Learning Patterns from Sequential and Network Data Using Probabilistic Models

Yin Cheng Ng

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of

University College London.

Department of Statistical Science

University College London

April 14, 2019

I, Yin Cheng Ng, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

The focus of this thesis is on developing probabilistic models for data observed over temporal and graph domains, and the corresponding variational inference algorithms. In many real-world phenomena, sequential data points that are observed closer in time often exhibit higher degrees of dependency. Similarly, data points observed over a graph domain (e.g., user interests in a social network) may exhibit higher dependencies with lower degrees of separation over the graph. Furthermore, the connectivity structures that define the graph domain can also evolve temporally (i.e., temporal networks) and exhibit dependencies over time. The data sets observed over temporal and graph domains often (but not always) violate the independent and identically distributed (i.i.d.) assumption made by many mathematical models. The works presented in this dissertation address various challenges in modelling data sets that exhibit dependencies over temporal and graph domains. In Chapter 3, I present a stochastic variational inference algorithm that enables factorial hidden Markov models for sequential data to scale up to extremely long sequences. In Chapter 4, I propose a simple but powerful Gaussian process model that captures the dependencies of data points observed on a graph domain, and demonstrate its viability in graph-based semi-supervised learning problems. In Chapter 5, I present a dynamical model for graphs that captures the temporal evolution of the connectivity structures as well as the sparse connectivity structures often observed in temporal real network data sets. Finally, I summarise the contributions of the thesis and propose several directions for future works that can build on the proposed methods in Chapter 6.

Impact Statement

The research results presented in this thesis build upon the contributions of other researchers in the machine learning and statistics communities. The completion of the works in this thesis would not have been possible without their contributions. In return, I hope that my work will benefit researchers working inside and outside of academia in the following ways.

1. As big data has become commonplace, the algorithm presented in Chapter 3 can benefit researchers who need to model big data with sequential dependency. While the algorithm was designed with a specific probabilistic model in mind, components of the algorithm can be adapted to develop scalable algorithms for other probabilistic sequential models.

2. The work should encourage researchers and practitioners to think more deeply about the dependency between data points and ways to exploit the dependency to construct better models under the probabilistic modelling framework. The result presented in Chapter 4 should serve as an example of one approach to model dependency in data sets with network structures.

3. The work presented in Chapter 5 aims to inspire researchers in social networks to develop dynamic network models that can better capture properties observed empirically in real temporal network data sets.

Acknowledgements

First of all, I would like to thank my supervisor, Ricardo Silva, for his support and generosity with his time. Our weekly meetings to discuss research ideas in the past three and a half years have truly expanded my mind and understanding of what research is all about. I would also like to thank University College London and the Department of Statistical Science for their financial support that allowed me unrestricted freedom to explore ideas in the past few years. I am grateful to Nicolò Colombo and Sandipan Roy for the many fruitful conversations about research and beyond.

I would like to give a special shout-out to Vincent Kan for his friendship and for keeping me grounded over the years. To Colleen, for making the past three years of my life so special and memorable. Finally, I am forever in debt to my family, for their unconditional support and encouragement.

Contents

List of Figures

List of Tables

1

Introduction

Data sets observed in the real-world can arise from many different complex processes. Some processes include dynamical components that result in observations which exhibit sequential dependency and repeatable patterns over time. Some other processes may include interactive components where interactions between entities in the system result in observations that exhibit dependency with respect to the patterns of interactions. The interactions between entities in the system form a network, and is often expressed as a graph. For example, data points observed on entities (i.e., node labels in a network) that have interacted can exhibit dependency as a result of the interactions. Additionally, networks that encode patterns of interactions between entities can also exhibit sequential dependency as a result of underlying dynamical components that drive the system. Sequential and network dependencies are very important features of many data sets. These features should be accounted for and exploited by statistical models whenever possible in order to adequately capture the patterns in the data sets.

Examples of sequential data are abundant in our everyday lives, ranging from the music that we listen to, to the DNA sequences that encode our biological traits. Another setting where important sequential data sets are collected is in the financial markets, where participants buy and sell financial products at prices informed by information available to the participants. The time series of transaction prices recorded

in the markets can exhibit both sequential and cross-sectional dependencies due to the varying speeds of market participants in receiving, processing and reacting to information, among many other reasons. As the technologies to capture, process and store sequential data improve, more and more sequential data sets that are extremely long and rich in structures have become available for analysis. A relevant example is the increasing adoptions of smart electricity meters which record household electricity consumption at minute intervals. The higher frequency at which the data is captured not only results in longer sequences, but also captures intra-day usage patterns which were not previously available. The availability of long sequential data sets with rich structures calls for powerful and flexible models that can scale up to the sizes of these data sets. Furthermore, sequential data sets are not restricted to vector-valued sequences. Another interesting type of sequential data sets that are often observed in the real-world are temporal sequences of network data, which I will discuss later in this section.

Apart from sequential dependency, another type of dependency that is both important and commonplace in real-world data sets is network dependency. In data sets where network structures exist, the data points are typically viewed as vertices[1] in the networks and the network edges[2] represent the relationships between the data points. The presence of an edge between two vertices (i.e., 1-hop neighbours) typically implies positive associations between the two data points on the vertices. While no theorem dictates that 1-hop neighbours are always positively associated, such positive associations are widely observed empirically, and is known as homophily in the literatures [Goldenberg et al., 2010]. The dependencies between vertices fade away as the numbers of hops between pairs of vertices increase. As such, it is natural to view the network structure as an irregular coordinate system in non-Euclidean space where distances between pairs of data points are measured in the numbers of hops separating the pairs. Looking at network data through this coordinate system view allows us to draw parallels between sequential and network dependencies, in

[1] Vertices are also referred to as nodes in the literatures.

[2] Edges are also referred to as links or connections in the literatures.

that sequential data consists of a list of data points ordered on a regular 1-dimensional grid in the Euclidean space, and the dependencies between pairs of data points in the sequence decrease as the Euclidean distances between them increase.

Examples of network data in the wild include communication networks, where vertices and edges represent individuals and their interactions respectively; computer networks where computers are connected through communication links; rail networks where train stations are connected by train tracks, and many others. A particularly illustrative example of network data are citation networks. Vertices in citation network are academic publications, and the edges in the network connect each vertex to a list of other publications cited in its bibliography. Each vertex may also include covariates, such as the bag-of-word representation of the publication, as well as a label that indicates the field/sub-field that the publication is associated with. As academic papers that belong to the same sub-fields are more likely to cite each others, the vertex labels of neighbours are typically positively associated, giving rise to dependency over the citation network connectivity structure. Clearly, the network dependency should be taken into account when building statistical models.

In addition to data sets with naturally observed network structures, practitioners have also invented various heuristics and algorithms to create networks for independently and identically distributed (i.i.d.) data sets that can suitably represent the relationships between the data points (e.g., Argyriou et al. [2006]). The derived network representations allow powerful algorithms that operate on networks to be applied to the data sets.

Network data sets are complex in nature, with a multitude of properties that can be difficult to capture by any single model. Understanding different aspects of a network data set often requires modellers to look at the data through many different lenses. The description of networks in the previous paragraphs assumed that the fundamental datum of the network data sets is the vertex. While many existing models and algorithms implicitly make the same assumption of treating vertices as the fundamental data unit, this is by no mean the only way to think about networks.

Given different modelling objectives, it may be more intuitive and productive to think of edges, triplets, hyper-edges or even paths in the networks as the fundamental units, as discussed in details in Crane [2018], and model them as such.

Moving Average Models: Sequences and Networks

Moving average process is one of the most fundamental and well-known stochastic process in the time series literature. Given a sequence indexed by positive integers $(y_1, y_2, y_3, \ldots)$, the moving average process models the dependency between data points in a sequence as

$$y_t = \sum_{s=1}^{q} a_s \varepsilon_{t-s} + \varepsilon_t \tag{1.1}$$

where $\varepsilon_t \sim \mathcal{N}(0, \sigma^2)$ is a sequence of i.i.d. Gaussian white noise and $a_1, \ldots, a_q$ are the model parameters. The model in Equation (1.1) is known as the $MA(q)$ process. The auto-covariance structure of the sequence as specified by $MA(q)$ can be easily derived from the first principle.

The moving average model can also be extended to data sets with network structure. Given data points $\{y_n | n \in 1, \ldots, N\}$ observed on the vertices of a network $\mathcal{G}$ with N vertices, one can specify

$$y_n = \sum_{s=1}^{q} \Big(\frac{a_s}{|Ne(s,n)|} \sum_{i \in Ne(s,n)} \varepsilon_i \Big) + \varepsilon_n \tag{1.2}$$

where $\varepsilon_n \sim \mathcal{N}(0, \sigma^2)$, $a_1, \ldots, a_q$ are model parameters and $Ne(s,n)$ is the set of indices denoting the $s-$hop neighbours of vertex n. For q=1, it is easy to see that

$$\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \mathbf{P}\mathbf{P}^\intercal) \tag{1.3}$$

where $\mathbf{y} = [y_1, \ldots, y_N]^\intercal$, $\mathbf{P} = (\mathbf{I} + \mathbf{D})^{-1}(\mathbf{I} + \mathbf{A})$, $\mathbf{A} \in \{0,1\}^{N \times N}$ is the adjacency matrix for the network $\mathcal{G}$ and $\mathbf{D} \in \mathbb{Z}^{N \times N}$ is the diagonal vertex degree matrix.

The models specified in Equation (1.1) and (1.2) are similar in that the data points are dependent on their adjacent data points. However, the notions of adjacency differ in the two scenarios. In the sequential data, adjacent data points are defined as the data points that precede the data point of interest. In the network, adjacent data points refer to the neighbours of the data point of interest at different hops. The two models presented above illustrated the conceptual similarities between modelling sequential and network data sets.

In addition to the data points observed on the vertices of network, the connectivity structure of the network itself is often of modelling interest. While the network connectivity often exhibits rich structure, it is difficult to gain deeper insights, such as understanding the community structure of the network, without models. Network data sets are intrinsically temporal as it is extremely rare to observe all the vertices and edges of a network at the same moment in time: vertices typically join the network at different time points while connections form and vanish over time. Therefore, it may be important to account for the temporal aspect of network data sets using dynamic network models whenever possible.

One example of the network data sets in which the temporal aspect is important is the communication network. In a communication network, users of the communication service (e.g., e-mails, telecommunications etc.) are represented by the vertices, and the communication records between pairs of users are the edges. The edges are typically time-stamped at the moment when the users communicate. Aggregating the edges that are observed daily, we can construct a sequence of communication networks in which each network in the sequence represents a snapshot of communications between the users during the day. The sequence of networks may be temporally correlated as users who had communicated previously are more likely to communicate again in the near-future.

The focus of this thesis is on the development of probabilistic models and variational inference algorithms that can capture the non-trivial dependency structures of sequences, networks and sequences of networks data sets. The probabilistic models of primary interest are probabilistic graphical models (PGMs) and Gaussian processes (GPs). These probabilistic methods provide a convenient, modular and powerful way to specify models that can capture different types of dependency structure intended by the modellers. However, performing Bayesian inference in complex probabilistic models has remained a difficult challenge without a satisfying universal solution, and requires bespoke approximate inference algorithms to be developed. Therefore, in addition to proposing novel probabilistic models for dependent data,

the development of suitable variational inference algorithms for the proposed models forms a key part of this thesis. A review of the core concepts, algorithms and probabilistic models that are the building blocks of the works presented in this thesis is available in Chapter 2.

1.1 Contributions

The contributions made in this dissertation are summarised as follow:

1. The proposal of a novel scalable variational inference algorithm for factorial hidden Markov models (FHMMs) in Chapter 3. The proposed algorithm extends the stochastic variational inference algorithm proposed in Hoffman et al. [2013] to FHMM, and takes advantage of a novel Gaussian-Bernoulli copula parameterization of Markov chain variational distribution that lends itself to amortized inference using feed-forward recognition neural networks. The computational complexity of the proposed algorithm is sub-linear with respect to the length of the data sequences, allowing FHMMs to be applied to very long sequences under limited computing budgets. This is a joint work with Pawel Chilinski.

2. The proposal of a data efficient Gaussian process model for semi-supervised learning on graphs in Chapter 4. The proposed model shows extremely competitive performance when compared to the state-of-the-art graph neural networks on semi-supervised learning benchmark experiments, and outperforms the neural networks in active learning experiments where labels are scarce. Furthermore, the model does not require a validation data set for early stopping to control over-fitting. The model can be viewed as an instance of empirical distribution regression weighted locally by network connectivity. Its intuitive construction is further motivated by a Bayesian linear model interpretation where the node features are filtered by an operator related to the graph Laplacian. The method can be easily implemented by adapting off-the-shelf scalable

variational inference algorithms for Gaussian processes.

3. The proposal of a dynamic edge exchangeable random network model that can capture sparse connections observed in real temporal networks, in contrast to existing dynamic models which can only model dense networks. The model achieved good link prediction accuracy on multiple data sets when compared to the benchmark models, and is able to extract interpretable time-varying community structures from the data. In addition to sparsity, the model accounts for the effect of social influence on vertices' future behaviours. Compared to the dynamic blockmodels, the proposed model has a smaller latent space. The compact latent space requires a smaller number of parameters to be estimated in variational inference and results in a computationally friendly inference algorithm.

The works presented in Chapter 3 and Chapter 4 have been published as the following self-contained articles in conference proceedings.

- Y.C. Ng, P. Chilinski, R. Silva. Scaling Factorial Hidden Markov Models: Stochastic Variational Inference without Messages. In NIPS, 2016.
- Y.C. Ng, N. Colombo, R. Silva. Bayesian Semi-supervised Learning with Graph Gaussian Processes. To appear in NIPS, 2018.

2

Background

In this chapter, I survey important concepts, models and algorithms that are the building blocks of the works presented in this dissertation, and provide relevant references to the literatures. The topics surveyed in this chapter are general and non-exhaustive. Technical concepts that are more specific to the three pieces of work presented in this dissertation are surveyed in the relevant chapters.

At the core of probabilistic models are joint probability distributions that specify the statistical dependencies between the observed data and a set of latent random variables that explain the data. In Section 2.1, I survey the probabilistic graphical model (PGM) as a flexible framework to compose joint probability distributions over a large number of random variables, with a focus on the directed PGMs. In Section 2.2, I introduce the dynamic directed graphical models, which are discrete-time stochastic processes constructed through recursively specified conditional distributions. The two types of dynamic PGM discussed are the hidden Markov Model (HMM) and the linear dynamical system (LDS). The HMM is relevant to the work presented in Chapter 3 while the LDS is related to the work in Chapter 5.

In Section 2.3, I discuss some graphical models for network data sets. The probabilistic graphical models for networks are related to the models proposed in Chapter 5. Starting with some discussions on the probabilistic graphical models for

static network, I then proceed to discuss their dynamic extensions through temporally coupling the graphical models for static network using LDS and HMM. The resulted dynamic network models can capture the temporal evolution of the connectivity in sequences of networks.

In addition to the probabilistic graphical models, I discuss the Gaussian process (GP) as a flexible Bayesian non-parametric model for supervised learning in Section 2.4. Using the vanilla GP as a building block, I propose a simple but effective Bayesian model for semi-supervised learning on graphs in Chapter 4.

Finally, I discuss the variational inference algorithms in Section 2.5. Variational inference serves as an important tool to approximate the intractable posterior distributions encountered in probabilistic models, and is extensively used in the works presented in the dissertation.

2.1 Probabilistic Graphical Models

A rich framework to construct high-dimensional joint probability distribution of data is the probabilistic graphical model (PGM). PGM allows a joint distribution to be decomposed into a product of factors. The decomposition allows for a more succinct representation of the probabilistic model compared to a direct specification of the joint distribution because of the inductive bias introduced in the decomposition process. Additionally, inference for PGMs can be performed using a suite of well-developed algorithms.

The two main categories of PGMs are the directed graphical models and the undirected graphical models. Directed graphical models are typically chosen when there is plausible hierarchical or causal structure in the data, such that the joint distributions can be factorised accordingly. The time series and network models of interest in this report mainly fall into the directed graphical model category because of the causal nature of time and the hierarchical structure in network interactions [Fox, 2009].

More formally, a directed graphical model is a directed acyclic graph (DAG) where each node in the DAG represents a random variable with directed incoming edges from its parent nodes and outgoing edges to its child nodes. The joint probability distribution for the random variables of interest can be expressed as a product of conditional distributions, with one conditional distribution for each node in the DAG and a conditioning set that corresponds to its parent nodes. Two nodes in the DAG that are independent a priori may be conditionally dependent when conditioned on their common descendants. This phenomenon is known as explaining away, and plays a critical role in determining the conditional independence relationships encoded by the directed graphical model. To efficiently determine the conditional independence relationships encoded by a directed graphical model, one can make use of the Bayes ball algorithm [Murphy, 2012].

Inference in PGMs amounts to computing the conditional probability distributions of the unobserved random variables conditioning on the observed data, known as the posterior distributions. For tree-structured PGMs, inference can be performed efficiently using dynamic programming algorithms that exploit the conditional independence structure of the graphical models [Barber, 2012]. However, computing the posterior distributions of general PGMs are typically difficult because of the intractable high-dimensional integrals and/or summations in the normalising constants that need to be evaluated. Therefore, practitioners resort to approximate inference algorithms to approximate the posterior distributions.

