dependencies are broken and thus all the variables are mutually independent.

same algorithm with only the first order information in later work (Asuncion et al., 2009). Sato and Nakagawa (2012) argued that the lower order approximation was less affected by the zero-forcing effect, which induces the mode-seeking property in statistical learning (Minka, 2005). Therefore, we employ the first order Taylor approximation throughout this thesis. Here we have for example,

$$\mathbb{E}_{q(\mathbf{z}^{\setminus t})}[\log(C^{\setminus t}_{z_{t-1},z_t} + \alpha)] \approx \log(\mathbb{E}_{q(\mathbf{z}^{\setminus t})}[C^{\setminus t}_{z_{t-1},z_t}] + \alpha). \tag{3.32}$$

Applying the first order Taylor approximation to all the expected log terms in (3.31), we obtain an approximate CVI update,

$$\begin{aligned} q(z_t) \approx\propto (\mathbb{E}[C^{\setminus t}_{z_{t-1},z_t}] + \alpha) \times \frac{\mathbb{E}[C^{\setminus t}_{z_t,z_{t+1}}] + \alpha + \mathbb{E}[\delta(z_{t-1}{=}z_t{=}z_{t+1})]}{\mathbb{E}[C^{\setminus t}_{z_t,\cdot}] + K\alpha + \mathbb{E}[\delta(z_{t-1}{=}z_t)]} \\ \times \frac{\mathbb{E}[C^{\setminus t}_{z_t,x_t}] + \beta}{\mathbb{E}[C^{\setminus t}_{\cdot,z_t}] + W\beta}. \end{aligned} \tag{3.33}$$

The extreme similarity between CGS comprising of actual counts (3.25), and CVI comprising of expected counts (3.33), confirms that the our first CVI algorithm indeed is the mean field version of collapsed Gibbs sampling.

The expected emission counts in (3.33) are easy to compute,

$$\begin{aligned} \mathbb{E}_{q(\mathbf{z}^{\setminus t})}[C^{\setminus t}_{k,w}] &= \sum_{t' \neq t} \mathbb{E}_{q(z_{t'})}[\delta(x_{t'} = w)\delta(z_{t'} = k)] \\ &= \sum_{t' \neq t} \delta(x_{t'} = w) q(z_{t'} = k) \end{aligned} \tag{3.34}$$

However, the strong dependencies in the true posterior brings an unforeseen challenge (only the variational posterior is factorized). Specifically, the expected transition counts involving z_{t-1} and z_{t+1} in (3.33) give rise to the difficulty: unlike the CGS update (3.25), here z_{t-1} and z_{t+1} are not fixed values but distributions. This challenge does not appear for i.i.d. latent variable models and the existing techniques can not be applied, and therefore we propose our approach.

To compute for example $\mathbb{E}[C^{\setminus t}_{z_{t-1},z_t}]$, we utilize the following equations that hold trivially,

$$C^{\setminus t}_{j,k} = C^{\setminus t-1,t}_{j,k} + \delta(z_{t-2} = j, z_{t-1} = k), \tag{3.35}$$

$$q(\mathbf{z}^{\setminus t}) = q(z_{t-1})q(\mathbf{z}^{\setminus t-1,t}), \tag{3.36}$$

where $\mathbf{z}^{\setminus t-1,t}$ denotes excluding both the $t-1^{th}$ and t^{th} hidden variables, and $C^{\setminus t-1,t}_{j,k}$ denotes excluding the following transitions: from $t-2$ to $t-1$, from $t-1$ to t and from t to $t+1$. Our definitions are consistent: since removing one hidden variable results in removing two transitions, removing two hidden variables results in removing three transitions.

We can compute $\mathbb{E}[C^{\setminus t}_{z_{t-1},z_t}]$ as follows,

$$\begin{aligned}
&\mathbb{E}_{q(\mathbf{z}^{\setminus t})}[C^{\setminus t}_{z_{t-1},z_t}] \\
&= \mathbb{E}_{q(\mathbf{z}^{\setminus t})}[C^{\setminus t-1,t}_{z_{t-1},z_t}] + \mathbb{E}_{q(\mathbf{z}^{\setminus t})}[\delta(z_{t-2} = z_{t-1}, z_{t-1} = z_t)] \quad \text{(by (3.35))} \\
&= \mathbb{E}_{q(z_{t-1})}[\mathbb{E}_{q(\mathbf{z}^{\setminus t-1,t})}[C^{\setminus t-1,t}_{z_{t-1},z_t}] \quad \text{(by (3.36))} \\
&\quad + \sum_{z_{t-2}} \sum_{z_{t-1}} q(z_{t-1})q(z_{t-2})\delta(z_{t-2} = z_{t-1}, z_{t-1} = z_t) \\
&= \sum_{z_{t-1}} \underbrace{q(z_{t-1})\mathbb{E}_{q(\mathbf{z}^{\setminus t-1,t})}[C^{\setminus t-1,t}_{z_{t-1},z_t}]}_{\text{most contributions weighted by } q(z_{t-1})} + \underbrace{q(z_{t-2} = z_t)q(z_{t-1} = z_t)}_{\text{one contribution from } t-2 \text{ to } t-1},
\end{aligned} \tag{3.37}$$

in which

$$\mathbb{E}_{q(\mathbf{z}^{\setminus t-1,t})}[C^{\setminus t-1,t}_{j,k}] = \sum_{\substack{t'=1 \\ t' \notin \{t-1,t,t+1\}}}^{T} q(z_{t'-1} = j, z_{t'} = k). \tag{3.38}$$

We can interpret our equation (3.37) as follows. The number of transitions from z_{t-1} to z_t depends on the specific state of z_{t-1}. Since we do not know the fixed state of z_{t-1}, by intuition we should sum over all the possible states and obtain a weighted average, which is precisely the first term in (3.37). The first term counts for most contributions, and the second term only counts for one contribution from the time step $t-2$ to $t-1$.

Similarly, we can compute the expected counts involving z_{t+1} in (3.33),

$$\begin{aligned}
&\mathbb{E}_{q(\mathbf{z}^{\setminus t})}[C^{\setminus t,t+1}_{z_t,z_{t+1}}(\mathbf{z}^{\setminus t})] \\
&= \sum_{z_{t+1}} \underbrace{q(z_{t+1})\mathbb{E}_{q(\mathbf{z}^{\setminus t,t+1})}[C^{\setminus t,t+1}_{z_t,z_{t+1}}]}_{\text{most contribution weighted by } q(z_{t+1})} + \underbrace{q(z_{t+1} = z_t)q(z_{t+2} = z_t)}_{\text{one contribution from } t+1 \text{ to } t+2}.
\end{aligned} \tag{3.39}$$

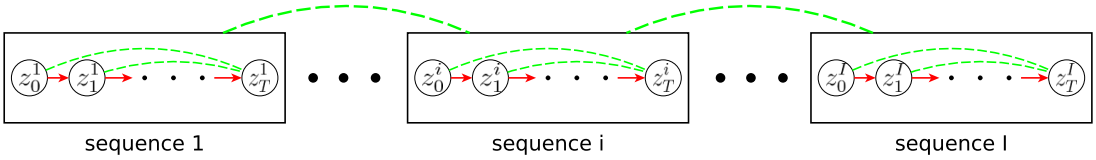

Figure 3.4: Graph depicting the true posterior $p(\mathbf{z}|\mathbf{x}, \alpha, \beta)$ in the multiple sequence setting. The sequence level dependencies between i and j mean that every variable in sequence i is dependent on every variable in sequence j. Other drawing conventions follow Figure 3.2.

The implementation for the first CVI algorithm simply keeps track of the global expected transition and emission counts $\mathbb{E}[C_{j,k}]$ and $\mathbb{E}[C_{k,w}]$, subtracting the local expected counts contributed by z_t (and z_{t-1} or z_{t+1}) when needed. After updating $q(z_t)$, the new expected counts around z_t are added back into the global counts. Each update of $q(z_t)$ has the computational complexity $O(K^2)$, which is the same as variational inference.

3.5 CVI Algorithm 2 for HMMs

The strong independence assumption in our first CVI algorithm has the potential to lead to inaccurate results. In this section and the next, we aim to break as few strong dependencies as possible. Our investigation of CVI for HMMs is inspired by large scale applications in computational linguistics. We note that a common feature of those problems is that there are usually many short sequences (i.e. sentences), where each sequence is drawn i.i.d. from the same set of parameters. In this section, we consider the multiple sequence setting: a collection of independent HMM sequences can be considered as a bag of clusters.

In the multiple sequence setting, we alter some notations while keeping most consistent with those in the previously described algorithms. We still use t to index the time step and T to denote the length of a sequence. For notational simplicity, we assume that all the sequences have the same length T. As with other parts of this thesis, we use i to index the data point. Altogether, we define $\mathbf{x} = \{\mathbf{x}^i\}_{i=1}^I$ and $\mathbf{z} = \{\mathbf{z}^i\}_{i=1}^I$, where $\mathbf{x}^i$ and $\mathbf{z}^i$ are the i^{th} observed and hidden sequences, respectively; and $\mathbf{x}^i = \{x_t^i\}_{t=1}^T$ and $\mathbf{z}^i = \{z_t^i\}_{t=1}^T$, where x_t^i and z_t^i are the t^{th} observed and hidden variables in the i^{th} sequence, respectively. In this setting, we alter the definitions of the counting functions accordingly: $C_{j,k}$ and $C_{k,w}$ denote

the transition and emission counts accumulated from all the sequences, respectively,

$$C_{j,k} = \sum_{i=1}^{I} \sum_{t=1}^{T} \delta(z_{t-1}^i = j, z_t^i = k), \tag{3.40}$$

$$C_{k,w} = \sum_{i=1}^{I} \sum_{t=1}^{T} \delta(z_t^i = k, x_t^i = w), \tag{3.41}$$

and $\backslash i$ denotes the exclusion of the contribution from the i^{th} sequence,

$$C_{j,k}^{\backslash i} = \sum_{\substack{i'=1 \\ i' \neq i}}^{I} \sum_{t=1}^{T} \delta(z_{t-1}^{i'} = j, z_t^{i'} = k), \tag{3.42}$$

$$C_{k,w}^{\backslash i} = \sum_{\substack{i'=1 \\ i' \neq i}}^{I} \sum_{t=1}^{T} \delta(z_t^{i'} = k, x_t^{i'} = w). \tag{3.43}$$

Our goal is to approximate the true posterior $p(\mathbf{z}|\mathbf{x}, \alpha, \beta)$ depicted in Figure 3.4. Again, it is a fully connected Markov random field and thus the exact computation of any marginal is intractable. As with our first CVI algorithm 1, we introduce an evidence lower bound $\mathcal{L}^{\text{CVI}}$ after marginalization of the parameters. For the variational posterior $q(\mathbf{z})$, this time we only break the sequence level dependencies, which are induced by marginalizing out parameters and thus are weak. Formally, we assume that $q(\mathbf{z})$ is factorized by,

$$q(\mathbf{z}) = \prod_{i=1}^{I} q(\mathbf{z}^i). \tag{3.44}$$

It might be worth emphasising that for each $q(\mathbf{z}^i)$, we do not assume any further factorizations, not even the first order Markov structures. That is each $q(\mathbf{z}^i)$ itself is a fully connected Markov random field. The graphical depiction is in Figure 3.5.

Let $\mathbf{x}^{\backslash i}$ denote the set of all observed sequences with $\mathbf{x}^i$ removed. Similarly, let $\mathbf{z}^{\backslash i}$ denote the set of all hidden sequences with $\mathbf{z}^i$ removed. Applying the structured mean field method at the cluster level, we obtain the update equation for $q(\mathbf{z}^i)$, which maximizes the lower bound $\mathcal{L}^{\text{CVI}}$,

$$\begin{aligned} q(\mathbf{z}^i) &\propto \exp(\mathbb{E}_{q(\mathbf{z}^{\backslash i})}[\log p(\mathbf{z}^i|\mathbf{x}, \mathbf{z}^{\backslash i})]) \\ &\propto \exp(\mathbb{E}_{q(\mathbf{z}^{\backslash i})}[\log p(\mathbf{x}^i, \mathbf{z}^i|\mathbf{x}^{\backslash i}, \mathbf{z}^{\backslash i})]) \end{aligned} \tag{3.45}$$

In our first CVI algorithm, we have revealed an important connection between CGS and CVI: a posterior predictive distribution used by CGS can be plugged into the corresponding

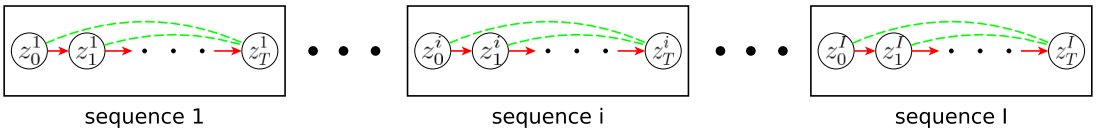

Figure 3.5: Graph depicting the variational posterior $q(\mathbf{z})$ assumed by the CVI algorithm 2. Only the sequence level weak dependencies are broken.

CVI update equation. Since now we update a cluster of variables simultaneously, it makes sense to first derive its joint posterior predictive. As we will see in more details in Chapter 4, blocked CGS and structured CVI share exactly the same spirit in updating a cluster of variables. By using the conjugacy property 4, with some algebraic manipulation we derive,

$$p(\mathbf{x}^i, \mathbf{z}^i | \mathbf{x}^{\setminus i}, \mathbf{z}^{\setminus i}) = \prod_{t=1}^{T} \frac{C^{\setminus i}_{z^i_{t-1}, z^i_t} + \sum_{t'=1}^{t-1} \delta(z^i_{t'-1} = z^i_{t-1}, z^i_{t'} = z^i_t) + \alpha}{C^{\setminus i}_{z^i_{t-1}, \cdot} + \sum_{t'=1}^{t-1} \delta(z^i_{t'-1} = z^i_{t-1}) + K\alpha} \times \frac{C^{\setminus i}_{z^i_t, x^i_t} + \sum_{t'=1}^{t-1} \delta(z^i_{t'} = z^i_t, x^i_{t'} = x^i_t) + \beta}{C^{\setminus i}_{\cdot, z^i_t} + \sum_{t'=1}^{t-1} \delta(z^i_{t'} = z^i_t) + W\beta}, \tag{3.46}$$

where $C^{\setminus i}_{j,k}$ and $C^{\setminus i}_{j,k}$ have been defined in (3.42) and (3.43).

At this stage, we face another inference challenge: if we proceed the same way as in the first algorithm (plugging (3.46) into (3.45) and applying a Taylor approximation), then the delta functions in (3.46) require at each time step t, looking up variables until the $t-1$ time step. This prohibits the use of efficient dynamic programming. Thus we seek to make an approximation before proceeding.

We propose to approximate (3.46) by removing all its delta functions. This approximation essentially ignores the local contribution from some parts of the i^{th} sequence to the global counts. Compared with contributions from all other sequences, the impact of these local counts is small. We empirically analyse our approximation in the next subsection.

By removing the delta functions, the approximate joint predictive is,

$$p(\mathbf{x}^i, \mathbf{z}^i | \mathbf{x}^{\setminus i}, \mathbf{z}^{\setminus i}) \approx \prod_{t=1}^{T} \frac{C^{\setminus i}_{z^i_{t-1}, z^i_t} + \alpha}{C^{\setminus i}_{z^i_{t-1}, \cdot} + K\alpha} \times \frac{C^{\setminus i}_{z^i_t, x^i_t} + \beta}{C^{\setminus i}_{\cdot, z^i_t} + W\beta}. \tag{3.47}$$

Substituting (3.47) into (3.45), and with the first order Taylor approximation, we derive $q(\mathbf{z}^i)$ that exhibits the desired first order Markov dependencies and output independence,

$$q(\mathbf{z}^i) \approx\propto \prod_{t=1}^{T} \bar{\theta}_{z^i_{t-1}, z^i_t} \bar{\phi}_{z^i_t, x^i_t}, \tag{3.48}$$

in which we define,

$$\bar{\theta}_{j,k} = \frac{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C_{j,k}^{\setminus i}] + \alpha}{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C_{j,\cdot}^{\setminus i}] + K\alpha}, \tag{3.49}$$

$$\bar{\phi}_{k,w} = \frac{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C_{k,w}^{\setminus i}] + \beta}{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C_{\cdot,k}^{\setminus i}] + W\beta}. \tag{3.50}$$

The similarity between (3.15) and (3.48) suggests that the dynamic programming used in the EM algorithm can be applied here to obtain the required marginals. In the E step, the EM algorithm uses the maximum likelihood parameters $(\theta^\star, \phi^\star)$ learnt from the M step; the VI algorithm uses the inverted expected natural parameters $(\tilde{\theta}, \tilde{\phi})$ after the variational M step; while our second CVI algorithm uses the surrogate parameters $(\bar{\theta}, \bar{\phi})$ based on the expected counts from all other sequences. The main difference with EM and VI is that the surrogate parameters in CVI change after updating each sequence; whereas EM and VI batch update the parameters in the M step after processing all sequences in the E step.

3.5.1 Empirical Validation

In this subsection, we empirically validate the approximation method we have proposed in our second CVI algorithm and discuss its application conditions. Then we generalize it to the conjugate exponential family case, which we use in Section 3.7.

To formalize our approximation method, we return to the notations in Section 2.3. Suppose $x^i \sim \text{Discrete}(\phi)$ for $i \in [1, I+M]$ and $\phi \sim \text{Dir}(\beta)$. As before, we let $C_k^{[1,I]}$ denote $\sum_{i=1}^{I} \delta(x^i = k)$. The conjugacy property 4 states that the joint posterior predictive is,

$$p(\{x^m\}_{m=I+1}^{I+M} | \{x^i\}_{i=1}^{I}, \beta) = \prod_{m=I+1}^{I+M} \frac{C_{x^m}^{[1,I]} + \sum_{j=I+1}^{m-1} \delta(x^j = x^m) + \beta_{x^m}}{m - 1 + \sum_k \beta_k}. \tag{3.51}$$

If we consider (3.51) as a product of predictives over x^m, then they are changing on the fly because of the delta functions ($m-1$ in the denominator can be written as $\sum_{i=1}^{m-1} \delta(1)$). Removing the delta functions allows (3.51) to be expressed as a product of predictives over x^m given the fixed $\{x^i\}_{i=1}^{I}$,

$$p(\{x^m\}_{m=I+1}^{I+M} | \{x^i\}_{i=1}^{I}, \beta) \approx \prod_{m=I+1}^{I+M} \frac{C_{x^m}^{[1,I]} + \beta_{x^m}}{I + \sum_k \beta_k}. \tag{3.52}$$

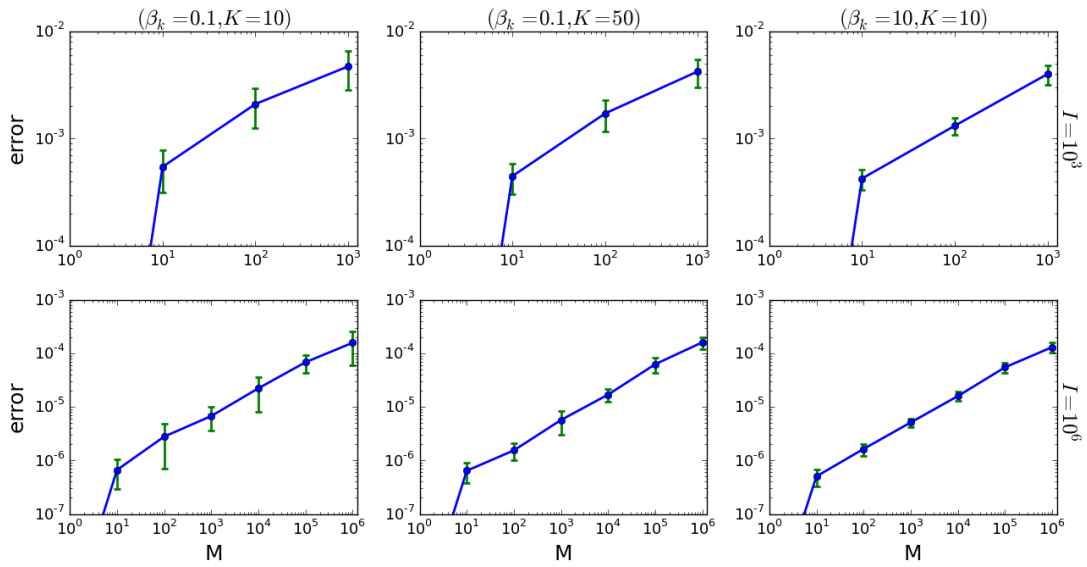

Figure 3.6: Empirical analysis of our approximation method. The approximation errors are averaged by 20 runs and their standard derivations are depicted by the vertical bars.

We define the average approximation error as,

$$\text{error} = \frac{1}{M}\sum_{m=I+1}^{I+M}\left|\frac{C_{x^m}^{[1,I]} + \sum_{j=I+1}^{m-1}\delta(x^j = x^m) + \beta_{x^m}}{m-1+\sum_k \beta_k} - \frac{C_{x^m}^{[1,I]} + \beta_{x^m}}{I+\sum_k \beta_k}\right|, \tag{3.53}$$

and conduct a comprehensive set of experiments to show the accuracy of our approximation. We vary $I \in \{1{,}000, 1{,}000{,}000\}$; and for each I, we vary $M \in \{10^0, 10^1, ..., I\}$. In other words, we start with one predictive data point that has no errors, and then increase the number of predictive data points exponentially until it is equal to I. By increasing the number of delta functions, we expect errors to increase as well. We assume the Dirichlet prior to be symmetric, since it is a common choice in practice when we do not have further prior information. We vary $(\beta_k, K) \in \{(0.1, 10), (0.1, 50), (10, 10)\}$ to investigate if there are any correlations between their values and errors. For each of the above combined settings, we run 20 experiments to reduce variance.

Figure 3.6 shows the averaged approximation errors and standard deviations. In each plot, we can see that a larger M leads to more errors, confirming our intuition. Comparing the top and bottom plots, we see a larger dataset can reduce errors dramatically. For example, when $I = 1{,}000{,}000$ and $M = 10$ (which is common in computational linguistics), the difference between the true and approximate distributions is indeed negligible (less than 10^{-6}). When $M = 1{,}000$, errors are also at an acceptable level (less than 10^{-5}). Finally,

comparing different values of (β_k, K), the correlations between their choices and errors are not obvious, and our approximation works well for all the tested combinations.

By the above empirically analysis, we can conclude that for our second CVI algorithm to be accurate (at least in discrete sequential datasets), we need to impose two application conditions. First, the length of each sequence is relatively small compared with the combined length of all sequences. Second, the combined length is large, which we assume to be true throughout this thesis (if datasets are small, then sampling is preferred). Both conditions are almost always satisfied in the field of computational linguistics (datasets often consist of many short sequences) and in many other areas including speech recognition, computer vision and data compression.

Below we generalize our approximation method to the CE family. In the rest of this thesis, we refer to this method as the approximation method to joint posterior predictives.

The Conjugate Exponential Family Using the same notations as in Section 2.3, we suppose that for $i \in [1, I+M]$, $\mathbf{x}^i$ is drawn from a distribution in the exponential family $p(\cdot|\eta)$ defined by (2.30) and the natural parameters η are drawn from the conjugate prior $p(\cdot|\lambda)$ defined by (2.33). As before, let $\lambda_1^A = \lambda_1 + \sum_{i\in A} s(\mathbf{x}^i)$ and $\lambda_2^A = \lambda_2 + |A|$, where A is a set of integers. The conjugacy property 4 states that the joint posterior predictive distribution is,

$$\begin{aligned}
& p(\{\mathbf{x}^m\}_{m=I+1}^{I+M} | \{\mathbf{x}^i\}_{i=1}^{I}, \lambda) \\
= & \Big(\prod_{m=I+1}^{I+M} h_l(\mathbf{x}^m) \Big) \exp\{a_g(\lambda^{[1,I+M]}) - a_g(\lambda^{[1,I]})\} \quad \text{(standard formalism)} \\
= & \prod_{m=I+1}^{I+M} \underbrace{h_l(\mathbf{x}^m) \exp\{a_g(\lambda^{[1,m]}) - a_g(\lambda^{[1,m-1]})\}}_{\text{predictive over } \mathbf{x}^m \text{ given } \{\mathbf{x}^i\}_{i=1}^{I} \text{ and } \{\mathbf{x}^j\}_{j=I+1}^{m-1}} \quad \text{(our formalism)}
\end{aligned} \tag{3.54}$$

By formalizing the joint distribution in this way, we can consider it as a product of M predictive distributions. Inspired by removing the delta functions in the Discrete-Dirichlet case, here we propose to remove the sufficient statistics $\sum_{j=I+1}^{m-1} s(\mathbf{x}^j)$ in both log normalizers in (3.54). Denoting the union of the set $[1, I]$ and the singleton $\{m\}$ by $[1, I] \cup \{m\}$, our approximation to the joint predictive in the CE family is,

$$p(\{\mathbf{x}^m\}_{m=I+1}^{I+M} | \{\mathbf{x}^i\}_{i=1}^{I}, \lambda) \approx \prod_{m=I+1}^{I+M} \underbrace{h_l(\mathbf{x}^m) \exp\{a_g(\lambda^{[1,I]\cup\{m\}}) - a_g(\lambda^{[1,I]})\}}_{\text{predictive over } \mathbf{x}^m \text{ given fixed } \{\mathbf{x}^i\}_{i=1}^{I}}. \tag{3.55}$$

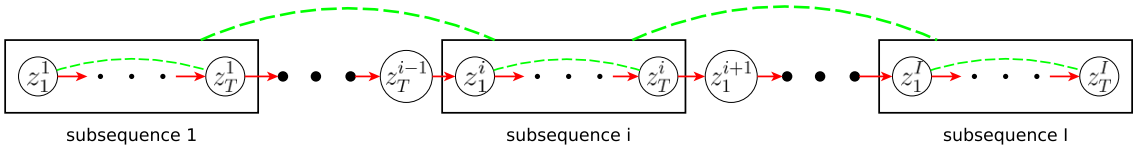

Figure 3.7: Graph depicting the true posterior $p(\mathbf{z}|\mathbf{x}, \alpha, \beta)$ in the one long sequence setting. Compared to Figure 3.4, in this setting there are strong dependencies at the boundaries of subsequences. Drawing notations follow Figure 3.4.

Provided that I is large and $I \gg M$, our approximation is expected to be accurate, since the contribution from $\{\mathbf{x}^j\}_{j=I+1}^{m-1}$ is negligible compared to the contribution from $\{\mathbf{x}^i\}_{i=1}^{I}$. In this thesis we focus on categorical datasets, and leave empirical analysis for other conjugate distribution pairs as future work.

3.6 CVI Algorithm 3 for HMMs

In this section, we propose our third CVI algorithm that is applicable to the single sequence setting. Our motivations are two-folds. First, for some applications of HMMs in the fields of biology and finance, datasets often consist of several very long time series (Foti et al., 2014). In this cases, our second CVI algorithm is inaccurate (for example, when there are two very long sequences) or even inapplicable (when there is only one sequence). Second, and more importantly, it is of theoretical interest to investigate how collapsed variational inference can be applied to truly chain-based structures, rather than a bag-of-chains[1]. This helps to shed lights on the general application of CVI to probabilistic graphical models.

In the one long sequence setting, we keep notations consistent with those in our second CVI algorithm. Here we do not have a collection of independent sequences. Instead, we consider a dataset (one long sequence) to be made of many *consecutive* subsequences. Using the same notations, we let $\mathbf{x} = \{\mathbf{x}^i\}_{i=1}^{I}$, where $\mathbf{x}^i$ is the i^{th} subsequence; and $\mathbf{x}^i = \{x_t^i\}_{t=1}^{T}$, where x_t^i is the t^{th} observation in the i^{th} subsequence. We decompose the corresponding hidden sequence into *consecutive* subsequences as well, namely $\mathbf{z} = \{\mathbf{z}^i\}_{i=1}^{I}$ and $\mathbf{z}^i = \{z_t^i\}_{t=1}^{T}$. The counting functions are also defined the same as in our second CVI

[1]Our first and second CVI algorithms were proposed in Wang and Blunsom (2013a), where an anonymous reviewer argued that 'what I consider as a main contribution in this area is to propose an approach for the case of one long sequence'. Our third algorithm was proposed in Wang and Blunsom (2015b).

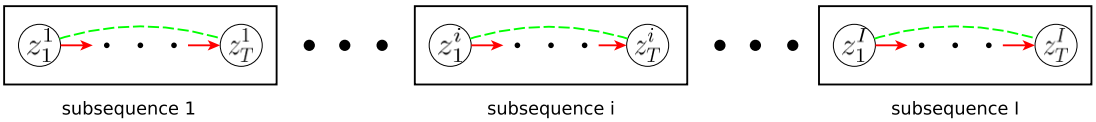

Figure 3.8: Graph depicting the variational posterior $q(\mathbf{z})$ assumed by the CVI algorithm 3. The subsequence level dependencies are broken including the strong ones at the boundaries.

algorithm (3.40-3.43) except for a minor change to $C^{\setminus i}_{j,k}$,

$$C^{\setminus i}_{j,k} = C_{j,k} - \delta(z^{i-1}_T{=}j, z^i_1{=}k) - \delta(z^i_T{=}j, z^{i+1}_1{=}k) - \sum_{t=2}^{T} \delta(z^i_{t-1}{=}j, z^i_t{=}k), \tag{3.56}$$

in which excluding the i^{th} subsequence of length T removes $T+1$ transitions.

In (3.56) and the rest of this section, the hidden variables z^{i-1}_T and z^{i+1}_1 are not convenient for writing concise mathematical expressions, and thus we rename them. Formally, for a subsequence $\mathbf{z}^i$ in the context of the whole sequence $\mathbf{z}$, the hidden variable preceding $\mathbf{z}^i$ is the last variable in the previous subsequence, z^{i-1}_T; and the hidden variable following it is the first variable in the next subsequence, z^{i+1}_1. We rename z^{i-1}_T to z^i_0 (the 0^{th} hidden variable of the i^{th} subsequence), and rename z^{i+1}_1 to z^i_{T+1} (the $T+1^{th}$ hidden variable of the i^{th} subsequence). By our renaming, (3.56) can be concisely written as,

$$C^{\setminus i}_{j,k} = C_{j,k} - \sum_{t=1}^{T+1} \delta(z^i_{t-1}{=}j, z^i_t{=}k). \tag{3.57}$$

As with our previous CVI algorithms, the true posterior $p(\mathbf{z}|\mathbf{x}, \alpha, \beta)$ shown in Figure 3.7 is intractable due to the graphical model's full connectivity. Instead we optimize the evidence lower bound $\mathcal{L}^{\text{CVI}}$. We introduce a variational distribution and assume that it factorizes as a product of local distributions over subsequences (shown in Figure 3.8),

$$q(\mathbf{z}) = \prod_{i=1}^{I} q(\mathbf{z}^i). \tag{3.58}$$

Compared with our first CVI algorithm, we keep most dependencies intact inside subsequences and our third algorithm is expected to be much more accurate. Compared with our second CVI algorithm, there exist strong dependencies at the boundaries of subsequences in the true posterior, which allows for message passing at the cluster level. Therefore, our third CVI algorithm works beyond the bag-of-clusters setting.

Similar to our second CVI algorithm, the update equation of $q(\mathbf{z}^i)$ maximizing $\mathcal{L}^{\text{CVI}}$ is given by,

$$q(\mathbf{z}_i) \propto \exp(\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[\log p(\mathbf{x}_i, \mathbf{z}_i | \mathbf{x}^{\setminus i}, \mathbf{z}^{\setminus i})]), \tag{3.59}$$

in which the joint posterior predictive is similarly derived as in (3.46),

$$\begin{aligned} p(\mathbf{x}^i, \mathbf{z}^i | \mathbf{x}^{\setminus i}, \mathbf{z}^{\setminus i}) = &\left(\overbrace{C^{\setminus i}_{z^i_0, z^i_1} + \alpha}^{\text{apply (3.61a)}} \right) \prod_{t=2}^{T+1} \frac{\overbrace{C^{\setminus i}_{z^i_{t-1}, z^i_t} + \sum_{t'=1}^{t-1} \delta(z^i_{t'-1} = z^i_{t-1}, z^i_{t'} = z^i_t) + \alpha}^{\text{apply (3.61b) for } t \in [2,T] \text{ and apply (3.61c) for } t = T+1}}{C^{\setminus i}_{z^i_{t-1}, \cdot} + \sum_{t'=1}^{t-1} \delta(z^i_{t'-1} = z^i_{t-1}) + K\alpha} \\ &\times \prod_{t=1}^{T} \underbrace{\frac{C^{\setminus i}_{z^i_t, x^i_t} + \sum_{t'=1}^{t-1} \delta(z^i_{t'} = z^i_t, x^i_{t'} = x^i_t) + \beta}{C^{\setminus i}_{z^i_t, \cdot} + \sum_{t'=1}^{t-1} \delta(z^i_{t'} = z^i_t) + W\beta}}_{\text{apply (3.61b)}}. \end{aligned} \tag{3.60}$$

If we proceed in the same way as our second CVI algorithm (removing the delta functions in (3.60), plugging the result into (3.59) and applying a Taylor approximation), then all the emission counts and most of the transition counts (for $t \in [2, T]$) work out in the same way as our second CVI algorithm. However, the transition counts at the boundaries ($t = 1$ and $t = T + 1$) turn out to be very problematic. To calculate their expected values, which will play important roles in message passing between clusters, we extend and utilize the techniques we have proposed in our previous CVI algorithms.