2.2 Probabilistic Time Series Models

In this section, I briefly review the hidden Markov model (HMM), the linear dynamical system (LDS) and the switching linear dynamical system (sLDS). These probabilistic time series models are stochastic processes that can be expressed as dynamic directed graphical models.

Building on the foundation for the HMM and LDS laid down in the following

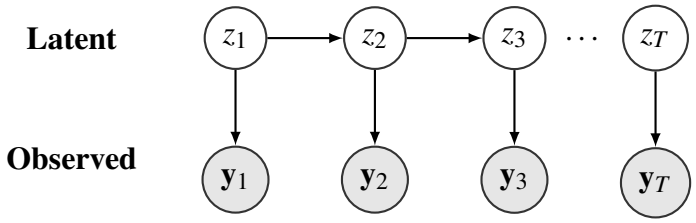

Figure 2.1: This figure shows a directed probabilistic graphical model of the hidden Markov model. This is also a valid graphical model for the linear dynamical system when the latent variables are continuous. The structure of the graphical model implies that conditioning on the present latent variable $z_{t'}$, the latent variables in the future $z_{t>t'}$ are independent of the latent variables in the past $z_{t<t'}$.

sub-sections, we describe a powerful extension of the HMM called the factorial HMM in chapter 3, and propose a scalable variational inference algorithm for the factorial HMM. In chapter 5, we propose a dynamic network model that leverages a variant of the LDS to model the temporal dependence in dynamic networks.

2.2.1 Hidden Markov Models

The hidden Markov model is a class of discrete time stochastic process with a latent discrete-valued Markov process where given the latent state at a particular time point, the observation is independent of other observations and distributed according to the state-specific probability distributions. The probability distributions for observations (i.e., the emission distributions), can either be discrete or continuous depending on the type of data observed. Intuitively, HMMs can be thought of as mixture models where the data point specific latent cluster assignments are distributed according to the underlying Markov process.

Given a sequence of observations $\mathbf{y}_1, ..., \mathbf{y}_T$, the joint probability distribution of the observed sequence and the corresponding latent states of the Markov process $z_1, ..., z_T$ parameterised by the model parameters θ are as follow.

$$p_\theta(\mathbf{y}_{1:T}, z_{1:T}) = p_\theta(z_1) p_\theta(\mathbf{y}_1|z_1) \prod_{t=2}^{T-1} p_\theta(z_t|z_{t-1}) p_\theta(\mathbf{y}_t|z_t) \tag{2.1}$$

The joint distribution in Equation (2.1) is a product of the *initial distribution*

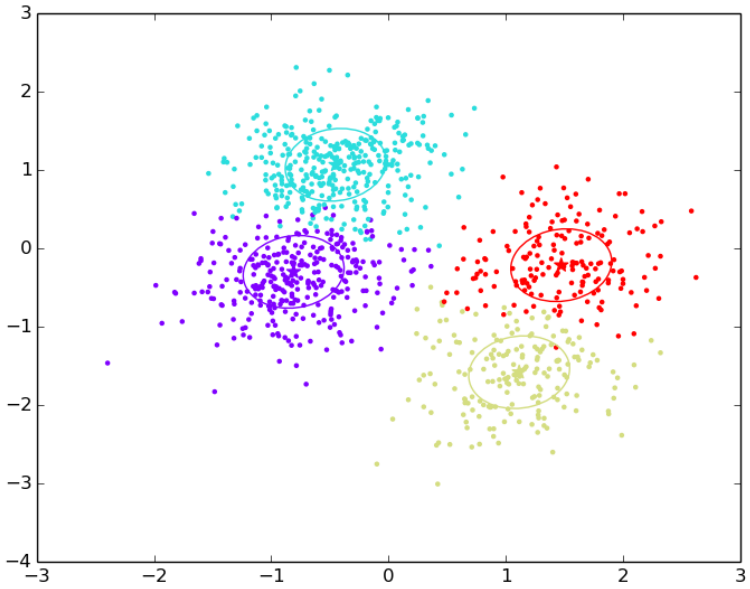

Figure 2.2: This figure shows some samples drawn from a 4-state HMM with 2-dimensional Gaussian emission distributions. The stars represent the means of the Gaussian distributions and the ellipses represent the covariance. The colored dots are the samples. The colors encode the 4 different states of the HMM.

$p_\theta(z_1)$, the *transition distribution* $p_\theta(z_t|z_{t-1})$ and the *emission distribution* $p_\theta(\mathbf{y}_t|z_t)$. The initial and transition distributions of the Markov process are typically chosen to be categorical distributions. The model parameters θ can be estimated from the data using the expectation-maximisation (EM) algorithm [Dempster et al., 1977].

Given the observed sequence $\mathbf{y}_{1:T}$, exact inference in HMM can be performed efficiently with dynamic programming algorithms as the corresponding graphical model has tree structure that implies conditional independence between the future and the past given the current latent state. The inference tasks of interest are filtering, smoothing and forecasting, which correspond to computing $p(z_\tau|\mathbf{y}_{1:\tau})$, $p(z_\tau|\mathbf{y}_{1:T})$ and $p(z_{T+n}|\mathbf{y}_{1:T})$ respectively where $\tau \in \{1,...,T-1\}$ and $n \in \mathbb{Z}^+$. The standard Forward-Backward algorithm to compute these distributions are detailed in standard machine learning textbooks such as [Murphy, 2012].

2.2.2 Linear Dynamical System

The linear dynamical systems are probabilistic time series models with a latent Gaussian Markov chain and observation probability distributions that are conditionally independent given the latent states at the corresponding time points. A LDS is essentially equivalent to a HMM with continuous latent states. Therefore, the LDS shares the same graphical model representation and conditional independence relationships as the HMM.

Given a sequence of observations $\mathbf{y}_1, ..., \mathbf{y}_T$, the joint probability distribution of the observed sequence and the corresponding real-valued multivariate state vectors $\mathbf{h}_1, ..., \mathbf{h}_T$ with the model parameters θ are as follow.

$$p_\theta(\mathbf{y}_{1:T}, \mathbf{h}_{1:T}) = p_\theta(\mathbf{h}_1) p_\theta(\mathbf{y}_1|\mathbf{h}_1) \prod_{t=2}^{T-1} p_\theta(\mathbf{h}_t|\mathbf{h}_{t-1}) p_\theta(\mathbf{y}_t|\mathbf{h}_t) \tag{2.2}$$

In contrast to the HMM, the initial distribution $p_\theta(\mathbf{h}_1)$ and transition distribution $p_\theta(\mathbf{h}_t|\mathbf{h}_{t-1})$ are chosen to be multivariate Gaussian distributions $\mathcal{N}(\mu_I, \Sigma_I)$ and $\mathcal{N}(\mathbf{A}\mathbf{h}_{t-1}, \Sigma_Q)$ respectively. The emission distribution $p_\theta(\mathbf{y}_t|\mathbf{h}_t)$ is typically a multivariate Gaussian distribution $\mathcal{N}(\mathbf{C}\mathbf{h}_t, \Sigma_R)$, but can be chosen to suit the sequences of modelling interest.

Inference in LDS is highly similar to inference in HMMs. The filtering, smoothing and predictive distributions, $p(\mathbf{h}_\tau|\mathbf{y}_{1:\tau})$, $p(\mathbf{h}_\tau|\mathbf{y}_{1:T})$ and $p(\mathbf{h}_{T+n}|\mathbf{y}_{1:T})$, can be computed using the same forward-backward algorithm, with the summations over the latent states replaced with integrals that can be solved analytically because of the model's linear Gaussian structure.

While the linear Gaussian structure of the model allows for tractable inference, it also imposes restrictions on the model's flexibility. Johnson et al. [2016] improved the flexibility of LDS at the expense of tractability by replacing the linear emission distribution mean $\mathbf{C}\mathbf{h}_t$ with a flexible black-box function $f_\theta(\mathbf{h}_t)$ and developed a tractable variational inference algorithm to approximate the posterior distributions.

Recent innovations in deep learning combined with advances in approximate inference techniques have opened up interesting opportunities to extend LDS for challenging modelling tasks.

2.2.3 Switching Linear Dynamical System

The switching linear dynamical system (sLDS) is an extension of LDS to allow the parameters of the transition and emission distributions at each time step to be chosen from a dictionary of parameters, each of which may fit particular segments of the time series data better than the other parameters. Therefore, sLDS have been applied to model complex time series that exhibit different properties over different segments [Barber et al., 2011].

To select a suitable set of parameters from the dictionary for each time step, a new discrete latent state s_t at each time step is introduced to the model such that the transition and emission distributions are conditional on s_t. The dynamics for the sequence of latent states $s_1, ..., s_T$ are typically modelled with a Markov process, but can be further adapted to modelling requirements [Linderman et al., 2016].

Expanding the LDS parameter notations $\theta = \{\mu_I, \Sigma_I, \mathbf{A}, \Sigma_Q, \mathbf{C}, \Sigma_R\}$ to sLDS with K parameter states (i.e., $s_t \in \{1, ..., K\}$), the model parameters for the sLDS are $\theta = \{\{\mu_I^{(1)}, ..., \mu_I^{(K)}\}, \{\Sigma_I^{(1)}, ..., \Sigma_I^{(K)}\}, \{\mathbf{A}^{(1)}, ..., \mathbf{A}^{(K)}\}, \{\Sigma_Q^{(1)}, ..., \Sigma_Q^{(K)}\}, \{\mathbf{C}^{(1)}, ..., \mathbf{C}^{(K)}\}, \{\Sigma_R^{(1)}, ..., \Sigma_R^{(K)}\}\}$. Given s_t, the transition and emission distributions are therefore $\mathcal{N}(\mathbf{A}^{(s_t)}\mathbf{h}_{t-1}, \Sigma_Q^{(s_t)})$ and $\mathcal{N}(\mathbf{C}^{(s_t)}\mathbf{h}_t, \Sigma_R^{(s_t)})$ respectively. The joint probability distribution of sLDS with a Markov process distributed parameter state-space is as follow

$$\begin{aligned} p_\theta(\mathbf{y}_{1:T}, \mathbf{h}_{1:T}, s_{1:T}) = & p_\theta(\mathbf{h}_1|s_1) p_\theta(\mathbf{y}_1|\mathbf{h}_1, s_1) p_\theta(s_1) \\ & \prod_{t=2}^{T-1} p_\theta(\mathbf{y}_t|\mathbf{h}_t, s_t) p_\theta(\mathbf{h}_t|\mathbf{h}_{t-1}, s_t) p_\theta(s_t|s_{t-1}). \end{aligned} \tag{2.3}$$

The expressiveness of sLDS comes at the cost of intractable exact inference,

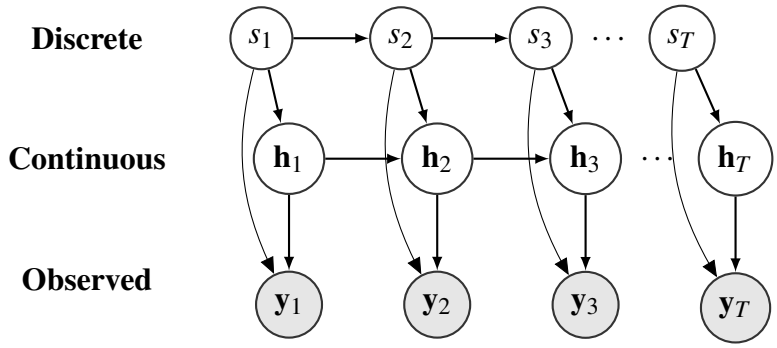

Figure 2.3: This figure shows a directed probabilistic graphical model of the switching linear dynamical system.

as the exact filtering distributions, which are mixtures of Gaussians, have numbers of components that increase exponentially with respect to their time indices. For example, the filtering distribution $p(\mathbf{h}_t|\mathbf{y}_{1:t}, s_t)$ for a sLDS with K components is a mixture of K^{t-1} Gaussians. The difficulty to keep track of the Gaussian mixtures demands inference algorithms that can sufficiently approximate the multi-modal posterior distributions without incurring exponential memory cost. Some pointers to tractable inference algorithm for sLDS can be found in [Barber et al., 2011, Murphy, 2012].

2.3 Probabilistic Network Models

Network data are data sets that express the relationships between multiple entities that are known as vertices or nodes. A network $\mathcal{G}$ consists of a set of N vertices $\mathcal{V}$, typically indexed by positive integers up to N (i.e., $V = \{1,...,N\}$), and a set of $2-$tuples $\mathcal{E}$ that specifies the relationships between the vertices in $\mathcal{V}$ (e.g., $\mathcal{E} = \{(1,3),(2,3),(1,2)\}$). In directed networks, the order of the vertices in tuple encodes the origin/destination relationship between two vertices, such that (i, j) implies a directed edge from vertex i to vertex j. In undirected networks, the ordering is simply ignored. A network can also be equivalently expressed using a $N \times N$ binary-valued adjacency matrix $\mathbf{Y}$, with $\mathbf{Y}_{ij} = 1$ if $(i, j) \in \mathcal{E}$ and 0 otherwise. The binary-valued matrix is also known as a graph, and in the case when its elements are random variables, it is called a random graph. Many probabilistic network models essentially define probability distributions on random graphs.

In this section, we briefly review some probabilistic network models and concepts that are relevant to the edge-exchangeable dynamic network models described in Chapter 5. We first describe the mixed-membership stochastic blockmodel and its dynamic network variant. Using the intuition developed, we then briefly describe exchangeable random graphs, in which mixed-membership stochastic blockmodel and many other probabilistic network models are special cases, and discuss their inherent limitations in modelling real-world network data. The discussions serve as a foundation for the dynamic network model that we propose in chapter 5. Finally, we review the concept of edge-exchangeability in network modelling. The model proposed in chapter 5 falls under the edge-exchangeable framework.

We refer readers who are interested in an in-depth survey of probabilistic network models to Goldenberg et al. [2010]. The excellent survey paper covers an extensive list of probabilistic network models and properties of networks that are commonly observed in the real-world.

2.3.1 Mixed-membership Stochastic Blockmodels

A common approach to model network data is to assume that the vertices in the data set can be categorised into several clusters, and the probability of interaction between two vertices is dependent on their unobserved cluster assignments. As the cluster assignments of the members are unobserved, they are assumed to be latent variables that can be inferred from the network data. This data generative process is known as stochastic blockmodel (SBM), first proposed by Nowicki and Snijders [2001].

Mixed-membership stochastic blockmodel (MMSB) [Airoldi et al., 2008] generalises the SBM to allow vertices to take on different cluster assignments depending on the vertices they are interacting with. Therefore, the cluster assignments of the vertices are unique to the vertices they are interacting with, and are sampled from from vertex-specific multinomial distributions. For example, in a directed network with 3 clusters, vertex A may belong to cluster 1 in a directed interaction from vertex A to B, and belong to cluster 2 in an interaction from A to C while belonging to cluster 3 in an interaction from B to A. Given the cluster assignments of two interacting vertices, the probability of forming a connection between the two vertices is the corresponding probability between the two clusters.

To sample a directed network $\mathcal{G}$ with N vertices and adjacency matrix $\mathbf{Y}$ from a MMSB with K clusters, one can draw samples using the generative process in Algorithm 1. The algorithm was first presented in Airoldi et al. [2008]. Modification of the procedure to sample undirected network requires only trivial changes to enforce symmetry to the cluster indicators.

The MMSB requires 3 hyper-parameters to be specified. These hyper-parameters are the number of vertices $N \in \mathbb{Z}_+$, the concentration hyper-parameter $\alpha \in \mathbb{R}_+^K$ for the Dirichlet prior distribution of the mixed-membership vector and the $\mathbf{B} \in [0,1]^{K \times K}$ matrix that describes the probabilities of connections between different mixed-membership clusters. The generative process implies a joint probability distribution as specified in Equation (2.4), which consists of $N^2 + N$ latent variables. In Equation 2.4, $Y_{pq} \in \{0,1\}$ is an entry in the adjancency matrix that describes

Algorithm 1 Data generating process for the MMSB.

Input: $N \in \mathbb{Z}_+$, $\alpha \in \mathbb{R}_+^K$, $\mathbf{B} \in [0,1]^{K \times K}$
Output: Adjacency matrix $\mathbf{Y} \in 0,1^{N \times N}$

for $p \leftarrow 1$ `to` N **do**
  `Sample a mixed-membership vector:` $\pi_p \sim Dirichlet(\alpha)$
for $p \leftarrow 1$ `to` N **do**
  for $q \leftarrow 1$ `to` N **do**
    `Sample a 1-hot column vector:` $\mathbf{z}_{p \to q} \sim Categorical(\pi_p)$
    `Sample a 1-hot column vector:` $\mathbf{z}_{p \leftarrow q} \sim Categorical(\pi_q)$
    `Sample an edge:` $\mathbf{Y}_{pq} \sim Bernoulli(\mathbf{z}_{p \to q}^\top \mathbf{B} \mathbf{z}_{p \leftarrow q})$

the network data, $\mathbf{z}_{p \to q}$ and $\mathbf{z}_{p \leftarrow q}$ are indicator random variables that indicate the community memberships of vertex p and q respectively as they interact with each other. These community assignment random variables are assumed to be sampled from vertex-specific categorical distributions parameterised by π_p.

$$p(Y, Z_{\to}, Z_{\leftarrow}, \pi_{1:N}) = \Big[\prod_{p,q} p(Y_{pq} | \mathbf{z}_{p \to q}, \mathbf{z}_{p \leftarrow q}, \mathbf{B}) p(\mathbf{z}_{p \to q} | \pi_p) p(\mathbf{z}_{p \leftarrow q} | \pi_q)\Big] \Big[\prod_p p(\pi_p | \alpha)\Big] \tag{2.4}$$

Exact posterior inference in MMSB is intractable because of the large number of latent variables that need to be marginalised. As a result, many approximate inference and sampling algorithms, such as the algorithms proposed in Airoldi et al. [2008], Gopalan et al. [2012] and Chang [2011], have been developed.