We first generalize our approximation to joint posterior predictives. In (3.60), instead of removing the delta functions only, we also remove the contribution from one more subsequence to the global counts. Explicitly, we present the following approximate relations,

$$C^{\setminus i} \approx C^{\setminus i-1,i}, \tag{3.61a}$$

$$C^{\setminus i} + \sum \delta(\cdot) \approx C^{\setminus i}, \tag{3.61b}$$

$$C^{\setminus i} + \sum \delta(\cdot) \approx C^{\setminus i,i+1}, \tag{3.61c}$$

where $C^{\setminus i-1,i}$ excludes the contribution from both the $i-1^{th}$ and i^{th} subsequences and $C^{\setminus i,i+1}$ excludes the contribution from both the i^{th} and $i+1^{th}$ subsequences. For simplicity, we write $\sum \delta(\cdot)$ to denote a summation of some delta functions, e.g., those in (3.60). Since $\sum \delta(\cdot)$ represents a partial contribution from the i^{th} subsequence, the largest difference between LHS and RHS in all the approximate relations of (3.61) is less than two subsequences' contribution, which is accurate by our empirical analysis in Subsection 3.5.1.

We apply the approximate relations (3.61) to (3.60) (details are given in (3.60)). Then we plug the result into (3.59) and apply a Taylor approximation, yielding,

$$q(\mathbf{z}^i) \approx\propto \underbrace{\left(\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C^{\setminus i-1,i}_{z_0^i,z_1^i}] + \alpha\right) \frac{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C^{\setminus i,i+1}_{z_T^i,z_{T+1}^i}] + \alpha}{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C^{\setminus i,i+1}_{z_T^i,\cdot}] + K\alpha}}_{\text{challenge: the expectations at boundary}} \underbrace{\prod_{t=2}^{T} \bar{\theta}_{z_{t-1}^i,z_t^i} \prod_{t=1}^{T} \bar{\phi}_{z_t^i,x_t^i}}_{\text{a standard HMM}}. \tag{3.62}$$

We have divided (3.62) into two parts. Its second part is similar to (3.48) in our second CVI algorithm: it is a standard HMM (except for the initial distribution) parametrized by the surrogate parameters,

$$\bar{\theta}_{j,k} = \frac{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C^{\setminus i}_{j,k}] + \alpha}{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C^{\setminus i}_{j,\cdot}] + K\alpha}, \tag{3.63}$$

$$\bar{\phi}_{k,w} = \frac{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C^{\setminus i}_{k,w}] + \beta}{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C^{\setminus i}_{\cdot,k}] + W\beta}. \tag{3.64}$$

For the first part of (3.62), since the required expectations are at boundary where strong dependencies are broken, we extend some technical tricks in our first CVI algorithm. To compute for example $\mathbb{E}[C^{\setminus i-1,i}_{z_0^i,z_1^i}]$, we utilize the following equations that hold trivially,

$$q(\mathbf{z}^{\setminus i}) = q(\mathbf{z}^{i-1})q(\mathbf{z}^{\setminus i-1,i}), \tag{3.65}$$

$$q(\mathbf{z}^{i-1}) = q(z_T^{i-1})q((\mathbf{z}^{i-1})^{\setminus T}|z_T^{i-1}) \tag{3.66}$$

$$\triangleq q(z_0^i)q((\mathbf{z}^{i-1})^{\setminus T}|z_0^i) \quad \text{(by our renaming)}, \tag{3.67}$$

in which $(\mathbf{z}^{i-1})^{\setminus T}$ denotes the set of all variables in $\mathbf{z}^{i-1}$ with z_T^{i-1} removed. Hence, $\mathbf{z}^{i-1} = (\mathbf{z}^{i-1})^{\setminus T} \cup \{z_T^i\}$. Subsequently, (3.66) holds trivially by the product rule of probability.

We can compute $\mathbb{E}[C^{\setminus i-1,i}_{z_0^i,z_1^i}]$ as follows,

$$\begin{aligned}
\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C^{\setminus i-1,i}_{z_0^i,z_1^i}] &= \sum_{\mathbf{z}^{i-1}} q(\mathbf{z}^{i-1})\mathbb{E}_{q(\mathbf{z}^{\setminus i-1,i})}[C^{\setminus i-1,i}_{z_0^i,,z_1^i}] \quad \text{(by (3.65))} \\
&= \sum_{z_0^i} \sum_{(\mathbf{z}^{i-1})^{\setminus T}} q(z_0^i)q((\mathbf{z}^{i-1})^{\setminus T}|z_0^i)\mathbb{E}_{q(\mathbf{z}^{\setminus i-1,i})}[C^{\setminus i-1,i}_{z_0^i,z_1^i}] \quad \text{(by (3.67))} \\
&= \sum_{z_0^i} q(z_0^i)\mathbb{E}_{q(\mathbf{z}^{\setminus i-1,i})}[C^{\setminus i-1,i}_{z_0^i,z_1^i}] \underbrace{\left(\sum_{(\mathbf{z}^{i-1})^{\setminus T}} q((\mathbf{z}^{i-1})^{\setminus T}|z_0^i)\right)}_{=1} \\
&= \sum_{z_0^i} \underbrace{q(z_0^i)\mathbb{E}_{q(\mathbf{z}^{\setminus i-1,i})}[C^{\setminus i-1,i}_{z_0^i,z_1^i}]}_{\text{transition counts weighted by } q(z_0^i)}.
\end{aligned} \tag{3.68}$$

As with our first CVI algorithm, $\mathbb{E}[C^{\backslash i-1,i}_{z_0^i,z_1^i}]$ requires a sum over all possible states of z_0^i and indeed is a weighted average. Similarly, we can compute the expectation involving z_{T+1}^i in (3.62),

$$\mathbb{E}_{q(\mathbf{z}^{\backslash i})}[C^{\backslash i,i+1}_{z_T^i,z_{T+1}^i}] = \sum_{z_{T+1}^i} \underbrace{q(z_{T+1}^i)\mathbb{E}_{q(\mathbf{z}^{\backslash i,i+1})}[C^{\backslash i,i+1}_{z_T^i,z_{T+1}^i}]}_{\text{transition counts weighted by } q(z_{T+1}^i)}. \tag{3.69}$$

Plugging (3.68) and (3.69) into (3.62), we have $q(\mathbf{z}^i)$ that consists of standard expected transition and emission counts,

$$q(\mathbf{z}^i) \approx\propto \underbrace{\left(\sum_{z_0^i} q(z_0^i)\mathbb{E}[C^{\backslash i-1,i}_{z_0^i,z_1^i}] + \alpha\right)\frac{\sum_{z_{T+1}^i} q(z_{T+1}^i)\mathbb{E}[C^{\backslash i,i+1}_{z_T^i,z_{T+1}^i}] + \alpha}{\mathbb{E}[C^{\backslash i,i+1}_{z_T^i,\cdot}] + K\alpha}}_{\text{challenge: the summations at boundary}}$$
$$\times \underbrace{\prod_{t=2}^{T}\bar{\theta}_{z_{t-1}^i,z_t^i}\prod_{t=1}^{T}\bar{\phi}_{z_t^i,x_t^i}}_{\text{a standard HMM}}. \tag{3.70}$$

Given the above update, the only work left now is to design an algorithm that computes marginals $q(z_t^i)$ and $q(z_{t-1}^i, z_t^i)$ tractably.

3.6.1 A Sum Product Algorithm

Having expanded the boundary expectations (3.68-3.69), the resultant summations at the boundaries (the first part of (3.70)) give rise to another inference challenge: the available forward backward algorithm used in the EM algorithm can not be directly applied to (3.70) for calculating its marginals. The challenge is not present in the bag-of-words setting nor in the bag-of-clusters setting (our second CVI algorithm).

We propose our approach with the aims of making use of most the forward-backward implementation while computing the required marginals correctly. Our essential idea is to define an expanded distribution[2] over $\{z_0^i, \mathbf{z}^i, z_{T+1}^i\}$,

$$q_2(z_0^i, \mathbf{z}^i, z_{T+1}^i) \propto f_0^i(z_0^i)\left(\prod_{t=1}^{T+1} f_t^i(z_{t-1}^i, z_t^i)\right) f_{T+2}^i(z_{T+1}^i), \tag{3.71}$$

such that with carefully defined potential functions $\{f_t^i\}_{t=0}^{T+2}$, our interested posterior $q(\mathbf{z}^i)$ in (3.70) is equal to $\sum_{z_0^i,z_{T+1}^i} q_2(z_0^i, \mathbf{z}^i, z_{T+1}^i)$ in (3.71). Consequently, the required marginals

[2]We use q_2 to emphasize that our definition treats it as an unrelated distribution to q.

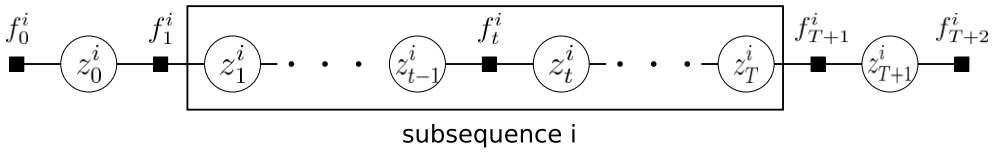

Figure 3.9: Factor graph depicting the explicit factorization in the expanded distribution q_2.

$q(z_{t-1}^i, z_t^i)$ and $q(z_t^i)$ are equal to $q_2(z_{t-1}^i, z_t^i)$ and $q_2(z_t^i)$, respectively. The reason for introducing the expanded distribution is that compared with (3.70), (3.71) does not contain complicated summations, and therefore we can design a propagation algorithm for computing its marginals that are equal to those of (3.70).

Formally, we employ a factor graph representation for $q_2(z_0^i, \mathbf{z}^i, z_{T+1}^i)$ shown in Figure 3.9. We define the potential functions as follow,

$$f_0^i(z_0^i) \triangleq q(z_0^i), \tag{3.72}$$

$$f_1^i(z_0^i, z_1^i) \triangleq \left(\mathbb{E}[C_{z_0^i, z_1^i}^{\backslash i-1, i}] + \frac{\alpha}{Kq(z_0^i)}\right)\bar{\phi}_{z_1^i, x_1^i}, \tag{3.73}$$

$$\text{for } t \in [2, T], \quad f_t^i(z_{t-1}^i, z_t^i) \triangleq \bar{\theta}_{z_{t-1}^i, z_t^i}\bar{\phi}_{z_t^i, x_t^i}, \tag{3.74}$$

$$f_{T+1}^i(z_T^i, z_{T+1}^i) \triangleq \frac{\mathbb{E}[C_{z_T^i, z_{T+1}^i}^{\backslash i, i+1}] + \frac{\alpha}{Kq(z_{T+1}^i)}}{\mathbb{E}[C_{z_T^i, \cdot}^{\backslash i, i+1}] + K\alpha}, \tag{3.75}$$

$$f_{T+2}^i(z_{T+1}^i) \triangleq q(z_{T+1}^i). \tag{3.76}$$

We can verify that with the above definitions, $\sum_{z_0^i, z_{T+1}^i} q_2(\mathbf{z}^i, z_0^i, z_{T+1}^i)/Z_2 = q(\mathbf{z}^i)/Z_1$, where Z_1 and Z_2 are their normalizers, respectively. To show $Z_1 = Z_2$, we sum over $\mathbf{z}^i$ on both sides, $\sum_{\mathbf{z}^i}\sum_{z_0^i, z_{T+1}^i} q_2(\mathbf{z}^i, z_0^i, z_{T+1}^i)/Z_2 = \sum_{\mathbf{z}^i} q(\mathbf{z}^i)/Z_1$. Since both q and q_2 are valid distributions, it follows $1/Z_2 = 1/Z_1$ and thus $Z_1 = Z_2$.

In Section 2.1, we have reviewed probabilistic graphical models and propagation algorithms. For any factor graph, we can design a sum product algorithm following the rules described in the subsection 2.1.1. In this specific example, we give details both for completeness and for comparison with the standard forward backward algorithm.

We first pick f_{T+2}^i as the root node and pass messages from the leaf node f_0^i, and then do it in the reverse direction. We present the messages from factor nodes to variable nodes only, and eliminate the messages from variable nodes to factor nodes, since variables nodes with only two neighbours perform no computations (Bishop, 2006, Chapter 8). Let $m_{f \to z}(z)$ denote the message from the factor node f to the variable node z, our sum product algorithm is constructed as follows.

The starting messages are given by,

$$m_{f_0^i \to z_0^i}(z_0^i) = f_0^i(z_0^i) \triangleq q(z_0^i), \tag{3.77}$$

$$m_{f_{T+2}^i \to z_{T+1}^i}(z_{T+1}^i) = f_{T+2}^i(z_{T+1}^i) \triangleq q(z_{T+1}^i). \tag{3.78}$$

The message recursions, similar to the forward backward algorithm are given by,

for $t = [1, T+1]$,

$$m_{f_t^i \to z_t^i}(z_t^i) = \sum_{z_{t-1}^i} m_{f_{t-1}^i \to z_{t-1}^i}(z_{t-1}^i) f_t^i(z_{t-1}^i, z_t^i), \tag{3.79}$$

for $t = [T, 0]$,

$$m_{f_{t+1}^i \to z_t^i}(z_t^i) = \sum_{z_{t+1}^i} m_{f_{t+2}^i \to z_{t+1}^i}(z_{t+1}^i) f_{t+1}^i(z_t^i, z_{t+1}^i). \tag{3.80}$$

After message passing in both directions, the required marginals are given by,

$$q_2(z_t^i) \propto m_{f_t^i \to z_t^i}(z_t^i) m_{f_{t+1}^i \to z_t^i}(z_t^i), \tag{3.81}$$

$$q_2(z_{t-1}^i, z_t^i) \propto f_t^i(z_{t-1}^i, z_t^i) m_{f_{t-1}^i \to z_{t-1}^i}(z_{t-1}^i) m_{f_{t+1}^i \to z_t^i}(z_t^i), \tag{3.82}$$

in which the normalization constant can be obtained by normalizing any of them.

Compared with the standard forward backward propagation, our sum product algorithm can be considered as an expanded version: there are additional messages on each side of a subsequence. In particular, we have made the standard propagation compatible with these additional messages and thus the propagation is expanded smoothly. Consequently, we can make use of most forward backward implementation (for $t \in [2, T]$) and only make a minor change at the subsequence boundaries.

In (3.77-3.78), we see that the initial messages for updating the i^{th} subsequence are the boundary marginals of its adjacent subsequences, since by our renaming $q(z_0^i)$ and $q(z_{T+1}^i)$ are actually the variational posteriors over z_T^{i-1} and z_1^{i+1}, respectively. After updating $q(\mathbf{z}^i)$, its boundary marginals become initial messages for updating its adjacent subsequences. In other words, the messages in our sum product algorithm are not locally restricted, but passed around in the whole sequence by adjacent subsequence communications. Therefore, our third CVI algorithm is different from naively breaking one long sequence into smaller pieces and ignoring their correlations, but provides the first message passing algorithm for graphical models in collapsed space.

3.6.2 Implications and Related Work

As we have reviewed via the HMM example in Section 3.3, a propagation algorithm for hidden variable posterior inference in EM can be directly reused in VI, by inverting expected natural parameters. In this subsection, we propose an analogous algorithmic shortcut from EM to CVI[3], following a systematic approach of reusing/redesigning propagation algorithms of EM, and then we discuss some related work.

In the frameworks of both EM and VI, we iteratively update the hidden variables and model parameters in turn (by either their values or their posterior distributions). In the CVI framework, unfortunately, the model parameters are integrated out and thus not available for iterating. To form an iterative method, it is logical to consider partitioning the set of all hidden variables into smaller subsets, namely clusters. If these clusters are disconnected in the original model (e.g., HMMs in the multiple sequence setting), then computing their posteriors is quite straightforward. We can express a cluster's posterior in terms of the surrogate parameters based on other clusters' statistics. Next, we run the existing propagation algorithm of EM under the surrogate parameters for obtaining the required marginals.

If the clusters are connected in the original model (e.g., HMMs in the single long sequence setting), then computing their posteriors requires additional considerations. Stemming from the standard structured mean field equation, the update for a cluster's posterior requires summing over its Markov Blanket (or integration for continuous hidden variables and Dirac delta for observed variables), and thus the existing propagation algorithm in EM is inapplicable. In this case, our propagation redesign approach breaks down into three steps: expand a cluster to include its MB, define the potential functions for the expanded distribution approximately such that summing over the MB reduces to the original distribution, and finally make modifications to the original propagation at the boundaries.

The summing over MB in our approach is reminiscent of the 'expected Markov Blanket' in the generalized mean field algorithm (Xing et al., 2003). Here our approach is specialized to collapsed space and primarily focuses on making local propagation compatible with incoming messages. An alternative way of deriving local propagation in a large graph was proposed by Foti et al. (2014) in the stochastic variational inference for HMMs, which we will compare. Their method propagates messages from a manually selected set of observed

[3]Our claim is restricted to the cases in which the model parameters are marginalized out and the conjugate analysis is available. We do not claim to have a method for marginalizing arbitrary variables.

variables until the MB of a targeted cluster is reached; then use those MB marginals as the initial messages. Hence, our approach is more mathematically principled and much more computationally efficient. On the other hand, there is extra memory cost of our approach: the MB marginals for each cluster (not each variable) need to be kept in memory.

3.7 CVI for the CE Family

In the previous sections (3.4-3.6), we have presented the novel aspects relating to handling the sequential dependencies. In this section we make contributions to collapsed variational inference in the conjugate exponential family, building on the work of Sung et al. (2008). In particular, the algorithm proposed by Sung et al. (2008) does not consider updating multiple variables nor updating statistics stochastically. Here we extend collapsed variational inference in these two directions.

To be concrete we continue working with our HMM example in the multiple sequence setting and focus on the emission part. This time we assume that the emission distributions are in the general exponential family and there is a conjugate prior. We use the definitions in Section 2.3. The emission distribution at the hidden state k over a possible observation $\mathbf{w}$ given the natural parameters η_k is defined by,

$$p(\mathbf{w}|\eta_k) = h_l(\mathbf{w}) \exp\{\eta_k^T s(\mathbf{w}) - a_l(\eta_k)\}, \tag{3.83}$$

and its conjugate prior parametrized by λ is defined by,

$$p(\eta_k|\lambda) = h_g(\eta_k) \exp\{\lambda_1^T \eta_k + \lambda_2^T(-a_l(\eta_k)) - a_g(\lambda)\}. \tag{3.84}$$

We also alter the definitions and notations relating to the emission part while keeping those relating to the transition part the same as in our second CVI algorithm. As before, we denote the set of all observed sequences by $\mathbf{x} = \{\mathbf{x}^i\}_{i=1}^T$ and the i^{th} sequence by $\mathbf{x}^i = \{\mathbf{x}_t^i\}_{t=1}^T$. Here each observation $\mathbf{x}_t^i \sim p(\cdot|\eta_{z_t^i})$ defined in (3.83) and is written in boldface, since it can be a vector. The counting functions $C_{k,w}$ (number of observation w at hidden state k) used in the Discrete-Dirichlet case are replaced by the general notions of sufficient statistics. We define $\lambda_k = \langle(\lambda_k)_1, (\lambda_k)_2\rangle$ to be the posterior hyperparameters (prior hyperparameters plus sufficient statistics from all sequences) at hidden state k as

follows,

$$(\lambda_k)_1 = \lambda_1 + \sum_{i=1}^{I}\sum_{t=1}^{T}\delta(z_t^i = k)s(\mathbf{x}_t^i), \tag{3.85}$$

$$(\lambda_k)_2 = \lambda_2 + \sum_{i=1}^{I}\sum_{t=1}^{T}\delta(z_t^i = k), \tag{3.86}$$

and $\lambda_k^{\setminus i}$ denotes the exclusion of the contribution from the i^{th} sequence.

In the following subsections, we focus on presenting collapsed variational inference for the emission part. This involves deriving CVI for multiple hidden variables and stochastic CVI in the general CE family setting.

3.7.1 CVI for Multiple Variables

As with our second CVI algorithm, we assume the variational distribution $q(\mathbf{z}) = \prod_{i=1}^{I} q(\mathbf{z}^i)$ and optimize $\mathcal{L}^{\text{CVI}}$. The update equation for $q(\mathbf{z}^i)$ is given by,

$$q(\mathbf{z}^i) \propto \exp\{\mathbb{E}[\log p(\mathbf{z}^i|\mathbf{z}^{\setminus i})]\} \times \exp\{\mathbb{E}[\log p(\mathbf{x}^i|\mathbf{x}^{\setminus i}, \mathbf{z})]\}, \tag{3.87}$$

in which the first term in RHS of (3.87) is well studied in Section 3.5, and here we solve the second term. By the conjugacy property 4 in Section 2.3, we derive the joint posterior predictive with some algebraic manipulation,

$$\begin{aligned} p(\mathbf{x}^i|\mathbf{x}^{\setminus i}, \mathbf{z}) \propto \prod_{t=1}^{T} h_l(\mathbf{x}_t^i) \exp\Big\{ & a_g\big(\lambda_{z_t^i}^{\setminus i} + \langle \textstyle\sum_{t'=1}^{t}\delta(z_{t'}^i = z_t^i)s(\mathbf{x}_{t'}^i), \sum_{t'=1}^{t}\delta(z_{t'}^i = z_t^i)\rangle\big) \\ & - a_g\big(\lambda_{z_t^i}^{\setminus i} + \langle \textstyle\sum_{t'=1}^{t-1}\delta(z_{t'}^i = z_t^i)s(\mathbf{x}_{t'}^i), \sum_{t'=1}^{t-1}\delta(z_{t'}^i = z_t^i)\rangle\big)\Big\}. \end{aligned} \tag{3.88}$$

By our approximation method to joint posterior predictives (3.55) in Subsection 3.5.1, we remove the small contribution from $\{\mathbf{x}_{t'}^i, \mathbf{z}_{t'}^i\}_{t'=1}^{t-1}$ in both log normalizers in (3.88),

$$p(\mathbf{x}^i|\mathbf{x}^{\setminus i}, \mathbf{z}) \approx \prod_{t=1}^{T} h_l(x_t^i) \exp\big\{a_g\big(\lambda_{z_t^i}^{\setminus i} + \langle s(\mathbf{x}_t^i), 1\rangle\big) - a_g\big(\lambda_{z_t^i}^{\setminus i}\big)\big\}, \tag{3.89}$$

which is then plugged into the second term in RHS of (3.87), yielding,

$$\exp\{\mathbb{E}[\log p(\mathbf{x}^i|\mathbf{x}^{\setminus i}, \mathbf{z})]\} \approx \prod_{t=1}^{T} h_l(\mathbf{x}_t^i) \exp\big\{\mathbb{E}\big[a_g\big(\lambda_{z_t^i}^{\setminus i} + \langle s(\mathbf{x}_t^i), 1\rangle\big)\big] - \mathbb{E}\big[a_g\big(\lambda_{z_t^i}^{\setminus i}\big)\big]\big\}. \tag{3.90}$$

The expected log normalizers are intractable to compute. We follow Sung et al. (2008) who applies the first order Taylor approximation to complete-data marginal. Here we apply the Taylor approximation to both log normalizers in (3.90),

$$a_g\big(\lambda^{\setminus i}_{z^i_t} + \langle s(\mathbf{x}^i_t), 1\rangle\big) \approx a_g\big(\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[\lambda^{\setminus i}_{z^i_t}] + \langle s(\mathbf{x}^i_t), 1\rangle\big) \tag{3.91}$$

$$+ (\lambda^{\setminus i}_{z^i_t} - \mathbb{E}_{q(\mathbf{z}^{\setminus i})}[\lambda^{\setminus i}_{z^i_t}]) a'_g\big(\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[\lambda^{\setminus i}_{z^i_t}] + \langle s(\mathbf{x}^i_t), 1\rangle\big), \tag{3.92}$$

$$a_g\big(\lambda^{\setminus i}_{z^i_t}\big) \approx a_g\big(\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[\lambda^{\setminus i}_{z^i_t}]\big) \tag{3.93}$$

$$+ (\lambda^{\setminus i}_{z^i_t} - \mathbb{E}_{q(\mathbf{z}^{\setminus i})}[\lambda^{\setminus i}_{z^i_t}]) a'_g\big(\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[\lambda^{\setminus i}_{z^i_t}]\big), \tag{3.94}$$

and plug the results into (3.90). Note that the first order terms of Taylor expansions, (3.92) and (3.94) become zero. Altogether, the update for $q(\mathbf{z}^i)$ is,

$$q(\mathbf{z}^i) \approx\propto \exp\{\mathbb{E}[\log p(\mathbf{z}^i|\mathbf{z}^{\setminus i})]\} \prod_{t=1}^{T} \bar{\eta}_{z^i_t, \mathbf{x}^i_t}, \tag{3.95}$$

in which we define,

$$\bar{\eta}_{k,\mathbf{w}} = h_l(\mathbf{w}) \exp\{a_g(\mathbb{E}[\lambda^{\setminus i}_k] + \langle s(\mathbf{w}), 1\rangle) - a_g(\mathbb{E}\big[\lambda^{\setminus i}_k\big])\}. \tag{3.96}$$

Implementing (3.96) is easy: it is a predictive distribution over $\mathbf{w}$ given expected sufficient statistics collected from all other sequences; and this distribution is expected to be known for any conjugate distribution pair. To list a few examples, in the Discrete-Dirichlet case, (3.96) is the same as (3.64); in the Gaussian-GaussianWishart case, (3.96) is the student distribution; in the Poisson-Gamma case, (3.96) is the negative binomial distribution.

In Sung et al. (2008)'s algorithm, instead of predictives or joint predictives, complete-data marginals are plugged into the mean field update,

$$q(\mathbf{z}^i) \approx \exp\{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[\log p(\mathbf{x}, \mathbf{z})]\}, \tag{3.97}$$

and then a Taylor approximation is employed, yielding an abstract update formula. However, the abstract formula is not implementable. For updating a single hidden variable, in the Mixture of Gaussian example (Sung et al., 2008, Section 4.1), their implementable update is the same posterior predictive as by our approach. For updating multiple hidden variables, they consider each data point to contribute one statistic to global statistics, no matter how complex a data point is. This bypasses the implementation details that involve complicated joint predictives, for which we propose a solution using our approximation method.

3.7.2 Stochastic CVI

In the previous subsection, we have presented a batch CVI algorithm for HMMs whose emission distributions are in the exponential family. One disadvantage of this algorithm is its memory requirement: we need to keep sufficient statistics for each sequence so that they can be subtracted or added when required. For example, $\lambda_k^{\setminus i}$ can be efficiently obtained by subtracting statistics of $\{\mathbf{x}^i, \mathbf{z}^i\}$ from λ_k. To circumvent this requirement, in this subsection we extend the batch CVI algorithm to the stochastic setting, deriving stochastic CVI for the CE family. Stochastic optimization is also useful for scaling batch algorithms to large datasets (Robbins and Monro, 1951).

Overall our algorithm for HMMs is a generalization of the SCVI algorithm for LDA at the variable level (Foulds et al., 2013) to the cluster level. Given $\mathbf{x}$, our algorithm examines one sequence $\mathbf{x}^i$ at a time to learn the local variational posterior $q(\mathbf{z}^i)$, and then it stochastically updates the transition counts and emission statistics. We first present the algorithm updates and then discuss its optimization objectives and convergence.

In the batch setting, we have derived the local update equation for $q(\mathbf{z}^i)$ in (3.95). In the stochastic setting, we do not keep local statistics, and thus we do not have access to $\mathbb{E}[C_{j,k}^{\setminus i}]$ and $\mathbb{E}[\lambda_k^{\setminus i}]$. Following Foulds et al. (2013), in (3.95) we replace $\mathbb{E}[C_{j,k}^{\setminus i}]$ with the stochastic approximation for the expected transition count, denoted by $N_{j,k}$. For the emission part, we propose to replace $\mathbb{E}[\lambda_k^{\setminus i}]$ with the stochastic approximation for the emission statistics, denoted by M_k. Altogether the local update of $q(\mathbf{z}^i)$ in the stochastic setting is,

$$q(\mathbf{z}^i) \approx\propto \prod_{t=1}^{T} \hat{\theta}_{z_{t-1}^i, z_t^i} \underbrace{\prod_{t=1}^{T} \hat{\eta}_{z_t^i, \mathbf{x}_t^i}}_{\text{emission part}}, \tag{3.98}$$

in which,

$$\hat{\theta}_{j,k} = \frac{N_{j,k} + \alpha}{N_{j,\cdot} + K\alpha} \tag{3.99}$$

$$\hat{\eta}_{k,\mathbf{w}} = h_l(\mathbf{w}) \exp\{a_g(M_k + \langle s(\mathbf{w}), 1\rangle) - a_g(M_k)\}. \tag{3.100}$$

Given (3.98), we compute its marginals $q(z_{t-1}^i, z_t^i)$ and $q(z_t^i)$ by the standard forward backward algorithm. Then we collect its local expected transition counts $\mathbb{E}[C_{j,k}^i]$ and emis-

sion statistics $\mathbb{E}[\lambda_k^i]$ defined as follows,

$$\mathbb{E}[C_{j,k}^i] = \sum_{t=1}^{T} q(z_{t-1}^i = j, z_t^i = k), \tag{3.101}$$

$$\mathbb{E}[\lambda_k^i] = \big\langle \sum_{t=1}^{T} q(z_t^i = k) s(\mathbf{x}_t^i), \sum_{t=1}^{T} q(z_t^i = k) \big\rangle. \tag{3.102}$$

Since $I\mathbb{E}[C_{j,k}^i]$ and $I\mathbb{E}[\lambda_k^i]$ are the stochastic estimates of $\mathbb{E}[C_{j,k}]$ and $\mathbb{E}[\lambda_k]$, respectively, we can update their stochastic approximations, $N_{j,k}$ and M_k by online averaging the new estimates and their previous values (Robbins and Monro, 1951),

$$N_{j,k} = (1 - \rho_i) N_{j,k} + \rho_i I\mathbb{E}[C_{j,k}^i], \tag{3.103}$$

$$M_k = (1 - \rho_i) M_k + \rho_i I\mathbb{E}[\lambda_k^i], \tag{3.104}$$

in which ρ_i is the step size satisfying $\sum_{i=1}^{\infty} \rho_i = \infty$ and $\sum_{i=1}^{\infty} \rho_i^2 < \infty$.

In our SCVI algorithm, if the emission distribution is discrete, then we can follow the same analysis in Foulds et al. (2013) to prove its convergence. The proof is based on the fact that SCVI is equivalent to online EM (Cappé and Moulines, 2007) with MAP estimation and incrementing Dirichlet hyperparameters by one. Nevertheless, this interpretation does not allow for encoding sparsity and can not explain other distributions.

For the general CE family case, we do not have a rigorous proof for its convergence, as with some other adaptations of variational inference. In particular, our algorithm is very similar to the locally stochastic collapsed variational inference (LSCVI) by Wang and Blei (2012) except that they drew samples to form an empirical estimation of $q(\mathbf{z}^i)$. They argue that LSCVI approximately optimizes the objective of expectation propagation algorithm (Minka, 2009), which may not converge. Sato et al. (2016) derived the same SCVI algorithm for the CE family as ours under a different framework. They interpreted this algorithm as an approximate solution to the predictive distribution (PD) optimization, arguing that SCVI can make better predictions about what is yet to be observed than SVI. Both work investigates the optimization objectives and thus sheds light on SCVI's convergence. Orthogonally, our thesis does not focus on theoretical analysis but on algorithmic contributions that allow CVI and SCVI to be generally applicable to a wide variety of graphical models in the conjugate exponential family.

3.8 Experiments

In this section, we empirically evaluate our proposed (stochastic) collapsed variational inference algorithms for HMMs. In Subsections 3.8.1 and 3.8.2, we compare our batch CVI algorithms with EM, VI and CGS on the task of Part-of-Speech induction, both in the semi-supervised and unsupervised settings, using a number of evaluation metrics. In Subsection 3.8.3, we apply our first CVI algorithm to the word alignment task. The material in Subsection 3.8.3 is mostly contributed by my co-author[4] and is included to provide another application of CVI. In Subsection 3.8.4, we compare our stochastic CVI algorithms with SVI (Hoffman et al., 2013) on the large scale textual prediction task. In particular, in the single long sequence setting, we compare our third algorithm that uses a propagation redesign with the SVI algorithm for HMMs that uses a buffering scheme (Foti et al., 2014).