Despite its popularity in the statistical network modelling community, MMSBs are not able to capture the sparsity of edges commonly observed in real network data. We discuss the limitation in more details in Section 2.3.3.

2.3.2 Dynamic Mixed-membership Stochastic Blockmodels

Networks that are observed in real-life often evolve over time, with new edges and vertices entering the networks at different time points while some existing edges disappear. Modelling these time-evolving networks can often reveal interesting time-varying patterns of the networks, and even allows us to make predictions. Many different probabilistic dynamic network models have been proposed by different authors [Ho et al., 2011, Xu and Hero, 2014, Heaukulani and Ghahramani, 2013,

Algorithm 2 Data generating process for the dM^{3}SB .

Input: $C, N, T \in \mathbb{Z}_+$, $\mathbf{v} \in \mathbb{R}^{K\times 1}$, $\Phi \in \mathbb{R}^{K\times K}$, $\Sigma_{1:C} \in \mathbb{R}^{K\times K}$, $\delta \in \Delta^K$, $\mathbf{B} \in [0,1]^{K\times K}$
Output: A sequence of adjacency matrices $[\mathbf{Y}^{(1)}, ..., \mathbf{Y}^{(T)}]$, where $\mathbf{Y}^{(t)} \in \{0,1\}^{N\times N}$

for $h \leftarrow 1$ `to` C **do** ▷ Sample a sequence of latent states
  `Sample` $\mu_h^{(1)} \sim \mathcal{N}(\mathbf{v}, \Phi)$
  for $t \leftarrow 2$ `to` T **do**
    `Sample` $\mu_h^{(t)} \sim \mathcal{N}(\mu_h^{(t-1)}, \Phi)$
for $t \leftarrow 1$ `to` T **do**
  for $p \leftarrow 1$ `to` N **do**
    `Sample` $c_p^{(t)} \sim Categorical(\delta)$ ▷ Sample mixture component indicator
    `Sample` $\gamma_p^{(t)} \sim \mathcal{N}(\mu_{c_p^{(t)}}^{(t)}, \Sigma_{c_p^{(t)}})$ ▷ Sample mixed-membership vector
    `Transform` $\pi_p^{(t)} \leftarrow Softmax(\gamma_p^{(t)})$
  for $p \leftarrow 1$ `to` N **do** ▷ Sample a network as in Algorithm 1
    for $q \leftarrow 1$ `to` N **do**
      `Sample` $\mathbf{z}_{p\to q}^{(t)} \sim Categorical(\pi_p^{(t)})$
      `Sample` $\mathbf{z}_{p\leftarrow q}^{(t)} \sim Categorical(\pi_q^{(t)})$
      `Sample` $\mathbf{Y}_{pq}^{(t)} \sim Bernoulli(\mathbf{z}_{p\to q}^{(t)\top} \mathbf{B} \mathbf{z}_{p\leftarrow q}^{(t)})$

Sarkar and Moore, 2006, Sewell et al., 2017, Durante and Dunson, 2014].

The review in this section focuses on the dynamic mixture of mixed-membership stochastic blockmodel (dM^{3}SB) proposed by Ho et al. [2011]. dM^{3}SB extends the MMSB in two directions. Firstly, dM^{3}SB replaces the Dirichlet prior of the mixed-membership vector with a mixture of logistic normal prior, such that the mixture model can capture both the covariance between different clusters and the multi-modal data densities. Secondly, dM^{3}SB models the time dependency of the networks by chaining the mixture of logistic normal priors at adjacent time points with a Gaussian random walk state-space.

The generative process to sample a time series of T networks $\mathcal{G}_1, ..., \mathcal{G}_T$ with adjacency matrices $\mathbf{Y}^{(1)}, ..., \mathbf{Y}^{(T)}$ among a set of N vertices $\mathcal{V}$ from a dM^{3}SB with K mixed-membership clusters and C state-space mixture components is as described in Algorithm 2.

As with MMSB, inference and parameter learning in dM^{3}SB are computation-

ally intractable. A structured mean-field variational inference and a variational EM algorithms were developed to allow tractable posterior inference and learning by Ho et al. [2011].

2.3.3 Exchangeable Random Graphs and Limitations

A network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with an adjacency matrix $\mathbf{Y}$ is invariant to the relabelling of the vertices $\mathcal{V}$. Roughly speaking, this means that renaming the vertices in $\mathcal{G}$ does not alter the information contained in the data. The invariant property, defined in Definition 2.3.1, also implies that simultaneously permuting the rows and columns of $\mathbf{Y}$ has no effect on the network represented by $\mathbf{Y}$. A random graph that is jointly exchangeable is called an exchangeable random graph.

Definition 2.3.1. *A random graph (Y_{ij}) is called **jointly exchangeable** if $(Y_{ij}) \stackrel{d}{=} (Y_{\pi(i)\pi(j)})$ for every permutation π of $\mathbb{Z}$.*

The jointly exchangeable property of $\mathbf{Y}$ is widely exploited to define probabilistic network models, including the MMSB [Lloyd et al., 2012]. It is easy to see that the relabelling of vertices has no effect on the MMSB joint probability distribution in Equation (2.4). Probabilistic models for jointly exchangeable random graph are characterised and unified by the following Aldous-Hoover representation theorem [Aldous, 1981, Hoover, 1979].

Theorem 2.3.1. *A random graph (Y_{ij}) is jointly exchangeable if and only if it can be represented as follows: There is a random function $F : [0,1]^3 \to \mathbf{Y}$ such that $(Y_{ij}) \stackrel{d}{=} (F(U_i, U_j, U_{i,j}))$, where $(U_i)_{i\in\mathbb{Z}}$ and $(U_{i,j})_{i,j\in\mathbb{Z}}$ are, respectively, a sequence and an array of i.i.d. $Uniform[0,1]$ random variables, which are independent of F.*

While the theorem above assumes that $(U_i)_{i\in\mathbb{Z}}$ and $(U_{\{i,j\}})_{i,j\in\mathbb{Z}}$ are i.i.d. $Uniform\big[0,1\big]$ distributed, the uniformly distributed assumption is not a necessary condition given the following conditions hold as detailed in Orbanz and Roy [2015].

1. Both $(U_i)_{i\in\mathbb{Z}}$ and $(U_{\{i,j\}})_{i,j\in\mathbb{Z}}$ are i.i.d.

2. The random variables are independent of the random function F.

3. The distributions of $(U_i)_{i\in\mathbb{Z}}$ and $(U_{\{i,j\}})_{i,j\in\mathbb{Z}}$ are non-atomic.

In the case where the random graph is symmetric (undirected network), the Aldous-Hoover theorem can be expressed as a symmetric graphon function $W : [0,1]^2 \to [0,1]$ [Lovász, 2012], such that $F(U_i, U_j, U_{i,j}) \overset{d}{=} \mathbf{1}(U_{i,j} < W(U_i, U_j)$. A random graph $\mathcal{G}_N$ with N vertices sampled from an exchangeable random graph model has an expected proportion of edges $p = \frac{1}{2}\int_{[0,1]^2} W(x,y)dxdy$. The expected number of edges in the sampled undirected graph is $\frac{N(N-1)}{2}p = \Theta(N^2)$ if $p > 0$. Therefore, networks sampled from exchangeable random graph models are guaranteed to be either trivially empty ($p = 0$) or dense ($p > 0, |\mathcal{E}| = \Theta(N^2)$). However, networks observed in the real-world are generally sparse (i.e., $|\mathcal{E}| = o(N^2)$) [Goldenberg et al., 2010]. Therefore, probabilistic network models and their dynamic variants that are built upon the exchangeable random graph assumption cannot fit sparse real network data well.

2.3.4 Edge Exchangeable Network Models

Definition 2.3.2. *Consider the random network sequence $(\mathcal{G}_n)_n$, where $\mathcal{G}_n$ has a set of edges $\mathcal{E}_n$ that are indexed by integers and $\mathcal{V}_n$ are the active vertices of $\mathcal{E}_n$ (i.e., $\mathcal{V}_n$ is the union of all vertices in $\mathcal{E}_n$). $(\mathcal{G}_n)_n$ is (infinitely) **edge exchangeable** if for every $n \in \mathbb{Z}$ and every permutation π of the edge indices in $\mathcal{E}_n$, $\mathcal{G}_n \overset{d}{=} \tilde{\mathcal{G}}_n$, where $\tilde{\mathcal{G}}_n$ is the resulted network from the permutation.*

The limitation of the exchangeable random graph approach in modelling sparse network data has called for a different framework to model sparse networks. To address the issue, Crane and Dempsey [2016], Cai et al. [2016] separately proposed to model sparse networks as edge exchangeable random networks, and showed that the edge exchangeable models do indeed produce sparse networks. The notion of

edge exchangeability is fundamentally different from the previously described joint exchangeability of random graphs and the Kallenberg exchangeability explored in Caron and Fox [2014]. We refer to the previously described joint exchangeability as vertex exchangeability to avoid confusion.

The vertex exchangeable random graphs, as defined in Definition 2.3.1, implicitly assumes that vertices are the statistical units of network data [Crane and Dempsey, 2016]. The assumption implies that as vertices are sampled or observed, the connectivity information between the set of sampled vertices (i.e., the edges and the lack of edges) become fully available. This is consistent with the approach of modelling networks as binary-valued adjacency matrices. However, many networks observed in the real world are only partial observations of the interactions. It is difficult to differentiate between non-interactions and unobserved interactions.

In contrast, the edge exchangeable models assume that edges are the statistical units of networks, and that the observed edges in networks are finite samples of an interaction process. With the edge exchangeable assumption, the models are invariant to the ordering of the edges. Therefore, the edges are drawn iid conditioning on an underlying distribution. Edge exchangeable networks are formally defined in Cai et al. [2016] in Definition 2.3.1. An in-depth discussion of the edge exchangeable models is presented in Janson [2017].

2.4 Gaussian Processes

In this section, I briefly review the powerful Gaussian process model. Gaussian process is a powerful tool in the probabilistic modelling tool kits. In chapter 4, we propose an extension of Gaussian process to model data points observed on the vertices of graphs.

Gaussian processes are a flexible class of stochastic processes that leverage the convenient properties of multivariate Gaussian distribution to specify prior probability distributions over functions $f : \mathcal{X} \to \mathbb{R}$ indexed by $\mathbf{x} \in \mathcal{X}$. The marginalisation

property of the multivariate Gaussian distribution implies that the marginal distribution for any finite subset of random variables from a collection of jointly multivariate Gaussian distributed random variables is also a multivariate Gaussian distribution, with the mean vector and the covariance matrix specified by the corresponding subsets of the original joint distribution parameters. In Gaussian process, the distribution over an infinite collection of Gaussian distributed random variables $f(\cdot)$ is fully specified by its mean function $\mu : \mathcal{X} \to \mathbb{R}$ and the positive semi-definite covariance kernel function $k_\theta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ parameterized by a small set of hyper-parameters θ. Therefore, a GP distributed random function can be denoted as

$$f(\cdot) \sim \mathcal{GP}(\mu(\cdot), k_\theta(\cdot,\cdot)). \tag{2.5}$$

In the simplest case of Gaussian process interpolation, a prior over the interpolating function f can be specified using a GP. Once a set of observations drawn from the underlying function $\mathcal{D}_I = \{(\mathbf{x}_1, f(\mathbf{x}_1)), \ldots, (\mathbf{x}_N, f(\mathbf{x}_N))\}$ is observed, the predictive distribution at arbitrary point $\mathbf{x}$ can be analytically computed as follow

$$p(f(\mathbf{x})|\mathcal{D}_I\} = \frac{p(f(\mathbf{x}), \mathcal{D}_I)}{p(\mathcal{D}_I)}. \tag{2.6}$$

The distributions $p(f(\mathbf{x}), \mathcal{D}_I)$ and $p(\mathcal{D}_I)$ are multivariate Gaussian distributions with the mean vectors and the covariance matrices specified by evaluating $\mu(\cdot)$ and $k_\theta(\cdot,\cdot)$ at the corresponding index sets.

The applications of GPs to model real-world data sets usually require the specification of per data point likelihood functions that are coupled via a GP prior for the likelihood parameters. One typical modelling scenario that requires the specification of likelihood functions is Gaussian process regression. Given continuous valued data points with additive noise $\mathcal{D}_R = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, a likelihood model $p(y_n|f(\mathbf{x}_n))$, where $f(\mathbf{x}_n)$ is the 'location' parameter of the distribution, can be specified for the data points to account for additive noise. Combining the likelihood model with a GP prior on $f(\mathbf{x})$, one can compute the posterior process of $f(\mathbf{x})$, $p(f(\mathbf{x})|\mathcal{D}_R)$,

and draw inference on the underlying function that generated the data set using the Bayes rule

$$p(f(\mathbf{x})|\mathcal{D}_R\} = \frac{p(f(\mathbf{x}))\prod_{n=1}^{N} p(y_n|f(\mathbf{x}_n))}{p(\mathcal{D}_R)}. \tag{2.7}$$

The most commonly used likelihood function for the GP regression model is the Gaussian distribution, which conveniently allows Equation (2.7) to be computed analytically. Other than the Gaussian distribution, likelihood functions that are suitable for regression tasks include the Laplace distributions and the Student-t distributions. However, the use of non-Gaussian likelihood functions results in intractable model that requires approximate inference algorithms to compute the posterior process.

Beyond regression, Gaussian processes are also powerful priors for building classification models and latent variable models [Rasmussen and Williams, 2006, Lawrence, 2004, Byron et al., 2009]. However, performing inference in these more complicated GP models is analytically intractable and requires suitable approximate inference algorithms. Another major drawback of the GP models is the computational complexity. Performing inference in GP models incurs a memory cost that scales $O(N^2)$ and a computational cost that scales $O(N^3)$, where N is the number of data points in the training data set $\mathcal{D}$, rendering the models to be computationally intractable in modelling big data sets. Fortunately, research breakthroughs in variational inference algorithms in recent years have provided feasible solutions that address both the analytically and computationally intractable aspects of Gaussian processes. I review the variational inference algorithms for GPs in Section 2.5.3.

The choice of the covariance function $k_\theta(\mathbf{x},\mathbf{x}')$, also known as the kernel function, is particularly important for Gaussian processes. The covariance function determines the prior covariances between pairs of random variables indexed by different values of $\mathbf{x}$, and therefore the prior probability density assigned to functions of different smoothness and trends. For example, the commonly used squared exponential (SE) covariance function $k_\theta(\mathbf{x},\mathbf{x}') = \sigma^2 e^{-\frac{(\mathbf{x}-\mathbf{x}')^\mathsf{T}(\mathbf{x}-\mathbf{x}')}{2l^2}}$ assumes that prior covariance between random variables decays exponentially with respect to the squared Euclidean

distance between their indices, with the rate of decay determined by the lengthscale parameter l. Therefore, SE covariance function with larger lengthscale parameter results in a GP that places higher density on smoother functions compared to another SE covariance GP with a smaller lengthscale parameter. In addition to smoothness, the covariance function can also be designed to incorporate other prior knowledge such as seasonal and linear trends. GPs with carefully designed kernels to capture specific known trends in the data are able to extrapolate sensibly, resulting in greater performances when compared to the other generic models. However, designing suitable kernel functions requires extensive expert knowledge, and the development of algorithms for automatic discovery of kernels remains an active research area [Duvenaud, 2014, Wilson, 2014].

Inputs to the kernel functions are not restricted to vector valued data only. Kernels for structured objects such as sequences, graphs and probability distributions have also been proposed, extending the scopes of kernel machines, including the Gaussian processes, to beyond vector valued data. In Chapter 4, I discuss a GP model with kernel function which operates on vertices in graphs that can also be viewed as a GP that operates on probability distributions.

As the kernel functions are critical components of Gaussian processes, I refer the readers to comprehensive reviews of different kernel functions in Duvenaud [2014] and Rasmussen and Williams [2006]. Discussions and reviews on the technical details of the Gaussian processes are available in Rasmussen and Williams [2006].

2.5 Variational Inference

Posterior inference in probabilistic models are typically intractable because computing the normalising constants of the posterior distributions involve integrating or summing over a large number of random variables. Variational inference provides a scalable solution to approximate the posterior distribution p with a tractable family of approximating distributions q such that the Kullback-Leibler (KL) divergence

between the intractable true posterior distribution and the approximating distribution $KL(q||p)$ is (indirectly) minimised.