3.8.1 Semi-Supervised PoS Induction

In this subsection and next, we evaluate our CVI algorithms on the task of learning syntactic categories for words in text, namely the Part-of-Speech induction. Our dataset is the Wall Street Journal (WSJ) treebank (Marcus et al., 1993), which consists of $49,208$ sentences with the average length roughly 22. Each token in WSJ has been assigned one of the 45 PoS tags based on the linguistic domain knowledge, and these tags are often called reference tags or gold tags. One important evaluation metric is to compute the accuracy of the hidden states produced by an inference algorithm against those reference tags.

We compare our first and second CVI algorithms with the EM, VI and CGS algorithms for HMMs. They can be considered as the two special cases of our third CVI algorithm (one breaks all strong dependencies and the other breaks none in the multiple sentence setting here). We leave the explicit evaluation of our third algorithm to the single long sequence setting in Subsection 3.8.4 when we compare it with a recent benchmark (Foti et al., 2014). Also we experiment with simple bi-gram taggers with the aim of understanding the properties of our proposed inference algorithms, rather than building a state-of-the-art (higher-gram) tagger (e.g. Berg-Kirkpatrick et al. (2010); Blunsom and Cohn (2011)). Finally, we conduct our experiments both in the semi-supervised and unsupervised settings.

[4]The technical manuscript is entitled 'Collapsed Variational Bayes for Inference in Word Alignment Models', in which Yarin Gal was the first author. My contributions were proposing CVI for the IBM word alignment model 1 (which is a simpler case of the HMM) and writing the algorithm section.

		EM		VB		Algorithm 1		Algorithm 2		CGS	
Size	Random	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
1K	65.1	81.0	1.2	79.2	1.3	82.9	0.2	84.2	0.2	**85.3**	0.5
2K	65.2	81.1	0.9	80.5	1.1	83.1	0.3	**85.5**	0.3	85.0	0.3
3K	65.1	81.1	0.8	80.5	1.0	83.1	0.2	**85.8**	0.3	85.0	0.2
5K	64.9	81.0	1.5	80.4	1.5	83.0	0.2	**85.6**	0.1	85.2	0.2
10K	64.7	81.4	1.7	80.7	1.2	83.4	0.2	**85.6**	0.1	85.0	0.2
All	64.8	81.4	0.9	81.4	1.1	83.7	0.1	**85.7**	0.1	84.6	0.1

Figure 3.10: The tagging accuracies and standard deviations of 10 random runs on various corpus sizes with a complete tag dictionary.

In the semi-supervised setting, we adopt the formulation of Merialdo (1994): given a raw corpus and a tag dictionary that defines legal syntactic categories for each word type, tag each token in the corpus with the goal of maximizing accuracy against a reference tagged corpus. The tag dictionary is constructed by collecting all the tags found for each word type in the entire WSJ. We conduct experiments for different corpus sizes, from 1K sentences to the entire treebank. For all the corpora, the percentages of ambiguous tokens is roughly 55% and the average number of tags per token in the dictionary is approximately 2.3. Later we gradually relax the tag dictionary constraints with increasing ambiguity towards the unsupervised PoS induction.

We conduct our experiments using the following experimental settings. We run 50 iterations for all the variational algorithms including EM, which is included to serve as a benchmark although it optimizes the likelihoods nor marginals. We run $20,000$ iterations for CGS with the annealing scheme designed by Goldwater and Griffiths (2007) (temperature incrementally decreased from 2.0 to 0.08). For each algorithm, we run it 10 times with random initializations[5], and the hyperparameters α and $\beta \in [0.003, 0.01, 0.03, 0.1, 0.3, 1.0]$ are optimized on held-out data. Finally, to compute tagging accuracies, we produce hidden states using the maximum posterior decoding, since we find that it almost always yields slightly better results than the Viterbi decoding for all algorithms.

Figure 3.10 presents the tagging accuracies and standard derivations achieved by each of the algorithms on the various corpora. Both of our first and second CVI algorithms (de-

[5]We thought that it was natural for EM and VI to randomly initialize model parameters. For CVI and CGS, which integrate out model parameters, we had no choices but initialize local posteriors, $q(z_t^i|z_{t-1}^i) \propto \mathcal{U}[0,1]$ and $z_t^i \sim \mathcal{U}[0,1]$, respectively. We suspect that randomizing initial parameters might cause higher variances as reported in Figures 3.10 and 3.12. In the remaining implementation, for 'absolutely fair' comparisons (or at least with same starting points), we initialize local posteriors for all the deterministic algorithms.

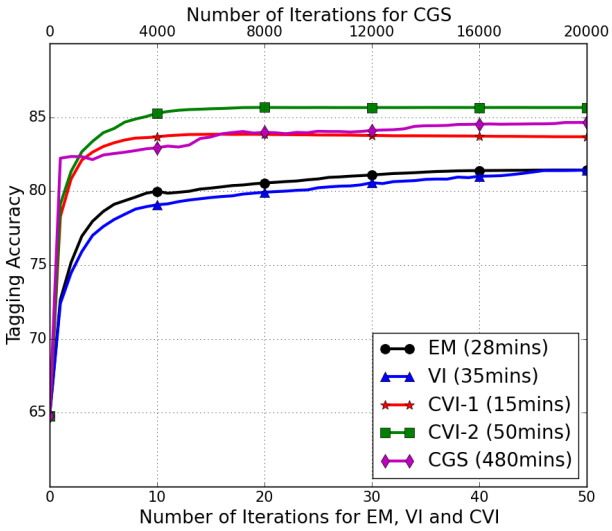

Figure 3.11: The average tagging accuracy over 10 runs versus the number of iterations for the entire treebank with a complete tag dictionary. All the variational algorithms were implemented in Python, and the CGS algorithm were implemented in C++. All the algorithms were run on an Intel Core i5 3.10GHZ computer.

noted by 'CVI-1' and 'CVI-2') outperform the standard VI algorithm, which in turn does not seem to have an advantage over EM. The first CVI algorithm surpasses VI by 1-3%, suggesting that the seemingly strong hidden variable independence assumption proves to be weaker than the assumptions in standard VI. Furthermore, we think given the highly peaked posterior, products of marginals might be good approximations to joint distributions. Our second CVI algorithm that takes advantages of collapsing without breaking the sequential dependencies yields the best results in most cases except for the 1K subset.

Figure 3.11 shows the convergence rates for all the algorithms. The variational algorithms converge after approximately 20 iterations, whereas the accuracy of CGS is near its maximum with 10K iterations. All the variational algorithms have the same computational complexity ($O(ITK^2)$), but their time usage varies in practice. The most efficient method is our first CVI algorithm, not EM, since EM is normally implemented in log space to avoid underflow[6]. VI requires to compute expensive digamma functions, while our second CVI algorithm calculates dynamic parameters; hence both are slower than EM. Finally, CGS

[6]In the semi-supervised setting, we implemented the deterministic algorithms in Python, used log space in computing the forward and backward probabilities to avoid underflow, and used dictionaries as main data structures. All these simple choices make our implementation quite slow, and as a result, we had to implement CGS in C++. In the remaining experiments in the unsupervised setting, we implemented all the algorithms in Cython, used rescaling factors to avoid underflow, and only used arrays as main data structures. These more sophisticated choices can give more than 100 times speed up for some algorithms.

			EM		VB		CVI-1		CVI-2		CGS	
d	tags/token	% ambig.	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
1	2.3	54	81.0	1.2	79.2	1.3	82.9	0.2	84.2	0.2	**85.3**	0.5
2	7.8	63	69.6	1.5	69.6	1.6	73.8	0.4	77.9	0.4	**79.0**	0.4
3	10.8	66	64.7	1.7	61.1	1.6	65.1	0.4	**74.1**	0.3	72.5	0.7
5	14.9	71	54.7	2.9	54.3	2.0	64.7	0.4	**65.0**	0.5	64.2	0.9
10	20.8	77	43.5	2.4	43.6	2.1	51.2	0.7	53.3	0.5	**55.7**	0.6

Figure 3.12: The tagging accuracies and standard deviations of 10 random runs on the 1K dataset with various incomplete dictionaries depending on values of d.

takes an order of magnitude more time than any of the deterministic algorithms. When CGS mixes quickly we would expect its performance to exceed all the other algorithms. From our experiments we see that this is the case for the smallest dataset, but as the data size increases CGS performs poorly. We see that the accuracies of our collapsed variational algorithms increase with the size of the data, countering the decreasing trend for CGS.

In practice, it is not always possible to build a complete tag dictionary, especially for the infrequent words. We investigate the effects of reducing dictionary information. Following Smith and Eisner (2005), we use the 1K dataset with various tag dictionaries depending on the value of d. Specifically, we define a word type to be frequent if the word's tokens appear at least d times in the training corpus, otherwise it is infrequent. For frequent word types the standard tag dictionary is available; whereas for infrequent word types, all the 45 tags are considered to be legal. Figure 3.12 presents the accuracies achieved by all the algorithms at various ambiguity levels. Because of the small data set, the collapsed Gibbs sampler performs best in most cases, although somewhat surprisingly in some cases the second CVI algorithm outperforms CGS even in this small corpus. We also find that with more ambiguity, the margins between the uncollapsed algorithms and collapsed algorithms increase. In particular, when $d = 3$, the margin is as large as 13%.

3.8.2 Unsupervised PoS Induction

In the unsupervised setting, without a tag dictionary, the hidden states are interchangeable and we have a label identifiability issue. In general, there are two kinds of methods to deal with this issue: one tries to identify mappings between hidden states and gold tags before evaluation; whereas the other uses information theoretic measures for calculating clustering distances. We first briefly describe four evaluation methods (two for each kind), and then

we present our experiment settings and evaluation results.

Many-to-one is probably the most commonly used evaluation method. It maps each hidden state to the gold tag it co-occurs most frequently and computes the tagging accuracy by that mapping. This method favours larger numbers of hidden states: in the extreme case, each token has its unique state, yielding 100% accuracy. This can be partially mitigated by cross validation: we identify a mapping in one half of the dataset and compute accuracy in the second half. Another approach for defining a mapping is called greedy one-to-one: it constrains the mapping so that at most one hidden state is mapped to any gold tag, by greedily assigning hidden states to tags until either is used up. We use the greedy one-to-one (denoted by '1-1') and cross validation many to one (denoted by 'M-1') in this experiment.

We also use entropy based measures for evaluation, which consider not only the majority mapped tag in each cluster, but also the reminders. Variance of information (denoted by 'V-I') is a popular clustering evaluation metric. It assesses the homogeneity and completeness of clusterings using the conditional entropies, $H(G|S)$ (entropy of gold tag distribution conditioned on state clustering), and $H(S|G)$ (entropy of state clustering conditioned on gold tag distribution), respectively. This evaluation favours smaller numbers of hidden states: in the extreme case, all tokens having the same state does disturbingly well (Johnson, 2007). This can be partially fixed by reweighing the conditional entropies, leading to the V-measure (denoted by 'V-M') (Rosenberg and Hirschberg, 2007), which is analogous to the standard F measure and is defined by,

$$\text{V-M} = \frac{2hc}{h+c} \qquad h = 1 - \frac{H(G|S)}{H(G)} \qquad c = 1 - \frac{H(S|G)}{H(S)}. \tag{3.105}$$

We conduct our experiments using the same experimental settings as in previous semi-supervised task, except for the followings: we run all the deterministic algorithms for 200 iterations; and the grid search of (α, β) is conducted w.r.t. each evaluation metric and each algorithm on held-out data. In addition, we find that occasionally, in some hyperparameter settings, some algorithms produce clusterings with less than five hidden states, which give very good V-I scores, but are poor by other metrics. Since these taggers should be considered as poor taggers in general, we exclude those settings for the corresponding algorithms.

Figure 3.13 presents the evaluations of all the algorithms by four metrics on two datasets. On the 1K sentence dataset, we see that our first CVI algorithm outperforms EM and VI by all the evaluation metrics. Hence, it seems that on a small dataset, the benefits of collapsing

		EM		VI		CVI-1		CVI-2		CGS	
Size	Metric	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
1K	1-1	26.3	2.4	24.9	2.8	30.4	2.7	**32.5**	2.8	30.3	2.0
1K	M-1	38.9	2.2	33.6	1.9	48.1	2.6	50.5	2.2	**51.4**	1.5
1K	V-I	6.55	.09	6.86	.06	5.46	.08	5.32	.08	**5.16**	.06
1K	V-M	31.0	1.0	27.9	0.6	42.8	0.9	44.4	0.8	**45.4**	0.5
All	1-1	37.6	2.1	42.0	1.9	39.5	2.0	**46.2**	2.6	41.3	1.9
All	M-1	59.7	2.2	56.4	1.9	59.2	0.8	63.5	1.2	**63.7**	0.7
All	V-I	4.65	.08	4.82	.07	4.31	.04	**4.10**	.06	4.19	.07
All	V-M	51.4	0.8	49.5	0.9	53.3	0.3	**56.7**	0.6	56.4	0.6
All	Time	95 mins		98 mins		86 mins		101 mins		565 mins	

Figure 3.13: Evaluation of algorithms on the 1K dataset and on the entire treebank, by four metrics in the unsupervised setting. For variation of information (V-I), lower score is better. The averages and standard deviations are over 10 random runs.

outweigh the benefits of modelling sequential dependencies perfectly. Our second CVI algorithm performs better than the first one, as expected; and its performance is comparable with CGS. In the entire WSJ, we see that our second CVI algorithm performs the best by three evaluation metrics except for many-to-one accuracy. In particular, it surpasses CGS by 5% in one-to-one accuracy. Our first CVI algorithm is still advantageous over EM and VI by the entropy based metrics. By our implementation of all algorithms in Cython and using rescaling factors to avoid numerical underflows, all the deterministic algorithms take between $85-105$ minutes, while CGS consumes about five times more training time.

3.8.3 Word Alignment

In this subsection, we apply our first CVI algorithm to the task of word alignment, which is often an important early step in training a statistical machine translation system. The traditional word alignment pipeline is made of the IBM model 1 (Brown et al., 1990), the HMM (Vogel et al., 1996), and the IBM models 3-6 reviewed by Och and Ney (2003). Each model's parameters are used for bootstrapping the parameters of the model following it, with the IBM model 1 being the starting point. Given a parallel corpora consisting of the source and target sentences, the IBM model 1 only takes into account the frequencies of occurrences for the source-target word pairs in those sentences. Thus it is a simpler case of the HMM without the alignment orders (i.e., transition information). Here we apply our first CVI algorithm to the IBM model 1 and HMM. Alternatively, a recent reparametrization

Metric	EM	VI	CVI-1	CGS	Model 4
BLEU (ZH-EN)	27.6	28.8	**29.2**	**29.2**	28.5
BLEU (EN-ZH)	14.2	14.2	14.3	**14.6**	14.6
AER	36.4	36.8	35.7	**34.6**	34.5

Figure 3.14: Evaluation of different algorithm trained models for word alignment on the Chinese FBIS data. Top row: BLEU scores of translating from Chinese to English. Middle row: BLEU scores of translating from English to Chinese. Bottom row: alignment qualities (AER). For references, the last column shows the results produced by the IBM model 4.

of the IBM Model 2 (Dyer et al., 2013) provides an efficient and strong benchmark for word alignment, although we have not explored this direction yet.

We compare our first CVI algorithm trained IBM model 1 and HMM (denoted by 'CVI-1') with the following benchmarks: the pipelined EM trained IBM model 1 and HMM as provided by Giza++ (Och and Ney, 2003) (denoted by 'EM'), the VI trained IBM model 1 and HMM (Riley and Gildea, 2012) (denoted by 'VI'), and the CGS trained hierarchical Pitman-Yor process (HPYP) IBM model 1 and HMM provided by Giza# (Gal and Blunsom, 2013) (denoted by 'CGS'). Note that the last benchmark uses the HPYP as the prior distribution and also models the phrasal dependencies, and thus CGS is essentially carried out on much more sophisticated models.

We conduct our experiments using the following experimental settings. Our dataset is the Chinese FBIS corpus that consists of paralleled Chinese and English sentences. MT02 was used for tuning and MT03 was used for evaluation. We run the EM trained Giza++ for 5 iterations as suggested by Och and Ney (2003) and the same is done for VI. On the validation data, we find that our CVI algorithm takes as little as 3 iterations to yield good results, and thus we choose 3 iterations. We run CGS for 50 iterations as suggested by Gal and Blunsom (2013). For the Dirichlet priors, we set the hyper-parameters to 0.1 in order to encourage sparsity. Finally, we use Moses (Koehn et al., 2007) for training the translation systems. We use the BLEU scores (Papineni et al., 2002) and alignment error rates (AER) as our evaluation metrics; and the results are reported by averaging 5 random runs.

In Figure 3.14, we can see that our CVI-1 trained model achieves higher BLEU scores than both the Giza++ and VI trained models. In particular, when translating from Chinese to English, it surpasses the Giza++ trained benchmark by 1.6 BLEU points and the IBM model 4 by 0.7 BLEU points; furthermore, the comparable results with the CGS trained HPYP IBM model are achieved by using only a fraction of its running time: CVI-1 takes

about 1 hour compared to the 11.5 hours for CGS. For both the BLEU scores of translating from English to Chinese and AER, the results also favour towards our CVI-1 trained model, with small improvements over the Giza++ and VI trained models. Given the performance patterns of all algorithms in the PoS induction task, one might expect that applying our second CVI algorithm can lead to more improvements. With the promising initial experiment results, we leave it as a future work to fully explore our CVI algorithms for the remaining IBM models and the reparametrized variants (Dyer et al., 2013).

3.8.4 Text Prediction

In this subsection, we evaluate our stochastic CVI algorithms on the text prediction tasks especially on large scales, by calculating the predictive likelihoods of unseen text. We first compare the stochastic version of our second CVI algorithm (denoted by 'SCVI-2') with the standard SVI algorithm (Hoffman et al., 2013) in the bag-of-sentences setting. Next, we compare the stochastic version of our third CVI algorithm (denoted by 'SCVI-3') with SVI for HMMs (Foti et al., 2014) in the long single sequence setting.

We create two datasets in addition to the standard WSJ. The New York Times (NYT) dataset is extracted from the English Gigaword (Graff, 2003) corpora. We select the articles tagged as 'story' from years 1994 to 1995. After cleaning the text, the final corpora consist of about 5M sentences with 105M tokens. This dataset is approximately 100 times larger than the WSJ. We also create a very long time series that is about 1.1 million in length, by concatenating all the sentences in WSJ. We split each dataset into a training set and a testing set at random, and compare the algorithms' predicative log likelihoods on the testing sets. For the WSJ (and the single long sequence), the ratio of the training set size and testing set size is set to $9:1$; and for the NYT, the ratio is set to $99:1$.

First, we conduct our experiments in the multiple sequence scenarios by the following experiment settings. For both SVI and SCVI-2, we set the hyper-parameters $\alpha = \beta = 0.1$, to encourage sparsity. In the stochastic setting, since we do not need to keep local statistics (and also, initializing local statistics on NYT is too computationally expensive), we directly initialize the global transition and emission counting matrices using the exponential distributions, as suggested by Hoffman et al. (2013),

$$N_{j,k} \sim \text{Exponential}(IT/K^2), \tag{3.106}$$

$$N_{k,w} \sim \text{Exponential}(IT/(KW)), \tag{3.107}$$

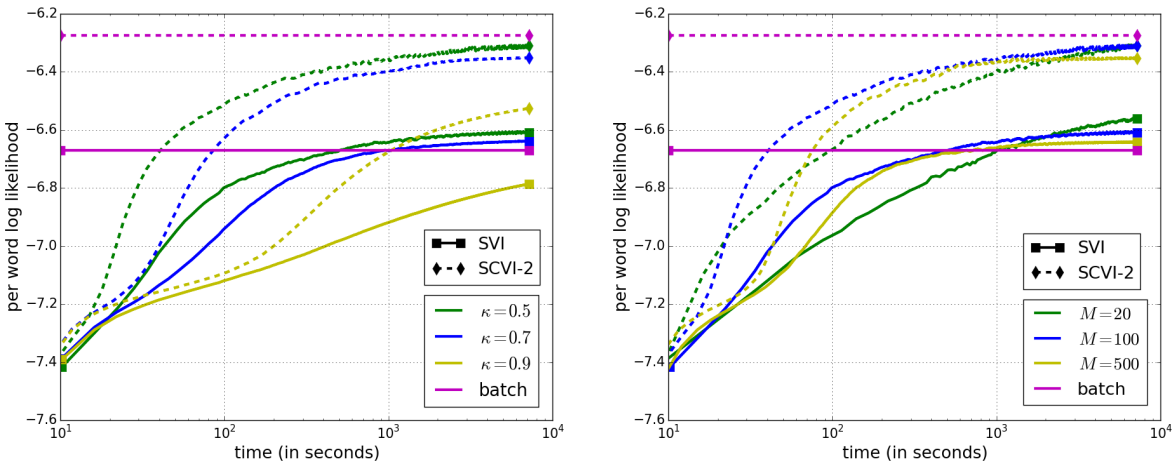

Figure 3.15: Evaluation of the stochastic algorithms on the WSJ treebank by their predictive performances. Left: we set the minibatch size $M = 100$ and vary forgetting rates. Right: we set the forgetting rate $\kappa = 0.5$ and vary minibatch sizes. For references, we include the results produced by the batch algorithms that run for 200 iterations.

and the resulting $N_{j,k}$ and $N_{k,w}$ are similar to those we would get by tallying the local statistics with randomly assigned hidden states. We use the per word predictive log likelihood as the evaluation metric. In a test set, $\mathbf{x}^{\text{test}} = \{\mathbf{x}^i\}_{i=1}^{I'}$, let $|\mathbf{x}^i|$ be the length of the i^{th} sentence, and the per word likelihood is defined as follows,

$$\text{likelihood}_{\text{pw}} = \frac{\sum_i \log \sum_{\mathbf{z}^i} \prod_t \hat{\theta}_{z^i_{t-1},z^i_t} \hat{\theta}_{z^i_t,x^i_t}}{\sum_i |\mathbf{x}^i|}, \tag{3.108}$$

in which

$$\begin{aligned} \hat{\theta}_{j,k} &= \frac{N_{j,k} + \alpha}{N_{j,\cdot} + K\alpha}, \\ \hat{\theta}_{k,w} &= \frac{N_{k,w} + \beta}{N_{\cdot,k} + W\beta}. \end{aligned} \tag{3.109}$$

On the WSJ, we investigate a range of learning rates for stochastic inference. Given the step sizes $\rho_m = (1+m)^{-\kappa}$ parametrized by forgetting rates $\kappa \in [0.5, 1.0)$ and the minibatch sizes M, we first set minibatch size $M = 100$ and vary $\kappa \in \{0.5, 0.7, 0.9\}$. Then we set $\kappa = 0.5$ (the best choice for both SVI and SCVI-2) and vary $M = \{20, 100, 500\}$. We run both algorithms for 2 hours and report their predictive log likelihoods versus wall-clock time. On the NYT, we set $\kappa = 0.5$ and $M = 1000$. This time we explore various values of $K \in \{20, 40, 80\}$, since the true number of hidden states on this corpora is undefined. We run both the SVI and SCVI-2 algorithms for 6 hours.

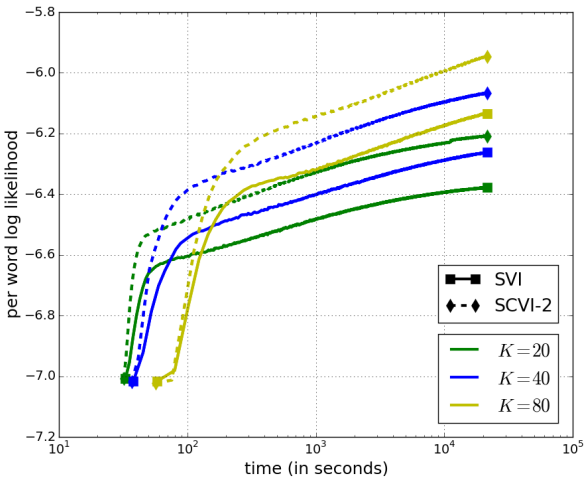

Figure 3.16: Evaluation of the stochastic algorithms on the NYT by their predictive performances. We set $\kappa = 0.5$, $M = 1000$, and vary the numbers of hidden states.

We present the predictive performances of SVI and SCVI-2 on the WSJ in Figure 3.15. We can see that under each explored setting, our SCVI-2 is significantly more accurate than SVI, extending the success of our CVI algorithms to the stochastic setting. For references, we also include the results produced by VI and CVI-2 that run for 200 iterations (~ 90 minutes). Both of their stochastic versions achieve comparable (or slightly worse for SCVI-2) results within 1000 seconds ($\sim$17 minutes), i.e., roughly one fifth of the batch training time. Despite the stochasticity, SVI performs slightly better than VI. Indeed VI is a special case of SVI, which takes a unit gradient in Riemannian space (Hoffman et al., 2013).

In our experiments, it is interesting to notice that fixing M, a smaller forgetting rate κ (i.e., larger step sizes) seems preferred both by SVI and SCVI-2. Our findings contradict with the empirical observations in LDA (a larger κ is preferred) (Hoffman et al., 2013), but confirms the those observations in HMMs (Foti et al., 2014; Zhang et al., 2016). We think that at least on this dataset, larger step sizes in early phase might be helpful for stochastic HMMs to escape from bad local optimums. On the ther hand, both algorithms are not too sensitive with minibatch sizes: even when $M = 20$, both algorithms perform surprisingly well. These preferences allow both algorithms to be scalable to very large datasets.

In Figure 3.15, we show the predictive performances of SVI and SCVI-2 on the NYT that has 105 million tokens and is roughly 100 times larger than the WSJ. In this dataset, batch algorithms are not scalable. We see that again our SCVI-2 outperforms SVI by large margins, while both algorithms favour more hidden states to represent the posteriors.

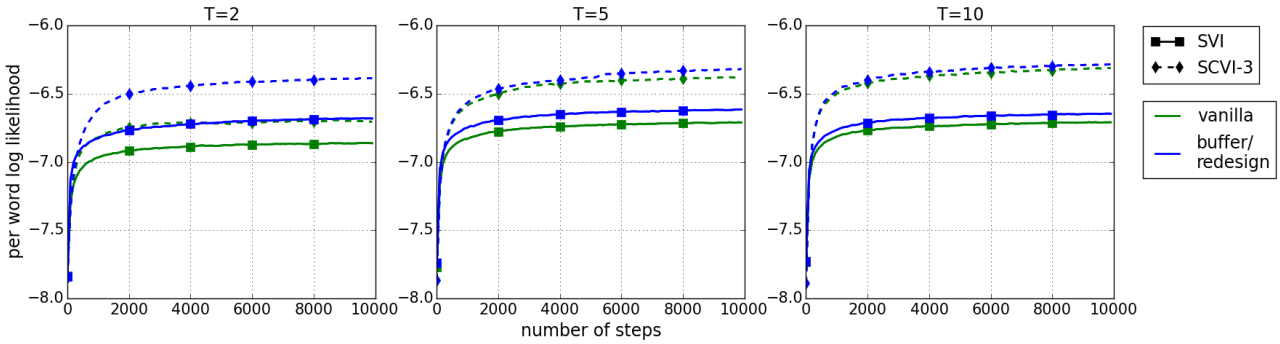

Figure 3.17: Evaluation of the stochastic algorithms in the single sequence setting by their predictive performances. We set $\kappa = 0.5$, $T \times M = 1000$, and vary subsequence lengths.

Next, we conduct our experiments in the single sequence setting to evaluate our third stochastic CVI algorithm (denoted by 'SCVI-3'). In this setting, Foti et al. (2014) proposed an SVI algorithm that buffers several observations on both sides of each subsequence, and demonstrated the utility of their buffering method. Here as well as comparing SCVI-3 with SVI, we also explicitly evaluate the utility of our propagation redesign method. Specifically, for both SCVI-3 with SVI, we create two benchmarks that do not use propagation redesign or buffering mechanisms, respectively. In these reduced algorithms (denoted by 'vanilla'), each subsequence starts with the initial distribution equal to the stationary distribution of the whole HMM sequence. For SVI with buffering, we pad 10 observations on both sides of each subsequence, which is sufficient by our empirical observations.

We use the same hyperparameters and initialization methods as previously in the multiple sequence setting. To fully study the effectiveness of our propagation redesign method, we also vary the subsequence lengths, since we expect that it will be most effective when subsequences are short. Following Foti et al. (2014), we fix the number of tokens in a mini-batch, $T \times M = 1000$, and vary $T \in \{2, 5, 10\}$. That is increasing T means decreasing M and vice-versa. We set the forgetting rate κ to 0.5. Finally, we run both SVI and SCVI-3 for 10000 time steps and report the results by averaging 5 random runs.

Figure 3.17 presents the predictive performances of SVI, SCVI-3 and their correspondingly reduced versions. We can see that by the per time step evaluation, our SCVI-3 also converges much more accurately than SVI in each setting, extending the success of collapsing to the single time series setting. For both algorithms, when T is small, there are significant improvements using respective methods for handling subsequence boundaries. For SCVI-3, we attribute such improvement to the inter subsequence communication by

the message passing in our redesigned propagation algorithm. We have tried to plot likelihoods versus wall clock time as with other stochastic experiments in this thesis, but found that the buffer/redesign increases can not be clearly illustrated. This is because for example when $T = 2$, SCVI-3 takes twice training time as its reduced version (1 hidden variable on each side); while SVI consumes 11 times training effort as its reduced version (10 observed variables on each side). Therefore, given a single time series, SCVI-3 can be much more computationally efficient than SVI depending on the subsequence lengths[7].

3.9 Summary

In this chapter, we have studied how we can apply collapsed variational inference beyond its current bag-of-words setting, and to probabilistic graphical models in general. Specifically, we have explored three approaches for marring the idea of collapsing with the sequential dependencies in HMMs, using from brute-forced mean field, softened structured mean field to the mixture of both in the single sequence setting. We can see that it is possible to derive sophisticated message passing procedures in collapsed space, as we have done when model parameters are provided. As a result, our study sheds lights on the general usage rules of CVI, as we have summarized in Subection 3.6.2. We also extend the CVI research for the conjugate exponential family (Sung et al., 2008). These extensions allow CVI to be applied to many probability distributions and in the stochastic setting. With the novel techniques we have developed during the algorithm derivations, we expect that many exciting applications of CVI will continue to come in the future.

We have conducted a comprehensive set of experiments in the domain of computational linguistics to evaluate our proposed CVI algorithms, from PoS induction, word alignment to textual prediction. The experiment results show that our CVI algorithms (especially the second and third ones) consistently and significantly outperform the standard variational inference algorithms. Our second CVI algorithm often performs comparably with CGS, which sometimes consumes up to an order of magnitude more training time. In Chapter 5, we will extend our CVI algorithms to HDP-HMMs that allow for using an infinite number of hidden states and learning the model hyper-parameters.

[7] In addition, our SCVI-3 algorithm does not compute the digamma functions required by SVI.

Chapter 4

Collapsed Variational Inference for PCFGs

Chapter Abstract

In this chapter, we extend collapsed variational inference along the Chomsky hierarchy, from hidden Markov models in the previous chapter to probabilistic context free grammars (PCFGs) (Wang and Blunsom, 2013b). This extension allows collapsed variational inference to be applied to statistical models whose model structures are random objects and thus can not be represented by graphical models. We evaluate our proposed algorithm both in vanilla PCFGs and their two variants, showing that our algorithm achieves close performance to the Metropolis-Hastings sampling algorithm while using an order of magnitude less training time; and outperforms the standard inside-outside algorithm and variational inference with similar training time.

4.1 Introduction

Probabilistic context free grammars (PCFGs) are ubiquitous for modelling syntactic structures in the field of computational linguistics, as well as other hierarchical structures in the fields such as speech recognition (Baker, 1979), computational biology (Sakakibara, 2005) and computer vision (Zhu and Mumford, 2007). In computational linguistics, PCFGs underlie most high-performance parsing and grammar induction systems, either in the semi-supervised setting (i.e., learning grammars from labelled parse trees) (Charniak and John-

son, 2005; Petrov et al., 2006), or in the unsupervised setting (i.e., learning grammars from raw text) (Headden III et al., 2009; Blunsom and Cohn, 2010). The traditional method for learning PCFGs is the inside-outside (IO) algorithm (Lari and Young, 1990). As an instance of the expectation maximization (EM) algorithm based on the principle of maximum likelihood estimation (MLE), the standard IO algorithm often learns relatively uniform distributions for grammars, whereas the true distributions can be highly skewed (Johnson et al., 2007). In order to encourage sparse distributions and avoid overfitting, recent research for learning PCFGs therefore has drifted away from MLE in favor of Bayesian inference.

Variational inference (VI) for PCFGs (Kurihara and Sato, 2006) generalizes the inside-outside algorithm by placing no constraints when updating parameters in the variational M step. As in the HMM case in the previous chapter, the performances of both EM and VI for PCFGs suffer from their local optimality issues. Johnson et al. (2007) proposed several Markov Chain Monte Carlo (MCMC) algorithms for PCFGs that can reach the global optimal solutions after convergences. In particular, their Metropolis-Hastings algorithm that makes use of both collapsing and blocking is probably the most effective sampling strategy for PCFGs. Nevertheless, still a large number of samples are often required in practical applications (e.g. our experiment in Subsection 4.5.1). Given the computational complexity of PCFGs that is two orders higher than LDA and HMMs in data size and the availability of large corpora, there is a more demand of inference algorithms for PCFGs that are efficient, accurate and can be used in various grammar induction tasks.

In this chapter, we extend the collapsed variational approach to PCFGs, using the algorithmic techniques we have developed in the previous chapter. Specifically, we follow the idea of our second CVI algorithm for HMMs and apply the structured mean field method after model parameters are integrated out. A notable difference between HMMs and PCFGs is that while HMMs can be represented by probabilistic graphical models (model structures are fixed, hidden variables are random); PCFGs can not. This is because for PCFGs, model structures are parse trees, and parse trees themselves are random objects as well as hidden variables conditioned on parse trees. In Section 4.4, we show that the joint posterior predictive distribution over random parse tree in PCFGs requires additional algebraic manipulation to be simplified than in HMMs, and derive our CVI algorithm for PCFGs that makes use of the traditional inside-outside procedure. Our work further enlarges the application domain of CVI to statistical models with random model structures.

In the following presentation, we employ the standard version of PCFGs in the Chomsky normal form. The extensions of all the reviewed algorithms and CVI to other versions or variants of PCFGs are either straightforward or standard, and thus we omit those elaborations. In Section 4.5, we conduct three grammar induction experiments and validate our CVI algorithm in standard PCFGs, PCFGs with latent annotations (PCFGs-LA) (Matsuzaki et al., 2005) and dependency model with valence (DMV) (Klein and Manning, 2004).

4.2 Definitions

A probabilistic context free grammar (PCFG) generalizes a probabilistic regular grammar, namely a hidden Markov model by relaxing the production constraints for non-terminals. Formally, adopting the similar notations to Johnson et al. (2007), we specify a PCFG by a tuple (T, N, S, R, θ), where T, N, R and θ are the finite sets of terminals (e.g., words in sentences), non-terminals (e.g., phrase labels in trees), production rules and parameters respectively, and $S \in N$ is the start symbol. Given the context free grammar $G = (T, N, S, R)$ in the Chomsky normal form, each rule $r \in R$ takes either the form $A \to BD$ or $A \to w$, where $A, B, D \in N$ and $w \in T$. Let $\theta_{A\to\beta}$ be the probability of rule $A \to \beta$, where β ranges over $(N \times N) \cup T$. Finally, in a Bayesian setting, we put symmetric Dirichlet priors with hyperparameters α on each $\theta_A = \{\theta_{A\to\beta}\}$.

Having defined the grammar, a parse tree and its yield (e.g., sentence) is generated by the following stochastic process: start with S and apply a rule $r_{S\to\beta}$ with probability $\theta_{S\to\beta}$; for the non-terminals in β, recursively apply rules and generate their subtrees; the process stops when all the parse tree's leaves are terminals. Furthermore, we consider the scenario of many parse trees and sentences, which is assumed by all the grammar induction tasks in computational linguistics. We have not encountered the scenario of learning probably very large and deep trees from a single long sentence in any application fields.

Given a corpus of sentences $\mathbf{x} = \{\mathbf{x}^i\}_{i=1}^I$ and the corresponding parse trees $\mathbf{z} = \{\mathbf{z}^i\}_{i=1}^I$, as with HMMs, we first define the counting functions for PCFGs. Let $C_r(\mathbf{x}, \mathbf{z})$ denote the count (or frequency) of the product rule r in all the parse trees, $C_r^i(\mathbf{x}^i, \mathbf{z}^i)$ denote the count of r in the i^{th} parse tree, and $C_r^{\backslash i}(\mathbf{x}^{\backslash i}, \mathbf{z}^{\backslash i})$ denote the count of r in all but the i^{th} parse tree. By our conventions, since the arguments to C can be inferred by the superscripts of C, we can safely omit the arguments for brevity. For example, C_r is short for $C_r(\mathbf{x}, \mathbf{z})$.

By the above definitions, the joint probability distribution of parameters and variables can be expressed as follows:

$$\begin{aligned} p(\mathbf{x}, \mathbf{z}, \theta|\alpha) &= p(\theta|\alpha) \prod_{i=1}^{I} P_G(\mathbf{x}^i, \mathbf{z}^i|\theta) \\ &= \Big(\prod_{A \in N} P_D(\theta_A|\alpha) \Big) \prod_{r \in R} \theta_r^{C_r}, \end{aligned} \tag{4.1}$$

in which,

$$P_D(\theta_A|\alpha) = \frac{\Gamma(|R_A|\alpha)}{\Gamma(\alpha)^{|R_A|}} \prod_{r \in R_A} \theta_r^{\alpha-1}, \tag{4.2}$$

$P_G(\cdot)$ denotes the distribution under the grammar G[1], $P_D(\cdot)$ denotes the Dirichlet distribution reviewed in Section 2.3, R_A denotes the set of rules with left-hand side A, and $|R_A|$ denotes the size of R_A.

4.3 Approximate Inference for PCFGs

In this section, we review the EM, VI and Hastings algorithms for PCFGs, which are widely used, served as benchmarks in the experiments and closely related to CVI. In particular, our CVI algorithm shares the same (approximation) spirit with the Hastings algorithm.

4.3.1 Variational Inference

The traditional inside-outside algorithm for PCFGs (Lari and Young, 1990) belongs to the general EM class, which is further a subclass of variational inference. As with HMMs in the previous chapter, we review both EM and VI for PCFGs in one framework. Specifically, VI seeks to maximize an evidence lower bound on the log marginal likelihood $\log p(\mathbf{x}|\alpha)$, denoted by $\mathcal{L}^{\text{VI}}$,

$$\log p(\mathbf{x}|\alpha) \geq \mathbb{E}_{q(\mathbf{z},\theta)}[\log p(\mathbf{x}, \mathbf{z}, \theta|\alpha)] - \mathbb{E}_{q(\mathbf{z},\theta)}[\log q(\mathbf{z}, \theta)] \triangleq \mathcal{L}^{\text{VI}}, \tag{4.3}$$

in which $q(\mathbf{z}, \theta)$ is an approximate posterior whose model parameters and hidden variables are assumed to be independent. That is it is factorized by,

$$q(\mathbf{z}, \theta) = q(\mathbf{z})q(\theta). \tag{4.4}$$

[1] $P_G(\mathbf{x}^i, \mathbf{z}^i|\theta) = 0$ if $\mathbf{z}^i$ can not be generated by the grammar G or yield$(\mathbf{z}^i) \neq \mathbf{x}^i$.

This strong independence assumption allows for the separate updates of $q(\mathbf{z})$ and $q(\theta)$, maximizing the lower bound $\mathcal{L}^{\text{VI}}$. For the traditional inside outside algorithm using maximum likelihood estimation, $q(\theta)$ is further assumed to be degenerate, i.e. $q(\theta) = \delta(\theta = \theta^\star)$. Hence, the IO algorithm iterates over the following expectation and maximization steps,

$$\text{E step: } q(\mathbf{z}) \propto P_G(\mathbf{x}, \mathbf{z}|\theta^\star), \tag{4.5}$$

$$\text{M step: } \theta^\star = \underset{\theta}{\operatorname{argmax}}\, \mathbb{E}_{q(\mathbf{z})}[P_G(\mathbf{x}, \mathbf{z}|\theta)]. \tag{4.6}$$

In the E step, we update each $q(\mathbf{z}^i)$ over random parse trees,

$$\begin{aligned} q(\mathbf{z}^i) &\propto P_G(\mathbf{x}^i, \mathbf{z}^i|\theta^\star) \\ &= \prod_{r \in R} (\theta_r^\star)^{C_r^i}. \end{aligned} \tag{4.7}$$

The distribution over parse trees $q(\mathbf{z}^i)$ is intractable to compute as its normalization requires summing over all possible parse trees producing $\mathbf{x}^i$. But the inside-outside dynamic programming algorithm can be used to compute the expected counts required in the M step,

$$\mathbb{E}[C_{A\to BD}^i] \propto \sum_{0 \le j < k < l \le |\mathbf{x}^i|} P_{\text{OUT}}(A, j, l)\, \theta_{A\to BD}\, P_{\text{IN}}(B, j, k)\, P_{\text{IN}}(D, k, l), \tag{4.8}$$

$$\mathbb{E}[C_{A\to w}^i] \propto \sum_{0 \le j \le |\mathbf{x}^i|} P_{\text{OUT}}(A, j)\, \theta_{A\to w} \delta(x_j^i = w), \tag{4.9}$$

in which the word positions in the sentence $\mathbf{x}^i$ are indexed $\{j, k, l\}$, e.g., x_j^i denotes the j^{th} word in the i^{th} sentence; the inside probability of observation $\{x_j^i, x_{j+1}^i ..., x_l^i\}$ given A is the root of the subtree is denoted by $P_{\text{IN}}(A, j, l)$, and the probability of A spanning (j, l), together with the rest of $\mathbf{x}^i$ is denoted by $P_{\text{OUT}}(A, j, l)$. Both the inside and outside probabilities are computed recursively in $O(|G||\mathbf{x}^i|^3)$ with $|G|$ the grammar size and $|\mathbf{x}^i|$ the sentence length. Details can be found in Lari and Young (1990).

In the M step, we find the optimal $\theta^\star$ based on the MLE principle,

$$\theta_{A\to\beta}^\star = \frac{\sum_{i=1}^I \mathbb{E}[C_{A\to\beta}^i]}{\sum_{i=1}^I \mathbb{E}[C_{A\to\cdot}^i]} = \frac{\mathbb{E}[C_{A\to\beta}]}{\mathbb{E}[C_{A\to\cdot}]}, \tag{4.10}$$

in which we have used the dot to denote the summed out right-hand side,

$$\mathbb{E}[C_{A\to\cdot}^i] = \sum_{A\to\beta' \in R_A} \mathbb{E}[C_{A\to\beta'}^i] \tag{4.11}$$

VI generalizes EM by allowing arbitrary parametric forms for $q(\theta)$ (Kurihara and Sato, 2006). Using the algorithmic shortcut from EM to VI (Beal, 2003), we can reuse the inside-outside implementation with a minor change in the variational M step, namely updating the following inverted expected natural parameters instead of the MLE ones,

$$\tilde{\theta}_{A\to\beta} = \frac{\exp(\psi(\mathbb{E}[C_{A\to\beta}] + \alpha))}{\exp(\psi(\mathbb{E}[C_{A\to\cdot}] + |R_A|\alpha))}. \tag{4.12}$$

4.3.2 Hastings Algorithm

The Hastings algorithm for PCFGs (Johnson et al., 2007) first integrates out the model parameters θ, and thus dealing with their dependencies on the parse trees in an exact fashion. By the conjugacy property 2 in Section 2.3, the complete data likelihood is given by,

$$\begin{aligned} p(\mathbf{x}, \mathbf{z}|\alpha) &= \int_\theta P_G(\mathbf{x}, \mathbf{z}|\theta) P_D(\theta|\alpha) d\theta \\ &= \prod_{A\in N} \frac{\Gamma(|R_A|\alpha)}{\Gamma(|R_A|\alpha + C_{A\to\cdot})} \Bigg(\prod_{A\to\beta\in R_A} \frac{\Gamma(\alpha + C_{A\to\beta})}{\Gamma(\alpha)} \Bigg). \end{aligned} \tag{4.13}$$

Then the Hastings algorithm draws samples in a component wise manner, i.e., by drawing each parse tree conditioned on the remaining ones in turn. Let $\mathbf{x}^{\backslash i}$ and $\mathbf{z}^{\backslash i}$ denote the set of all sentences and parse trees with the i^{th} ones removed, respectively. By the conjugacy property 4 in Section 2.3, the joint posterior predictive distribution over each random parse tree $\mathbf{z}^i$ is given by,

$$\begin{aligned} p(\mathbf{z}^i|\mathbf{x}, \mathbf{z}^{\backslash i}, \alpha) &\propto \frac{p(\mathbf{x}, \mathbf{z}|\alpha)}{p(\mathbf{x}^{\backslash i}, \mathbf{z}^{\backslash i}|\alpha)} \\ &= \prod_{A\in N} \frac{\Gamma(|R_A|\alpha + C^{\backslash i}_{A\to\cdot})}{\Gamma(|R_A|\alpha + C_{A\to\cdot})} \Bigg(\prod_{A\to\beta\in R_A} \frac{\Gamma(\alpha + C_{A\to\beta})}{\Gamma(\alpha + C^{\backslash i}_{A\to\beta})} \Bigg). \end{aligned} \tag{4.14}$$

Johnson et al. (2007) suggests that sampling a parse tree from the above distribution is difficult, since the counts of the current parse tree C^i_r is included in C_r, which is in the RHS of (4.14). This prohibits the efficient dynamic programming techniques. Although it is essentially the same problem we have faced in our second CVI algorithm for HMMs, the problem for PCFGs is a little more complicated. Due to the stochastic model structures, it requires additional calculations to derive exactly how the rule parameters change on the fly inside a parse tree, which we will show in the next section.

In order to solve this problem, Johnson et al. (2007) introduced a proposal distribution using an alternative set of rule probabilities, denoted by θ^{H},

$$\mathbf{z}^i \sim p(\mathbf{z}^i|\mathbf{x}^i, \theta^{\mathrm{H}}), \tag{4.15}$$

$$\theta^{\mathrm{H}}_{A\to\beta} \triangleq \frac{C^{\backslash i}_{A\to\beta} + \alpha}{C_{A\to\cdot} + |R_A|\alpha}. \tag{4.16}$$

Since θ^{H} are based on the counts collected from all other parse trees, they are fixed for the conditional distribution of the current parse tree. Therefore, by using a variant of inside parsing and outside sampling algorithm (Goodman, 1998), one can efficiently sample an entire parse tree, which is either accepted or rejected by the Metropolis choice.

In theory, in a Metropolis Hastings algorithm any proposal distribution is valid. That is there are no approximations no matter how far the proposal distribution is from the true conditional. Empirically, we conduct the experiment of learning Sesotho language as in Johnson et al. (2007) in Subsection 4.5.1, and we find that the sampled trees based on θ^{H} never get rejected (i.e., no waste computations at all), illustrating an acceptance rate close to 100%, and meaning that the Metropolis proposal in (4.15) is very close to the true conditional in (4.14). This is not surprising, because as we will see shortly, the difference between (4.14) and (4.15) is precisely our approximation method to joint posterior predictives, which has been empirically validated in Subsection 3.5.1.

4.4 CVI for PCFGs

To combine the best of both VI (efficiency) and the Hastings algorithm (accuracy), in this section, we propose collapsed variational inference for PCFGs. Specifically, CVI integrates out the model parameters and introduce a model evidence lower bound on the log marginal likelihood $\log p(\mathbf{x})$, denoted by $\mathcal{L}^{\mathrm{CVI}}$,

$$\log p(\mathbf{x}|\alpha) \geq \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}, \mathbf{z}|\alpha)] - \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] \triangleq \mathcal{L}^{\mathrm{CVI}}. \tag{4.17}$$

We follow the idea of our second CVI algorithm for HMMs, and only break the sentence level dependencies, which are induced by marginalizing out model parameters and thus are weak. Formally, the variational distribution $q(\mathbf{z})$ is assumed to factorize as follows,

$$q(\mathbf{z}) = \prod_{i=1}^{I} q(\mathbf{z}^i). \tag{4.18}$$

To maximize the model evidence lower bound $\mathcal{L}^{\text{CVI}}$, we apply the structured mean field method, and the update equation for each $q(\mathbf{z}^i)$ is given by,

$$q(\mathbf{z}^i) \propto \exp\{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[\log p(\mathbf{z}^i|\mathbf{x}, \mathbf{z}^{\setminus i}, \alpha)]\}. \tag{4.19}$$

As with the Hastings algorithm, the intractable conditional distribution $p(\mathbf{z}^i|\mathbf{x}, \mathbf{z}^{\setminus i}, \alpha)$ in (4.14) is also required in the above update. But for CVI, replacing the intractable distribution with a tractable one (e.g., (4.15)) means making hard approximations, and thus we need to understand and validate them if possible. In the following, we show that the bridge from (4.14) to (4.15) is simply made of some algebraic manipulation and our approximation method to joint posterior predictives in Subsection 3.5.1.

Using the property of the Gamma function $\Gamma(x+1) = x\Gamma(x)$, we can simplify (4.14) into the following form,

$$p(\mathbf{z}^i|\mathbf{x}, \mathbf{z}^{\setminus i}, \alpha) \propto \prod_{A \in N} \frac{\prod_{A \to \beta \in R_A} \overbrace{\prod_{m=0}^{C^i_{A\to\beta}-1} (C^{\setminus i}_{A\to\beta} + m + \alpha)}^{\text{a product of } C^i_{A\to\beta} \text{ terms}}}{\underbrace{\prod_{n=0}^{C^i_{A\to\cdot}-1} (C^{\setminus i}_{A\to\cdot} + n + |R_A|\alpha)}_{\text{a product of } C^i_{A\to\cdot} \text{ terms}}}. \tag{4.20}$$

A key insight in the above equation is that the number of terms to be multiplied in the numerator is equal to the number of terms to be multiplied in the denominator: both have $C^i_{A\to\cdot}$ terms. By this observation, we seek to rearrange all these terms so that they form a product of $C^i_{A\to\cdot}$ fractions, and each fraction takes one term from the numerator and one term from the denominator. To do this, let there be an arbitrary ordering in the production rules, for example, $A \to \beta$ precedes $A \to \beta'$, which precedes $A \to \beta''$, and so on. We use $A \to < \beta$ to denote summing over all the rules preceding $A \to \beta$. That is,

$$C^i_{A\to<\beta} = \sum_{A\to\beta'<A\to\beta} C^i_{A\to\beta'}. \tag{4.21}$$

Then we can rearrange (4.20) into the following form,

$$\begin{aligned} p(\mathbf{z}^i|\mathbf{x}, \mathbf{z}^{\setminus i}, \alpha) &\propto \prod_{A\in N} \prod_{A\to\beta\in R_A} \prod_{m=0}^{C^i_{A\to\beta}-1} \frac{C^{\setminus i}_{A\to\beta} + m + \alpha}{C^{\setminus i}_{A\to\cdot} + \left(\sum_{A\to\beta'<A\to\beta} C^i_{A\to\beta'}\right) + m + |R_A|\alpha} \\ &= \prod_{A\to\beta\in R} \prod_{m=0}^{C^i_{A\to\beta}-1} \frac{C^{\setminus i}_{A\to\beta} + m + \alpha}{C^{\setminus i}_{A\to\cdot} + C^i_{A\to<\beta} + m + |R_A|\alpha}. \end{aligned} \tag{4.22}$$

The above equation explicitly expresses that how the rule probabilities change on the fly inside a parse tree, and we can recognize that our approximation method to joint posterior predictives in Subsection 3.5.1 can be applied. Note that the ordering assumed in the production rules is arbitrary and does not matter, since different orderings just express different ways of changing rule probabilities. By removing the local contributions from the current parse tree $\mathbf{z}^i$ in both the numerator and denominator in (4.22), we have,

$$p(\mathbf{z}^i|\mathbf{x}, \mathbf{z}^{\setminus i}, \alpha) \approx\propto \prod_{A\to\beta\in R} \left(\frac{C^{\setminus i}_{A\to\beta} + \alpha}{C^{\setminus i}_{A\to\cdot} + |R_A|\alpha}\right)^{C^i_{A\to\beta}}, \tag{4.23}$$

which is the proposal distribution in the Hastings algorithm in (4.15).

Since we have validated that our approximation method to joint posterior predictives is very accurate, we plug (4.23) into the structured mean field update in (4.19). Then we apply the first order Taylor approximation. In total, we have,

$$\begin{aligned}
q(\mathbf{z}^i) &\propto \exp\left\{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}\left[\log \prod_{A\to\beta\in R} \left(\frac{C^{\setminus i}_{A\to\beta} + \alpha}{C^{\setminus i}_{A\to\cdot} + |R_A|\alpha}\right)^{C^i_{A\to\beta}}\right]\right\} \\
&= \prod_{A\to\beta\in R} \exp\left\{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}\left[\log\left(\frac{C^{\setminus i}_{A\to\beta} + \alpha}{C^{\setminus i}_{A\to\cdot} + |R_A|\alpha}\right)\right]\right\}^{C^i_{A\to\beta}} \\
&\approx \prod_{A\to\beta\in N} \left(\frac{\mathbb{E}[C^{\setminus i}_{A\to\beta}] + \alpha}{\mathbb{E}[C^{\setminus i}_{A\to\cdot}] + |R_A|\alpha}\right)^{C^i_{A\to\beta}} \\
&\triangleq \prod_{A\to\beta\in N} \left(\bar{\theta}_{A\to\beta}\right)^{C^i_{A\to\beta}},
\end{aligned} \tag{4.24}$$

where in the last line, we have defined the surrogate rule parameters of CVI, denoted by $\bar{\theta}$. The similarity between (4.7) and (4.24) indicates that the inside-outside dynamic programming algorithm used in both EM and VI can take over from now. That is our CVI algorithm runs the inside-outside algorithm based on the surrogate parameters $\bar{\theta}$ computed from the expected counts of all other sentences, in the same ways as EM based on $\theta^\star$ and VI based on $\tilde{\theta}$. The main difference with EM and VI is that CVI updates parameters after processing each sentence, and thus information can be propagated even sooner.

In the previous chapter, we have developed three CVI approaches for HMMs. It might be worth mentioning that our first and third approaches are applicable to PCFGs with fixed shapes, e.g., PCFGs-LA (see definition in Subsection 4.5.3). By fixing the shape of a parse tree, our first CVI approach would break all the dependencies despite whether they are

strong (for adjacent nodes) or weak (for other). On the other hand, in the above derivations we have followed our second CVI approach, and this approach is more widely applicable: by simplifying and rearranging terms in the joint posterior predictive distribution, our derivations for PCFGs with random structures finally arrive at an elegant update equation in (4.24) as with those models with fixed structures, and therefore the application domain of collapsed variational inference has been further expanded.

4.5 Experiments

In this section, we conduct three experiments to evaluate our CVI algorithm for PCFGs. In Subsection 4.5.1, we illustrate the significantly reduced training time of our CVI algorithm compared to the Hastings algorithm for standard PCFGs; and in later two subsections, we demonstrate the increased performance of our CVI algorithm compared to the corresponding EM and VI algorithms for two variants of PCFGs. The experiments in Subsection 4.5.2 are conducted by my co-author and the rest is my work.

4.5.1 Standard PCFGs

Firstly, we conduct the same experiment of inferring sparse grammars describing the morphology of the Sesotho language as in Johnson et al. (2007). We employ a similar corpus that consists of $1{,}671$ unsegmented Sesotho verb types extracted from CHILDES (MacWhinney and Snow, 1985), and define a CFG grammar by allowing each non-terminal to emit any substrings in the corpus as terminals plus five predefined production rules as follows,

$$\text{Word} \to \text{V} \tag{4.25}$$

$$\text{Word} \to \text{V M} \tag{4.26}$$

$$\text{Word} \to \text{SM V M} \tag{4.27}$$

$$\text{Word} \to \text{SM T V M} \tag{4.28}$$

$$\text{Word} \to \text{SM T OM V M}. \tag{4.29}$$

We randomly withhold 10% of the verb types from the corpus for testing, and use the rest 90% for training. Both algorithms are evaluated by their per word test perplexity on the test data set with prior set to 10^{-5} as suggested by Johnson et al. (2007). We run five times with random starts, and report the averaged results in Figure 4.1 (left). We can see that the

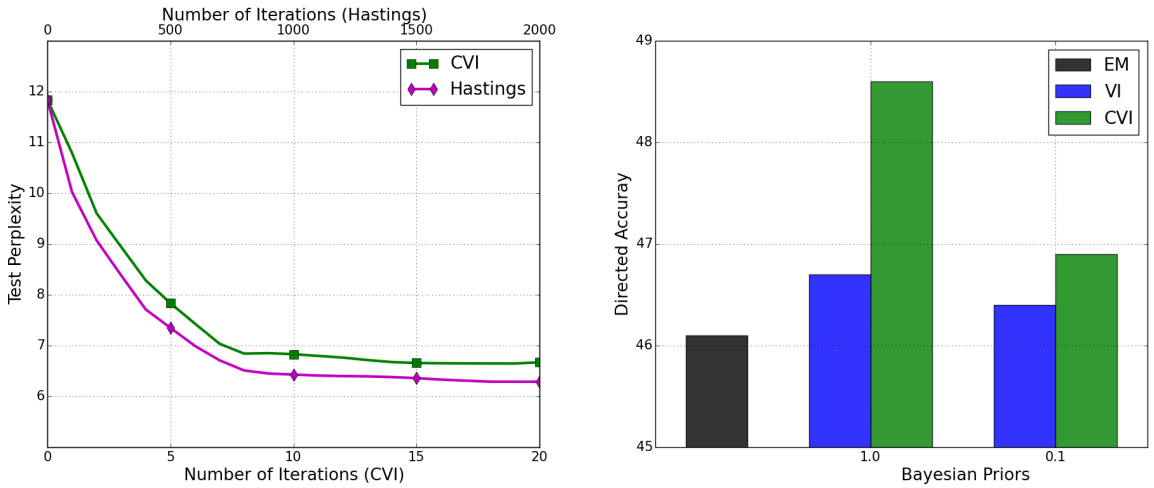

Figure 4.1: Left: Evaluation of the CVI and Hastings algorithms for standard PCFGs in the task of learning Sotho verbs, by the test perplexities. Right: Evaluation of the EM, VI and CVI algorithms for Dependency Model with Valence in the task of unsupervised grammar induction, by the directed accuracy scores on section 23, WSJ.

Hastings algorithm[2] takes roughly $1{,}000$ iterations to converge, while our CVI algorithm reaches the convergence in 10 iterations, consuming only a fraction of training time (CVI: 1.5 minutes; Hastings: 20 minutes). As well as little difference margin in final perplexities, we also evaluate their segmentation qualities using the F1 scores, and again the difference is trivial (CVI: 29.8%, Hastings: 31.3%).

4.5.2 Dependency Model with Valence

As a second empirical validation of our CVI algorithm, we apply it to unsupervised grammar induction with the popular Dependency Model with Valence (DMV) (Klein and Manning, 2004). Although the original maximum likelihood formulation of this model has long since been surpassed by more advanced models, all of the state-of-the-art approaches to unsupervised dependency parsing still have DMV at their core (Headden III et al., 2009; Blunsom and Cohn, 2010). As such we believe demonstrating improved inference on this core model will enable future improvements to more complex models.

We evaluate a Dirichlet-Multinomial formulation of DMV in the standard fashion by training on sections 2-21 and testing on section 23 of the Wall Street Journal treebank (Marcus et al., 1993). We initialize our models using the original harmonic initialiser (Klein and

[2] Annealing is not used in order to facilitate the perplexity calculation in the test set.

	H=2				H=4	H=8	H=16
	LP	LR	F1	Exact	F1	F1	F1
EM	75.84	72.92	74.35	11.13	82.59	85.48	87.42
VI	76.98	73.32	75.11	11.49	82.62	85.46	87.33
CVI	78.85	76.98	77.90	12.56	83.36	86.55	88.06

Figure 4.2: Evaluation of the EM, VI and CVI algorithms for PCFGs-LA in the task of semi-supervised grammar induction on section 23, WSJ. We report the precision, recall, F1 and exact match scores for $H = 2$ and F1 scores for $H = \{4, 8, 16\}$.

Manning, 2004). Figure 4.1 (right) displays the directed accuracy results for DMV models trained with CVB and VB with hyperparameters $\alpha \in \{0.1, 1\}$, as well as the previously reported MLE result. In both cases we see superior results for CVB inference, providing evidence that CVB may be a better choice of inference algorithm for Bayesian formulations of generative grammar induction models such as DMV.

4.5.3 PCFGs with Latent Annotations

It is known that the standard PCFGs estimated by simply taking the empirical rule frequencies off treebanks are not accurate models to capture the syntactic structures in most natural languages, as demonstrated by Charniak (1997) and Klein and Manning (2003). Our third experiment is to apply our CVI algorithm to the PCFGs with latent annotations (PCFGs-LA) Matsuzaki et al. (2005), where each non-terminal symbol is augmented with hidden variables (or subtypes). Given a parsed corpus, training a PCFG-LA yields a finer grammar with the automatically induced features represented by the subtypes. For example, an augmented binary rule takes the form $A[a] \rightarrow B[b]D[d]$, where $a, b, d \in [1, H]$ are the hidden subtypes, and H denotes the number of subtypes for each non-terminal.

In our experiments, we adopt the methods in Petrov et al. (2006) to process the WSJ treebank: right binarizing the training trees and replacing infrequent words with the generic unknown word marker for English; for initialization, we add 1% randomness to the parameters θ to start the EM training, and we calculate the expected counts to initialize both VI and CVI. For the hyperparameters, we use a grid search for $\alpha \in [1, 0.1, 0.01, 0.001]$ on the development set (section 22) for both VI and CVI. Finally, we report the results on the section 23, using the best grammar tested on the development set from 5 random runs.

In Figure 4.2, when each non-terminal is split into 2 hidden subtypes, we show that our CVI algorithm outperforms the EM and VB algorithms in terms of all the evaluation

objectives. We also investigate the hidden state space with higher dimensions (4,8,16 subtypes), and find our CVI algorithm retains the advantages over the other two, whereas the VI algorithm fails to surpass its more restricted version, the EM algorithm.

4.6 Summary

In this chapter we have presented a collapsed variational inference algorithm for PCFGs. By making use of the common scenario where data consists of multiple sentences, we apply our second CVI approach in the previous chapter. A notable difference between HMMs and PCFGs is that the model structures of PCFGs are random objects, which require additional algebraic manipulation before our approximation method to joint posterior predictives can be applied, demonstrating the applicability of CVI to random model structures. In a range of parsing applications, our CVI algorithm produces more accurate results than standard VI, and close results to sampling with significantly less training time.

While not state-of-the-art, the models we have illustrated our CVI algorithm on are the core building blocks for a number of high performance parsers (Blunsom and Cohn, 2010; Petrov et al., 2006; Petrov and Klein, 2007; Liang et al., 2007). Therefore, our future work naturally extends to employing our CVI algorithm in more advanced models such as hierarchical splitting and merging system used in Berkeley parser (Petrov and Klein, 2007), and generalizing our CVI algorithm to the non-parametric models such as tree substitution grammars (Blunsom and Cohn, 2010) and infinite PCFGs (Liang et al., 2007). In particular, the generalization to infinite PCFGs can be easily achieved by following the algorithmic techniques we will develop in the next chapter.

Chapter 5

Collapsed Variational Inference for HDP-HMMs

Chapter Abstract

In this chapter, we derive a collapsed variational inference algorithm for hierarchical Dirichlet process hidden Markov models (HDP-HMMs), allowing for an unbounded number of states and statistical sharing. Essential to our algorithm is a set of novel techniques for computing the probabilities of each transition's existence, which are important ingredients of the expected table counts (Teh et al., 2008). Specifically, we propose an exact but expensive calculation method and some approximate techniques, which are empirically evaluated against the exact one (Wang and Blunsom, 2015a). Our techniques are widely applicable to many HDP-based computational linguistic models. In our experiments, our CVI algorithm for HDP-HMMs reliably determines the effective number of hidden states and significantly outperforms other uncollapsed variational benchmarks and (in large datasets) beam sampling as well.

5.1 Introduction

The hierarchical Dirichlet process (HDP) (Teh et al., 2006) is a Bayesian nonparametric prior for drawing multiple random measures on the same countably infinite set of supports with different weights. The infinite capacity and statistical sharing mechanism provided by the HDP makes it an ideal tool for managing related groups of data that share clusters, a

recurring theme in Bayesian statistics. In the past decade, the HDP has been widely applied to a number of mixed membership models including hidden Markov models reviewed by Airoldi et al. (2014). HDP-HMMs (Teh et al., 2006), otherwise known as infinite HMMs (Beal et al., 2002) are Bayesian nonparametric extensions of the finite state HMMs we have studied in Chapter 3. In practical applications of time series, we often do not know the number of states a priori. The HDP-HMMs can determine this automatically as well as improve the performance of HMMs by optimizing the HDP-related parameters.

The hierarchical structure of the HDP prior coupled with the sequential dependencies of HMMs pose a significant challenge to the posterior inference. Collapsed Gibbs sampling and beam sampling are two popular inference methods for HDP-HMMs. CGS (Teh et al., 2006) is the first algorithm for the HDP inference based on the Chinese restaurant franchise representation. For HDP-HMMs, it draws each hidden state variable separately conditioned on others, which can be slow to mix due to the strong coupling (Scott, 2002). Beam sampling (Van Gael et al., 2008) combines the slice sampling with dynamic programming to draw whole state trajectories. Although beam sampling often mixes relatively faster than CGS, both algorithms are not as efficient as the deterministic methods.