Consider a data set $X = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ with N data points that are assumed to be generated from a model with joint probability distribution $p_\theta(X,Z) = \prod_{n=1}^{N} p_\theta(\mathbf{x}_n|\mathbf{z}_n)p_\theta(\mathbf{z}_n)$, where $Z = \{\mathbf{z}_1, ..., \mathbf{z}_N\}$ are K dimensional latent variables drawn from the prior distribution $p_\theta(Z) = \prod_{n=1}^{N} p_\theta(\mathbf{z}_n)$ and θ denotes the set of model parameters. The goal of variational inference is to approximate the intractable posterior distribution $p(Z|X) = \frac{p_\theta(X,Z)}{p_\theta(X)}$ with a family of distribution $q_\lambda(Z) = \prod_{n=1}^{N} q_{\lambda_n}(\mathbf{z}_n)$ parameterised by the set of variational parameters $\lambda = \{\lambda_1, ..., \lambda_N\}$. The conventional choice of the family of approximating variational distributions $q_{\lambda_n}(\mathbf{z}_n)$ is the fully-factorised exponential family distributions $\prod_{k=1}^{K} q_{\lambda_{nk}}(z_{nk})$, called the mean-field distributions. The mean-field distributions are primarily chosen for the easiness to compute the expectations of the sufficient statistics, which allow the variational inference objective function for conditionally conjugate models to be evaluated in a tractable way. However, recent advances in integrating sampling algorithms into variational inference have allowed richer non-factorial variational distributions that are easy to sample from to be used instead [Kingma and Welling, 2013, Ranganath et al., 2014, Kucukelbir et al., 2016].

Variational inference seeks to optimise the variational parameters λ such that $KL(q_\lambda(Z)||p(Z|X))$ is minimized. However, the KL divergence itself is again intractable because it requires the evaluation of the intractable model evidence $p(X)$. Therefore, the *evidence lower bound (ELBO)*, which is equivalent to the KL divergence up to a normalising constant is optimized instead. The ELBO is derived by lower bounding the model's log-evidence using the Jensen's inequality

$$\ln \int_Z p_\theta(X,Z) \geq \int_Z q_\lambda(Z) \ln \frac{p_\theta(X,Z)}{q_\lambda(Z)}. \tag{2.8}$$

The r.h.s. of Equation (2.8) is the variational objective function ELBO to be maximised with respect to λ. If the actual posterior distribution falls inside the

family of variational distributions, the value of ELBO at its global maxima is equal to the log-evidence and the bound is tight.

The ELBO, as stated in Equation (2.9), and its gradients with respect to λ can be evaluated analytically when the model is conditionally conjugate and the variational distributions are from the mean-field family. The gradients can then be used in gradient ascent or coordinate ascent algorithms to optimise the variational parameters to convergence. However, in many interesting probabilistic models, such as the Bayesian logistic regression model and the variational autoencoder (VAE), the models are not conditionally conjugate and require further approximations of the expected log-complete likelihood term $E_{q_\lambda}[\ln p_\theta(X,Z)]$. The intractable expectation can be approximated by linearising the log-complete likelihood such that its expectation can be analytically evaluated [Jaakkola and Jordan, 1997, Johnson et al., 2016, Wang and Blei, 2013]. More recently, black-box variational inference methods that attempt to approximate ELBO and its gradients using Monte Carlo samples from the variational distributions have also been successfully applied in Titsias and Lázaro-Gredilla [2014], Ranganath et al. [2014], Kucukelbir et al. [2016], Kingma and Welling [2013].

$$ELBO(\lambda) = E_{q_\lambda}[\ln p_\theta(X,Z)] - E_{q_\lambda}[\ln q_\lambda(Z)] \tag{2.9}$$

An intuitive way to interpret the ELBO is to rearrange Equation (2.9) as a sum of the expected model log-likelihood and $-KL(q_\lambda(Z)||p_\theta(Z))$, as shown in Equation (2.10). The ELBO objective function can be interpreted as a trade-off between modelling the data well by maximising $E_{q_\lambda}[\ln p_\theta(X|Z)]$ and minimizing the KL distance to the prior:

$$ELBO(\lambda) = -KL(q_\lambda(Z)||p_\theta(Z)) + E_{q_\lambda}[\ln p_\theta(X|Z)]. \tag{2.10}$$

While the review above derived the ELBO based on data points that are i.i.d.

samples drawn from the model, Equation (2.9) is also valid for dependent data. However, the fully factorised assumption of the mean-field distributions is often too restrictive for dependent data and may result in highly biased approximations. Therefore, further assumptions, such as time dependency, are often taken into account to partially factorise the $q_\lambda(X)$. The resulted variational distributions are the structured mean-field distributions. We review the structured mean-field variational inference for probabilistic models with latent Markov processes in Section 2.5.2.

2.5.1 Variational Expectation-Maximisation Algorithm

Probabilistic models are often parameterised with model parameters or hyper-parameters that can be learned from the data. In a Bayesian inference setting, prior distributions are placed on the parameters and their posterior distributions can be inferred using the suite of tools for Bayesian inference, including variational inference. Otherwise, the parameters are often learned by maximising the log-evidence of the probabilistic models $\int_Z p_\theta(X,Z)$ with the expectation-maximisation (EM) algorithm under the maximum likelihood principle [Dempster et al., 1977]. However, the EM algorithm requires access to the exact posterior distributions of the latent variables and computing the exact posterior distributions is intractable for many models of interest as discussed in Section 2.5.

The variational EM algorithm attempts to approximate the maximum likelihood model parameters $\hat{\theta}_{ML}$ by maximising the lower bound of the log-evidence as derived in Equation (2.9). Expanding the arguments of the ELBO function to include model parameters θ, the optimal variational EM parameter setting $\hat{\theta}_{VEM}$ is

$$\hat{\theta}_{VEM} = \underset{\theta}{\operatorname{argmax}}\, ELBO(\lambda,\theta) = \underset{\theta}{\operatorname{argmax}}\, E_{q_\lambda}[\ln p_\theta(X,Z)]. \tag{2.11}$$

The variational EM solution $\hat{\theta}_{VEM}$ is not guaranteed to converge to $\hat{\theta}_{ML}$ as the set of local maxima of ELBO may not include $\hat{\theta}_{ML}$ if the family of variational distributions does not include the true posterior distributions. As with the EM algorithm,

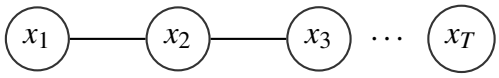

Figure 2.4: This figure shows an undirected graphical model that encodes the conditional independence structure of the smoothing posterior distribution resulted from the variational Kalman filter algorithm.

the variational EM solution may also converge to a sub-optimal local maxima as the ELBO function is non-convex and the converged solutions are dependent on the starting points as well as the optimisation algorithms. Therefore, the application of the variational EM algorithm often requires multiple random restarts and careful selection of the optimizer's parameter settings to find a good solution.

2.5.2 Variational Inference for Latent Markov Models

Hidden Markov models and linear dynamical systems are probabilistic time series models with tractable posterior inference. However, many of their more complex extensions do not share the same computational convenience. Some examples of these intractable latent Markov models include the factorial hidden Markov models [Ghahramani and Jordan, 1997], dynamic topic models [Blei and Lafferty, 2006], dynamic word embedding [Bamler and Mandt, 2017] and dynamic MMSB [Ho et al., 2011].

In factorial hidden Markov models, the multiple latent Markov processes are coupled at each time point upon conditioning on the observations, resulting in posterior computations that scale exponentially with the number of latent Markov chains. A naive application of the completely factorised mean-field variational inference algorithm to the model results in highly biased approximation because of the strong sequential dependency. Therefore, a structured mean-field approach that preserves the time dependency within each Markov chain but ignores the dependency across chains was proposed. The variational distribution for each of the Markov chain preserves the conditional independence assumption of Markov chains where the future is independent of the past when conditioned on the present.

In models like the dynamic topic models and the dynamic MMSB where posterior inference is intractable because of the complex likelihood functions, variants of the variational Kalman filter algorithm are typically applied [Blei and Lafferty, 2006]. The variational Kalman filter algorithms result in smoothing Gaussian variational distributions with Markovian conditional independence structure. As such, the smoothing Gaussian variational distributions can be equivalently parameterised by directly specifying a block-tridiagonal precision matrix [Archer et al., 2015]. Another alternative but equivalent parameterisation of the variational distribution is by the following normalised product of bi-variate distributions for adjacent pairs of latent variables x_t, x_{t+1} in the state-space

$$q(x_1, ..., x_T) = \frac{\prod_{t=1}^{T-1} q(x_t, x_{t+1})}{\prod_{t=2}^{T-1} q(x_t)}. \tag{2.12}$$

The pairwise variational distributions $q(x_t, x_{t+1})$ are bi-variate Gaussians with mean and covariance variational parameters and $q(x_t)$ are the corresponding Gaussian marginals. This alternative parameterisation allows the first and second moments of the distributions, which are often required in variational EM algorithm to be easily computed from the parameters.

Combining the directly parameterised Gaussian Markov chain variational distributions with the reparameterisation trick allows complex probabilistic models with latent Markov process to be treated as convenient black-boxes when variational inference is applied.

2.5.3 Variational Inference for Gaussian Processes

Posterior inference in Gaussian processes poses two main challenges to the practitioners. Firstly, the computational complexity of computing the posterior distribution is $O(N^3)$, where N is the number of data points in the training data set $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$. The computational bottleneck arises from inverting the kernel matrix of the GP prior and is a ubiquitous problem that also applies to GP

models that are analytically tractable. Secondly, posterior inference for many GP models with non-conjugate likelihoods is analytically intractable. Therefore, suitable approximate inference methods are required in order to work with the intractable GP models.

Algorithms such as the Laplace approximation [Williams and Barber, 1998], the Fully Independent Training Conditional (FITC) algorithm [Snelson and Ghahramani, 2006], the expectation-propagation (EP) algorithm [Kim and Ghahramani, 2006, Li et al., 2015a], the elliptical slice sampler [Murray et al., 2010] and many others have been proposed by various authors over the years to address the two challenges. A sparse approximation scheme proposed in Titsias [2009] resolves the two sources of intractability elegantly under the variational inference framework. The approximation scheme gives practitioners the ability to trade off the fidelity of the approximations and the computational complexity, allowing GPs to scale up to larger data sets. The complexity of the variational algorithm is $O(NM^2)$, where M is a hyper-parameter that can be chosen based on computational requirements. Additionally, it side-steps the analytical intractability by turning the intractable normalizing constant in Equation (2.7) for models with non-conjugate likelihood functions into a series of 1-dimensional integrals that can be efficiently approximated using well-known numerical methods. In Bauer et al. [2016], the authors showed that this variational approximation method is superior to the widely adopted FITC algorithm, in that the variational algorithm attempts to approximate the true posterior processes while the FITC algorithm approximates a model different from the one intended. This variational inference algorithm for GPs is the template of the variational algorithm used in Chapter 4. I describe the sparse variational inference algorithm in the following paragraphs. The algorithm described does not assume specific likelihood functions, and is applicable to both regression and classification tasks.

Given training data set $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, where $\mathbf{y} = [y_1, \ldots, y_N]^\mathsf{T}$ are labels for classification or regression tasks and $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\mathsf{T}$ are the corresponding features in the feature space $\mathcal{X}$, a Gaussian process model for the data set can be

defined as

$$p(\mathbf{y},\mathbf{f}|\mathbf{X}) = p(\mathbf{f}|\mathbf{X})\prod_{n=1}^{N} p(y_n|f_n) \tag{2.13}$$

where $p(\mathbf{f}|\mathbf{X})$ is the joint probability distribution for $\mathbf{f} = [f(\mathbf{x}_1),\ldots,f(\mathbf{x}_N)]^\mathsf{T}$ and $f(\cdot)$ is a GP distributed random function with the kernel function $k_\theta(\cdot,\cdot)$, such that $p(\mathbf{f}|\mathbf{X})$ is a zero mean multivariate Gaussian distribution with a covariance matrix $\mathbf{K}_{XX}$,

$$p(\mathbf{f}|\mathbf{X}) = \mathcal{N}(\mathbf{0},\mathbf{K}_{XX}). \tag{2.14}$$

The posterior distribution of interest is

$$p(\mathbf{f}|\mathbf{y},\mathbf{X}) = \frac{p(\mathbf{y},\mathbf{f}|\mathbf{X})}{\int p(\mathbf{y},\mathbf{f}|\mathbf{X})d\mathbf{f}}. \tag{2.15}$$

As described in Section 2.4, Equation (2.15) is analytically intractable if $p(y_n|f_n)$ is non-Gaussian, and expensive to compute for big data sets in the Gaussian case. To approximate the posterior distribution in Equation (2.15), Titsias [2009] introduced a set of M inducing points $\mathbf{u} = [u_1,\ldots,u_M]^\mathsf{T} = [f(\mathbf{z}_1),\ldots,f(\mathbf{z}_M)]^\mathsf{T}$, where $\mathbf{Z} = [\mathbf{z}_1,\ldots,\mathbf{z}_M]^\mathsf{T}$ are inducing inputs in $\mathcal{X}$. The inducing inputs $\mathbf{Z}$ are variational parameters that can be optimized to achieve good approximation quality. In addition, a variational distribution $q(\mathbf{u},\mathbf{f}) = q(\mathbf{u})p(\mathbf{f}|\mathbf{u})$ is introduced to obtain a variational lower bound of the GP model

$$\begin{aligned}\log p(\mathbf{y}) &\geq \int q(\mathbf{u})p(\mathbf{f}|\mathbf{u})\log\frac{\prod_{n=1}^{N} p(y_n|f_n)\cancel{p(\mathbf{f}|\mathbf{u})}p(\mathbf{u}|\mathbf{Z})}{q(\mathbf{u})\cancel{p(\mathbf{f}|\mathbf{u})}}d\mathbf{u}d\mathbf{f}\\ &= \sum_{n=1}^{N}\int p(f_n|\mathbf{u})\log p(y_n|f_n)df_n - KL(q(\mathbf{u})||p(\mathbf{u}|\mathbf{Z})).\end{aligned} \tag{2.16}$$

For models with Gaussian likelihood functions, the optimal $q(\mathbf{u})$ can be obtained using the calculus of variation, resulting in an optimal multivariate Gaussian variational distribution. For models with intractable likelihood functions, a multivariate Gaussian variational distribution $q(\mathbf{u}) = \mathcal{N}(\mathbf{m},\Sigma)$ is typically assumed, and the variational parameters $\mathbf{m},\Sigma$ and $\mathbf{Z}$ can be estimated by maximizing Equation (2.16) using a gradient ascent algorithm. The 1-dimensional integrals $\int p(f_n|\mathbf{u})\log p(y_n|f_n)df_n$

can be efficiently approximated using numerical methods if no analytical solution is available for the specified likelihood functions. The computational complexity of optimizing the bound in Equation (2.16) is $O(NM^2)$. The predictive distribution $p(y_*|\mathbf{Y})$ at a test point $\mathbf{x}_*$ is as follow.

$$p(y_*|\mathbf{Y}) \approx \int p(y_*|f_*) \int p(f_*|\mathbf{u})q(\mathbf{u})d\mathbf{u}df_* \tag{2.17}$$

The distribution is analytically intractable for non-Gaussian likelihood functions, but can be approximated using Monte Carlo method.

As the number of data points N becomes large, the bound in Equation (2.16) can be optimized stochastically using only mini-batches of the data points, such that the computational cost of optimizing the bound can be further reduced. The stochastic optimization approach is described in Hensman et al. [2015a].

2.5.4 Challenges and Innovations in Variational Inference

Despite the wide-spread use of variational inference in the machine learning community, there remains significant shortcomings and challenges that require methodological innovations to overcome. We briefly highlight the key challenges and point to some recent literatures that attempt to address them.

Scalability

Computing the ELBO function and its gradients often become computationally challenging as the data sets increase in size. In i.i.d. data setting, the computational complexity of ELBO scales linearly with the number of data points. In time series models, the computational complexity is often linear with respect to the number of time points. In certain network models, such as the MMSB, the computational complexity is quadratic with respect to the number of vertices in the networks. Various stochastic variational inference algorithms have been proposed to reduce the computational cost of variational inference by optimising a noisy but unbiased

surrogate of the ELBO function through data sub-sampling [Hoffman et al., 2013, Foti et al., 2014, Gopalan et al., 2012]. By randomly selecting a subset of the data at each iteration of the variational inference algorithm and optimise the noisy approximation of the ELBO with stochastic gradient ascent algorithm, the algorithm will converge to the optima of the ELBO function while reducing the computational complexity to sub-linear [Robbins and Monro, 1951].

The number of variational parameters λ that need to be stored also becomes large as the size of the data set increases. To reduce the memory requirements, amortised variational inference algorithms that reparameterise the variational parameters as outputs of expressive neural networks have been successfully applied to both large i.i.d data sets and long time series data [Kingma and Welling, 2013, Ng et al., 2016].

Tractability

The parameterisations of complex probabilistic models often include probability distributions that are not log-linear. The class of models includes those with logistic, softmax and neural network functions in the likelihoods [Jaakkola and Jordan, 1997, Blei and Lafferty, 2006, Kingma and Welling, 2013, Johnson et al., 2016] and the latent states [Linderman et al., 2016]. The ELBO functions of this class of models are intractable because of the non-linear expectations $E_{q_\lambda}[\ln p_\theta(X,Z)]$ that can no longer be analytically evaluated. Applying variational inference to these models requires either deterministic linear approximations, as is the case in Jaakkola and Jordan [1997], Blei and Lafferty [2006], Kingma and Welling [2013], Johnson et al. [2016], or Monte Carlo approximations of the expectations with variance reduction techniques [Kingma and Welling, 2013, Ranganath et al., 2014, Titsias and Lázaro-Gredilla, 2014].

Approximation Accuracy

Despite the attractive computational property of variational inference algorithms, the resulted approximations are known to be biased because of mismatch between the

simple variational distributions and complex true posterior distributions. The choice of $KL(q||p)$ as the divergence measure also exacerbate the problem as the approximations resulted by minimising $KL(q||p)$ are known to severely under-estimate the posterior uncertainties [Minka et al., 2005]. The bias in the approximated posteriors also result in biased model parameter estimates when combined with the variational EM algorithm for parameter learning. Several methodological innovations that seek to enrich the expressiveness of the variational distributions in order to reduce approximation bias have been successfully applied in iid data settings [Ranganath et al., 2016, Kingma et al., 2016, Rezende and Mohamed, 2015, Tran et al., 2015b,a, Gershman et al., 2012]. Additionally, different choices of divergence measures had also been proposed to address the under-estimations of posterior uncertainties in variational inference [Li and Turner, 2016, Dieng et al., 2016]. However, much remains to be explored and studied to address this significant disadvantage of variational inference.

3

Scalable Variational Inference for Factorial Hidden Markov Models

Factorial Hidden Markov Models (FHMMs) are powerful models for sequential data but they do not scale well with long sequences. We propose a scalable inference and learning algorithm for FHMMs that draws on ideas from the stochastic variational inference, neural network and copula literatures. Unlike the existing approaches, the proposed algorithm requires no message passing procedure among latent variables and can be distributed to a network of computers to speed up learning. Our experiments corroborate that the proposed algorithm does not introduce further approximation bias compared to the proven structured mean-field algorithm, and achieves better performance with long sequences and large FHMMs.

Breakthroughs in modern technology have allowed more sequential data to be collected in higher resolutions. The resulted sequential data sets are often extremely long and high-dimensional, exhibiting rich structures and long-range dependency that can only be captured by fitting large models to the sequences, such as Hidden Markov Models (HMMs) with a large state space. The standard methods of learning and performing inference in the HMM class of models are the Expectation-Maximization (EM) and the Forward-Backward algorithms. The Forward-Backward and EM algorithms are prohibitively expensive for long sequences and large models because of their linear and quadratic computational complexity with respect to sequence length and state space size respectively.

The work presented in this chapter was completed in collaboration with Pawel Chilinski.

To rein in the computational cost of inference in HMMs, several variational inference algorithms that trade-off inference accuracy in exchange for lower computational cost have been proposed in the literatures. Variational inference is a deterministic approximate inference technique that approximates posterior distribution p by minimizing the Kullback-Leibler divergence $KL(q||p)$, where q lies in a family of distributions selected to approximate p as closely as possible while keeping the inference algorithm computationally tractable [Wainwright and Jordan, 2008]. Despite its biased approximation of the actual posteriors, the variational inference approach has been proven to work well in practice [Teh et al., 2007].

Variational inference has also been successfully scaled to tackle problems with large data sets through the use of stochastic gradient descent (SGD) algorithms [Hoffman et al., 2013]. However, applications of such techniques to models where the data is dependent (i.e., non-i.i.d.) require much care in the choice of the approximating family and parameter update schedules to preserve dependency structure in the data [Gopalan and Blei, 2013]. More recently, developments of stochastic variational inference algorithms to scale models for non-i.i.d. data to large data sets have been increasingly explored [Foti et al., 2014, Gopalan and Blei, 2013].

We propose a stochastic variational inference algorithm to scale up inference for a flexible class of hidden Markov models called the factorial hidden Markov models (FHMM). Despite its flexibility, the existing inference algorithms for FHMM that rely on passing messages between the latent Markovian random variables do not scale well to long sequences. The proposed inference algorithm approximates the posterior distributions with chains of bivariate Gaussian copulas. Unlike the existing variational inference algorithms, the proposed approach eliminates the need for explicit message passing between latent variables and allows computations to be distributed to multiple computers. To scale the variational distribution to long sequences, we reparameterise the bivariate Gaussian copula chain parameters with feed-forward recognition neural networks that are shared by copula chain parameters across different time points. The use of recognition networks in variational inference

has been well-explored in models in which data is assumed to be i.i.d. [Kingma and Welling, 2013, Hinton et al., 1995]. To the best of our knowledge, the use of recognition networks to decouple inference in non-factorised stochastic process of unbounded length has not been well-explored. In addition, both the FHMM parameters and the parameters of the recognition networks are learnt in conjunction by maximising the stochastic lower bound of the log-marginal likelihood, computed based on randomly sampled subchains from the full sequence of interest. The combination of recognition networks and stochastic optimisations allow us to scale the Gaussian copula chain variational inference approach to very long sequences.

3.1 Factorial Hidden Markov Models

Building on the background survey of the hidden Markov models in Section 2.2.1, we briefly introduce the factorial hidden Markov models, the existing variational inference algorithm, and its limtations.

Factorial hidden Markov models (FHMMs) are a class of HMMs consisting of M latent variables $\mathbf{s}_t = (s_t^1, \cdots, s_t^M)$ at each time point, and observations y_t where the conditional emission probability of the observations $p(y_t|\mathbf{s}_t, \eta)$ is parameterised through factorial combinations of $\mathbf{s}_t$ and emission parameters η. Each of the latent variables s_t^m evolves independently in time through discrete-valued Markov chains governed by transition matrix A_m [Ghahramani and Jordan, 1997]. For a sequence of observations $\mathbf{y} = (y_1, \cdots, y_T)$ and corresponding latent variables $\mathbf{s} = (\mathbf{s}_1, \cdots, \mathbf{s}_T)$, the joint distribution can be written as follow.

$$p(\mathbf{y}, \mathbf{s}) = \prod_{m=1}^{M} p(s_1^m) p(y_1|\mathbf{s}_1, \eta) \prod_{t=2}^{T} p(y_t|\mathbf{s}_t, \eta) \prod_{m=1}^{M} p(s_t^m|s_{t-1}^m, A_m) \tag{3.1}$$

Depending on the state of latent variables at a particular time point, different subsets of emission parameters η can be selected, resulting in a dynamic mixture of distributions for the data. The factorial representation of state space reduces the

required number of parameters to encode transition dynamics compared to regular HMMs with the same number of states. As an example, a state space with 2^M states can be encoded by M binary transition matrices with a total of $2M$ parameters while a regular HMM requires a transition matrix with $2^M \times (2^M - 1)$ parameters to be estimated.

We specify a FHMM with $D-$dimensional Gaussian emission distributions and M binary hidden Markov chains. The emission distributions share a covariance matrix Σ across different states while the mean is parameterised as a linear combination of the latent variables

$$\mu_t = \mathbf{W}^\top \hat{\mathbf{s}}_t, \tag{3.2}$$

where $\hat{\mathbf{s}}_t = [s_t^1, \cdots, s_t^M, 1]^\top$ is a $M+1$-dimensional binary vector and $\mathbf{W} \in \mathbb{R}^{(M+1)\times D}$. The FHMM model parameters $\Gamma = (\Sigma, \mathbf{W}, A_1, \cdots, A_M)$ can be estimated with the EM algorithm. Note that to facilitate optimisations, we reparameterised Σ as $\mathbf{L}\mathbf{L}^\top$ where $\mathbf{L} \in \mathbb{R}^{D\times D}$ is a lower-triangular matrix.

The computational complexity for exact inference in FHMM with M K-state hidden Markov chains is $O(TMK^{M+1})$. The exponential cost with respect to the number of chains is induced by the coupling of the Markov chains in the posterior, and renders exact inference computationally intractable as the number of chains increases. A structured mean-field (SMF) variational inference approach proposed in [Ghahramani and Jordan, 1997] approximates the posterior distribution with M independent Markov chains and reduces the complexity to $O(TMK^2)$ in models with linear-Gaussian emission distributions. While the reduction in complexity is significant, inference and learning with SMF remain insurmountable in the presence of extremely long sequences. In addition, SMF requires the storage of $O(2TMK)$ variational parameters in-memory per training sequence. Such computational requirements remain expensive to satisfy even in the age of cloud computing.

3.2 Review: Gaussian Copulas, Stochastic Variational & Amortised Inference

In this section, we review the key concepts that form the basis of the novel scalable variational inference algorithm will be introduced in Section 3.3. These important concepts are Gaussian copula, stochastic variational inference and amortised inference. In the next section, we describe a novel variational distribution with Markovian structure that is constructed by chaining bivariate Gaussian copulas. The structure of the variational distribution allows stochastic variational inference with amortising to be readily applicable to allow scalable inference.

3.2.1 Gaussian Copulas

Gaussian copulas are a family of multivariate cumulative distribution functions (CDFs) that capture linear dependency structure between random variables with potentially different marginal distributions. Given two random variables X_1, X_2 with their respective marginal CDFs F_1, F_2, their Gaussian copula joint CDF can be written as

$$\Phi_\rho(\phi^{-1}(F_1(x_1)), \phi^{-1}(F_2(x_2))) \tag{3.3}$$

where ϕ^{-1} is the quantile function of the standard Gaussian distribution, and Φ is the CDF of the standard bivariate Gaussian distribution with correlation ρ. In a bivariate setting, the dependency between X_1 and X_2 is captured by ρ. The bivariate Gaussian copula can be easily extended to multivariate settings through a correlation matrix. For an in-depth introduction of copulas, please refer to [Nelsen, 2013, Elidan, 2013].

3.2.2 Stochastic Variational Inference

Variational inference is a class of deterministic approximate inference algorithms that approximate intractable posterior distributions $p(\mathbf{s}|\mathbf{y})$ of latent variables $\mathbf{s}$ given data $\mathbf{y}$ with a tractable family of variational distributions $q_\beta(\mathbf{s})$ parameterised by variational parameters β. The variational parameters are fitted to approximate the posterior

distributions by maximising the evidence lower bound of log-marginal likelihood (ELBO) [Wainwright and Jordan, 2008]. By applying the Jensen's inequality to $\int p(\mathbf{y},\mathbf{s})d\mathbf{s}$ the ELBO can be expressed as

$$ELBO = \mathbb{E}_q[\log p(\mathbf{y},\mathbf{s})] - \mathbb{E}_q[\log q(\mathbf{s})]. \tag{3.4}$$

The ELBO can also be interpreted as the negative KL-divergence $KL(q_\beta(\mathbf{s})||p(\mathbf{s}|\mathbf{y}))$ up to a constant. Therefore, variational inference results in variational distribution that is the closest to p within the approximating family as measured by KL.

Maximising ELBO in the presence of large data set is computationally expensive as it requires the ELBO to be computed over all data points. Stochastic variational inference (SVI) [Hoffman et al., 2013] successfully scales the inference technique to large data sets using subsampling based stochastic gradient descent algorithms [Duchi et al., 2011].

3.2.3 Amortised Inference and Recognition Neural Networks

The many successes of neural networks in tackling certain supervised learning tasks have generated much research interest in applying neural networks to unsupervised learning and probabilistic modelling problems [Stuhlmüller et al., 2013, Gershman and Goodman, 2014, Rezende et al., 2014, Kingma and Welling, 2013]. A recognition neural network was initially proposed in [Hinton et al., 1995] to extract underlying structures of data modelled by a generative neural network. Taking the observed data as input, the feed-forward recognition network learns to predict a vector of unobserved code that the generative neural network initially conjectured to generate the observed data.

More recently, a recognition network was applied to variational inference for latent variable models [Kingma and Welling, 2013, Gershman and Goodman, 2014]. Given data, the latent variable model and an assumed family of variational distributions, the recognition network learns to predict optimal variational parameters

for the specific data points. As the recognition network parameters are shared by all data points, information learned by the network on a subset of data points are shared with other data points. This inference process is aptly named amortised inference. In short, recognition network can simply be thought of as a feed-forward neural network that learns to predict optimal variational parameters given the observed data, with ELBO as its utility function.

3.3 Message-free Stochastic Variational Inference

While the structured mean-field (SMF) variational inference and its associated EM algorithms are effective tools for inference and learning in FHMMs with short sequences, they become prohibitively expensive as the sequences grow longer. For example, one iteration of SMF forward-backward message passing for FHMM of 5 Markov chains and 10^6 sequential data points takes hours of computing time on a modern 8-cores workstation, rendering SMF unusable for large scale problems. To scale FHMMs to long sequences, we resort to stochastic variational inference.

The proposed variational inference algorithm approximates posterior distributions of the M hidden Markov chains in FHMM with M independent chains of bivariate Gaussian-Bernoulli copulas. The bivariate Gaussian-Bernoulli copula extends the Gaussian copula to incorporate Bernoulli marginal distributions while keeping a Gaussian correlation structure, and is mathematically defined in Equation 3.6. The computational cost of optimising the variational parameters is managed by a subsampling-based stochastic gradient ascent algorithm similar to SVI. In addition, parameters of the copula chains are reparameterised using feed-forward recognition neural networks to improve efficiency of the variational inference algorithm.

In contrast to the EM approach for learning FHMM model parameters, our approach allows for both the model parameters and variational parameters to be learnt in conjunction by maximising the ELBO with a stochastic gradient ascent algorithm. In the following sections, we describe the variational distributions and

recognition networks, and derive the stochastic ELBO for SGD.

3.3.1 Variational Chains of Bivariate Gaussian Copulas

Similar to the SMF variational inference algorithm proposed by [Ghahramani and Jordan, 1997], we aim to preserve posterior dependency of latent variables within the same hidden Markov chains by introducing chains of bivariate Gaussian copulas. The chain of bivariate Gaussian copulas variational distribution can be written as the product of bivariate Gaussian copulas divided by the marginals of latent variables at the intersection of the pairs

$$q(\mathbf{s}^m) = \frac{\prod_{t=2}^{T} q_t(s_{t-1}^m, s_t^m)}{\prod_{t=2}^{T-1} q_t(s_t^m)} \tag{3.5}$$

where $q_t(s_{t-1}^m, s_t^m)$ is the joint probability mass function of a bivariate Gaussian copula as defined in Equation 3.6. We suppress the t subscript in q_t from here on for succinctness.

The copula parameterization in Equation (3.5) offers several advantages. Firstly, the overlapping bivariate copula structure enforces coherence of $q(s_t^m)$ such that $\sum_{s_{t-1}^m} q(s_{t-1}^m, s_t^m) = \sum_{s_{t+1}^m} q(s_t^m, s_{t+1}^m)$. Secondly, the chain structure of the distribution restricts the growth in the number of variational parameters to only two parameters per chain for every increment in the sequence length. The two additional parameters encode the correlation of the random variables in the final two time points, and the marginal variational distribution of the last random variable. Finally, the Gaussian copula allows marginals and dependence structure of the random variables to be modelled separately [Elidan, 2013]. The decoupling of the marginal and correlation parameters allows these parameters to be predicted separately using feed-forward recognition neural networks, and the neural network parameters can be optimised jointly to form a good variational approximation.

We propose to approximate the posteriors of adjacent FHMM latent variables

with the following bivariate Gaussian-Bernoulli copula probability mass function

$$\begin{aligned} q(s_{t-1}^m = 0, s_t^m = 0) &= q_{00_t}^m \\ q(s_{t-1}^m = 1, s_t^m = 0) &= 1 - \theta_{t,m} - q_{00_t}^m \\ q(s_{t-1}^m = 0, s_t^m = 1) &= 1 - \theta_{t-1,m} - q_{00_t}^m \\ q(s_{t-1}^m = 1, s_t^m = 1) &= \theta_{t,m} + \theta_{t-1,m} + q_{00_t}^m - 1 \end{aligned} \tag{3.6}$$

where $q_{00_t}^m = \Phi_{\rho_{t,m}}(\phi^{-1}(1-\theta_{t-1,m}), \phi^{-1}(1-\theta_{t,m}))$ and $q(s_t^m = 1) = \theta_{t,m}$ is a Bernoulli distribution with parameter $\theta_{t,m} \in [0,1]$. The Gaussian-Bernoulli copula can be easily extended to multinomial random variables.

Assuming independence between random variables in different hidden chains, the posterior distribution of $\mathbf{s}$ can be factorised by chains and approximated by

$$q(\mathbf{s}) = \prod_{m=1}^{M} q(\mathbf{s}^m). \tag{3.7}$$

3.3.2 Feed-forward Recognition Neural Networks

The number of variational parameters in the chains of bivariate Gaussian copulas scales linearly with respect to the length of the sequence as well as the number of sequences in the data set. While it is possible to directly optimise these variational parameters, the approach quickly becomes infeasible as the size of data set grows. We propose to circumvent the challenging scalability problem by reparameterising the variational parameters with rolling feed-forward recognition neural networks that are shared among variational parameters within the same chain. The marginal variational parameters $\theta_{t,m}$ and copula correlation variational parameters $\rho_{t,m}$ are parameterised with different recognition networks as they are parameters of a different nature.