Variational inference is a promising efficient alternative, but it has been less well developed for HDP-HMMs until recently. This is arguably because the 'standard' variational inference[1] doest not exist due to the non-conjugate structure of the HDP, and thus some approximation must be introduced. To avoid non-conjugacy, Liang et al. (2007) and Johnson and Willsky (2014) proposed a point estimate strategy for the top level DP. Recently, with the aim of developing a full posterior estimation, Hughes et al. (2015b) proposed a surrogate variational bound for tractable inference; while Zhang et al. (2016) applied a two level stick breaking construction (Wang et al., 2011) to HDP-HMMs. Both algorithms require additional approximation to derive close-form updates.

Despite the empirical superiority of CVI over VI, the current variational algorithms do not take advantages of collapsing, i.e., marginalizing out model parameters. In this chapter, we explore this direction and propose a (stochastic) collapsed variational algorithm for HDP-HMMs, which is as efficient as the previous variational approaches, but significantly more accurate due to collapsing. Our algorithm builds on the work of CVI for HDP-LDA by Teh et al. (2008), who proposed a set of useful techniques to variationally deal with the

[1]By 'standard', we mean that given a factorization $q(\mathbf{y}) = \prod_{j \in \mathcal{J}} q(\mathbf{y}_j)$, the standard variational updates $q(\mathbf{y}_j) \propto \exp\{\mathbb{E}[\log p(\mathbf{y})]\}$ have closed form solutions without using any approximation.

HDP in collapsed space, including the direct assignment truncation, introducing auxiliary variables and approximating digamma functions. We divide our presentation into two parts. In Section 5.3, we derive closed-form updates (for batch CVI) and gradients (for stochastic CVI), employing the techniques in Teh et al. (2008). In Section 5.4, we address an inference challenge that arises because of the sequential dependencies of HDP-HMMs.

Putting it simple, the challenge is to compute a function's expectation under a Markov chain distribution. Jointly factorizing the target function and the Markov chain, we propose a message passing algorithm that is exact but has high computational cost. Then we propose three efficient approximate methods. First, we replace the Markov chain distribution with a mean field approximation, reducing the high computational cost by two orders of magnitude. Second, we estimate the expectation by sampling, using a form of ancestral sampling. Thirdly, we propose a heuristic method, which is intuitively interpretable, computationally efficient and very accurate evaluated against the exact method. Though we focus on HDP-HMMs, our techniques are generally applicable to a variety of HDP-based computational linguistic models such as HDP-PCFGs (Liang et al., 2007), sticky HDP-HMMs (Fox et al., 2008) and word segmentation models (Goldwater et al., 2009).

In Section 5.5, we validate our CVI algorithm for HDP-HMMs in several sequential analysis tasks. First, we create synthetic datasets so that the numbers of states are known. We truncate our CVI algorithm to larger values, and most of the time it converges to the correct number of states. Second, we compare our CVI algorithm with beam sampling (Van Gael et al., 2008) and two recent variational benchmarks (Johnson and Willsky, 2014; Zhang et al., 2016), showing that our algorithm performs the best in most cases. Finally, we revisit the textual datasets in Chapter 3. We compare our CVI algorithm for HDP-HMMs with the one for finite HMMs, demonstrating the usefulness of sharing statistical strength and optimizing concentration parameters of the HDP.

5.2 Definitions

The hierarchical Dirichlet process (Teh et al., 2006) is a distribution over a global random measure G and a set of secondary level random measures G_j coupled through the global one. Formally, the global random measure is distributed according a Dirichlet process with

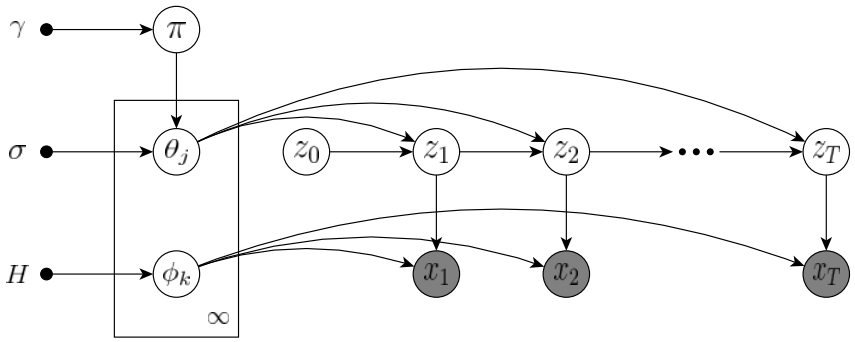

Figure 5.1: Graphical model representation of an HDP-HMM.

concentration parameter γ and base measure H,

$$G \sim \text{DP}(\gamma, H), \tag{5.1}$$

and each secondary level random measure is distributed according to the Dirichlet process with concentration parameter σ and the same global measure G,

$$G_j \sim \text{DP}(\sigma, G) \qquad \text{for } j \in \mathcal{J}. \tag{5.2}$$

In the stick breaking representation of the HDP, the random measures can be expressed explicitly as follows: $G = \sum_{k=1}^{\infty} \pi_k \delta_{\phi_k}$ and $G_j = \sum_{k=1}^{\infty} \theta_{j,k} \delta_{\phi_k}$, where $\pi = (\pi_1, \pi_2, ...) \sim \text{GEM}(\gamma)$, $\phi_k \sim H$ and $\theta_j \sim \text{DP}(\alpha, \pi)$. The GEM distribution is defined in Section 2.4,

$$\pi_k = \tilde{\pi}_k \prod_{l<k} (1 - \tilde{\pi}_l), \qquad \tilde{\pi}_k \sim \text{Beta}(1, \alpha), \qquad \text{for } k = 1, 2, ... \tag{5.3}$$

We see that all G_j have supports on the same set of points $\{\phi_k\}_{k=1}^{\infty}$ as G_0, which is the key to define HDP-HMMs. Without tying G_j, a state trajectory would keep entering into a new state that has never been entered before. The concentration parameter σ controls the variability of each θ_j around π, with a smaller value implying greater variability.

Let $\mathbf{x} = \{\mathbf{x}_t\}_{t=1}^T$ be a sequence of observations and $\mathbf{z} = \{z_t\}_{t=1}^T$ be the corresponding hidden sequence. Let the base measure H be a conjugate prior on emission distributions F parametrized by ϕ_k at each state k. By identifying each G_j as describing from state j, the probabilities of transitioning into each state k, namely $\theta_{j,k}$ and its associated emission

distribution $F(\phi_k)$, we can define the generative process of an HDP-HMM as follows,

$$\pi \sim \text{GEM}(\alpha), \tag{5.4}$$

$$\theta_j \sim \text{DP}(\alpha, \pi) \quad \text{for } j = 0, 1, 2, ..., \tag{5.5}$$

$$\phi_k \sim H \quad \text{for } k = 1, 2, ..., \tag{5.6}$$

$$z_t | z_{t-1}, \theta \sim \text{Discrete}(\theta_{z_{t-1}}) \quad \text{for } t = [1, T], \tag{5.7}$$

$$\mathbf{x}_t | z_t, \phi \sim F(\phi_{z_t}) \quad \text{for } t = [1, T]. \tag{5.8}$$

The graphical model representation is given in Figure 5.1. The joint probability of the parameters and variables is,

$$p(\mathbf{x}, \mathbf{z}, \theta, \phi | \gamma, \sigma) = p(\tilde{\pi}|\gamma) p(\theta|\tilde{\pi}, \sigma) p(\phi) \prod_{t=1}^{T} p(z_t | z_{t-1}, \theta) p(\mathbf{x}_t | z_t, \phi). \tag{5.9}$$

To be concrete, we focus on the discrete HDP-HMMs in the multiple sequence setting. The single long sequence setting is covered in Section 3.6, and the general CE family case is covered in Section 3.7. We briefly review our notations that are used in Chapter 3: for discrete emission, we have $H = \text{Dir}(\beta)$ and $F(\phi_k) = \text{Discrete}(\phi_k)$ with $\phi_{k,w} = p(x_t = w | z_t = k)$. In the multiple sequence setting, we have $\mathbf{x} = \{\mathbf{x}^i\}_{i=1}^I$ and $\mathbf{z} = \{\mathbf{z}^i\}_{i=1}^I$, where $\mathbf{x}^i$ and $\mathbf{z}^i$ are the i^{th} observed and hidden sequences, respectively; and $\mathbf{x}^i = \{x_t^i\}_{t=1}^T$ and $\mathbf{z}^i = \{z_t^i\}_{t=1}^T$, where x_t^i and z_t^i are their t^{th} observed and hidden variables, respectively. As usual we use $C_{j,k}$ and $C_{k,w}$ to denote the transition and emission counts, respectively; and $C^{\setminus i}$ to denote the exclusion of the contribution from the i^{th} subsequence.

Unlike the previous two chapters, although we use the beam sampler (Van Gael et al., 2008) and variational inference algorithms (Johnson and Willsky, 2014; Zhang et al., 2016) as comparing benchmarks, they do not share the same derivations with our CVI algorithm. Hence, we skip reviewing them here.

5.3 CVI for HDP-HMMs

In this section and next, we present collapsed variational inference for HDP-HMMs. For a concise presentation, we focus on the aspects relating to the HDP, avoiding the duplicate materials relating to finite HMMs in Chapter 3. Further, our attention is specifically given to the novel aspects of our algorithm; omitting the derivation steps that use existing methods with algebraic manipulation. The detailed derivations can be found in Appendix 5.A.

The goal of collapsed variational inference is to approximate the true posterior in collapsed space, $p(\mathbf{z}, \tilde{\pi}|\mathbf{x}, \gamma, \sigma)$ that is intractable to compute. We also aim to update the concentration parameters so that our algorithm is not sensitive to their initial values. We start with integrating out the parameters θ and ϕ. By the conjugacy property 2 in Section 2.3, the complete-data marginal is as follows,

$$p(\mathbf{x}, \mathbf{z}|\tilde{\pi}, \sigma) = \prod_{j=0}^{K} \frac{\Gamma(\sigma)}{\Gamma(\sigma + C_{j,\cdot})} \left(\prod_{k=1}^{K} \frac{\Gamma(\sigma\pi_k + C_{j,k})}{\Gamma(\sigma\pi_k)} \right) \times p(\mathbf{x}|\mathbf{z}), \tag{5.10}$$

in which K is a function of $\mathbf{z}$ and defined to be the index such that $\forall i, t,\ z_t^i \leq K$. We do not expand $p(\mathbf{x}|\mathbf{z})$, since here we focus on the HDP part.

Teh et al. (2008) pointed out that the Gamma functions in (5.10) can be inconvenient for deriving certain variational posteriors. Following Teh et al. (2008), we replace some of them with summing over the auxiliary variables $\mathbf{s} = \{s_{j,k}\}$,

$$\frac{\Gamma(\sigma\pi_k + C_{j,k})}{\Gamma(\sigma\pi_k)} = \sum_{s_{j,k}=1}^{C_{j,k}} \begin{bmatrix} C_{j,k} \\ s_{j,k} \end{bmatrix} (\sigma\pi_k)^{s_{j,k}}, \tag{5.11}$$

where $\begin{bmatrix} C \\ s \end{bmatrix}$ are Stirling numbers of the first kind. The auxiliary variables have interpretable meanings in the Chinese Restaurant Franchise representation: $s_{j,k} \in [0, C_{j,k}]$ is the random number of tables serving the dish k in the j^{th} restaurant. The above equation (5.11) holds by properties of Stirling number of the first kind, included in Appendix 5.A.

Then the expanded probability distribution is,

$$p(\mathbf{x}, \mathbf{z}, \mathbf{s}|\sigma, \pi) = \prod_{j=0}^{K} \left(\frac{\Gamma(\sigma)}{\Gamma(\sigma + C_{j,\cdot})} \prod_{k=1}^{K} \begin{bmatrix} C_{j,k} \\ s_{j,k} \end{bmatrix} (\sigma\pi_k)^{s_{j,k}} \right) \times p(\mathbf{x}|\mathbf{z}). \tag{5.12}$$

We can verify that summing over $\mathbf{s}$ reduces (5.12) to (5.10). Including the auxiliary variables $\mathbf{s}$, we introduce a variational distribution $q(\mathbf{z}, \tilde{\pi}, \mathbf{s})$ and form the following evidence lower bound on the log marginal likelihood,

$$\log p(\mathbf{x}) \geq \mathbb{E}[\log p(\mathbf{x}, \mathbf{z}, \mathbf{s}|\tilde{\pi}, \sigma)] + \mathbb{E}[\log p(\tilde{\pi}|\gamma)] - \mathbb{E}[\log q(\mathbf{z}, \tilde{\pi}, \mathbf{s})] \triangleq \mathcal{L}^{\text{CVI}}. \tag{5.13}$$

We assume that the variational distribution factorizes as follows,

$$q(\mathbf{z}, \tilde{\pi}, \mathbf{s}) = \prod_{i=1}^{I} q(\mathbf{z}^i) \prod_{j=0}^{\infty} \prod_{k=1}^{\infty} q(s_{j,k}|\mathbf{z}) \prod_{k=1}^{\infty} \text{Beta}(\tilde{\pi}_k|u_k, v_k). \tag{5.14}$$

We have modelled the exact dependence of $\mathbf{s}$ on $\mathbf{z}$. This is equivalent to marginalize out $\mathbf{s}$ with respect to $\mathbf{z}$ (Teh et al., 2007). For tractable inference, we truncate the variational distribution. We choose to use the direct assignment truncation strategy (Teh et al., 2008), which has a number of advantages over other choices. Specifically, we truncate the hidden sequence distributions by fixing $q(\mathbf{z} = 0)$ when any $z_t^i > K$, where K is the truncation level we choose. With this assumption, the global variables $(\tilde{\pi}, \mathbf{s}, \theta, \phi)$ with indices greater than K are conditionally independent of the observations; and thus we can ignore updating their posteriors. This strategy is also nested over K: q with truncation $K+1$ can represent another q with truncation K by setting its final component to zero. Hence, K can be treated as a parameter and optimized, for example, by the split-merge search method (Bryant and Sudderth, 2012), though we have not explored this direction yet.

Having defined $\mathcal{L}^{\text{CVI}}$ and truncated q, we are ready to derive a set of update equations that optimize $\mathcal{L}^{\text{CVI}}$. For each update, we state our derivation approach and key results. The complete derivation steps are provided in Appendix 5.A.

Concentration Parameters We derive the partial gradients of $\mathcal{L}^{\text{CVI}}$ w.r.t. concentration parameters γ and σ, which are used in the stochastic CVI algorithm. Setting the gradients to zeros yields closed form solutions that are used in the batch CVI algorithm. Central to our gradient derivation is simplifying the relative terms in $\mathcal{L}^{\text{CVI}}$ so that they become explicit functions of the target parameters. In $\mathcal{L}^{\text{CVI}}$, the terms relating to γ can be simplified to,

$$\begin{aligned}\mathbb{E}[\log p(\tilde{\pi}|\gamma)] = K \log \Gamma(1+\gamma) - K \log \Gamma(\gamma) \\ + (\gamma - 1) \sum_{k=1}^{K} (\psi(v_k) - \psi(u_k + v_k)).\end{aligned} \tag{5.15}$$

Taking the partial gradient w.r.t. γ, we have,

$$\frac{\partial \mathcal{L}}{\partial \gamma} = \frac{K}{\gamma} + \sum_{k=1}^{K} \psi(v_k) - \psi(u_k + v_k). \tag{5.16}$$

Setting the gradient to zero yields,

$$\gamma = \frac{K}{\sum_{k=1}^{K} \psi(u_k + v_k) - \psi(v_k)}. \tag{5.17}$$

In $\mathcal{L}^{\text{CVI}}$, the terms relating to σ can be simplified to (const denotes the dropped terms that are constant w.r.t. σ),

$$\begin{aligned}\mathbb{E}[\log p(\mathbf{x}, \mathbf{z}, \mathbf{s}|\sigma, \tilde{\pi})] = &\sum_{j=0}^{K}(\log\Gamma(\sigma) - \mathbb{E}[\log\Gamma(\sigma + C_{j,\cdot})]) \\ &+ \log\sigma \sum_{j=0}^{K}\sum_{k=1}^{K}\mathbb{E}[s_{j,k}] + \text{const.}\end{aligned} \tag{5.18}$$

Taking the partial gradient w.r.t. σ, we have,

$$\frac{\partial\mathcal{L}}{\partial\sigma} = \sum_{j=0}^{K}(\psi(\sigma) - \mathbb{E}[\psi(\sigma + C_{j,\cdot})]) + \frac{1}{\sigma}\sum_{j=0}^{K}\sum_{k=1}^{K}\mathbb{E}[s_{j,k}]. \tag{5.19}$$

Setting the gradient to zero yields,

$$\sigma = \frac{\sum_{j=0}^{K}\sum_{k=1}^{K}\mathbb{E}[s_{j,k}]}{\sum_{j=0}^{K}\mathbb{E}[\psi(\sigma + C_{j,\cdot})] - \psi(\sigma)}. \tag{5.20}$$

Top Level Stick As with the concentration parameters, we derive the partial gradients of $\mathcal{L}^{\text{CVI}}$ w.r.t. u_k and v_k. A more standard approach for deriving the batch CVI updates is taking the functional derivative of $\mathcal{L}^{\text{CVI}}$ w.r.t. $q(\tilde{\pi}_k)$ and equating it to zero. We derive the Euclidean gradients to give both the batch and stochastic CVI updates at the same time. In $\mathcal{L}^{\text{CVI}}$, the terms relating to u_k and v_k can be simplified to,

$$\begin{aligned}\mathbb{E}[\log p(\mathbf{x}, \mathbf{z}, \mathbf{s}|\sigma, \tilde{\pi})] =& \mathbb{E}[s_{\cdot,k}](\psi(u_k) - \psi(u_k + v_k)) \\ &+ \mathbb{E}[s_{\cdot,>k}](\psi(v_k) - \psi(u_k + v_k)) + \text{const},\end{aligned} \tag{5.21}$$

$$\begin{aligned}\mathbb{E}[\log q(\tilde{\pi}_k)] =& \log\Gamma(u_k + v_k) - \log\Gamma(u_k) - \log\Gamma(v_k) \\ &+ (u_k - 1)(\psi(u_k) - \psi(u_k + v_k)) \\ &+ (v_k - 1)(\psi(v_k) - \psi(u_k + v_k)),\end{aligned} \tag{5.22}$$

in which dot denotes the summed out column, and $> k$ denotes summing over the indices that are greater than k. Taking the partial gradients w.r.t. u_k and v_k, we have,

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial u_k} =& \psi'(u_k + v_k)(u_k + v_k - \mathbb{E}[s_{\cdot,\geq k}] - \gamma - 1) \\ &- \psi'(u_k)(u_k - \mathbb{E}[s_{\cdot,k}] - 1),\end{aligned} \tag{5.23}$$

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial v_k} =& \psi'(u_k + v_k)(u_k + v_k - \mathbb{E}[s_{\cdot,\geq k}] - \gamma - 1) \\ &- \psi'(v_k)(v_k - \mathbb{E}[s_{\cdot,>k}] - \gamma).\end{aligned} \tag{5.24}$$

Setting the gradients to zeros yields,

$$u_k = 1 + \mathbb{E}[s_{\cdot,k}], \tag{5.25}$$

$$v_k = \gamma + \mathbb{E}[s_{\cdot,>k}]. \tag{5.26}$$

Auxiliary Variables The conditional posterior $q(\mathbf{s}|\mathbf{z})$ are used for calculating the expected number of tables $\mathbb{E}[s_{j,k}]$. We take the functional derivative of $\mathcal{L}^{\text{CVI}}$ w.r.t. $q(\mathbf{s}|\mathbf{z})$ and equate it to zero (the Lagrange multiplier term enforcing normalisation is omitted for brevity),

$$\begin{aligned}\frac{\partial \mathcal{L}^{\text{CVI}}}{\partial q(\mathbf{s}|\mathbf{z})} &= \sum_{\mathbf{z}} q(\mathbf{z}) \mathbb{E}_{q(\tilde{\pi})}[\log p(\mathbf{x}, \mathbf{z}, \mathbf{s}|\tilde{\pi}, \sigma)] - \sum_{\mathbf{z}} q(\mathbf{z}) \log q(\mathbf{s}|\mathbf{z}) - 1 \\ &= 0. \end{aligned} \tag{5.27}$$

Solving the above equation, we derive,

$$q(s_{j,k}|\mathbf{z}) = \frac{\Gamma(\sigma \mathbb{G}[\pi_k])}{\Gamma(\sigma \mathbb{G}[\pi_k] + C_{j,k})} \begin{bmatrix} C_{j,k} \\ s_{j,k} \end{bmatrix} (\sigma \mathbb{G}[\pi_k])^{s_{j,k}}, \tag{5.28}$$

in which $\mathbb{G}[x]$ denotes geometric expectation. $\mathbb{G}[\pi_k]$ can be computed as follows: $\mathbb{G}[\pi_k] = \mathbb{G}[\tilde{\pi}_k] \prod_{l<k} \mathbb{G}[1-\tilde{\pi}_l]$; using properties of the Beta distribution, each $\mathbb{G}[\tilde{\pi}_k] = \exp\{\psi(u_k) - \psi(u_k + v_k)\}$ and $\mathbb{G}[1-\tilde{\pi}_k] = \exp\{\psi(v_k) - \psi(u_k + v_k)\}$.

Expected Digamma Functions The expected table counts, $\mathbb{E}[s_{j,k}]$ are used for updating σ, u_k and v_k. Since they are w.r.t. both $q(\mathbf{z})$ and $q(\mathbf{s}|\mathbf{z})$, we first compute $\mathbb{E}_{q(\mathbf{s}|\mathbf{z})}[s_{j,k}]$ using a property (in 5.87) of Stirling numbers of the first kind,

$$\begin{aligned} &\mathbb{E}_{q(\mathbf{z})}[\mathbb{E}_{q(\mathbf{s}|\mathbf{z})}[s_{j,k}]] \\ &= \mathbb{E}_{q(\mathbf{z})}\left[\sum_{s_{j,k}=1}^{C_{j,k}} \frac{\Gamma(\sigma \mathbb{G}[\pi_k])}{\Gamma(\sigma \mathbb{G}[\pi_k] + C_{j,k})} \begin{bmatrix} C_{j,k} \\ s_{j,k} \end{bmatrix} (\sigma \mathbb{G}[\pi_k])^{s_{j,k}} s_{j,k} \right] \quad \text{(by 5.28)} \\ &= \sigma \mathbb{G}[\pi_k] (\mathbb{E}_{q(\mathbf{z})}[\psi(\sigma \mathbb{G}[\pi_k] + C_{j,k})] - \psi(\sigma \mathbb{G}[\pi_k])) \quad \text{(by 5.87)} \end{aligned} \tag{5.29}$$

Since the expected digamma function in (5.29) is intractable, we seek to use the Taylor approximation, as we have done several times for the log functions in Chapter 3. Nevertheless, a complication for the digamma function pointed out by Teh et al. (2008) is that, when x approaches zero, $\psi(x)$ diverges more quickly than $\log x$ and thus the Taylor approximation is inappropriate. Following Teh et al. (2008), we consider two sets of $\mathbf{z}$,

$$A \triangleq \{\mathbf{z} : C_{j,k}(\mathbf{z}) = 0\}, \tag{5.30}$$

$$B \triangleq \{\mathbf{z} : C_{j,k}(\mathbf{z}) > 0\}, \tag{5.31}$$

and introduce the following shorthand notations,

$$q(C_{j,k} = 0) \triangleq \sum_{\mathbf{z} \in A} q(\mathbf{z}), \tag{5.32}$$

$$q(C_{j,k} > 0) \triangleq \sum_{\mathbf{z} \in B} q(\mathbf{z}), \tag{5.33}$$

$$\mathbb{E}_+[C_{j,k}] \triangleq \frac{\mathbb{E}[C_{j,k}]}{q(C_{j,k} > 0)}. \tag{5.34}$$

To compute the expected digamma function in (5.29), we treat $\mathbf{z} \in A$ exactly and apply the first order Taylor approximation when $\mathbf{z} \in B$ (i.e., $C_{j,k}(\mathbf{z}) \geq 1$), deriving,

$$\begin{aligned} &\mathbb{E}_{q(\mathbf{z})}[\mathbb{E}_{q(\mathbf{s}|\mathbf{z})}[s_{j,k}]] \\ &\approx \sigma \mathbb{G}[\pi_k] q(C_{j,k} > 0)(\psi(\sigma \mathbb{G}[\pi_k] + \mathbb{E}_+[C_{j,k}]) - \psi(\sigma \mathbb{G}[\pi_k])). \end{aligned} \tag{5.35}$$

We see that the quantity, $q(C_{j,k} > 0)$ (or $q(C_{j,k} = 0)$), roughly interpreted as the likelihood of whether (or not) there exists a transition from state j to state k, is an important ingredient of the expected table count, $\mathbb{E}[s_{j,k}]$. However, unlike HDP-LDA, this quantity in HDP-HMMs is not straightforward to compute due to the sequential dependencies. We investigate its exact computation and some approximate alternatives in the next section.

In the update equations for σ (5.19-5.20), we have also encountered expected digamma functions, $\mathbb{E}[\psi(\sigma + C_{j,\cdot})]$. Following a similar approach (splitting $\mathbf{z}$ into two cases: $C_{j,\cdot}(\mathbf{z}) = 0$ and $C_{j,\cdot}(\mathbf{z}) > 0$, and applying the Taylor approximation to the second case), the expected digamma functions in (5.19-5.20) can be computed as follows,

$$\psi(\sigma) - \mathbb{E}[\psi(\sigma + C_{j,\cdot})] \approx q(C_{j,\cdot} > 0)(\psi(\sigma) - \psi(\sigma + \mathbb{E}_+[C_{j,\cdot}])), \tag{5.36}$$

where $q(C_{j,\cdot} > 0) \triangleq \sum_{\mathbf{z} \in B'} q(\mathbf{z})$, $B' \triangleq \{\mathbf{z} : C_{j,\cdot}(\mathbf{z}) > 0\}$, and $\mathbb{E}_+[C_{j,\cdot}] \triangleq \frac{\mathbb{E}[C_{j,\cdot}]}{q(C_{j,\cdot} > 0)}$. The quantity, $q(C_{j,\cdot} > 0)$ can be computed using the same approaches for $q(C_{j,k} > 0)$.

Hidden Sequences It has been proved that given two sets of hidden variables, D and E, preserving the exact dependence of D on E is equivalently marginalizing out D (Teh et al., 2007). Therefore, we consider $\mathbf{s}$ together with (θ, ϕ) to be integrated out w.r.t. $\mathbf{z}$ and derive the variational posterior for each hidden sequence,

$$q(\mathbf{z}^i) \propto \prod_{t=1}^{T} \bar{\theta}_{z^i_{t-1}, z^i_t} \bar{\phi}_{z^i_t, x^i_t}, \tag{5.37}$$

in which we define the surrogate parameters as follows,

$$\bar{\theta}_{j,k} = \frac{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C_{j,k}^{\setminus i}] + \sigma\mathbb{G}[\pi_k]}{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C_{j,\cdot}^{\setminus i}] + \sigma}, \tag{5.38}$$

$$\bar{\phi}_{k,w} = \frac{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C_{k,w}^{\setminus i}] + \beta}{\mathbb{E}_{q(\mathbf{z}^{\setminus i})}[C_{\cdot,k}^{\setminus i}] + W\beta}. \tag{5.39}$$

We can see that the above updates only differ from those (3.48-3.50) for finite HMMs in priors of transition parameters $\bar{\theta}_{j,k}$: symmetric α are replaced by asymmetric $\sigma\mathbb{G}[\pi_k]$.

Stochastic CVI We extend our CVI algorithm for HDP-HMMs to the stochastic setting. In Section 3.7, for finite HMMs, we have proposed to iterate between learning local posteriors by (3.98- 3.100) and updating global counts by (3.103-3.104). For the HMM part, the only modification required for HDP-HMMs is replacing the symmetric α with asymmetric $\sigma\mathbb{G}[\pi_k]$ in surroagate transition parameters $\bar{\theta}_{j,k}$. Hence, we do not repeat the details here.

For the HDP-related hyperparameters (γ, σ) and variational distribution $q(\tilde{\pi}_k)$, they are conditionally independent of data sizes given those expected digamma functions in (5.35-5.36). Therefore, stochastic inference is not necessary for them, and we can batch update them after processing all sequences. Alternatively, we propose to optimize the variational bound by gradient ascent after processing a minibatch of sequences. Specifically, we treat u_k and v_k parametrizing $q(\tilde{\pi}_k)$ as hyperparameters too. Let (m) denote the m^{th} iteration (or minibatch). For any hyperparameter $x \in \{\gamma, \sigma\} \cup \{u_k\}_{k=1}^K \cup \{v_k\}_{k=1}^K$,

$$x^{(m+1)} = x^{(m)} + \rho_m \frac{\partial \mathcal{L}^{\text{CVI}}}{\partial x^{(m)}}, \tag{5.40}$$

where the partial gradients have been derived in (5.16), (5.19) and (5.23-5.24).

In terms of implementation, the expected digamma functions in (5.35-5.36) require the following statistics, $\mathbb{E}[C_{j,k}] = \sum_m \mathbb{E}[C_{j,k}^m]$, $q(C_{j,k} > 0) = 1 - \prod_m q(C_{j,k}^m = 0)$ and $q(C_{j,\cdot} > 0) = 1 - \prod_m q(C_{j,\cdot}^m = 0)$, where those with the superscript m denote the corresponding local statistics for the m^{th} minibatch. The decomposition of $q(C_{j,k} > 0)$ and $q(C_{j,\cdot} > 0)$ are explained in the next section. For each minibatch, we keep track of these local statistics, which are replaced by their new values after processing that minibatch.

5.4 Algorithms for Computing $q(C_{j,k} > 0)$

In this section, we address the challenge of computing $q(C_{j,k} > 0)$ and $q(C_{j,\cdot} > 0)$, which is the only remaining problem in our CVI algorithm for HDP-HMMs. Since $q(C_{j,\cdot} > 0)$ is a special case of $q(C_{j,k} > 0)$ as we will see, it is sufficient to focus on $q(C_{j,k} > 0)$[2]. We first formalize the problem by investigating the structure of $q(\mathbf{z})$. Then we propose an exact algorithm and several approximate techniques that are empirically analysed.

5.4.1 Problem Formalization

By our definitions in (5.30-5.34), since the set A is the complement of the set B, the following series of equations hold trivially,

$$q(C_{j,k} > 0) \triangleq \sum_{\mathbf{z} \in B} q(\mathbf{z}) = 1 - \sum_{\mathbf{z} \in A} q(\mathbf{z}) \triangleq 1 - q(C_{j,k} = 0). \tag{5.41}$$

Let $C^i_{j,k}(\mathbf{z}^i)$ denote the number of transitions from state j to state k in the i^{th} sequence. The membership condition of an arbitrary $\mathbf{z}$ belonging to the set A, $\delta(C_{j,k}(\mathbf{z}) = 0)$ can be factorized as follows,

$$\delta(C_{j,k}(\mathbf{z}) = 0) = \prod_{i}^{I} \delta(C^i_{j,k}(\mathbf{z}^i) = 0), \tag{5.42}$$

where each term in RHS can be further factorized as follows,

$$\delta(C^i_{j,k}(\mathbf{z}^i) = 0) = \prod_{t=1}^{T} \delta(z^i_{t-1}, z^i_t \neq j, k). \tag{5.43}$$

We will use (5.41-5.43) to decompose $q(C_{j,k} > 0)$. Before getting into it, we briefly show why $q(C_{j,\cdot} > 0)$ is a special case of $q(C_{j,k} > 0)$. By its definition,

$$q(C_{j,\cdot} > 0) \triangleq \sum_{\mathbf{z} \in B'} q(\mathbf{z}) = 1 - \sum_{\mathbf{z} \in A'} q(\mathbf{z}), \tag{5.44}$$

and the membership condition of an arbitrary $\mathbf{z}$ belonging to the set A', $\delta(C_{j,\cdot}(\mathbf{z}) = 0)$ can similarly factorized,

$$\delta(C_{j,\cdot}(\mathbf{z}) = 0) = \prod_{i}^{I} \delta(C^i_{j,\cdot}(\mathbf{z}^i) = 0) = \prod_{i=1}^{I} \prod_{t=1}^{T} \delta(z^i_{t-1}, z^i_t \neq j, -1). \tag{5.45}$$

[2]If the concentration parameters are set to be fixed, we do not need to compute $q(C_{j,\cdot} > 0)$.

It is clear that (5.45) is a special case of (5.43), and therefore $q(C_{j,\cdot} > 0)$ is a special case of $q(C_{j,k} > 0)$.