Given observed sequence $\mathbf{y} = (y_1, \ldots, y_T)$, the marginal and correlation recognition networks for hidden chain m compute the variational parameters $\theta_{t,m}$ and $\rho_{t,m}$ by performing a forward pass on a window of observed data $\Delta\mathbf{y}_t =$

$(y_{t-\frac{1}{2}\Delta t}, \ldots, y_t, \ldots, y_{t+\frac{1}{2}\Delta t})$

$$\theta_{t,m} = f_\theta^m(\Delta \mathbf{y}_t) \qquad \rho_{t,m} = f_\rho^m(\Delta \mathbf{y}_t) \tag{3.8}$$

where $\Delta t + 1$ is the user selected size of rolling window, f_θ^m and f_ρ^m are the marginal and correlation recognition networks for hidden chain m with parameters $\omega_m = (\omega_{\theta,m}, \omega_{\rho,m})$. The output layer non-linearities of f_θ^m and f_ρ^m are chosen to be the sigmoid and hyperbolic tangent functions respectively to match the range of $\theta_{t,m}$ and $\rho_{t,m}$.

The recognition network hyperparameters, such as the number of hidden units, non-linearity, and the window size Δt can be chosen based on computing budget and empirical evidence. In our experiments with shorter sequences where ELBO can be computed within a reasonable amount of time, we did not observe any significant difference in the convergence of the ELBO among different choices of non-linearity. However, we observed that the converged value of the ELBO is sensitive to the number of hidden units and the number of hidden units needs to be adapted to the data set and computing budget. Recognition networks with larger hidden layers have larger capacity to approximate the posterior distributions as closely as possible but require more computing budget to learn. Similarly, the choice of Δt determines the amount of information that can be captured by the variational distributions as well as the computing budget required to learn the recognition network parameters. As a rule of thumb, we recommend the number of hidden units and Δt to be chosen as large as the computing budget allows in long sequences. We emphasise that the range of posterior dependency captured by the correlation recognition networks is not limited by Δt, as the recognition network parameters are shared across time, allowing dependency information to be encoded in the network parameters. For FHMMs with large number of hidden chains, various schemes to share the networks' hidden layers can be devised to scale the method to FHMMs with a large state space. This presents another trade-off between computational requirements and goodness of posterior approximations.

In addition to scalability, the use of recognition networks also allows our approach to perform fast inference at run-time, as computing the posterior distributions only require forward passes of the recognition networks with data windows of interest. The computational complexity of the recognition network forward pass scales linearly with respect to Δt. As with other types of neural networks, the computation is highly data-parallel and can be massively sped up with GPU. In comparison, computation for a stochastic variational inference algorithm based on a message passing approach also scales linearly with respect to Δt but is not data-parallel [Foti et al., 2014]. Subchains from long sequences, together with their associated recognition network computations, can also be distributed across a cluster of computers to improve learning and inference speed.

However, the use of recognition networks is not without its drawbacks. Compared to message passing algorithms, the recognition networks approach cannot handle missing data gracefully by integrating out the relevant random variables. The fidelity of the approximated posterior can also be limited by the capacity of the neural networks and bad local minimas. The posterior distributions of the random variables close to the beginning and the end of the sequence also require special handling, as the rolling window cannot be moved any further to the left or right of the sequences. In such scenarios, the posteriors can be computed by adapting the structured mean-field algorithm proposed in [Ghahramani and Jordan, 1997] to the subchains at the boundaries. The importance of the boundary scenarios in learning the FHMM model parameters diminishes as the data sequence becomes longer.

3.3.3 Learning Recognition Network and Model Parameters

Given sequence $\mathbf{y}$ of length T, the M-chain FHMM parameters Γ and recognition network parameters $\Omega = (\omega_1, \ldots, \omega_M)$ need to be adapted to the data by maximising the ELBO as expressed in Equation (3.4) with respect to Γ and Ω. Note that the distribution $q(\mathbf{s}^m)$ is now parameterised by the recognition network parameters ω_m. For notational simplicty, we do not explicitly express the parameterisation of

$q(\mathbf{s}^m)$ in our notations. Plugging in the FHMM joint distribution in Equation (3.1) and variational distribution in Equation (3.7), the FHMM ELBO $\mathbb{L}(\Gamma,\Omega)$ for the variational chains of bivariate Gaussian copula is approximated as

$$\begin{aligned}\mathbb{L}(\Gamma,\Omega) \approx & \sum_{t=\frac{1}{2}\Delta t+1}^{T-\frac{1}{2}\Delta t-1} \left\langle \log p(y_t|s_t^1,\ldots,s_t^M)\right\rangle_q \\ & + \sum_{m=1}^{M} \left\langle \log p(s_t^m|s_{t-1}^m)\right\rangle_q + \left\langle \log q(s_t^m)\right\rangle_q - \left\langle \log q(s_t^m,s_{t+1}^m)\right\rangle_q. \end{aligned} \tag{3.9}$$

Equation (3.9) is only an approximation of the ELBO as the variational distribution of s_t^m close to the beginning and the end of the sequence cannot be computed using the recognition networks, as the recognition networks require data points before and after t in the input to compute the variational parameters at time point t. The FHMM initial distribution $\prod_{m=1}^{M} p(s_1^m)$ cannot be learned using our approach as the initial distribution cannot be incorporated into Equation 3.9. However, they can be approximated by the stationary distribution of the transition matrices as T become large assuming that the sequence is close to stationary [Foti et al., 2014]. Comparisons to SMF in our experiment results suggest that the error caused by the approximations is negligible.

The log-transition probability expectations and variational entropy in Equation (3.9) can be easily computed as they are simply sums over pairs of Bernoulli random variables. The expectations of log-emission distributions can be efficiently computed for certain distributions, such as multinomial and multivariate Gaussian distributions.

Stochastic Gradient Descent & Subsampling Scheme

We propose to optimise Equation (3.9) with SGD by computing noisy unbiased gradients of ELBO with respect to Γ and Ω based on contributions from subchains of length $\Delta t+1$ randomly sampled from $\mathbf{y}$ [Duchi et al., 2011, Hoffman et al., 2013]. Multiple subchains can be sampled in each of the learning iterations to form a mini-batch of subchains, reducing variance of the noisy gradients. Noisy gradients with

high variance can cause the SGD algorithm to converge slowly or diverge [Duchi et al., 2011]. The subchains should also be sampled randomly without replacement until all subchains in $\mathbf{y}$ are depleted to speed up convergence. To ensure unbiasedness of the noisy gradients, the gradients computed in each iteration need to be multiplied by a batch factor

$$c = \frac{T - \Delta t}{n_{minibatch}} \tag{3.10}$$

where $n_{minibatch}$ is the number of subchains in each mini-batch. The scaled noisy gradients can then be used by SGD algorithm of choice to optimise $\mathbb{L}$. In our implementation of the algorithm, gradients are computed using the Python automatic differentiation tool [Maclaurin et al., 2015] and the optimisation is performed using Rmsprop [Tieleman and Hinton, 2012].

3.4 Related Work

Copulas have previously been adapted in variational inference literatures as a tool to model posterior dependency in models with i.i.d. data assumption [Tran et al., 2015a, Han et al., 2015]. However, the previously proposed approaches cannot be directly applied to HMM class of models without addressing parameter estimation issues as the dimensionality of the posterior distributions grow with the length of sequences. The proposed formulation of the variational distribution circumvents the problem by exploiting the chain structure of the model, coupling only random variables within the same chain that are adjacent in time with a bivariate Gaussian-Bernoulli copula, leading to a coherent chain of bivariate Gaussian copulas as the variational distribution.

On the other hand, a stochastic variational inference algorithm that also aims to scale HMM class of models to long sequences has previously been proposed in [Foti et al., 2014]. Our proposed algorithm differs from the existing approach in that it does not require explicit message passing to perform inference and learning.

Applying the algorithm proposed in [Foti et al., 2014] to FHMM requires multiple message passing iterations to determine the buffer length of each subchain in the mini batch of data, and the procedure needs to be repeated for each FHMM Markov chain. The message passing routines can be expensive as the number of Markov chains grows. In contrast, the proposed recognition network approach eliminates the need for iterative message passing and allows the variational distributions to be learned directly from the data using gradient descent. The use of recognition networks also allows fast inference at run-time with modern parallel computing hardware.

The use of recognition networks as inference devices for graphical models has received much research interest recently because of its scalability and simplicity. Similar to our approach, the algorithms proposed in [Fan et al., 2016, Johnson et al., 2016] also make use of the recognition networks for inference, but still rely on message passing to perform certain computations. In addition, [Archer et al., 2015] proposed an inference algorithm for state space models using a recognition network. However, the algorithm cannot be applied to models with non-Gaussian posteriors.

Finally, the proposed algorithm is analogous to composite likelihood algorithms for learning in HMMs in that the data dependency is broken up according to subchains to allow tractable computations [Gao and Song, 2011]. The EM-composite likelihood algorithm in [Gao and Song, 2011] partitions the likelihood function according to subchains, bounding each subchain separately with a different posterior distribution that uses only the data in that subsequence. Our recognition models generalise that.

3.5 Experiments

We evaluate the validity of our algorithm and the scalability claim with experiments using real and simulated data.

3.5.1 Algorithm Validation

To validate the algorithm, we learn FHMMs on simulated and real data using the proposed algorithm and the existing SMF-EM algorithm. The models learned using the two approaches are compared with log-likelihood (LL). In addition, we compare the learned FHMM parameters to parameters used to simulate the data. The validation experiments ensure that the proposed approach does not introduce further approximation bias compared to SMF.

Simulated Data: We simulate a $1,000$ time steps long 2-dimensional sequence from a FHMM with 2 hidden binary chains and Gaussian emission, and attempt to recover the true model parameters with the proposed approach. The proposed algorithm successfully recovers the true model parameters from the simulated data. The LL of the learned model also compared favorably to FHMM learned using the SMF-EM algorithm, showing no visible further bias compares to the proven SMF-EM algorithm. The LL of the proposed algorithm and SMF-EM are shown in Table 3.1. The learned emission parameters, together with the training data, are visualised in Figure 3.1.

Bach Chorales Data Set [Lichman, 2013]: Following the experiment in [Ghahramani and Jordan, 1997], we compare the proposed algorithm to SMF-EM based on LL. The training and testing data consist of 30 and 36 sequences from the Bach Chorales data set respectively. FHMMs with various numbers of binary hidden Markov chains are learned from the training data with both algorithms. The log-likelihoods, tabulated in Table 3.1, show that the proposed algorithm is competitive with SMF-EM on a real data set in which FHMM is proven to be a good model, and show no further bias. Note that the training log-likelihood of the FHMM with 8 chains trained using the proposed algorithm is smaller than the FHMM with 7 chains, showing that the proposed algorithm can be trapped in bad local minima.

	Proposed Algo.		SMF	
n_{chain}	LL_{train}	LL_{test}	LL_{train}	LL_{test}
Simulated Data				
2	-2.320	-2.332	-2.315	-2.338
Bach Chorales				
2	-7.241	-7.908	-7.172	-7.869
3	-6.627	-7.306	-6.754	-7.489
4	-6.365	-7.322	-6.409	-7.282
5	-6.135	-6.947	-5.989	-7.174
6	-5.973	-6.716	-5.852	-7.008
7	-5.754	-6.527	-5.771	-6.664
8	-5.836	-6.722	-5.675	-6.697

Table 3.1: Log-likelihoods from the validation experiments. The results demonstrate that the proposed algorithm is competitive with SMF.

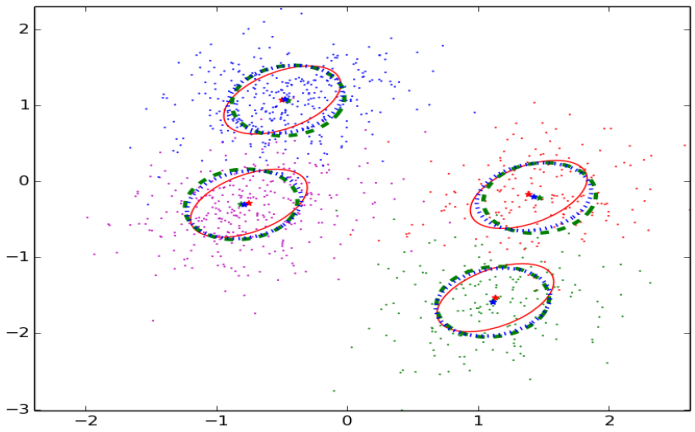

Figure 3.1: This figure shows the simulated data set in validation experiments with the emission parameters from simulation (red), learned by the proposed algorithm (green) and the SMF-EM algorithm (blue). The emission means are depicted as stars and standard deviations as elliptical contour at 1 standard deviation.

3.5.2 Scalability Verification

To verify the scalability claim, we compare the LL of FHMMs with different numbers of hidden chains learned on simulated sequences of increasing length using the proposed and SMF-based EM algorithms. Two sets of experiments are conducted to showcase scalability with respect to sequence length and the number of hidden Markov chains. To simulate real-world scenarios where computing budget is constrained, both algorithms are given the same fixed computing budget. The learned FHMMs are compared after the computing budget is depleted. Finally, we

demonstrate the scalability of the proposed algorithm by learning a 10 binary hidden Markov chains FHMM on long time series recorded in a real-world scenario.

Simulated Data: This experiment consists of two parts to verify scalability with respect to sequence length and the state space size. In the first component, we simulate 2-dimensional sequences of varying length from a FHMM with 4-binary chains using an approach similar to the validation experiment. Given fixed computing budget of 2 hours per sequence on a 24 cores Intel i7 workstation, both SMF-EM and the proposed algorithm attempt to fit 4-chain FHMMs to the sequences. Two testing sequences of length $50,000$ are also simulated from the same model. In the second component, we keep the sequence length to $15,000$ and attempt to learn FHMMs with various numbers of chains with computing budget of $1,000$s. The computing budget in the second component is scaled according to the sequence length. Log-likelihoods are computed with the last available learned parameters after computing time runs out. The proposed algorithm is competitive with SMF-EM when sequences are shorter and state space is smaller, and outperforms SMF-EM in longer sequences and larger state space. The results in Figure 3.2 and Figure 3.3 both show the increasing gaps in the log-likelihoods as sequence length and state space size increased. The recognition networks in the experiments have 1 hidden layer with 30 *tanh* hidden units, and rolling window size of 5. The marginal and correlation recognition networks for latent variables in the same FHMM Markov chain share hidden units to reduce memory and computing requirements as the number of Markov chains increases.

Household Power Consumption Data Set [Lichman, 2013]: We demonstrate the applicability of our algorithm to long sequences in which learning with SMF-EM using the full data set is simply intractable. The power consumption data set consists of a 9-dimensional sequence of $2,075,259$ time steps. After dropping the date/time series and the current intensity series that is highly correlated with the power consumption series, we keep the first 10^6 data points of the remaining 6 dimensional sequence for training and set aside the remaining series as test data. A FHMM

with 10 hidden Markov chains is learned on the training data using the proposed algorithm. In this particular problem, we force all 20 recognition networks in our algorithm to share a common tanh hidden layer of 200 units. The rolling window size is set to 21 and we allow the algorithm to complete $150,000$ SGD iterations with 10 subchains per iteration before terminating. To compare, we also learned the 10-chain FHMM with SMF-EM on the last $5,000$ data points of the training data. The models learned with the proposed algorithm and SMF-EM are compared based on the Mean Squared Error (MSE) of the smoothed test data (i.e., learned emission means weighted by latent variable posterior). As shown in Table 3.2, the test MSEs of the proposed algorithm are lower than the SMF-EM algorithm in all data dimensions. The result shows that learning with more data is indeed advantageous, and the proposed algorithm allows FHMMs to take advantage of the large data set.

Dim.	MSE_{SMF}	$MSE_{Proposed}$
1	0.155	0.082
2	0.084	0.055
3	0.079	0.027
4	0.466	0.145
5	0.121	0.062
6	0.202	0.145

Table 3.2: Test mean squared errors (MSEs) of the SMF-EM and the proposed algorithm for each dimension in the household power consumption data set. The results show that the proposed algorithm is able to take advantage of the full data set to learn a better model because of its scalability.