Now we decompose $q(C_{j,k} > 0)$ using (5.41-5.42) as follows,

$$\begin{aligned}
q(C_{j,k} > 0) &= 1 - \sum_{\mathbf{z} \in A} q(\mathbf{z}) \quad \text{(by (5.41))} \\
&= 1 - \sum_{\mathbf{z}} q(\mathbf{z}) \delta(C_{j,k}(\mathbf{z}) = 0) \quad \text{(by (5.30))} \\
&= 1 - \sum_{\mathbf{z}} \prod_{i=1}^{I} \underbrace{q(\mathbf{z}^i) \delta(C^i_{j,k}(\mathbf{z}^i) = 0)}_{\text{a function of } \mathbf{z}^i} \text{(by (5.14), (5.42))} \\
&= 1 - \prod_{i=1}^{I} \sum_{\mathbf{z}^i} q(\mathbf{z}^i) \delta(C^i_{j,k}(\mathbf{z}) = 0) \\
&\triangleq 1 - \prod_{i=1}^{I} q(C^i_{j,k} = 0) \qquad (5.46)
\end{aligned}$$

We see that the global statistic $q(C_{j,k} > 0)$ is composed of local statistics $q(C^i_{j,k} = 0)$ that we define in the last line of (5.46). In (5.46), if we replace the superscript i (indexing sequences) with m (indexing minibatches), a similar decomposition will follow. Hence, as we have mentioned in the previous section, $q(C_{j,k} > 0) = 1 - \prod_m q(C^m_{j,k} = 0)$ holds.

We proceed to compute $q(C^i_{j,k} = 0)$. We start with investigating the structure of $q(\mathbf{z}^i)$, whose update equation is given by,

$$q(\mathbf{z}^i) \propto \prod_{t=1}^{T} \bar{\theta}_{z^i_{t-1}, z^i_t} \bar{\phi}_{z^i_t, x^i_t}. \qquad (5.47)$$

One fast way to realize the structure of $q(\mathbf{z}^i)$ is as follows: we treat $\bar{\theta}_{z^i_{t-1}, z^i_t}$ and $\bar{\phi}_{z^i_t, x^i_t}$ as potential functions over a Markov random field and run a junction tree algorithm (Jensen, 1996). It is known that the resulting $q(\mathbf{z}^i)$ will be a product of cluster marginals divided by a product of separator marginals. Then it follows immediately that $q(\mathbf{z}^i)$ is a Markov chain. Expressing this idea mathematically, we have,

$$q(\mathbf{z}^i) = \frac{\prod_{t=1}^{T} q(z^i_{t-1}, z^i_t)}{\prod_{t=1}^{T-1} q(z^i_t)} = q(z^i_1) \prod_{t=2}^{T} q(z^i_t | z^i_{t-1}). \qquad (5.48)$$

Alternatively, we can formally show the structure of $q(\mathbf{z}^i)$. Recall that $q(\mathbf{z}^i)$ is computed by the standard forward backward algorithm in the multiple sequence setting in Section 3.5. We define the forward variable $\alpha^i_t(z^i_t) \triangleq p(z^i_t, \mathbf{x}^i_{1:t} | \bar{\theta}, \bar{\phi})$, and the backward variable

$\beta_t^i(z_t^i) \triangleq p(\mathbf{x}_{t+1:T}^i | z_t^i, \bar{\theta}, \bar{\phi})$. Let Z denote the normalization constant of (5.47). Our claim can be deduced by the following arguments ('$\Leftrightarrow$' reads as 'is equivalent to show'),

$$
\begin{aligned}
&\text{Claim: } q(\mathbf{z}^i) = q(z_1^i) \prod_{t=2}^{T} q(z_t^i | z_{t-1}^i) \\
&\Leftrightarrow q(\mathbf{z}^i) = q(z_1^i) \prod_{t=2}^{T} \frac{q(z_{t-1}^i, z_t^i)}{q(z_{t-1}^i)} \\
&\Leftrightarrow q(\mathbf{z}^i) = q(z_1^i) \prod_{t=2}^{T} \frac{\cancel{\alpha_{t-1}^i(z_{t-1}^i)} \bar{\theta}_{z_{t-1}^i, z_t^i} \bar{\phi}_{z_t^i, x_t^i} \beta_t^i(z_t^i) Z}{\cancel{\alpha_{t-1}^i(z_{t-1}^i)} \beta_{t-1}^i(z_{t-1}^i) Z} \quad \text{(by (3.16)-(3.17))} \\
&\Leftrightarrow q(\mathbf{z}^i) = q(z_1^i) \prod_{t=2}^{T} \bar{\theta}_{z_{t-1}^i, z_t^i} \bar{\phi}_{z_t^i, x_t^i} \prod_{t=2}^{T} \frac{\beta_t^i(z_t^i)}{\beta_{t-1}^i(z_{t-1}^i)} \\
&\Leftrightarrow q(\mathbf{z}^i) = \frac{\alpha_1^i(z_1^i) \cancel{\beta_1^i(z_1^i)}}{Z} \prod_{t=2}^{T} \bar{\theta}_{z_{t-1}^i, z_t^i} \bar{\phi}_{z_t^i, x_t^i} \frac{\beta_T^i(z_T^i)}{\cancel{\beta_1^i(z_1^i)}} \\
&\Leftrightarrow q(\mathbf{z}^i) = \frac{\bar{\theta}_{z_1^i, z_0^i} \bar{\phi}_{z_1^i, x_1^i}}{Z} \prod_{t=2}^{T} \bar{\theta}_{z_{t-1}^i, z_t^i} \bar{\phi}_{z_t^i, x_t^i} \\
&\Leftrightarrow q(\mathbf{z}^i) = \frac{1}{Z} \prod_{t=1}^{T} \bar{\theta}_{z_{t-1}^i, z_t^i} \bar{\phi}_{z_t^i, x_t^i}. \quad \square
\end{aligned} \tag{5.49}
$$

It might be worth mentioning that in the single long sequence setting, the posterior of each subsequence, $q(\mathbf{z}^i)$ is also a Markov chain. This can be proved by similar arguments, where the forward and backward variables are replaced by the analogous messages in our sum product algorithm in Section 3.6. Therefore, our problem is to compute an expected delta function w.r.t. a distribution over Markov chains,

$$
q(C_{j,k}^i = 0) \triangleq \sum_{\mathbf{z}^i} q(\mathbf{z}^i) \delta(C_{j,k}^i(\mathbf{z}^i) = 0). \tag{5.50}
$$

5.4.2 An Exact Algorithm

To compute (5.50), one naive approach is summing over all possible Markov chains. This approach has the exponential computational complexity $O(K^T)$ and thus is intractable. In this subsection and next, we exploit some dynamic programming tricks.

We consider two cases: $j = 0$ and $j \in [1, K]$. When $j = 0$ (the initial state), the delta function in (5.50) is true if and only if $z_1^i \neq k$. Thus in this case, (5.50) is straightforward

to compute as follows,

$$\sum_{\mathbf{z}^i} q(\mathbf{z}^i)\delta(C^i_{0,k}(\mathbf{z}^i)=0) = \sum_{z^i_1} q(z^i_1)\delta(z^i_1 \neq k) = 1 - q(z^i_1 = k). \tag{5.51}$$

When $j \in [1, K]$, we have shown that the delta function in (5.50) can be factorized as a product of deltas over smaller pieces, i.e., pairs of variables,

$$\delta(C^i_{j,k}(\mathbf{z}^i)=0) = \prod_{t=2}^{T} \delta(z^i_{t-1}, z^i_t \neq j, k). \tag{5.52}$$

Then we can jointly factorize $q(\mathbf{z}^i)$ and $\delta(C^i_{j,k}(\mathbf{z})=0)$ in (5.50) as follows,

$$\begin{aligned}
&\sum_{\mathbf{z}^i} q(\mathbf{z}^i)\delta(C^i_{j,k}(\mathbf{z})=0) \\
&= \sum_{\mathbf{z}^i} q(z^i_1)\prod_{t=2}^{T} q(z^i_t|z^i_{t-1}) \prod_{t=2}^{T} \delta(z^i_{t-1}, z^i_t \neq j, k) \quad \text{(by (5.49), (5.52))} \\
&= \sum_{\mathbf{z}^i} q(z^i_1)\prod_{t=2}^{T} \underbrace{q(z^i_t|z^i_{t-1})\delta(z^i_{t-1}, z^i_t \neq j, k)}_{\triangleq f^i_{t-1}(z^i_{t-1}, z^i_t)} \\
&\triangleq \sum_{\mathbf{z}^i} q(z^i_1)\prod_{t=2}^{T} f^i_{t-1}(z^i_{t-1}, z^i_t).
\end{aligned} \tag{5.53}$$

By the distributive property of multiplication over addition, which is the essential idea of all sum product algorithms, we manipulate (5.53) as follows,

$$\begin{aligned}
&\sum_{\mathbf{z}^i} q(z^i_1)\prod_{t=2}^{T} f^i_{t-1}(z^i_{t-1}, z^i_t) \\
&= \sum_{z^i_T}\sum_{z^i_{T-1}}\cdots\sum_{z^i_2}\sum_{z^i_1} f^i_{T-1}(z^i_{T-1}, z^i_T) f^i_{T-2}(z^i_{T-2}, z^i_{T-1})\cdots f^i_1(z^i_1, z^i_2) q(z^i_1) \\
&= \sum_{z^i_T}\left[\sum_{z^i_{T-1}} f^i_{T-1}(z^i_{T-1}, z^i_T)\cdots\left[\sum_{z^i_2} f^i_2(z^i_2, z^i_3)\left[\underbrace{\sum_{z^i_1} f^i_1(z^i_1, z^i_2) q(z^i_1)}_{m^i_2(z^i_2)}\right]\right]\cdots\right].
\end{aligned} \tag{5.54}$$

We recognize that the above computations can be performed in a message passing style, with the starting message indicated in the last line of (5.54). In general, the t^{th} message, $m^i_t(z^i_t)$ is computed recursively as follows,

$$m^i_t(z^i_t) = \sum_{z^i_{t-1}} f^i_{t-1}(z^i_{t-1}, z^i_t) m^i_{t-1}(z^i_{t-1}), \tag{5.55}$$

and our goal, $q(C^i_{j,k} = 0)$ is achieved by the final summation,

$$q(C^i_{j,k} = 0) = \sum_{z^i_T} m^i_T(z^i_T). \tag{5.56}$$

Finally, we work out the computational cost of our message passing (or sum-product) algorithm. To compute a message, $m^i_t(z^i_t)$, we perform K^2 multiplications to obtain a table of $K \times K$ numbers and sum over z^i_{t-1} for each value of z^i_t. Thus the computational cost for each message is $O(K^2)$. Since we need to perform T such operations to get the final result in (5.56), the computational cost is $O(TK^2)$. Therefore, the total computational cost for $q(C_{j,k} > 0) = 1 - \prod_i q(C^i_{j,k} = 0)$ for all (j, k) pairs is $O(ITK^4)$.

5.4.3 Approximate Algorithms

The exact algorithm can be computationally too expensive for practical applications, especially when large truncations are used (normally, we choose truncation levels that are larger than our expected number of states). In this subsection, we propose three computationally efficient algorithms for approximating $q(C^i_{j,k} = 0)$. The first two are more mathematically principled, using two well acknowledged approximations, namely the mean field approximation and sampling approximation, respectively. The third algorithm, which surprisingly performs the best is based on a simple heuristic idea.

Mean Field Approximation For $q(\mathbf{z}^i)$, we assume a simpler structure than Markov chain (first order dependence), namely the complete factorization,

$$q(\mathbf{z}^i) \approx \prod_{t=1}^{T} q(z^i_t). \tag{5.57}$$

Given the empirical performances of our first CVI algorithm for HMMs in Section 3.4, this mean field approximation has potential to be accurate. Following similar calculations as in (5.53) and (5.54), again we can compute $q(C^i_{j,k} = 0)$ in a message passing style,

$$\begin{aligned}
&q(C^i_{j,k} = 0) \\
&= \sum_{\mathbf{z}^i} \prod_{t=1}^{T} q(z^i_t) \prod_{t=2}^{T} \underbrace{q(z^i_t)\delta(z^i_{t-1}, z^i_t \neq j, k)}_{\triangleq g^i_t(z^i_{t-1}, z^i_t)} \\
&= \sum_{z^i_T} \Bigg[\sum_{z^i_{T-1}} g^i_{T-1}(z^i_{T-1}, z^i_T) \cdots \Bigg[\sum_{z^i_2} g^i_2(z^i_2, z^i_3) \Bigg[\underbrace{\sum_{z^i_1} g^i_1(z^i_1, z^i_2) q(z^i_1)}_{m^i_2(z^i_2)} \Bigg] \Bigg] \cdots \Bigg],
\end{aligned} \tag{5.58}$$

in which the starting message is indicated in the last line. In general, the t^{th} message can be computed recursively as follows,

$$m_t^i(z_t^i) = \sum_{z_{t-1}^i} g_{t-1}^i(z_{t-1}^i, z_t^i) m_{t-1}^i(z_{t-1}^i) \tag{5.59}$$

Currently, this algorithm has the same computational cost as the exact algorithm. We simplify the above message expression to reduce the computational cost,

$$\begin{aligned}
&\sum_{z_{t-1}^i} g_{t-1}^i(z_{t-1}^i, z_t^i) m_{t-1}^i(z_{t-1}^i) \\
\triangleq &\sum_{z_{t-1}^i} q(z_t^i)\delta(z_{t-1}^i, z_t^i \neq j, k) m_{t-1}^i(z_{t-1}^i) \\
= &\sum_{z_{t-1}^i} q(z_t^i)(1 - \delta(z_{t-1}^i = j)\delta(z_t^i = k)) m_{t-1}^i(z_{t-1}^i) \\
= &q(z_t^i)\big(\sum_{z_{t-1}^i} m_{t-1}^i(z_{t-1}^i)\big) - q(z_t^i) m_{t-1}^i(z_{j-1}^i = j)\delta(z_t^i = k).
\end{aligned} \tag{5.60}$$

We can see that to obtain $m_t^i(z_t^i)$, we need to compute a sum of the previous message. Then for each value of z_t^i, we only need to perform constant operations. Hence, we have reduced the computational cost by $O(K)$. To inspire a further reduction, we represent the message $m_t^i(z_t^i)$ in a K dimensional vector (each entry k' corresponds to $m_t^i(z_t^i = k')$) as follows (the k^{th} entry is different from other entries by (5.60)),

$$\begin{bmatrix}
q(z_t^i = 1) \sum_{z_{t-1}^i} m_{t-1}^i(z_{t-1}^i) \\
q(z_t^i = 2) \sum_{z_{t-1}^i} m_{t-1}^i(z_{t-1}^i) \\
\vdots \\
q(z_t^i = k) \sum_{z_{t-1}^i} m_{t-1}^i(z_{t-1}^i) - q(z_t^i = k) m_{t-1}^i(z_{t-1}^i = j) \\
\vdots \\
q(z_t^i = K) \sum_{z_{t-1}^i} m_{t-1}^i(z_{t-1}^i)
\end{bmatrix}. \tag{5.61}$$

Since we are interested in the final summation in (5.58),

$$q(C_{j,k}^i = 0) = \sum_{z_T^i} m_T^i(z_T^i), \tag{5.62}$$

and in (5.61), the only quantities required from the previous message are $\sum_{z_{t-1}^i} m_{t-1}^i(z_{t-1}^i)$ and $m_{t-1}^i(z_{t-1}^i = j)$; it is not necessary to compute the entire message. Formally, we define

a recursion over message sums, instead of messages: summing over all the rows in (5.61) yields,

$$\sum_{z_t^i} m_t^i(z_t^i) = \sum_{z_{t-1}^i} m_{t-1}^i(z_{t-1}^i) - q(z_t^i = k) m_{t-1}^i(z_{t-1}^i = j). \tag{5.63}$$

In other words, at each time step t, instead of computing K values in (5.61); we simply compute two quantities, namely $\sum_{z_t^i} m_t^i(z_t^i)$ in (5.63) and $m_t^i(z_t^i = j)$ given as follows,

$$m_t^i(z_t^i = j) = \begin{cases} q(z_t^i = j) \sum_{z_{t-1}^i} m_{t-1}^i(z_{t-1}^i) & \text{if } j \neq k, \\ q(z_t^i = j) \sum_{z_{t-1}^i} m_{t-1}^i(z_{t-1}^i) - q(z_t^i = j) m_t^i(z_t^i = j) & \text{else} \end{cases}. \tag{5.64}$$

We have reduced the computational cost by another $O(K)$, and thus the computational cost for $q(C_{j,k}^i = 0)$ is $O(T)$. In total, the computational cost of $q(C_{j,k} > 0)$ for all (j, k) pairs is $O(ITK^2)$, which is the same as the computational complexity of finite HMMs.

Sampling Sampling is a useful tool for computing expectations that are either intractable or too expensive if solved exactly. Given a collection of S samples $\{(\mathbf{z}^i)^s\}_{s=1}^S$ that are drawn i.i.d. from $q(\mathbf{z}^i)$, $q(C_{j,k}^i = 0)$ is approximated by,

$$q(C_{j,k}^i = 0) \triangleq \sum_{\mathbf{z}^i} q(\mathbf{z}^i) \delta(C_{j,k}^i(\mathbf{z}^i) = 0) \approx \frac{1}{S} \sum_{s=1}^{S} \delta(C_{j,k}^i((\mathbf{z}^i)^s) = 0). \tag{5.65}$$

To obtain a sample from the joint distribution $q(\mathbf{z}^i)$ that is a Markov chain, we employ the ancestral sampling method (Bishop, 2006). Specifically, we start with drawing a sample from $p(z_1^i)$, which we call $\hat{z}_1^i$; then we draw a sample from $p(z_2^i|\hat{z}_1^i)$, which we call $\hat{z}_2^i$; we keep this procedure until we draw a sample from $p(z_T^i|\hat{z}_{T-1}^i)$. It is easy to verify that this series $\{\hat{z}_1^i, \hat{z}_2^i, ..., \hat{z}_T^i\}$ is one sample from $q(\mathbf{z}^i)$. Alternatively, we can embed a sampling procedure in our CVI algorithm: after a forward pass, we backward sample state sequences using the technique detailed in Scott (2002).

Both algorithms have the same computational cost $O(TK)$ to draw one sample. With either method, we do not need to repeatedly draw samples for each $q(C_{j,k}^i = 0)$: the drawn samples can be reused for computing all $q(C_{j,k}^i = 0)$. Therefore, the total computational cost for computing $q(C_{j,k} > 0)$ for all (j, k) pairs is $O(ITKS)$.

Heuristic Finally, we propose a heuristic algorithm. Given the problem,

$$q(C^i_{j,k}=0) \triangleq \sum_{\mathbf{z}^i} q(\mathbf{z}^i)\delta(C^i_{j,k}(\mathbf{z}^i)=0)$$
$$= \mathbb{E}_{q(\mathbf{z}^i)}[\prod_{t=2}^{T}\delta(z^i_{t-1}, z^i_t \neq j,k)], \tag{5.66}$$

our intuition is as follows: instead of considering a sequence of variables, $\mathbf{z}^i = \{z^i_t\}_{t=1}^T$, we assume it to be replaced by a collection of variable pairs $\{(z^i_1, z^i_2), (z^i_2, z^i_3), ..., (z^i_{t-1}, z^i_t)\}$, which are mutually *independent* from each other. Our heuristic has potential to work well, since those overlapping pairs can capture the crucial transition information induced by the first order dependence on a Markov chain. On the other hand, the independence assumption allows us to move the product operation in (5.66) before the expectation, yielding,

$$q(C^i_{j,k}=0) \approx \prod_{t=2}^{T}\mathbb{E}_{q(\mathbf{z}^i)}[\delta(z^i_{t-1}, z^i_t \neq j,k)]$$
$$= \prod_{t=2}^{T}\mathbb{E}_{q(z^i_{t-1}, z^i_t)}[\delta(z^i_{t-1}, z^i_t \neq j,k)]$$
$$= \prod_{t=2}^{T}(1-q(z^i_{t-1}, z^i_t = j,k)). \tag{5.67}$$

In (5.67), it is easy to identify that the cost for computing each $q(C^i_{j,k}=0)$ is $O(T)$. Hence, the total computational cost for $q(C_{j,k}>0)$ for all (j,k) pairs is $O(ITK^2)$, which is the same as the computational cost of using our mean field approximation method. Although we find it difficult to prove exactly what approximations are introduced by such heuristic, the empirical evaluations suggest that our intuition of using '*the overlapping but independent pairs*' can produce very accurate results compared with the exact algorithm.

5.4.4 Empirical Validation

In this subsection, we empirically evaluate the approximate methods against the exact one and study the behaviour of $q(C_{j,k}>0)$ both by simulation and in real datasets. Our study also sheds lights on understanding the mechanism for determining the number of effective hidden states provided by our CVI algorithm for HDP-HMMs.

To begin our investigation, we define the average approximation error as,

$$\text{error} = \frac{1}{K\times K}\sum_{j=1}^{K}\sum_{k=1}^{K}|q(C_{j,k}>0) - q^{\text{approx}}(C_{j,k}>0)|, \tag{5.68}$$

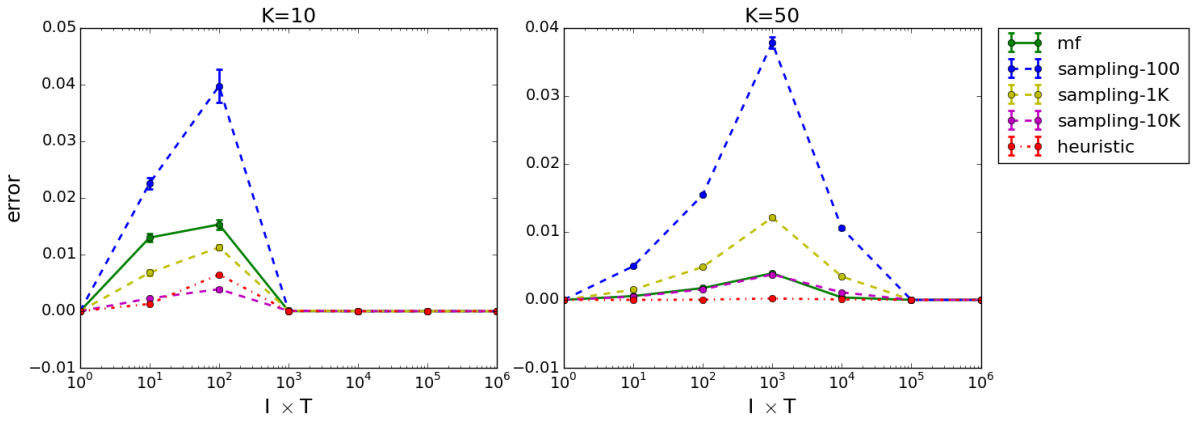

Figure 5.2: Empirical evaluation of the approximate algorithms against the exact method in simulated data. The results are averaged by 5 runs.

in which $q(C_{j,k} > 0)$ are the results produced by the exact algorithm in Subsection 5.4.2, and $q^{\text{approx}}(C_{j,k} > 0)$ are the results produced by each of the approximate algorithms in Subsection 5.4.3. For the sampling method, we use 100, 1000 and 10000 samples to show that how many samples are required for accurate approximation.

Simulation We conduct the following experiment to show the errors introduced by the approximate algorithms with the aim of finding some patterns. For each $q(\mathbf{z}^i)$, we simulate it by creating a heterogeneous series of discrete distributions: each $q(z_t^i|z_{t-1}^i) \propto \mathcal{U}[0,1]$, where $\mathcal{U}[0,1]$ is the uniform distribution between 0 and 1. The simulation is equivalent to the standard initialization method for CVI, which we have used in Chapter 3. Although we keep in mind that $q(\mathbf{z}^i)$ will evolve with algorithm iterations, the simulation provides a rough estimation about the errors in early iterations and reveals some patterns. We vary the total length of Markov chains, namely $I \times T \in \{10^0, 10^1, ..., 10^6\}$, and we expect errors to increase with larger total length (since errors accumulate). For $I \times T \leq 100$, we set $I = 1$; for $I \times T > 100$, we create multiple sequences, each of which is 100 in length. We also try different values for $K \in \{10, 50\}$. Under each setting and for each approximate algorithm, we run the simulation 5 times to reduce variances.

Figure 5.2 shows the average performance with standard deviation error bars for each approximate algorithm. In both the left and right plots, we can see that for the sampling method, the errors decrease by using more samples; the heuristic algorithm performances surprisingly well, and even better than drawing $10,000$ samples. We also notice that for all

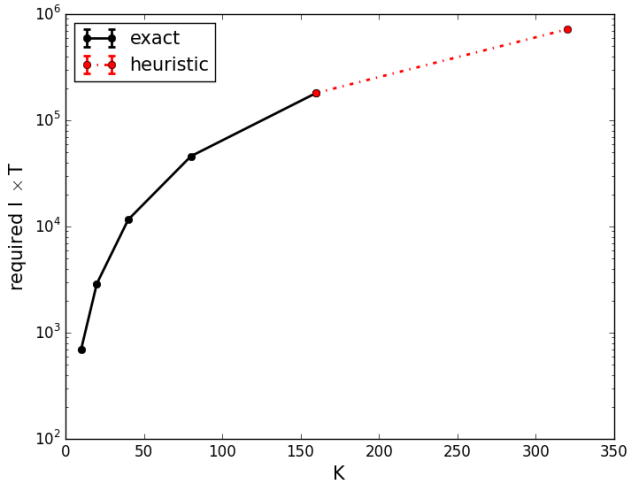

Figure 5.3: Empirical behaviour of $q(C_{j,k} > 0)$ in simulated data: for each K in x-axis, the y-axis shows the required $I \times T$ so that all $q(C_{j,k} > 0) > 0.999$. The results are averaged by 5 runs (the exact method does not scale for $K \geq 160$ and the heuristic one is used).

algorithms, initially the errors increase as $I \times T$ gets large, as we expected; but then the errors diminish after $I \times T$ is large enough. This is because with the increasing total length of all Markov chains, it becomes more and more likely that at least one transition of each type will occur; and when $q(C_{j,k} > 0)$ is close to 1 (which we will verify soon), there is little slack for approximation methods to make mistakes. Finally, we empirically observe that for the deterministic approximation methods (mean field method and heuristic), their errors decrease as K increases, making them more suitable when truncation levels are set large; while the sampling method is less sensitive to the value of K.

We investigate the behaviour of $q(C_{j,k} > 0)$ as $I \times T$ increases. In particular, we study whether $q(C_{j,k} > 0) > 0.999$ for all (j, k) pairs when $I \times T$ is large enough. Comparing two plots in Figure 5.2, we suspect that the required length for all $q(C_{j,k} > 0) > 0.999$ depends on K, and thus we vary $K \in \{10, 20, 40, 80, 160, 320\}$. The results averaged over 5 runs are given in Figure 5.3. We see that given a sufficiently large dataset, $q(C_{j,k} > 0)$ can be very close to 1 for all (j, k) pairs. Therefore, we initially think that it is reasonable to assume $q(C_{j,k} > 0) \approx 1$ in large datasets; and subsequently, we do not have to compute this quantity at all[3]. Nevertheless, as the variational posteriors $q(\mathbf{z}^i)$ evolve with algorithm iterations in real data, such assumption becomes inaccurate.

[3]With a simple assumption $q(C_{j,k} > 0) \approx 1$, our submission to NIPS 2016 was criticized by an anonymous reviewer, who strongly argued the necessity for computing $q(C_{j,k} > 0)$.

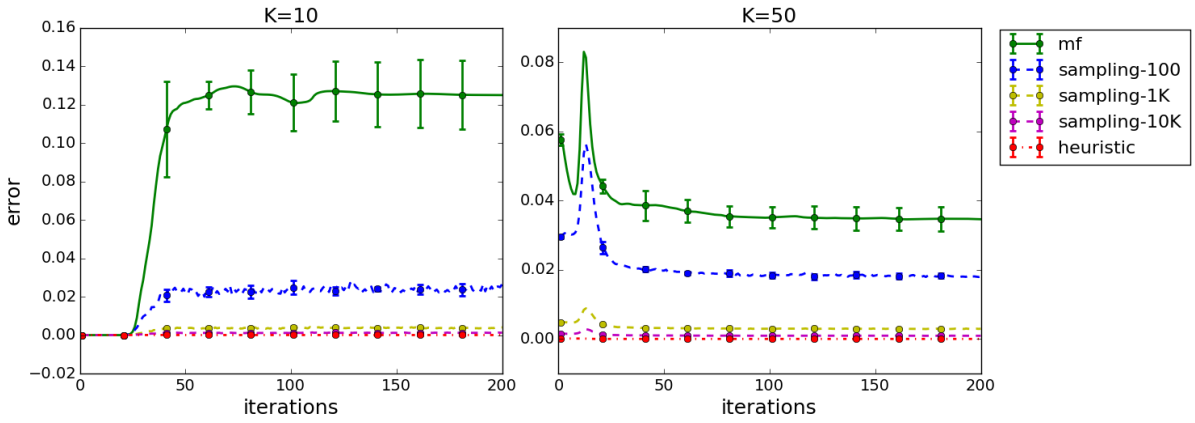

Figure 5.4: Empirical evaluation of the approximate algorithms against the exact method in Alice's Adventures in Wonderland. The results are averaged by 5 runs.

Real Data We also empirically evaluate the approximate algorithms in real datasets and study the behaviour of $q(C_{j,k} > 0)$ as variational posteriors change over time. We use a textual dataset taken from 'Alice's Adventures in Wonderland'. Specifically, we filter out all the symbols except for the 26 characters and white space. Then we choose the first $10,000$ characters and split it into 100 sequences, each of which is 100 in length. This gives the the training data in which we run our CVI algorithm for HDP-HMMs. We use the following experiment settings: $\gamma = 5$, $\sigma = 3$, $\beta = 1/27$; and we initialize $q(z_t^i | z_{t-1}^i) \propto \mathcal{U}[0, 1]$, $\mathbb{G}[\pi_k] = 1/K$, where K is the truncation level. For each iteration, we compute the quantity $q(C_{j,k} > 0)$ by all the exact and approximate algorithms; and for each approximate algorithm, we compute its approximation error as defined in (5.68). We repeat the experiment 5 times with random starts to reduce variance.

Figure 5.4 shows each approximate algorithm' performance as a function of iterations (i.e., as variational posteriors $q(\mathbf{z}^i)$ change). In both the left and right plots, we empirically observe that the mean field approximation, which works well in the simulated data, performs quite badly in the real dataset here; the sampling method with 1K or 10K samples are reasonably accurate; and finally, the heuristic algorithm performs again surprisingly the best, with the approximation errors less than 0.001 throughout all iterations. For $K = 10$, in the first 20 iterations, the approximations errors are negligible for all the approximate algorithms due again to the fact that $q(C_{j,k} > 0)$ is very close to 1.

To illustrate the behaviour of $q(C_{j,k} > 0)$, we draw the heat maps both after the first and final iterations of our CVI algorithm. In Figure 5.5, in each heat map, the color of its

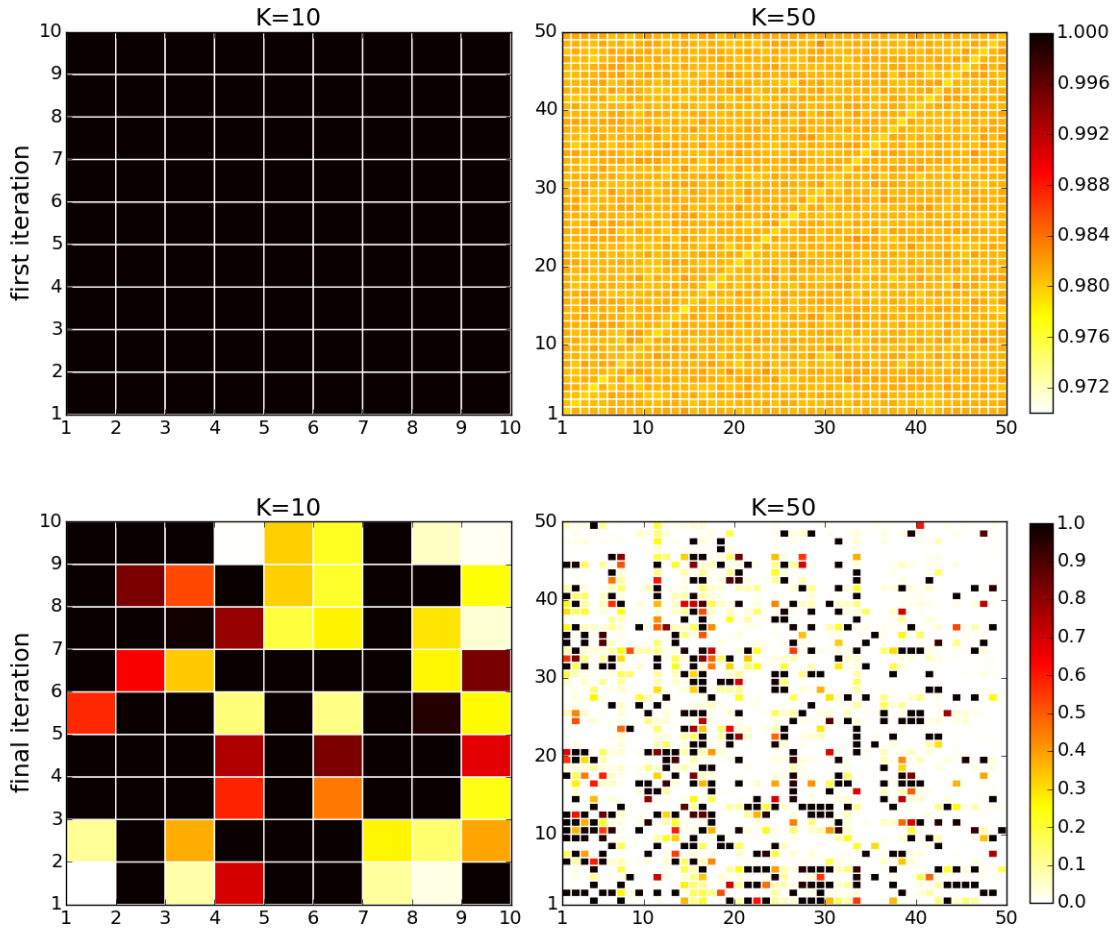

Figure 5.5: Empirical behaviour of $q(C_{j,k} > 0)$ in Alice's Adventures in Wonderland. Top row: heat map for $q(C_{j,k} > 0)$ after the first iteration of the CVI algorithm for HDP-HMMs. Bottom row: heat map for $q(C_{j,k} > 0)$ after the final iteration.

(j, k) entry represents the value of the corresponding $q(C_{j,k} > 0)$. As we can see, after the first iteration where the symmetries in count matrices are probably not much broken, $q(C_{j,k} > 0)$ is almost 1 for $K = 10$, and is about 0.985 in average with small variances for $K = 50$. After the final iteration, it is clear that not all $q(C_{j,k} > 0)$ are close to 1, and thus assuming that $q(C_{j,k} > 0) \approx 1$ is no longer reasonable. Even for $K = 10$, there are some cells filled by white colors, meaning that the corresponding $q(C_{j,k} > 0) \approx 0$. For $K = 50$, we see that about half cells are white; and by (5.35), the corresponding table counts $\mathbb{E}[s_{j,k}]$ are almost zeros, resulting in very small priors for certain states, and therefore, our CVI algorithm finds a smaller number of effective states than the truncation level.

Finally, we evaluate the approximate methods in another dataset, the Wall Street Journal to ensure that the astonishing performances of the heuristic method is consistent. Specifi-

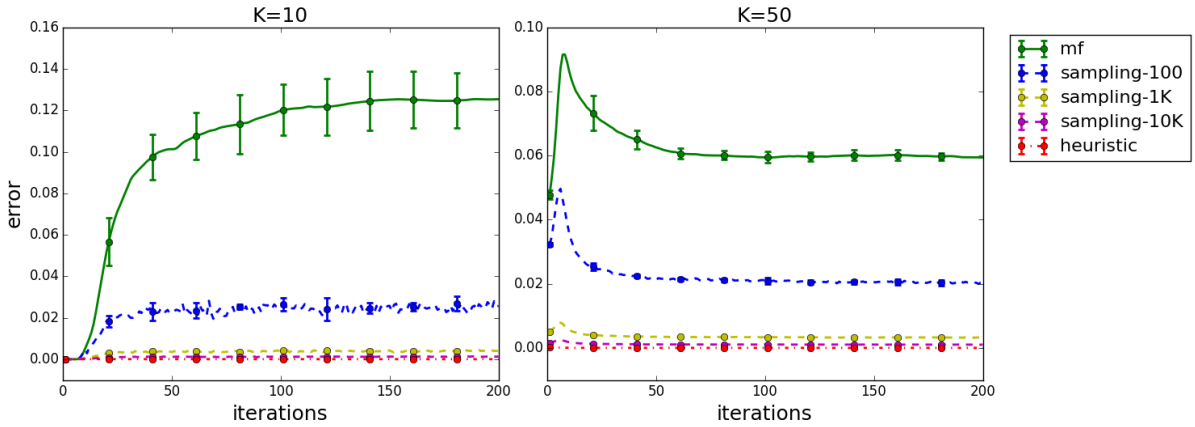

Figure 5.6: Empirical analysis of the approximate algorithms against the exact method in the Wall Street Journal. The results are averaged by 5 runs.

cally, we use the first 450 sentences as the training data, with the average length is roughly 22. We run our CVI algorithm and for each iteration, record the approximation errors by each approximate algorithm. Figure 5.6 shows the similar performances for each algorithm as they are in Alice's Adventures in Wonderland. Given the consistent performances of the heuristic method and its computational efficiency, we will use it in our experiments, to compare our CVI algorithm for HDP-HMMs with other benchmarks.

5.5 Experiments

In this section, we evaluate our CVI algorithm and its stochastic version for HDP-HMMs, in terms of learning effective states, learning the HDP-related parameters as well as competing with other benchmarks including two recent stochastic variational algorithms.

5.5.1 Finding Numbers of States

Since the HDP defines distributions on an infinite set of supports, in this subsection, we let our algorithm to determine the number of effective states given that we set the truncation levels to larger values. In order to achieve this, to the best of our knowledge, some ad hoc methods need to be used. For example, for variational algorithms, there are several ways of pruning ineffective states: Liang et al. (2007) cut the PCFG rules if they are below 10^{-5}; Zhang et al. (2016) used the minimum number of states to cover 99.5% of data. On the other hand, for sampling methods, we often get around with this issue by just using one

sample. In this subsection and Subsection 5.5.3, we count the number of effective states to be the minimum number of states that cover 99% of the data[4].

We employ the similar datasets and the same experiment settings described in the previous papers. Specifically, we first follow Zhang et al. (2016) and generate datasets of 100 sequences (each is 100 in length) from the following two transition matrices θ_{pos} and θ_{neg}, respectively, as well as the shared emission matrix,

$$\theta_{\text{pos}} = \begin{bmatrix} 0.99 & 0.01 & 0 & 0 \\ 0 & 0.99 & 0.01 & 0 \\ 0 & 0 & 0.99 & 0.01 \\ 0.01 & 0 & 0 & 0.99 \end{bmatrix}, \qquad \theta_{\text{neg}} = \begin{bmatrix} 0.01 & 0.99 & 0 & 0 \\ 0 & 0.01 & 0.99 & 0 \\ 0 & 0 & 0.01 & 0.99 \\ 0.99 & 0 & 0 & 0.01 \end{bmatrix}, \tag{5.69}$$

$$\phi = \frac{1}{3} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}. \tag{5.70}$$

We also follow Beal et al. (2002) and generate datasets of 100 sequences (each is length 100) using the following regular grammar. A sequence can be one of the three types: the first type is a substring of $(abc)^*$, where $*$ denotes the infinite repetition; the second type is a substring of $(acb)^*$, and the third type is a substring of $(a^*b^*)^*$ where a and b are symbols emitted stochastically with probability 0.5 each. Given the grammar rules, we expect an HMM with seven hidden states to model the generated data (i.e., three states for $(abc)^*$, three states for $(acb)^*$, and one self-transitioning state for $(a^*b^*)^*$).

For each scenario, we run our CVI algorithm 10 times with newly generated datasets and report the averaged results. In the datasets generated by (5.69-5.70), we set the truncation level $K = 10$, as suggested in Zhang et al. (2016); and in the datasets generated by the regular grammar, we set $K = 12$, as suggested in Beal et al. (2002). Both choices allow for some redundancy. We also include the results of beam sampling, since it is known as a reliable algorithm for recovering data structures. For both algorithms, we fix $\gamma = 1$, $\sigma = 1$, $\beta = 1$. For our CVI algorithm, we initialize $q(z_t^i | z_{t-1}^i) \propto \mathcal{U}[0, 1]$; for beam sampling, we randomly assign a hidden state from $\{1, 2, ..., K\}$ to each observation.

Figure 5.7 shows the numbers of effective states recovered both by our CVI algorithm and beam sampling on the artificial datasets. In the left plot, we see that our CVI algorithm

[4]This is simply achieved by reordering the state labels such that $\mathbb{E}[C_{\cdot,1}] > \mathbb{E}[C_{\cdot,2}] > \cdots > \mathbb{E}[C_{\cdot,K}]$ as in Teh et al. (2008).

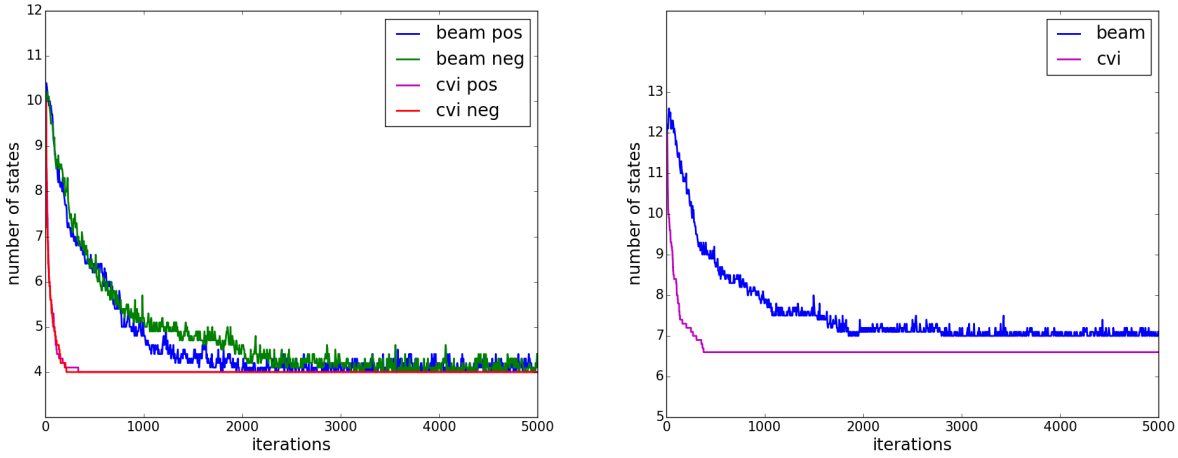

Figure 5.7: Demonstration of finding the correct numbers of hidden states. Left: results on the datasets generated by a diagonally dominant HMM and a cycled HMM in (Zhang et al., 2016). Right: results on the datasets generated by a regular grammar in Beal et al. (2002).

uses about 300 iterations (~ 27 seconds) to converge to exactly four states, whereas beam sampling takes more than $2,500$ iterations (~ 180 seconds) to stabilize close to the correct number. In the right plot, our CVI algorithm again converges much faster and it finds a slightly smaller number (6.6 in average) than the correct number of states, seven. On the other hand, beam sampling recovers a little the more than seven states using about $3,000$ samples. Therefore, it is fair to claim that both algorithms have abilities to recover hidden data structures to certain extents, but our CVI algorithm is significantly more efficient.

5.5.2 Comparison with Other Benchmarks

In this subsection, we compare our CVI algorithm and its stochastic version with other inference algorithms for HDP-HMMs on the Million Song Dataset, created by Zhang et al. (2016). Initially, the dataset consists of music pieces. By extracting their audio features and clustering them using the K-means algorithm, the resulting training dataset consists of 125K discretized sequences (each is 50 in length); and the number of possible observations is 256. There is also a testing dataset of 634 sequences being withheld for calculating the predictive log likelihoods.

First, we compare our batch CVI algorithm with beam sampling. We follow exactly the same experiment settings as in Zhang et al. (2016): we set the truncation level $K = 500$ (or the initial number of states for beam sampling); we fix $\gamma = 20$, $\sigma = 3$ and $\beta = 0.1$. We

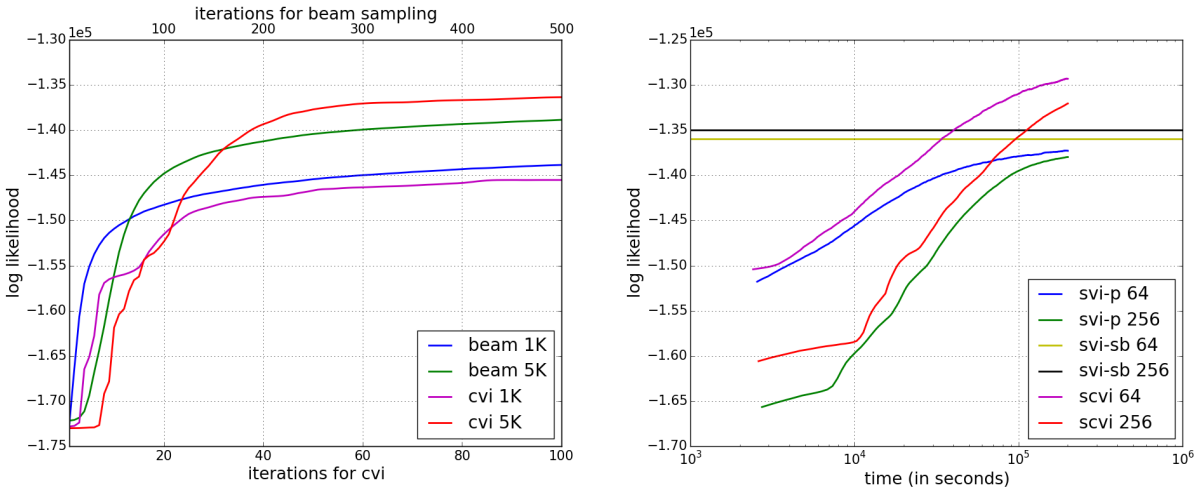

Figure 5.8: Comparison of our CVI algorithm and other benchmarks for HDP-HMMs on the Million Song Dataset. Left: comparison of CVI and beam sampling in the batch setting. Right: comparison of SCVI and SVI in the stochastic setting; the horizontal lines denote the final results produced by the 'svi-sb' algorithm after processing 1 million sequences.

run our CVI algorithm for 100 iterations and beam sampling for 500 iterations[5]. The left plot of Figure 5.8 presents their predictive log likelihood results on two training datasets with different numbers of sequences. We can see that although beam sampling is slightly better for the 1K sequence dataset, our CVI algorithm outperforms beam sampling when there are 5K sequences. In the 5K dataset, our CVI algorithm occupies around 310 states in its posterior while the beam sampling uses about 330 states.

Next, we compare our stochastic CVI algorithm with two recent stochastic variational inference algorithms. Specifically, the SVI algorithm using a point estimate for the top level stick (Johnson and Willsky, 2014) is denoted by 'svi-p', whereas the SVI algorithm using a full posterior estimate based on the two level stick breaking contribution is denoted by 'svi-sb'. As suggested by Zhang et al. (2016), we set the learning rate as $\rho_m = (100 + m)^{-0.6}$, instead of conducting further explorations; and vary $M \in \{64, 256\}$, since by both mini-batch sizes, 'svi-sb' converge to the best (and very similar) results. The right plot of Figure 5.8 shows the predictive log likelihoods for the three stochastic algorithms, all of which have processed around 1 million sequences. The results of 'scvi' and 'svi-p' are based on our implementation, and for 'svi-sb' we reuse the reported results in Zhang et al. (2016). As we can see, our SCVI algorithm significantly outperforms both SVI algorithms, extending

[5]The beam sampling results based on our implementation are slightly better than those reported in Zhang et al. (2016), where the authors report results using wall clock time and the iteration numbers are unknown.

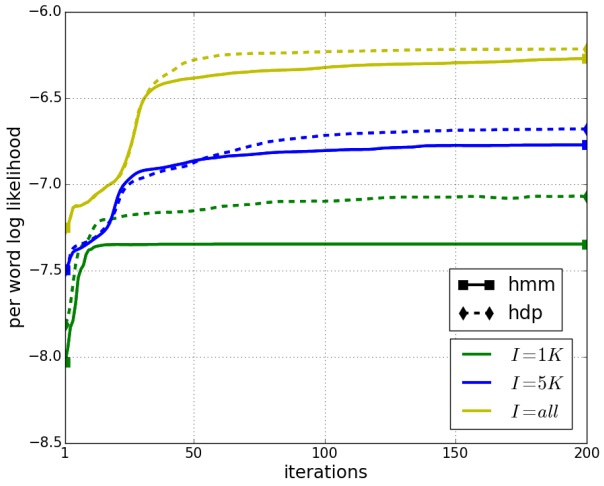

Figure 5.9: Comparison of CVI for HMMs and CVI for HDP-HMMs on the WSJ. We use various corpus sizes and the results are reported by averaging 5 random runs.

the success of SCVI for finite models to nonparametric models. In addition, we notice that our SCVI algorithm seems to work better when the minibatch size is smaller, whereas the SVI algorithms are less sensitive.

5.5.3 Hyperparameter Optimization

The HDP provides a natural mechanism for sharing statistical strength among clusters, and such mechanism often improves model performances. In this subsection, we demonstrate the benefits of sharing statistical strength and optimizing concentration parameters. Specifically, unlike the previous two subsections, here we do not prune ineffective states, but instead we keep the truncation levels in CVI for HDP-HMMs equal to the numbers of states in CVI for finite HMMs. Then we compare these two CVI algorithms. Furthermore, for HDP-HMMs, as well as learning the asymmetric global transition distribution determined by $\{\gamma\} \cup \{\sigma\} \cup \{u_k, v_k\}_{k=1}^K$, we also let $\phi_k \sim \text{Dir}(\beta\tau)$, where $\tau = \{\tau_1, ..., \tau_W\}$ is the asymmetric global emission distribution over the vocabulary and β is its concentration parameter. We update β and τ following the same approach in Teh et al. (2008).

We revisit the text prediction tasks on the Wall Street Journal in Chapter 3, and use the same initializations and experiment settings. In the batch setting, we initialize the local statistics $q(z_t^i|z_{t-1}^i) \propto \mathcal{U}[0, 1]$. In the stochastic setting, we directly initialize the global statistics using exponential distributions defined in (3.106-3.107). For finite HMMs, we set both the transition and emission priors to be $\text{Dir}(0.1)$, in order to encourage sparsity.

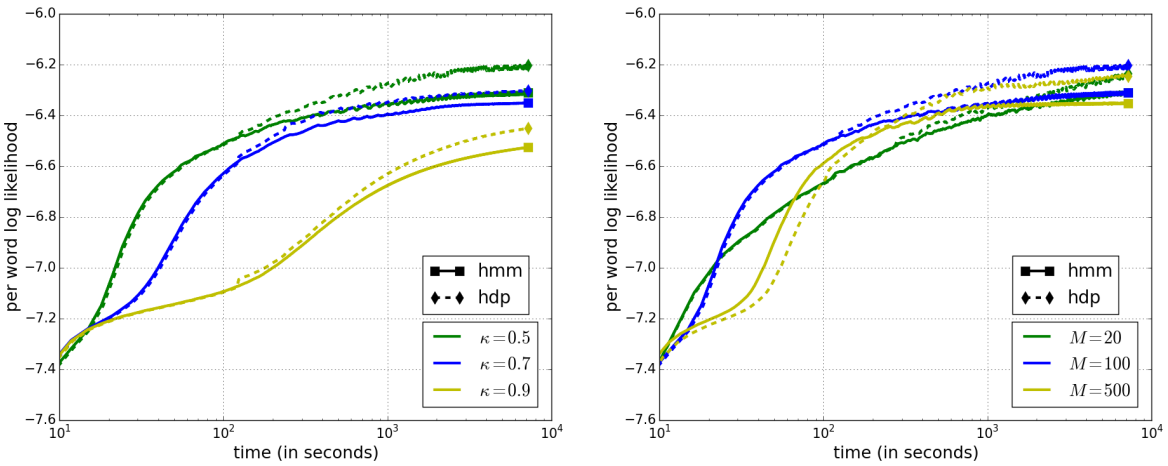

Figure 5.10: Comparison of SCVI for HMMs and SCVI for HDP-HMMs on the WSJ. Left: we set minibatch size $M = 100$ and vary forgetting rates. Right: we set the forgetting rate $\kappa = 0.5$ and vary the minibatch size. Both algorithms run for 2 hours.

For HDP-HMMs, we add the same prior counts (i.e., 0.1 in each dimension) for the first iteration for fair comparisons. We set the the concentration parameters to the initial values: $\gamma = 10$, $\sigma = 5$ and $\beta = 100$ and batch update them after processing all the sentences[6].

First, we compare CVI for finite HMMs and CVI for HDP-HMMs on the Wall Street Journal in the batch setting. As before, we set $K = 45$ (the number of states for HMMs or the truncation level for HDP-HMMs). We run each algorithm for 200 iterations and vary the dataset sizes. Figure 5.9 shows their predictive log likelihood results. It is not so surprising to notice that the hyperparameter optimization is most effective on the smallest 1K sentence dataset. We think this is because with a smaller dataset, the hyperparameters plays a comparatively more important roles. In the 5K dataset and the whole WSJ, we can see that there are still noticeable benefits of including the hyperparameter optimization.

Next, we compare SCVI for finite HMMs and SCVI for HDP-HMMs. In Chapter 3, we have explored a range of settings, showing that for finite HMMs, SCVI is consistently more accurate than SVI. Here, we test the SCVI algorithm for HDP-HMMs under each of the previous settings. Specifically, we first fix the minibatch size $M = 100$ and vary

[6]These values were never tweaked, and they were optimized each time a dataset was gone through in any way. For γ and σ, we can also set their values by our prior expectations about the number of states: take γ for an example, we first compute the expected $(\pi_1, \pi_2, ...)$, where $\tilde{\pi}_k \sim \text{Beta}(1, \gamma)$. If we define the number of states to be those that occupy 99% of the stick, and we expect there might be 30 to 60 states; then γ should fall in $[5.9, 12.5]$. Similarly, $\sigma \in [2.6, 5.9]$ if we expect 15 to 30 states in the second level DPs. Therefore, our choices of γ and σ are reasonable. We explored $\beta \in [10, 100, 1000]$ as well as $\beta = 0.1 \times W$ (the same as in finite HMMs), the final results were insensitive (no more than 0.01 per word log likelihood differences).

$\kappa = \{0.5, 0.7, 0.9\}$; and then we fix $\kappa = 0.5$ and vary $M \in \{20, 100, 500\}$. Under each of the setting, we run both algorithms for 2 hours. Figure 5.10 shows optimizing the hyper-parameters also improves upon the finite HMMs by certain margins, thus transferring the performance gains to the stochastic setting and making better use of the training time.

5.6 Summary

In this chapter, we have presented and evaluated a new (stochastic) collapsed variational inference algorithm for HDP-HMMs. On the algorithmic side, our main contribution is solving the inference challenge caused by coupling the hierarchical structure of the HDP and sequential dependencies of HMMs. Such coupling makes computing the probabilities of CRP tables' existences much more challenging than in the HDP-LDA. We propose an exact algorithm and evaluate several approximate techniques. Our algorithmic contribution allows CVI for HDP to be widely applied beyond the bag-of-words setting, including to a number of HDP-based computational linguistic models. On the application side, our algorithm often outperforms the previous benchmarks including beam sampling and two uncollapsed variational algorithms for HDP-HMMs as well as our CVI for finite HMMs, and therefore provides a new strong benchmark for time series analysis.

5.A Derivation Details in Section 5.3

In this appendix, we provide the detailed derivation steps that are omitted in Section 5.3. Recall the evidence lower bound is,

$$\mathcal{L}^{\text{CVI}} \triangleq \mathbb{E}[\log p(\mathbf{x}, \mathbf{z}, \mathbf{s}|\tilde{\pi}, \sigma)] + \mathbb{E}[\log p(\tilde{\pi}|\gamma)] - \mathbb{E}[\log q(\mathbf{z}, \tilde{\pi}, \mathbf{s})], \tag{5.71}$$

in which the variational distribution factorizes as follows,

$$q(\mathbf{z}, \tilde{\pi}, \mathbf{s}) = \prod_{i=1}^{I} q(\mathbf{z}^i) \prod_{j=0}^{\infty} \prod_{k=1}^{\infty} q(s_{j,k}|\mathbf{z}) \prod_{k=1}^{\infty} \text{Beta}(\tilde{\pi}_k|u_k, v_k).$$

Given the direct assignment truncation strategy, the indices greater than K are independent of observations, and thus we do not include them in our derivations.

Concentration Parameters In $\mathcal{L}^{\text{CVI}}$, the terms relating to γ can be simplified as follows,

$$\begin{aligned}
&\mathbb{E}_{q(\tilde{\pi})}[\log p(\tilde{\pi}|\gamma)] \\
&= \sum_{k=1}^{K} \mathbb{E}_{q(\tilde{\pi}_k)}[\log p(\tilde{\pi}_k|\gamma)] \\
&= \sum_{k=1}^{K} \mathbb{E}_{q(\tilde{\pi}_k)}[\log \frac{\Gamma(1+\gamma)}{\Gamma(1)\Gamma(\gamma)} \tilde{\pi}_k^{1-1}(1-\tilde{\pi}_k)^{\gamma-1}] \\
&= K \log \Gamma(1+\gamma) - K \log \Gamma(\gamma) + (\gamma - 1) \sum_{k=1}^{K} \mathbb{E}[\log(1-\tilde{\pi}_k)] \\
&= K \log \Gamma(1+\gamma) - K \log \Gamma(\gamma) + (\gamma - 1) \sum_{k=1}^{K} (\psi(v_k) - \psi(u_k + v_k)).
\end{aligned} \tag{5.72}$$

Taking the partial gradient w.r.t. γ and using the fact $\psi(x+1) = \psi(x) + \frac{1}{x}$, we have,

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \gamma} &= K\psi(\gamma+1) - K\psi(\gamma) + \sum_{k=1}^{K} \psi(v_k) - \psi(u_k + v_k) \\
&= \frac{K}{\gamma} + \sum_{k=1}^{K} \psi(v_k) - \psi(u_k + v_k).
\end{aligned} \tag{5.73}$$

Setting the gradient to zero yields,

$$\gamma = \frac{K}{\sum_{k=1}^{K} \psi(u_k + v_k) - \psi(v_k)}. \tag{5.74}$$

In $\mathcal{L}^{\text{CVI}}$, the terms relating to σ can be simplified as follows (const denotes the dropped terms that are constant w.r.t. σ),

$$\begin{aligned}
&\mathbb{E}_{q(\mathbf{z})q(\mathbf{s}|\mathbf{z})q(\tilde{\pi})}[\log p(\mathbf{x}, \mathbf{z}, \mathbf{s}|\sigma, \tilde{\pi})] \\
=&\mathbb{E}_{q(\mathbf{z})q(\mathbf{s}|\mathbf{z})q(\tilde{\pi})}[\log \prod_{j=0}^{K}\prod_{k=1}^{K} \sigma^{s_{j,k}}] + \text{const} \\
=&\sum_{j=0}^{K} \mathbb{E}[\log \frac{\Gamma(\sigma)}{\Gamma(\sigma + C_{j,\cdot})}] + \sum_{j=0}^{K}\sum_{k=1}^{K} \mathbb{E}[\log \sigma^{s_{j,k}}] + \text{const.} \\
=&\sum_{j=0}^{K} (\log \Gamma(\sigma) - \mathbb{E}[\log \Gamma(\sigma + C_{j,\cdot})]) + \log \sigma \sum_{j=0}^{K}\sum_{k=1}^{K} \mathbb{E}[s_{j,k}] + \text{const.}
\end{aligned} \tag{5.75}$$

Taking the partial gradient w.r.t. σ, we have,

$$\frac{\partial \mathcal{L}}{\partial \sigma} = \sum_{j=0}^{K} (\psi(\sigma) - \mathbb{E}[\psi(\sigma + C_{j,\cdot})]) + \frac{1}{\sigma} \sum_{j=0}^{K}\sum_{k=1}^{K} \mathbb{E}[s_{j,k}]. \tag{5.76}$$

Setting the gradient to zero yields,

$$\sigma = \frac{\sum_{j=0}^{K}\sum_{k=1}^{K} \mathbb{E}[s_{j,k}]}{\sum_{j=0}^{K} \mathbb{E}[\psi(\sigma + C_{j,\cdot})] - \psi(\sigma)}. \tag{5.77}$$

Top Level Stick In $\mathcal{L}^{\text{CVI}}$, the terms relating to u_k and v_k can be simplified as follows,

$$\begin{aligned}
&\mathbb{E}_{q(\mathbf{z})q(\mathbf{s}|\mathbf{z})q(\tilde{\pi})}[\log p(\mathbf{x}, \mathbf{z}, \mathbf{s}|\sigma, \tilde{\pi})] \\
=&\mathbb{E}_{q(\mathbf{z})q(\mathbf{s}|\mathbf{z})q(\tilde{\pi})}[\log \prod_{j=0}^{K}\prod_{k=1}^{K} \pi_k^{s_{j,k}}] + \text{const} \\
=&\mathbb{E}_{q(\mathbf{z})q(\mathbf{s}|\mathbf{z})q(\tilde{\pi})}[\log \prod_{j=0}^{K}\prod_{k=1}^{K} \tilde{\pi}_k^{s_{j,k}} \prod_{l<k}(1 - \tilde{\pi}_l)^{s_{j,k}}] + \text{const} \\
=&\sum_{k=1}^{K}\sum_{j=0}^{K} \mathbb{E}[s_{j,k}]\mathbb{E}[\log \tilde{\pi}_k] + \sum_{k=1}^{K}\sum_{j=0}^{K}\sum_{l<k} \mathbb{E}[s_{j,k}]\mathbb{E}[\log(1 - \tilde{\pi}_l)] + \text{const} \\
=&\sum_{k=1}^{K} \mathbb{E}[s_{\cdot,k}]\mathbb{E}[\log \tilde{\pi}_k] + \sum_{k=1}^{K} \mathbb{E}[s_{\cdot,>k}]\mathbb{E}[\log(1 - \tilde{\pi}_k)] + \text{const} \\
=&\mathbb{E}[s_{\cdot,k}](\psi(u_k) - \psi(u_k + v_k)) + \mathbb{E}[s_{\cdot,>k}](\psi(v_k) - \psi(u_k + v_k)) + \text{const},
\end{aligned} \tag{5.78}$$

$$
\begin{aligned}
&\mathbb{E}_{q(\tilde{\pi}_k)}[\log q(\tilde{\pi}_k)] \\
=&\mathbb{E}_{q(\tilde{\pi}_k)}[\log \frac{\Gamma(u_k+v_k)}{\Gamma(u_k)\Gamma(v_k)}\tilde{\pi}_k^{u_k-1}(1-\tilde{\pi}_k)^{v_k-1}] \\
=&\log\Gamma(u_k+v_k)-\log\Gamma(u_k)-\log\Gamma(v_k) \\
&+(u_k-1)\mathbb{E}[\log\tilde{\pi}_k]+(v_k-1)\mathbb{E}[\log(1-\tilde{\pi}_k)] \\
=&\log\Gamma(u_k+v_k)-\log\Gamma(u_k)-\log\Gamma(v_k) \\
&+(u_k-1)(\psi(u_k)-\psi(u_k+v_k))+(v_k-1)(\psi(v_k)-\psi(u_k+v_k)).
\end{aligned} \tag{5.79}
$$

Taking the partial gradients w.r.t. u_k and v_k, we have,

$$
\begin{aligned}
\frac{\partial\mathcal{L}}{\partial u_k}=&(r-1)(-\psi'(u_k+v_k)) \\
&-\psi(u_k+v_k)-\psi(u_k)+(\psi(u_k)-\psi(u_k+v_k)) \\
&+(u_k-1)(\psi'(u_k)-\psi'(u_k+v_k))+(v_k-1)(-\psi'(u_k+v_k)) \\
&+\mathbb{E}[s_{\cdot,k}](\psi'(u_k)-\psi'(u_k+v_k))+\mathbb{E}[s_{\cdot,>k}](-\psi'(u_k+v_k)) \\
=&\psi'(u_k+v_k)(u_k+v_k-\mathbb{E}[s_{\cdot,\geq k}]-\gamma-1) \\
&-\psi'(u_k)(u_k-\mathbb{E}[s_{\cdot,k}]-1),
\end{aligned} \tag{5.80}
$$

$$
\begin{aligned}
\frac{\partial\mathcal{L}}{\partial v_k}=&(r-1)(\psi'(v_k)-\psi'(u_k+v_k)) \\
&-\psi(u_k+v_k)-\psi(v_k)+(u_k-1)(-\psi'(u_k+v_k)) \\
&+(v_k-1)(\psi'(v_k)-\psi'(u_k+v_k))+(\psi(v_k)-\psi(u_k+v_k)) \\
&+\mathbb{E}[s_{\cdot,k}](-\psi'(u_k+v_k))+\mathbb{E}[s_{\cdot,>k}](\psi'(v_k)-\psi'(u_k+v_k)) \\
=&\psi'(u_k+v_k)(u_k+v_k-\mathbb{E}[s_{\cdot,\geq k}]-\gamma-1) \\
&-\psi'(v_k)(v_k-\mathbb{E}[s_{\cdot,>k}]-\gamma).
\end{aligned} \tag{5.81}
$$

Setting the gradients to zeros yields,

$$
u_k=1+\mathbb{E}[s_{\cdot,k}], \tag{5.82}
$$

$$
v_k=\gamma+\mathbb{E}[s_{\cdot,>k}]. \tag{5.83}
$$

Stirling Numbers Before we derive the conditional variational posterior $q(\mathbf{s}|\mathbf{z})$, we review the definition of Stirling numbers of the first kind and their properties. Denoted by $\begin{bmatrix} C \\ s \end{bmatrix}$, it

counts the number of permutations of C elements with s disjoint cycles. It can be defined in terms of rising factorials as follows,

$$x^{(C)} \triangleq x(x+1)(x+2)\cdots(x+C-1) \triangleq \sum_{s=1}^{C} \begin{bmatrix} C \\ s \end{bmatrix} x^s. \tag{5.84}$$

Using the fact that $\Gamma(x+1) = x\Gamma(x)$,

$$x^{(C)} = \frac{\Gamma(x+C)}{\Gamma(x)}. \tag{5.85}$$

Therefore, we have the following property,

$$\sum_{s=1}^{C} \begin{bmatrix} C \\ s \end{bmatrix} x^s = \frac{\Gamma(x+C)}{\Gamma(x)}. \tag{5.86}$$

Taking the derivatives of both sides of (5.86) w.r.t. x, and then multiplying each side by x, we have the second useful property,

$$\sum_{s=1}^{C} \frac{\Gamma(x)}{\Gamma(x+C)} \begin{bmatrix} C \\ s \end{bmatrix} x^s s = x(\psi(x+C) - \psi(x)). \tag{5.87}$$

We will use these two properties in deriving the conditional variational posterior $q(\mathbf{s}|\mathbf{z})$ and computing the expectation $\mathbb{E}[s_{j,k}]$.