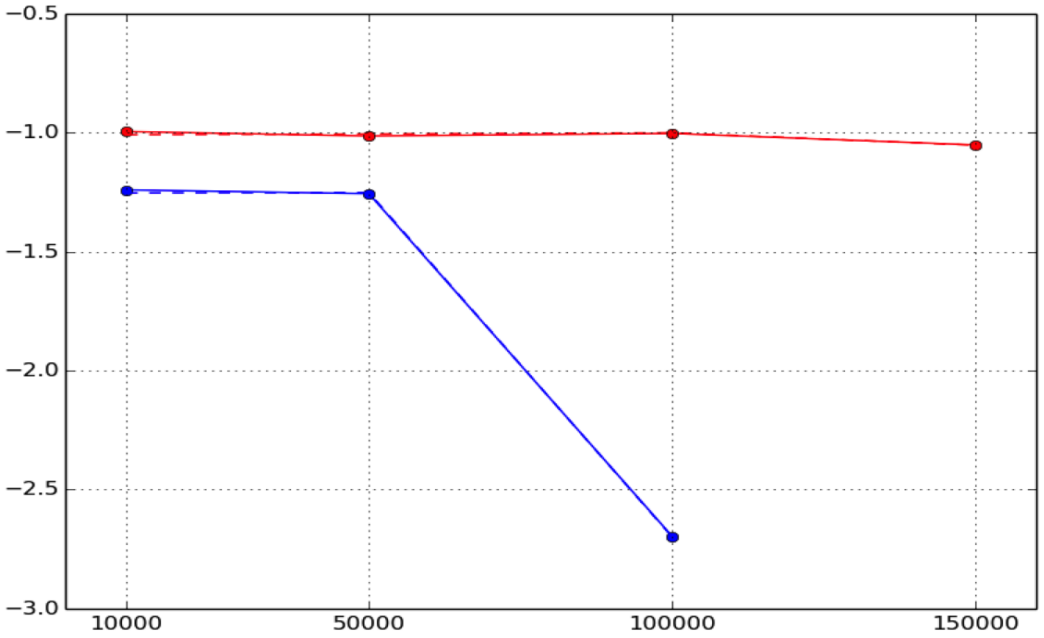

Figure 3.2: The red and blue lines show the train (solid) and test (dashed) log-likelihood (y-axis) results from the proposed and SMF-EM algorithms in the scalability experiments as the sequence length (x-axis) increases. Both algorithms are given 2hr computing budget per data set. SMF-EM failed to complete a single iteration for length of $150,000$.

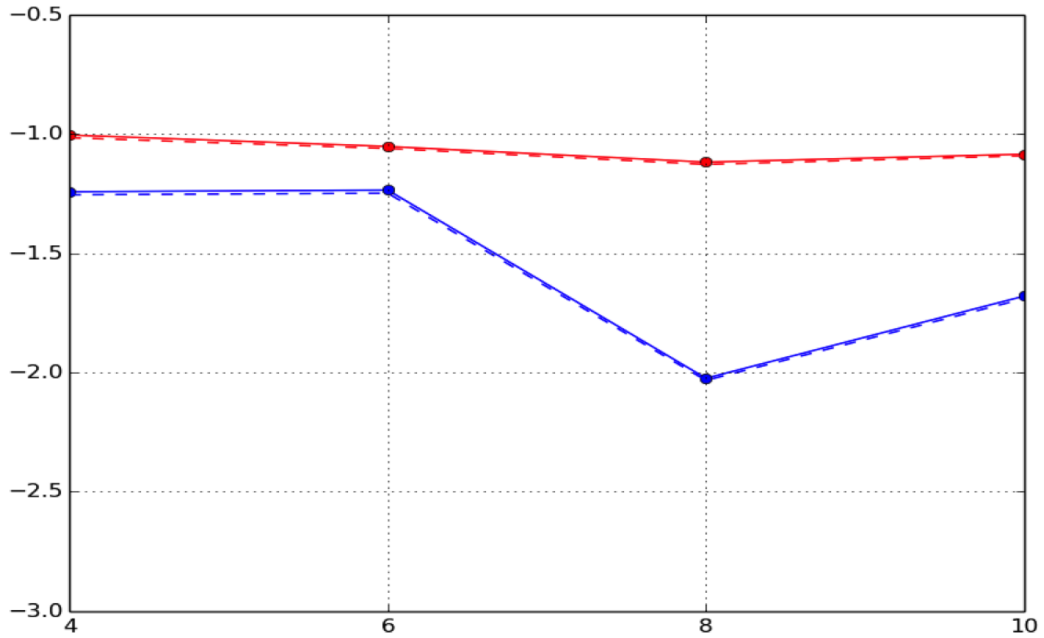

Figure 3.3: The red and blue lines show the train (solid) and test (dashed) log-likelihood (y-axis) results from the proposed and SMF-EM algorithms in the scalability experiments as the number of hidden Markov chain (x-axis) increases. Both algorithms are given $1,000$s computing budget per data set.

3.6 Discussions

In this chapter, I presented a novel stochastic variational inference and learning algorithm that does not rely on message passing to scale FHMM to long sequences and large state space. The proposed algorithm achieves competitive results when compared to structured mean-field on short sequences, and outperforms structured mean-field on longer sequences with a fixed computing budget that resembles a real-world model deployment scenario. The applicability of the algorithm to long sequences where the structured mean-field algorithm is infeasible is also demonstrated. The proposed scalable algorithm opens up new opportunities to apply FHMMs to long sequential data with rich structures that could not be previously modelled using existing algorithms.

4

Gaussian Processes for Bayesian Semi-supervised Learning on Graphs

In this chapter, I present a data-efficient Gaussian process-based Bayesian approach to the semi-supervised learning problem on graphs. The proposed model shows extremely competitive performance when compared to the state-of-the-art graph neural networks on semi-supervised learning benchmark experiments, and outperforms the neural networks in active learning experiments where labels are scarce. Furthermore, unlike the graph neural networks, the model does not require a validation data set for early stopping to control over-fitting. The proposed model can be viewed as an instance of empirical distribution regression weighted locally by network connectivity. Furthermore, I motivate the intuitive construction of the model with a Bayesian linear model interpretation where the node features are filtered by an operator related to the graph Laplacian. The model can be easily implemented by adapting an off-the-shelf scalable variational inference algorithm for Gaussian processes.

Data sets with network and graph structures that describe the relationships between the data points (nodes) are abundant in the real world. Examples of such data sets include friendship graphs on social networks, citation networks of academic papers, web graphs and many others. The relational graphs often provide rich information in addition to the node features that can be exploited to build better predictive models of the node labels, which can be costly to collect. In scenarios where there are not enough resources to collect sufficient labels, it is important to design data-efficient models that can generalize well with few training labels. The class of learning

problems where a relational graph of the data points is available is referred to as graph-based semi-supervised learning in the literature [Chapelle et al., 2009, Zhu, 2005].

Many of the successful graph-based semi-supervised learning models are based on graph Laplacian regularization or learning embeddings of the nodes. While these models have been widely adopted, their predictive performance leaves room for improvement. More recently, powerful graph neural networks that surpass Laplacian and embedding based methods in predictive performance have become popular. However, neural network models require relatively larger number of labels to prevent over-fitting and work well. We discuss the existing models for graph-based semi-supervised learning in detail in Section 4.3.

We propose a new Gaussian process model for graph-based semi-supervised learning problems that can generalize well with few labels, bridging the gap between the simpler models and the more data intensive graph neural networks. The proposed model is also competitive with graph neural networks in settings where there are sufficient labelled data. While posterior inference for the proposed model is intractable for classification problems, scalable variational inducing point approximation method for Gaussian processes can be directly applied to perform inference. Despite the potentially large number of inducing points that need to be optimized, the model is protected from over-fitting by the variational lower bound, and does not require a validation data set for early stopping. We refer to the proposed model as the graph Gaussian process (GGP).

4.1 Background

In this section, we briefly review key concepts in Gaussian processes and the relevant variational approximation technique. Additionally, we review the graph Laplacian, which is relevant to the alternative view of the model that we describe in Section 4.2.1. The reviews in this section serve as a starting point to introduce the proposed model

and introduce the notations used across the remaining of this chapter, which may differ from the notations used in Chapter 2.

4.1.1 Gaussian Processes

A Gaussian process (GP) $f(\mathbf{x})$ is an infinite collection of random variables, of which any finite subset is jointly Gaussian distributed. Consequently, a GP is completely specified by its mean function $m(\mathbf{x})$ and covariance kernel function $k_\theta(\mathbf{x},\mathbf{x}')$, where $\mathbf{x},\mathbf{x}' \in \mathcal{X}$ denote the possible inputs that index the GP and θ is a set of hyper-parameters parameterizing the kernel function. We denote the GP as follows

$$f(\mathbf{x}) \sim \mathcal{GP}\big(m(\mathbf{x}), k_\theta(\mathbf{x},\mathbf{x}')\big). \tag{4.1}$$

GPs are widely used as priors on functions in the Bayesian machine learning literatures because of their wide support, posterior consistency, tractable posterior in certain settings and many other good properties. Combined with a suitable likelihood function as specified in Equation (4.2), one can construct a regression or classification model that probabilistically accounts for uncertainties and control over-fitting through Bayesian smoothing. However, if the likelihood is non-Gaussian, such as in the case of classification, inferring the posterior process is analytically intractable and requires approximations. The GP is connected to the observed data via the likelihood function

$$y_n \mid f(\mathbf{x}_n) \sim p(y_n|f(\mathbf{x}_n)) \quad \forall n \in \{1,\ldots,N\}. \tag{4.2}$$

The positive definite kernel function $k_\theta(\mathbf{x},\mathbf{x}') : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ is a key component of GP that specifies the covariance of $f(\mathbf{x})$ *a priori*. While $k_\theta(\mathbf{x},\mathbf{x}')$ is typically directly specified, any kernel function can be expressed as the inner product of features maps $\langle\phi(\mathbf{x}),\phi(\mathbf{x}')\rangle_{\mathcal{H}}$ in the Hilbert space $\mathcal{H}$. The dependency of the feature map on θ is implicitly assumed for conciseness. The feature map $\phi(\mathbf{x}) : \mathcal{X} \to \mathcal{H}$ projects $\mathbf{x}$ into a typically high-dimensional (possibly infinite) feature space such that linear models

in the feature space can model the target variable y effectively. Therefore, GP can equivalently be formulated as

$$f(\mathbf{x}) = \phi(\mathbf{x})^{\mathsf{T}}\mathbf{w}, \tag{4.3}$$

where $\mathbf{w}$ is assigned a multivariate Gaussian prior distribution and marginalized. We assume the index set to be $\mathcal{X} = \mathbb{R}^{D\times 1}$ without loss of generality.

For a detailed review of the GP and the kernel functions, please refer to [Williams and Rasmussen, 2006].

4.1.2 Scalable Variational Inference for Gaussian Processes

Despite the flexibility of the GP prior, there are two major drawbacks that plague the model. First, if the likelihood function in Equation (4.2) is non-Gaussian, posterior inference cannot be computed analytically. Secondly, the computational complexity of the inference algorithm is $O(N^3)$ where N is the number of training data points, rendering the model inapplicable to large data sets.

Fortunately, modern variational inference provides a solution to both problems by introducing a set of M inducing points $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_M]^{\mathsf{T}}$, where $\mathbf{z}_m \in \mathbb{R}^{D\times 1}$. The inducing points, which are variational parameters, index a set of random variables $\mathbf{u} = [f(\mathbf{z_1}), \ldots, f(\mathbf{z}_M)]^{\mathsf{T}}$ that is a subset of the GP function $f(\mathbf{x})$. Through conditioning and assuming $m(\mathbf{x})$ is zero, the conditional GP can be expressed as

$$f(\mathbf{x}) \mid \mathbf{u} \sim \mathcal{GP}(\mathbf{k}_{\mathbf{zx}}^{\mathsf{T}}\mathbf{K}_{\mathbf{zz}}^{-1}\mathbf{u}, k_\theta(\mathbf{x},\mathbf{x}) - \mathbf{k}_{\mathbf{zx}}^{\mathsf{T}}\mathbf{K}_{\mathbf{zz}}^{-1}\mathbf{k}_{\mathbf{zx}}) \tag{4.4}$$

where $\mathbf{k}_{\mathbf{zx}} = [k_\theta(\mathbf{z}_1,\mathbf{x}), \ldots, k_\theta(\mathbf{z}_M,\mathbf{x})]$ and $[\mathbf{K}_{\mathbf{zz}}]_{ij} = k_\theta(\mathbf{z}_i,\mathbf{z}_j)$. Naturally, $p(\mathbf{u}) = \mathcal{N}(\mathbf{0},\mathbf{K}_{\mathbf{zz}})$. The variational posterior distribution of $\mathbf{u}$, $q(\mathbf{u})$ is assumed to be a multivariate Gaussian distribution with mean $\mathbf{m}$ and covariance matrix $\mathbf{S}$. Following the standard derivation of variational inference, the Evidence Lower Bound (ELBO)

objective function is

$$\mathcal{L}(\theta, \mathbf{Z}, \mathbf{m}, \mathbf{S}) = \sum_{n=1}^{N} \mathbb{E}_{q(f(\mathbf{x}_n))}[\log p(y_n | f(\mathbf{x}_n))] - \mathrm{KL}[q(\mathbf{u}) || p(\mathbf{u})]. \tag{4.5}$$

The variational distribution $q(f(\mathbf{x}_n))$ can be easily derived from the conditional GP in Equation (4.4) and $q(\mathbf{u})$, and its expectation can be approximated effectively using 1-dimensional quadratures. We refer the readers to [Matthews, 2016] for detailed derivations and results.

4.1.3 The Graph Laplacian

Given adjacency matrix $\mathbf{A} \in \{0,1\}^{N \times N}$ of an undirected binary graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ without self-loop, the corresponding graph Laplacian is defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{A}, \tag{4.6}$$

where $\mathbf{D}$ is the $N \times N$ diagonal node degree matrix. The graph Laplacian can be viewed as an operator on the space of functions $g : \mathcal{V} \to \mathbb{R}$ indexed by the graph's nodes such that

$$\mathbf{L}g(n) = \sum_{v \in Ne(n)} [g(n) - g(v)], \tag{4.7}$$

where $Ne(n)$ is the set containing neighbours of node n. Intuitively, applying the Laplacian operator to the function g results in a function that quantifies the variability of g around the nodes in the graph.

The Laplacian's spectrum encodes the geometric properties of the graph that are useful in crafting graph filters and kernels [Shuman et al., 2013, Vishwanathan et al., 2010, Bronstein et al., 2017, Chung, 1997]. As the Laplacian matrix is real symmetric and diagonalizable, its eigen-decomposition exists. We denote the decomposition as

$$\mathbf{L} = \mathbf{U}\Lambda\mathbf{U}^{\mathsf{T}}, \tag{4.8}$$

where the columns of $\mathbf{U} \in \mathbb{R}^{N\times N}$ are the eigenfunctions of $\mathbf{L}$ and the diagonal $\Lambda \in \mathbb{R}^{N\times N}$ contains the corresponding eigenvalues. Therefore, the Laplacian operator can also be viewed as a filter on function g re-expressed using the eigenfunction basis. Regularization can be achieved by directly manipulating the eigenvalues of the system [Smola and Kondor, 2003].

We refer the readers to Bronstein et al. [2017], Shuman et al. [2013] and Chung [1997] for comprehensive reviews of the graph Laplacian and its spectrum.

4.2 Graph Gaussian Processes

Given a data set of size N with D-dimensional features $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\mathsf{T}$, a symmetric binary adjacency matrix $\mathbf{A} \in \{0,1\}^{N\times N}$ that represents the relational graph of the data points and labels for a subset of the data points, $\mathcal{Y}_o = [y_1, \ldots, y_O]$, with each $y_i \in \{1, \ldots, K\}$, we seek to predict the unobserved labels of the remaining data points $\mathcal{Y}_U = [y_{O+1}, \ldots y_N]$. We denote the set of all labels as $\mathbf{Y} = \mathcal{Y}_O \cup \mathcal{Y}_U$.

The GGP specifies the conditional distribution $p_\theta(\mathbf{Y}|\mathbf{X},\mathbf{A})$, and predicts $\mathcal{Y}_U$ via the predictive distribution $p_\theta(\mathcal{Y}_U|\mathcal{Y}_O,\mathbf{X},\mathbf{A})$. The joint model is specified as the product of the conditionally independent likelihood $p(y_n|h_n)$ and the GGP prior $p_\theta(\mathbf{h}|\mathbf{X},\mathbf{A})$ with hyper-parameters θ. The latent likelihood parameter vector $\mathbf{h} \in \mathbb{R}^{N\times 1}$ is defined in the next paragraph.

The joint distribution of the model factorizes as

$$p_\theta(\mathbf{Y},\mathbf{h}|\mathbf{X},\mathbf{A}) = p_\theta(\mathbf{h}|\mathbf{X},\mathbf{A}) \prod_{n=1}^{N} p(y_n|h_n), \tag{4.9}$$

where for the multi-class classification problem that we are interested in, $p(y_n \mid h_n)$ is given by the robust-max likelihood [Matthews, 2016, Girolami and Rogers, 2006, Kim and Ghahramani, 2006, Hernández-Lobato et al., 2011, Hensman et al., 2015b]. We construct the GGP prior from a Gaussian process distributed latent function $f(\mathbf{x}) : \mathbb{R}^{D\times 1} \to \mathbb{R}$, $f(\mathbf{x}) \sim \mathcal{GP}\big(\mathbf{0}, k_\theta(\mathbf{x},\mathbf{x}')\big)$, where the key assumption is that the

likelihood parameter h_n for data point n is an average of the values of f over its 1-hop neighbourhood $Ne(n)$ as given by $\mathbf{A}$:

$$h_n = \frac{f(\mathbf{x}_n) + \sum_{l \in Ne(n)} f(\mathbf{x}_l)}{1 + D_n} \tag{4.10}$$

where $Ne(n) = \{l : l \in \{1,\ldots,N\}, \mathbf{A}_{nl} = 1\}$, $D_n = |Ne(n)|$. We further motivate this key assumption in Section 4.2.1.