Auxiliary variables We take the functional derivative of $\mathcal{L}^{\text{CVI}}$ w.r.t. $q(\mathbf{s}|\mathbf{z})$ and equate it to zero (the Lagrange multiplier term enforcing normalisation is omitted for brevity),

$$\begin{aligned} \frac{\partial \mathcal{L}^{\text{CVI}}}{\partial q(\mathbf{s}|\mathbf{z})} &= \sum_{\mathbf{z}} q(\mathbf{z}) \mathbb{E}_{q(\tilde{\pi})}[\log p(\mathbf{x}, \mathbf{z}, \mathbf{s}|\tilde{\pi}, \sigma)] - \sum_{\mathbf{z}} q(\mathbf{z}) \log q(\mathbf{s}|\mathbf{z}) - 1 \\ &= 0, \end{aligned} \tag{5.88}$$

which implies,

$$
\begin{aligned}
q(\mathbf{s}|\mathbf{z}) &\propto \exp\{\mathbb{E}_{q(\tilde{\pi})}[\log \prod_{j=0}^{K}\prod_{k=1}^{K} \begin{bmatrix} C_{j,k} \\ s_{j,k} \end{bmatrix} (\sigma\pi_k)^{s_{j,k}}]\} \\
&= \prod_{j=0}^{K}\prod_{k=1}^{K} \exp\{\mathbb{E}_{q(\tilde{\pi})}[\log \begin{bmatrix} C_{j,k} \\ s_{j,k} \end{bmatrix}]\} \exp\{\mathbb{E}_{q(\tilde{\pi})}[\log(\sigma\pi_k)^{s_{j,k}}]\} \\
&= \prod_{j=0}^{K}\prod_{k=1}^{K} \begin{bmatrix} C_{j,k} \\ s_{j,k} \end{bmatrix} \exp\{\mathbb{E}_{q(\tilde{\pi})}[\log(\sigma\pi_k)^{s_{j,k}}]\} \\
&= \prod_{j=0}^{K}\prod_{k=1}^{K} \begin{bmatrix} C_{j,k} \\ s_{j,k} \end{bmatrix} (\sigma \exp\{\mathbb{E}_{q(\tilde{\pi})}[\log \pi_k]\})^{s_{j,k}} \\
&= \prod_{j=0}^{K}\prod_{k=1}^{K} \begin{bmatrix} C_{j,k} \\ s_{j,k} \end{bmatrix} (\sigma\mathbb{G}[\pi_k])^{s_{j,k}},
\end{aligned} \tag{5.89}
$$

which then implies,

$$
q(s_{j,k}|\mathbf{z}) \propto \begin{bmatrix} C_{j,k} \\ s_{j,k} \end{bmatrix} (\sigma\mathbb{G}[\pi_k])^{s_{j,k}}. \tag{5.90}
$$

In (5.91), $\mathbb{G}[\pi_k]$ is a geometric expectation and it can be computed as follows: $\mathbb{G}[\pi_k] = \mathbb{G}[\tilde{\pi}_k]\prod_{l<k}\mathbb{G}[1-\tilde{\pi}_l]$; using properties of the Beta distribution, each $\mathbb{G}[\tilde{\pi}_k] = \exp\{\psi(u_k) - \psi(u_k+v_k)\}$ and $\mathbb{G}[1-\tilde{\pi}_k] = \exp\{\psi(v_k) - \psi(u_k+v_k)\}$.

The normalization constant of $q(s_{j,k}|\mathbf{z})$ can be found be summing over $s_{j,k}$. Using the property of Stirling numbers of the first kind in (5.86), we have,

$$
q(s_{j,k}|\mathbf{z}) = \frac{\Gamma(\sigma\mathbb{G}[\pi_k])}{\Gamma(\sigma\mathbb{G}[\pi_k]+C_{j,k})} \begin{bmatrix} C_{j,k} \\ s_{j,k} \end{bmatrix} (\sigma\mathbb{G}[\pi_k])^{s_{j,k}}. \tag{5.91}
$$

Expected Digamma Functions Since $\mathbb{E}[s_{j,k}]$ is w.r.t. both $q(\mathbf{z})$ and $q(\mathbf{s}|\mathbf{z})$, we first compute $\mathbb{E}_{q(\mathbf{s}|\mathbf{z})}[s_{j,k}]$ by using the conditional posterior in (5.91) and a property of Stirling numbers of the first kind in (5.87),

$$
\begin{aligned}
&\mathbb{E}_{q(\mathbf{z})}[\mathbb{E}_{q(\mathbf{s}|\mathbf{z})}[s_{j,k}]] \\
&= \mathbb{E}_{q(\mathbf{z})}\Bigg[\sum_{s_{j,k}=1}^{C_{j,k}} \frac{\Gamma(\sigma\mathbb{G}[\pi_k])}{\Gamma(\sigma\mathbb{G}[\pi_k]+C_{j,k})} \begin{bmatrix} C_{j,k} \\ s_{j,k} \end{bmatrix} (\sigma\mathbb{G}[\pi_k])^{s_{j,k}} s_{j,k}\Bigg] \quad \text{(by 5.91)} \\
&= \mathbb{E}_{q(\mathbf{z})}[\sigma\mathbb{G}[\pi_k](\psi(\sigma\mathbb{G}[\pi_k]+C_{j,k}) - \psi(\sigma\mathbb{G}[\pi_k]))] \quad \text{(by 5.87)} \\
&= \sigma\mathbb{G}[\pi_k](\mathbb{E}_{q(\mathbf{z})}[\psi(\sigma\mathbb{G}[\pi_k]+C_{j,k})] - \psi(\sigma\mathbb{G}[\pi_k]))
\end{aligned} \tag{5.92}
$$

Then we deal with the expected digamma function in (5.92). We reproduce our definitions here for convenience. Specifically, we have defined two disjoint sets of $\mathbf{z}$,

$$A \triangleq \{\mathbf{z} : C_{j,k}(\mathbf{z}) = 0\}, \tag{5.93}$$

$$B \triangleq \{\mathbf{z} : C_{j,k}(\mathbf{z}) > 0\}, \tag{5.94}$$

and introduced some shorthand notations,

$$q(C_{j,k} = 0) \triangleq \sum_{z \in A} q(\mathbf{z}), \tag{5.95}$$

$$q(C_{j,k} > 0) \triangleq \sum_{z \in B} q(\mathbf{z}), \tag{5.96}$$

$$\mathbb{E}_+[C_{j,k}] \triangleq \frac{\mathbb{E}[C_{j,k}]}{q(C_{j,k} > 0)}. \tag{5.97}$$

Given the definitions, we have the following trivial relations,

$$q(C_{j,k} > 0) = 1 - q(C_{j,k} = 0), \tag{5.98}$$

$$\sum_{\mathbf{z} \in B} q(\mathbf{z}) C_{j,k}(\mathbf{z}) = \sum_{\mathbf{z}} q(\mathbf{z}) C_{j,k}(\mathbf{z}), \tag{5.99}$$

where (5.98) holds by the fact that set A is the complement of set B and (5.99) holds by the definition of the set A, which implies $\sum_{\mathbf{z} \in A} q(\mathbf{z}) C_{j,k}(\mathbf{z}) = 0$.

We continue to compute (5.92) using the two relations (5.98-5.99),

$$\begin{aligned}
&\sigma\mathbb{G}[\pi_k](\mathbb{E}_{q(\mathbf{z})}[\psi(\sigma\mathbb{G}[\pi_k] + C_{j,k})] - \psi(\sigma\mathbb{G}[\pi_k])) \\
=&\sigma\mathbb{G}[\pi_k](\sum_{\mathbf{z} \in A} q(\mathbf{z}) \psi(\sigma\mathbb{G}[\pi_k]) + \sum_{\mathbf{z} \in B} q(\mathbf{z}) \psi(\sigma\mathbb{G}[\pi_k] + C_{j,k}) - \psi(\sigma\mathbb{G}[\pi_k])) \\
=&\sigma\mathbb{G}[\pi_k](\sum_{\mathbf{z} \in B} q(\mathbf{z}) \psi(\sigma\mathbb{G}[\pi_k] + C_{j,k}) - q(C_{j,k} > 0) \psi(\sigma\mathbb{G}[\pi_k])).
\end{aligned} \tag{5.100}$$

Since the $C_{j,k}$ in (5.100) is always great than 0 (due to $\mathbf{z} \in B$), we can apply the Taylor approximation to the first digamma function in (5.100), expanding at $\sigma\mathbb{G}[\pi_k] + \mathbb{E}_+[C_{j,k}]$,

$$\begin{aligned}
\psi(\sigma\mathbb{G}[\pi_k] + C_{j,k}) \approx &\psi(\sigma\mathbb{G}[\pi_k] + \mathbb{E}_+[C_{j,k}]) \\
&+ (C_{j,k} - \mathbb{E}_+[C_{j,k}]) \psi'(\sigma\mathbb{G}[\pi_k] + \mathbb{E}_+[C_{j,k}]).
\end{aligned} \tag{5.101}$$

Plugging (5.101) into (5.100) and applying (5.99) (the first order Taylor term becomes zero), we have,

$$\mathbb{E}[s_{j,k}] \approx \sigma\mathbb{G}[\pi_k] q(C_{j,k} > 0)(\psi(\sigma\mathbb{G}[\pi_k] + \mathbb{E}_+[C_{j,k}]) - \psi(\sigma\mathbb{G}[\pi_k])). \tag{5.102}$$

In the update equations for σ (5.76-5.77), we have also encountered expected digamma functions, $\mathbb{E}[\psi(\sigma + C_{j,\cdot})]$. Following a similar approach, we define two disjoint sets of $\mathbf{z}$,

$$A' \triangleq \{\mathbf{z} : C_{j,\cdot}(\mathbf{z}) = 0\}, \tag{5.103}$$

$$B' \triangleq \{\mathbf{z} : C_{j,\cdot}(\mathbf{z}) > 0\}, \tag{5.104}$$

and introduce the following shorthand notations

$$q(C_{j,\cdot} = 0) \triangleq \sum_{z \in A'} q(\mathbf{z}), \tag{5.105}$$

$$q(C_{j,\cdot} > 0) \triangleq \sum_{z \in B'} q(\mathbf{z}), \tag{5.106}$$

$$\mathbb{E}_+[C_{j,\cdot}] \triangleq \frac{\mathbb{E}[C_{j,\cdot}]}{q(C_{j,\cdot} > 0)}. \tag{5.107}$$

The expected digamma functions in (5.76-5.77) is computed as follows,

$$\begin{aligned}
&\mathbb{E}_{q(\mathbf{z})}[\psi(\sigma + C_{j,\cdot})] - \psi(\sigma) \\
=& \sum_{\mathbf{z} \in A'} q(\mathbf{z})\psi(\sigma) + \sum_{\mathbf{z} \in B'} q(\mathbf{z})\psi(\sigma + C_{j,\cdot}) - \psi(\sigma) \\
=& \sum_{\mathbf{z} \in B'} q(\mathbf{z})\psi(\sigma + C_{j,\cdot}) - q(C_{j,\cdot} > 0)\psi(\sigma) \\
\approx& \sum_{\mathbf{z} \in B'} q(\mathbf{z})\big(\psi(\sigma + \mathbb{E}_+[C_{j,\cdot}]) + (C_{j,\cdot} - \mathbb{E}_+[C_{j,\cdot}])\psi'(\sigma + \mathbb{E}_+[C_{j,\cdot}])\big) \\
&- q(C_{j,\cdot} > 0)\psi(\sigma) \quad \text{(by Taylor approximation)} \\
=& q(C_{j,\cdot} > 0)(\psi(\sigma + \mathbb{E}_+[C_{j,\cdot}]) - \psi(\sigma)).
\end{aligned} \tag{5.108}$$

Chapter 6

Conclusion and Future Work

Chapter Abstract

In this chapter, we conclude this thesis by overviewing the journey and progress made towards the goals of developing both accurate and efficient inference algorithms for the field of computational linguistics and enlarging the application domain of collapsed variational inference. We also suggest several future directions relating to the research presented in this thesis.

6.1 Conclusion

This thesis was motivated by the demand of developing both accurate and efficient approximate inference algorithms, since current inference tools have been found lacking due to their respective drawbacks in the large scale applications in computational linguistics (in terms of both data size and model complexity). To evaluate the proposed algorithms, we consider the deterministic and fast variational inference as the reference point of efficient algorithms, and the MCMC sampling with as many samples as possible in a computational feasible time as the reference point of accurate algorithms. Given that collapsed variational inference combines the efficiency of VI and accuracy of MCMC, we embarked on an exploration of applying CVI to models in computational linguistics, and our exploration greatly enlarges the application domain of CVI so that the combined advantages may benefit many models and applications in other fields as well.

Our journey started in Chapter 2, in which we interpreted collapsed models as fully connected Markov random fields to explain why collapsed variational inference is usually

considered as an inflexible framework (Wang and Blei, 2012). Due to the full connectivity, exact inference is clearly intractable, and even the naive mean field method requires a further Taylor approximation to arrive at tractable update equations in the simple bag-of-words setting, let alone developing more sophisticated algorithms in more complex models. Nevertheless, given the attractive empirical advantages of CVI, it is likely that we can make positive progress to address the demand in computational linguistics, if the challenges raised in extending CVI to more complex models are handled 'correctly' (e.g., by introducing accurate approximation methods, by algebraic manipulation and so on).

As an initial step, we began with hidden Markov models that are widely used and form one of the most important cornerstones in computational linguistics. We employed the naive mean field method to derive the first CVI algorithm. Although breaking the sequential dependencies seems a strong assumption, it is not a priori obvious whether it is stronger than the assumption of VI. Indeed, our experiment results suggested that most of time the benefits of collapsing outweigh the benefits of modelling sequential dependencies perfectly. It is not too surprising that our first CVI algorithm works well, since it is reminiscent of the first mean field algorithm that was proposed for the Ising model: the counts here are analogous to the model parameters in the Ising model; the sequential dependencies are analogous to the pairwise dependencies; and to update a variable's posterior, summing over its previous and next variables in our technique of computing the transition counts is analogous to the summation of neighbouring variables in the standard mean field equation.

Modelling larger substructures can yield more accurate approximation and there is no 'overfitting' regarding over-complex factorizations. Hence, we employed the structured mean field method to derive the second CVI algorithm. By taking advantages of both collapsing and modelling sequential dependencies perfectly, our experiment results showed that this algorithm significantly outperforms EM and VI and achieves comparable or even better results than MCMC; using the same forward-backward implementation as its core, it is as efficient as VI. Hence, we derived a both accurate and efficient inference algorithm for HMMs. The success was partly credited to using our approximation method to the joint posterior predictive distributions, which is very accurate by empirical validation.

We took a further step to relax the previous bag-of-clusters condition. Given a single long sequence, we broke it into subsequences in the variational distribution and applied the structured mean field method to derive the third CVI algorithm. Essential to this algorithm

is a propagation redesign technique that makes the incoming messages to a subsequence compatible with its local message propagation, providing consistent and commutable message passing between subsequences. This technique, combined with our contributions to CVI in the CE family building on the work of Sung et al. (2008), allows CVI to be generally applicable to a wide variety of probabilistic graphical models, by employing a systematic approach to reusing/redesigning the propagation algorithms of EM. This algorithmic shortcut from EM to CVI was detailed in Subsection 3.6.2 and parallels the previous algorithmic shortcut from EM to VI developed by Beal (2003).

The next generalization of HMMs in the Chomsky hierarchy, namely probabilistic context free grammars play a central role in the modelling of syntactic and semantic structures in computational linguistics. We applied the structured mean field method and derive a CVI algorithm for PCFGs. In several grammar induction tasks, we illustrated that our CVI algorithm can combine the advantages of VI and MCMC, not only for standard PCFGs, but also for the two popular variants that underline most high-performance parsers. Our derivation of CVI for PCFGs whose model structures are random objects demonstrates that CVI can be applied to a range of computational linguistic models with random latent structures such as tree substitution grammars (Cohn and Blunsom, 2010), adding an accurate and efficient technique to the computational linguists' toolbox.

Last we arrive at Bayesian Nonparametrics. The Bayesian nonparametric extensions of HMMs, namely HDP-HMMs, provide an ideal framework for managing related groups of data and are particularly useful when one can not decide a priori the number of states in computational linguistic models or other applications. We derived a CVI algorithm for HDP-HMMs by employing the existing algorithmic techniques in CVI for HDP-LDA (Teh et al., 2008) and proposing our algorithms for computing the probabilities of each transition's existence, which are essential but complicated due to the sequential dependencies of HMMs. In particular, our simple and efficient heuristic algorithm is very accurate, even more so than drawing $10{,}000$ samples. Such heuristic fits naturally for many computational linguistic models using HDP constructions for example HDP-PCFGs, since its essential insight is to replace a large structure with a set of overlapping but independent substructures. The empirical evaluation of our CVI algorithm showed that it reliably determines the number of effective states, and outperforms several recent benchmarks including beam sampling, two VI algorithms, and our CVI algorithm for finite HMMs. Thus it provides a

new scalable and strong benchmark for time series analysis.

With all the above achievements in our exploration, we conclude that we have made positive progress in the development of both accurate and efficient proximate inference algorithms for the field of computational linguistics and in the endeavour of enlarging the application domain of collapsed variational inference.

6.2 Future Work

This concludes our current research, but we believe that the exploration of CVI will continue in future. Our suggestions for the future directions are as follows:

1. To apply CVI to more complex models or systems in computational linguistics. In particular, the two variants of PCFGs we demonstrated CVI on are the core building blocks for most high-performance parsers that are often trained by EM. Applying CVI to those systems might yield state-of-art results. Another work is to develop CVI for Hierarchical Pitman-Yor Process (HPYP), which generalizes HDP and produces the power-law distributions that are often seen in natural language, but currently can only be trained by relatively slower MCMC sampling.

2. To develop automated system for deriving CVI algorithms. Utilizing our exploration of various factorization strategies in the collapsed space, and in particular our systematic propagation redesign method, we believe such automated procedure can be embedded in general purpose software such as VIBES (Winn and Bishop, 2005).

3. To use the split and merge mechanism in the CVI framework. Given the encouraging results in VI and memorized VI (Bryant and Sudderth, 2012; Hughes and Sudderth, 2013; Hughes et al., 2015a), we expect applying their split and merge mechanism can also help CVI to escape local optima and further improve its performance.

4. To apply CVI to deep architectures. The variational idea has been exploited in neural networks with several successful examples, including Kingma and Welling (2013) and Mnih and Gregor (2014). A recent work on CVI for the sum-product networks (Zhao et al., 2016) may have just opened the door for CVI in this new territory.

Bibliography

Edoardo M. Airoldi, David Blei, Elena A. Erosheva, and Stephen E. Fienberg. *Handbook of Mixed Membership Models and Their Applications*. Chapman Hall, 2014.

Arthur Asuncion, Max Welling, Padhraic Smyth, and Yee Whye Teh. On Smoothing and Inference for Topic Models. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, 2009.

James K. Baker. Trainable Grammars for Speech Recognition. *The Journal of the Acoustical Society of America*, 1979.

Leonard E. Baum and Ted Petrie. Statistical Inference for Probabilistic Functions of Finite State Markov Chains. *The Annals of Mathematical Statistics*, 1966.

Leonard E. Baum, Ted Petrie, George Soules, and Norman Weiss. A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Function of Markov Chains. *The Annals of Mathematical Statistics*, 1970.

Matthew Beal. *Variational Algorithms for Approximate Bayesian Inference*. PhD thesis, The Gatsby Computational Neuroscience Unit, University College London, 2003.

Matthew J. Beal, Zoubin Ghahramani, and Carl E. Rasmussen. The Infinite Hidden Markov Model. In *Advances in Neural Information Processing Systems*, 2002.

Taylor Berg-Kirkpatrick, Alexandre Bouchard-Côté, John DeNero, and Dan Klein. Painless Unsupervised Learning with Features. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2010.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

David M. Blei and Michael I. Jordan. Variational Inference for Dirichlet Process Mixtures. *Bayesian Analysis*, 2005.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 2003.

Phil Blunsom and Trevor Cohn. Unsupervised Induction of Tree Substitution Grammars for Dependency Parsing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 2010.

Phil Blunsom and Trevor Cohn. A Hierarchical Pitman-Yor Process HMM for Unsupervised Part of Speech Induction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011.

Peter F. Brown, John Cocke, Stephen Della Pietra, Vincent J. Della Pietra, Frederick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, 1990.

Michael Bryant and Erik B. Sudderth. Truly Nonparametric Online Variational Inference for Hierarchical Dirichlet Processes. In *Advances in Neural Information Processing Systems*, 2012.

Olivier Cappé and Eric Moulines. Online EM Algorithm for Latent Data Models. *Journal of the Royal Statistical Society*, 2007.

Eugene Charniak. Statistical Parsing with a Context-free Grammar and Word Statistics. In *Proceedings of the fourteenth national conference on artificial intelligence and ninth conference on Innovative applications of artificial intelligence*, 1997.

Eugene Charniak and Mark Johnson. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, 2005.

Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. Two Decades of Unsupervised POS Induction: How Far Have We Come? In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 2010.

Trevor Cohn and Phil Blunsom. Blocked inference in Bayesian tree substitution grammars. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, page To Appear, Uppsala, Sweden, 2010.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistics Society*, 1977.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. A simple, fast, and effective reparameterization of IBM Model 2. In *In Proc. NAACL*, 2013.

Michael D. Escobar and Mike West. Bayesian Density Estimation and Inference Using Mixtures. *Journal of the American Statistical Association*, 1994.

Thomas S. Ferguson. A Bayesian Analysis of Some Nonparametric Problems. *The Annals of Statistics*, 1973.

Nicholas Foti, Jason Xu, Dillon Laird, and Emily Fox. Stochastic Variational Inference for Hidden Markov Models. In *Advances in Neural Information Processing Systems*, 2014.

James R. Foulds, L. Boyles, C. DuBois, Padhraic Smyth, and Max Welling. Stochastic Collapsed Variational Bayesian Inference for Latent Dirichlet Allocation. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2013.

Emily B. Fox, Erik B. Sudderth, Michael I. Jordan, and Alan S. Willsky. An HDP-HMM for Systems with State Persistence. In *Proceedings of the 25th International Conference on Machine Learning*, 2008.

Yarin Gal and Phil Blunsom. A systematic Bayesian treatment of the IBM alignment models. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2013.

Jianfeng Gao and Mark Johnson. A Comparison of Bayesian Estimators for Unsupervised Hidden Markov Model PoS Taggers. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, 2008.

Zoubin Ghahramani and Michael I. Jordan. Factorial Hidden Markov Models. *Machine Learning*, 1997.

Walter R. Gilks. *Markov Chain Monte Carlo in Practice*. Chapman and Hall, London, 1996.

S. Goldwater, T. L. Griffiths, and M. Johnson. A Bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112:21–54, 2009.

Sharon Goldwater and Tom Griffiths. A fully Bayesian approach to unsupervised part-of-speech tagging. In *Proc. of the 45th Annual Meeting of the ACL (ACL-2007)*, pages 744–751, Prague, Czech Republic, June 2007.

Joshua T. Goodman. *Parsing inside-out*. PhD thesis, Harvard University, Cambridge, MA, USA, 1998.

David Graff. English Gigaword. 2003.

Thomas Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America*, 2004.

William P. Headden III, Mark Johnson, and David McClosky. Improving unsupervised dependency parsing with richer contexts and smoothing. In *Proceedings of the 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2009.

Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *J. Mach. Learn. Res.*, 14(1):1303–1347, May 2013. ISSN 1532-4435.

Michael C. Hughes and Erik B. Sudderth. Memoized online variational inference for dirichlet process mixture models. In *Advances in Neural Information Processing Systems*, 2013.

Michael C. Hughes, Dae Il Kim, and Erik B. Sudderth. Reliable and scalable variational inference for the hierarchical dirichlet process. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, 2015a.

Michael C Hughes, William T Stephenson, and Erik Sudderth. Scalable adaptation of state complexity for nonparametric hidden markov models. In *Advances in Neural Information Processing Systems 28*, 2015b.

William H. Jefferys and James O. Berger. Ockham's Razor and Bayesian Analysis. *American Scientist*, 1992.

Finn V. Jensen. *Introduction to Bayesian Networks*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1996.

Mark Johnson. Why doesnt EM find good HMM POS-Taggers. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL). Prague, Czech Republic: Association for Computational Linguistics*, pages 296–305, 2007.

Mark Johnson, Thomas Griffiths, and Sharon Goldwater. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *Proc. of the 7th International Conference on Human Language Technology Research and 8th Annual Meeting of the NAACL (HLT-NAACL 2007)*, pages 139–146, Rochester, New York, April 2007.

Matthew Johnson and Alan Willsky. Stochastic variational inference for Bayesian time series models. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1854–1862. JMLR Workshop and Conference Proceedings, 2014.

Michael I. Jordan, editor. *Learning in Graphical Models*. MIT Press, Cambridge, MA, USA, 1999.

Michael I. Jordan, Zoubin Ghahramani, and Lawrence K. Saul. Hidden markov decision trees. In *Advances in Neural Information Processing Systems*, 1997.

Biing-Hwang Juang and Lawrence. R. Rabiner. Hidden Markov models for speech recognition. *Technometrics*, 33(3):pp. 251–272, 1991.

Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 2000. ISBN 0130950696.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *Proceeings of the 1st International Conference on Learning Representations*, 2013.

Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, 2003.

Dan Klein and Christopher D. Manning. Corpus-based induction of syntactic structure: models of dependency and constituency. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 478, 2004.

Reinhard Kneser and Hermann Ney. Improved back off for m gram language modelling. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 1995.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Proc. of the 45th Annual Meeting of the ACL (ACL-2007)*, Prague, 2007.

F. R. Kschischang, B. J. Frey, and H. A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. Inf. Theor.*, 47(2):498–519, September 2001.

K. Kurihara, M. Welling, and Y. W. Teh. Collapsed Variational Dirichlet Process Mixture Models. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2007.

Kenichi Kurihara and Taisuke Sato. Variational bayesian grammar induction for natural language. In *Proceedings of the 8th international conference on Grammatical Inference: algorithms and applications*, 2006.

Karim Lari and Steve J Young. The estimation of stochastic context-free grammars using the Inside-Outside algorithm. *Computer Speech and Language*, 1990.

S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. In *Readings in Uncertain Reasoning*. Morgan Kaufmann Publishers Inc., 1988.

Percy Liang, Slav Petrov, Michael Jordan, and Dan Klein. The infinite PCFG using hierarchical Dirichlet processes. In *Proc. of the 2007 Conference on Empirical Methods in Natural Language Processing (EMNLP-2007)*, pages 688–697, Prague, Czech Republic, 2007.

David MacKay. Ensemble learning for hidden Markov models. Technical report, Cavendish Laboratory, University of Cambridge, 1997.

Brian MacWhinney and Catherine Snow. The child language data exchange system. *Child Language*, 1985.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of English: the Penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.

Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. Probabilistic cfg with latent annotations. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, 2005.

Bernard Merialdo. Tagging English text with a probabilistic model. *Comput. Linguist.*, 1994.

Thomas Minka. Divergence measures and message passing. Technical report, Microsoft Research, 2005.

Thomas P. Minka. Estimating a Dirichlet distribution. Technical report, Massachusetts Institute of Technology, 2009.

Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. In *Proceedings of the 31th International Conference on Machine Learning*, 2014.

Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

Radford M. Neal. Probabilistic inference using markov chain monte carlo methods. Technical report, University of Toronto, CA, 1993.

Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–52, 2003.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, 2002.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

Slav Petrov and Dan Klein. Improved inference for unlexicalized parsing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, 2007.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, 2006.

Carl Edward Rasmussen. The infinite gaussian mixture model. In *In Advances in Neural Information Processing Systems 12*, pages 554–560. MIT Press, 2000.

Darcey Riley and Daniel Gildea. Improving the ibm alignment models using variational bayes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, 2012.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 1951.

Andrew Rosenberg and Julia Hirschberg. V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2007.

Yasubumi Sakakibara. Grammatical inference in bioinformatics. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2005.

Issei Sato and Hiroshi Nakagawa. Rethinking collapsed variational Bayes inference for LDA. In *Proceedings of the 29th International Conference on Machine Learning*, 2012.

L Scott. Bayesian methods for hidden Markov models: Recursive computing in the 21st century. *Journal of the American Statistical Association*, 97:337–351, 2002.

J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.

Noah A. Smith and Jason Eisner. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 354–362, Ann Arbor, Michigan, June 2005.

Padhraic Smyth, David Heckerman, and Michael I. Jordan. Probabilistic independence networks for hidden Markov probability models. *Neural Comput.*, 9(2):227–269, February 1997. ISSN 0899-7667.

Jaemo Sung, Zoubin Ghahramani, and Sung-Yang Bang. Latent-space variational Bayes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(12), December 2008.

Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

Y. W. Teh, K. Kurihara, and M. Welling. Collapsed Variational Inference for HDP. In *Advances in Neural Information Processing Systems*, 2008.

Yee Whye Teh. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, 2006.

Yee Whye Teh, David Newman, and Max Welling. A collapsed variational Bayesian inference algorithm for latent Dirichlet allocation. In *In Advances in Neural Information Processing Systems, volume 19*, 2007.

Jurgen Van Gael, Yunus Saatci, Yee W. Teh, and Zoubin Ghahramani. Beam sampling for the infinite hidden markov model. In *ICML '08: Proceedings of the 25th international conference on Machine learning*, 2008.

S. Vogel, H. Ney, and C. Tillmann. HMM-based word alignment in statistical translation. In *Proc. of the 16th International Conference on Computational Linguistics (COLING '96)*, pages 836–841, Copenhagen, Denmark, August 1996.

Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, 2008.

Chong Wang and David Blei. Truncation-free stochastic variational inference for Bayesian nonparametric models. In *Advances in Neural Information Processing Systems*, 2012.

Chong Wang, John William Paisley, and David M. Blei. Online variational inference for the hierarchical dirichlet process. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, pages 752–760, 2011.

Pengyu Wang and Phil Blunsom. Collapsed Variational Bayesian Inference for Hidden Markov Models. In *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Scottsdale, AZ, USA, 2013a.

Pengyu Wang and Phil Blunsom. Collapsed Variational Bayesian Inference for PCFGs. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning (CONLL)*, Sofia, Bulgaria, 2013b. Association for Computational Linguistics.

Pengyu Wang and Phil Blunsom. Stochastic Collapsed Variational Inference for Sequential Data. In *NIPS Workshop on Advances in Approximate Bayesian Inference*, Montreal, Canada, 2015a.

Pengyu Wang and Phil Blunsom. Stochastic Collapsed Variational Inference for Hidden Markov Models. In *NIPS Workshop on Time Series Analysis*, Montreal, Canada, 2015b.

Niclas Wiberg. *Codes and Decoding on General Graphs*. PhD thesis, Linköping University, 1996.

Wim Wiegerinck. Variational approximations between mean field theory and the junction tree algorithm. In *In Uncertainty in Artificial Intelligence*, pages 626–633. Morgan Kaufmann, 2000.

John Winn and Christopher M. Bishop. Variational Message Passing. *Journal of Machine Learning Research*, 6:661–694, 2005.

Eric P. Xing, Michael I. Jordan, and Stuart Russell. A generalized mean field algorithm for variational inference in exponential families. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence*, pages 583–591, 2003.

Aonan Zhang, San Gultekin, and John Paisley. Stochastic variational inference for the HDP-HMM. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, AISTATS 2016, Cadiz, Spain, May 9-11, 2016*, 2016.

Han Zhao, Tameem Adel, Geoffrey J. Gordon, and Brandon Amos. Collapsed variational inference for sum-product networks. In *Proceedings of the 33th International Conference on Machine Learning*, 2016.

SongChun Zhu and David Mumford. A stochastic grammar of images. *Foundations and Trends in Computer Graphics and Vision*, 2007.