As $f(\mathbf{x})$ has a zero mean function, the GGP prior can be succinctly expressed as a multivariate Gaussian random field

$$p_\theta(\mathbf{h}|\mathbf{X},\mathbf{A}) = \mathcal{N}(\mathbf{0}, \mathbf{P}\mathbf{K}_{\mathbf{XX}}\mathbf{P}^\mathsf{T}), \tag{4.11}$$

where $\mathbf{P} = (\mathbf{I} + \mathbf{D})^{-1}(\mathbf{I} + \mathbf{A})$ and $[\mathbf{K}_{\mathbf{XX}}]_{ij} = k_\theta(\mathbf{x}_i, \mathbf{x}_j)$. A suitable kernel function $k_\theta(\mathbf{x}_i, \mathbf{x}_j)$ for the task at hand can be chosen from the suite of well-studied existing kernels, such as those described in [Duvenaud, 2014]. We refer to the chosen kernel function as the *base kernel* of the GGP. The $\mathbf{P}$ matrix is sometimes known as the random-walk matrix in the literatures [Chung, 1997]. A graphical model representation of the proposed model is shown in Figure 4.1.

The covariance structure specified in Equation (4.11) is equivalent to the pairwise covariance

$$\begin{aligned} Cov(h_m, h_n) &= \frac{1}{(1+D_m)(1+D_n)} \sum_{i \in \{m \cup Ne(m)\}} \sum_{j \in \{n \cup Ne(n)\}} k_\theta(\mathbf{x}_i, \mathbf{x}_j) \\ &= \langle \frac{1}{1+D_m} \sum_{i \in \{m \cup Ne(m)\}} \phi(\mathbf{x}_i), \frac{1}{1+D_n} \sum_{j \in \{n \cup Ne(n)\}} \phi(\mathbf{x}_j) \rangle_{\mathcal{H}} \end{aligned} \tag{4.12}$$

where $\phi(\cdot)$ is the feature map that corresponds to the base kernel $k_\theta(\cdot,\cdot)$. Equation (4.12) can be viewed as the inner product between the empirical kernel mean embeddings that correspond to the bags of node features observed in the 1-hop neighborhood sub-graphs of node m and n, relating the proposed model to the Gaussian process distribution regression model presented in e.g. [Flaxman et al., 2015].

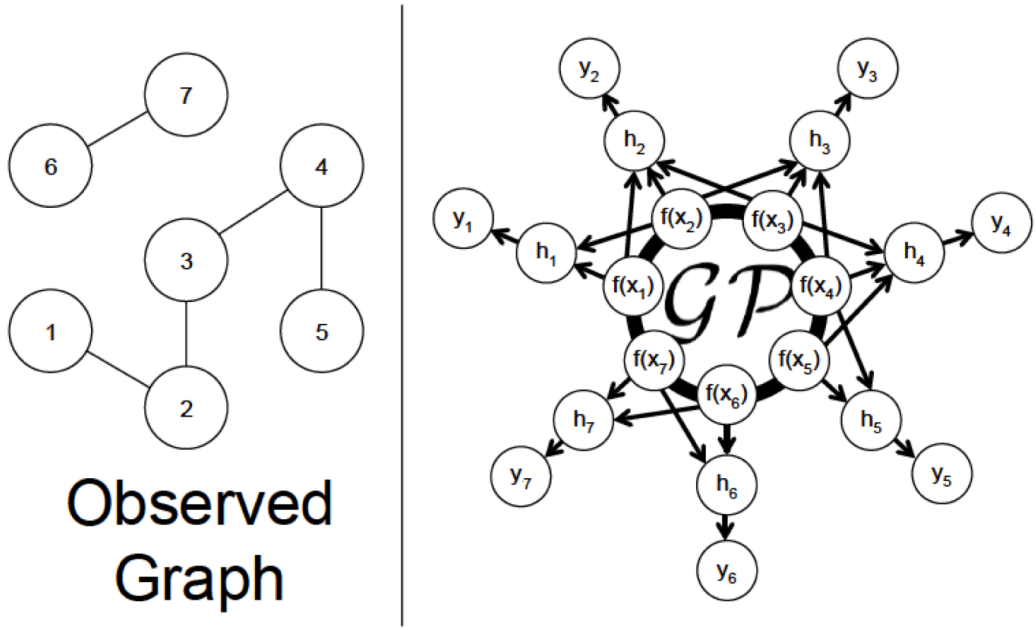

Figure 4.1: The figure depicts a relational graph (left) and the corresponding GGP represented as a graphical model (right). The thick circle represents a set of fully connected nodes.

More specifically, we can view the GGP as a distribution classification model for the labelled bags of node features $\{(\{\mathbf{x}_i | i \in \{n \cup Ne(n)\}\}, y_n)\}_{n=1}^{O}$, such that the unobserved distribution P_n that generates $\{\mathbf{x}_i | i \in \{n \cup Ne(n)\}\}$ is summarized by its empirical kernel mean embedding

$$\hat{\mu}_n = \frac{1}{1+D_n} \sum_{j \in \{n \cup Ne(n)\}} \phi(\mathbf{x}_j). \tag{4.13}$$

The prior on $\mathbf{h}$ can equivalently be expressed as $\mathbf{h} \sim \mathcal{GP}(0, \langle \hat{\mu}_m, \hat{\mu}_n \rangle_{\mathcal{H}})$. For detailed reviews of the kernel mean embedding and distribution regression models, we refer the readers to [Muandet et al., 2017] and [Szabó et al., 2016] respectively.

One main assumption of the 1-hop neighbourhood averaging mechanism is homophily - i.e., nodes with similar covariates are more likely to form connections with each others [Goldenberg et al., 2010]. The assumption allows us to approximately treat the node covariates from a 1-hop neighbourhood as samples drawn from the same data distribution, in order to model them using distribution regression.

While it is perfectly reasonable to consider multi-hops neighbourhood averaging, the homophily assumption starts to break down if we consider 2-hop neighbours which are not directly connected. Nevertheless, it is interesting to explore non-naive ways to account for multi-hop neighbours in the future, such as stacking 1-hop averaging graph GPs in a structure similar to that of the deep Gaussian processes [Damianou and Lawrence, 2013, Salimbeni and Deisenroth, 2017], or having multiple latent GPs for neighbours of different hops that are summed up in the likelihood functions.

4.2.1 An Alternative View of the Graph Gaussian Processes

In this section, we present an alternative formulation of the GGP, which results in an intuitive interpretation of the model. The alternative formulation views the GGP as a Bayesian linear model on feature maps of the nodes that have been transformed by a function related to the graph Laplacian $\mathbf{L}$.

As we reviewed in Section 4.1.1, the kernel matrix $\mathbf{K}_{\mathbf{XX}}$ in Equation (4.11) can be written as the product of feature map matrix $\Phi_{\mathbf{X}}\Phi_{\mathbf{X}}^{\mathsf{T}}$ where row n of $\Phi_{\mathbf{X}}$ corresponds to the feature maps of node n, $\phi(\mathbf{x}_n) = [\phi_{n1}, \ldots, \phi_{nQ}]^{\mathsf{T}}$. Therefore, the covariance matrix in Equation (4.11), $\mathbf{P}\Phi_{\mathbf{X}}\Phi_{\mathbf{X}}^{\mathsf{T}}\mathbf{P}^{\mathsf{T}}$, can be viewed as the product of the transformed feature maps

$$\widehat{\Phi}_{\mathbf{X}} = \mathbf{P}\Phi_{\mathbf{X}} = (\mathbf{I}+\mathbf{D})^{-1}\mathbf{D}\Phi_{\mathbf{X}} + (\mathbf{I}+\mathbf{D})^{-1}(\mathbf{I}-\mathbf{L})\Phi_{\mathbf{X}}. \tag{4.14}$$

where $\mathbf{L}$ is the graph Laplacian matrix as defined in Equation (4.6). Isolating the transformed feature maps for node n (i.e., row n of $\hat{\Phi}_{\mathbf{X}}$) gives

$$\hat{\phi}(\mathbf{x}_n) = \frac{D_n}{1+D_n}\phi(\mathbf{x}_n) + \frac{1}{1+D_n}[(\mathbf{I}-\mathbf{L})\Phi_{\mathbf{X}}]_n^{\mathsf{T}}, \tag{4.15}$$

where D_n is the degree of node n and $[\cdot]_n$ denotes row n of a matrix. The proposed GGP model is equivalent to a supervised Bayesian linear classification model with a feature pre-processing step that follows from the expression in Equation (4.15). For isolated nodes ($D_n = 0$), the expression in Equation (4.15) leaves the node feature

maps unchanged ($\hat{\phi} = \phi$).

The $(\mathbf{I} - \mathbf{L})$ term in Equation (4.15) can be viewed as a spectral filter $\mathbf{U}(\mathbf{I} - \Lambda)\mathbf{U}^{\top}$, where $\mathbf{U}$ and Λ are the eigenmatrix and eigenvalues of the Laplacian as defined in Section 4.1.3. For connected nodes, the expression results in new features that are weighted averages of the original features and features transformed by the spectral filter. The alternative formulation opens up opportunities to design other spectral filters with different regularization properties, such as those described in [Smola and Kondor, 2003], that can replace the $(\mathbf{I} - \mathbf{L})$ expression in Equation (4.15). We leave the exploration of this research direction to future work.

In addition, it is well-known that many graphs and networks observed in the real world follow the power-law node degree distributions [Goldenberg et al., 2010], implying that there are a handful of nodes with very large degrees (known as hubs) and many with relatively small numbers of connections. The nodes with few connections (small D_n) are likely to be connected to one of the handful of heavily connected nodes, and their transformed node feature maps are highly influenced by the features of the hub nodes. On the other hand, individual neighbours of the hub nodes have relatively small impact on the hub nodes because of the large number of neighbours that the hubs are connected to. This highlights the asymmetric outsize influence of hubs in the proposed GGP model, such that a mis-labelled hub node may result in a more significant drop in the model's accuracy compared to a mis-labelled node with much lower degree of connections.

4.2.2 Variational Inference with Inducing Points

Posterior inference for the GGP is analytically intractable because of the non-conjugate likelihood. We approximate the posterior of the GGP using a variational inference algorithm with inducing points similar to the inter-domain inference algorithm presented in [van der Wilk et al., 2017]. Implementing the GGP with its variational inference algorithm amounts to implementing a new kernel function that

follows Equation (4.12) in the GPflow Python package.[1]

We introduce a set of M inducing random variables $\mathbf{u} = [f(\mathbf{z}_1), \ldots, f(\mathbf{z}_M)]^\mathsf{T}$ indexed by inducing points $\{\mathbf{z}_m\}_{m=1}^M$ in the same domain as the GP function $f(\mathbf{x}) \sim \mathcal{GP}\big(\mathbf{0}, k_\theta(\mathbf{x}, \mathbf{x}')\big)$. As a result, the inter-domain covariance between h_n and $f(\mathbf{z}_m)$ is

$$Cov(h_n, f(\mathbf{z}_m)) = \frac{1}{D_n + 1}\Big[k_\theta(\mathbf{x}_n, \mathbf{z}_m) + \sum_{l \in Ne(n)} k_\theta(\mathbf{x}_l, \mathbf{z}_m)\Big]. \tag{4.16}$$

Additionally, we introduce a multivariate Gaussian variational distribution $q(\mathbf{u}) = \mathcal{N}(\mathbf{m}, \mathbf{S}\mathbf{S}^\mathsf{T})$ for the inducing random variables with variational parameters $\mathbf{m} \in \mathbb{R}^{M \times 1}$ and the lower triangular $\mathbf{S} \in \mathbb{R}^{M \times M}$. Through Gaussian conditioning, $q(\mathbf{u})$ results in the variational Gaussian distribution $q(\mathbf{h})$ that is of our interest. The variational parameters $\mathbf{m}, \mathbf{S}, \{\mathbf{z}_m\}_{m=1}^M$ and the kernel hyper-parameters θ are then jointly fitted by maximizing the ELBO function in Equation (4.5).

4.2.3 Computational Complexity

The computational complexity of the inference algorithm is $O(|\mathcal{Y}_o|M^2)$. In the experiments, we chose M to be the number of labelled nodes in the graph $|\mathcal{Y}_o|$, which is small relative to the total number of nodes. Computing the covariance function in Equation (4.12) incurs a computational cost of $O(D_{max}^2)$ per labelled node, where D_{max} is the maximum node degree. In practice, the computational cost of computing the covariance function is small because of the sparse property of graphs typically observed in the real-world [Goldenberg et al., 2010].

4.3 Related Work

Graph-based learning problems have been studied extensively by researchers from both machine learning and signal processing communities, leading to many models and algorithms that are well-summarized in review papers [Bronstein et al., 2017,

[1] https://github.com/markvdw/GPflow-inter-domain

Sandryhaila and Moura, 2014, Shuman et al., 2013].

Gaussian process-based models that operate on graphs have previously been developed in the closely related relational learning discipline, resulting in the mixed graph Gaussian process (XGP) [Silva et al., 2008] and relational Gaussian process (RGP) [Chu et al., 2007]. Additionally, the renowned Label Propagation (LP)[Zhu et al., 2003a] model can also be viewed as a GP with its covariance structure specified by the graph Laplacian matrix [Zhu et al., 2003b]. The GGP differs from the previously proposed GP models in that the local neighbourhood structures of the graph and the node features are directly used in the specification of the covariance function, resulting in a simple model that is highly effective.

Models based on Laplacian regularization that restrict the node labels to vary smoothly over graphs have also been proposed previously. The LP model can be viewed as an instance under this framework. Other Laplacian regularization based models include the deep semi-supervised embedding [Weston et al., 2012] and the manifold regularization [Belkin et al., 2006] models. As shown in the experimental results in Table 4.2, the predictive performance of these models fall short of other more sophisticated models.

Additionally, models that extract embeddings of nodes and local sub-graphs which can be used for predictions have also been proposed by multiple authors. These models include DeepWalk [Perozzi et al., 2014], node2vec [Grover and Leskovec, 2016], planetoid [Yang et al., 2016] and many others. The proposed GGP is related to the embedding based models in that it can be viewed as a GP classifer that takes empirical kernel mean embeddings extracted from the 1-hop neighbourhood sub-graphs as inputs to predict node labels.

Finally, many geometric deep learning models that operate on graphs have been proposed and shown to be successful in graph-based semi-supervised learning problems. The earlier models including [Li et al., 2015b, Scarselli et al., 2009, Gori et al., 2005] are inspired by the recurrent neural networks. On the other hand,

convolution neural networks that learn convolutional filters in the graph Laplacian spectral domain have been demonstrated to perform well. These models include the spectral CNN [Bruna et al., 2013], DCNN [Atwood and Towsley, 2016], ChebNet [Defferrard et al., 2016] and GCN [Kipf and Welling, 2016]. Neural networks that operate on the graph spectral domain are limited by the graph-specific Fourier basis. The more recently proposed MoNet [Monti et al., 2017] addressed the graph-specific limitation of spectral graph neural networks. The idea of filtering in graph spectral domain is a powerful one that has also been explored in the kernel literatures [Smola and Kondor, 2003, Vishwanathan et al., 2010]. We draw parallels between our proposed model and the spectral filtering approaches in Section 4.2.1, where we view the GGP as a standard GP classifier operating on feature maps that have been transformed through a filter that can be related to the graph spectral domain.

Our work has also been inspired by literatures in Gaussian processes that mix GPs via an additive function, such as [Byron et al., 2009, Duvenaud et al., 2011, van der Wilk et al., 2017].

4.4 Experiments

We present two sets of experiments to benchmark the predictive performance of the GGP against existing models under two different settings. In Section 4.4.1, we demonstrate that the GGP is a viable and extremely competitive alternative to the graph convolutional neural network (GCN) in settings where there are sufficient labelled data points. In Section 4.4.2, we test the models in an active learning experimental setup, and show that the GGP outperforms the baseline models when there are few training labels.

4.4.1 Semi-supervised Classification on Graphs

The semi-supervised classification experiments in this section exactly replicate the experimental setup in Kipf and Welling [2016], where the GCN is known to perform

	Type	N_{nodes}	N_{edges}	$N_{label\_cat.}$	$D_{features}$	Label Rate
Cora	Citation	2,708	5,429	7	1,433	0.052
Citeseer	Citation	3,327	4,732	6	3,703	0.036
Pubmed	Citation	19,717	44,338	3	500	0.003

Table 4.1: A summary of the benchmark data sets for the semi-supervised classification experiment.

	Cora	Citeseer	Pubmed
GGP	80.9%	69.7%	77.1%
GGP-X	84.7%	75.6%	82.4%
GCN[Kipf and Welling, 2016]	81.5%	70.3%	79.0%
DCNN[Atwood and Towsley, 2016]	76.8%	-	73.0%
MoNet[Monti et al., 2017]	81.7%	-	78.8%
DeepWalk[Perozzi et al., 2014]	67.2%	43.2%	65.3%
Planetoid[Yang et al., 2016]	75.7%	64.7%	77.2%
ICA[Lu and Getoor, 2003]	75.1%	69.1%	73.9%
LP[Zhu et al., 2003a]	68.0%	45.3%	63.0%
SemiEmb[Weston et al., 2012]	59.0%	59.6%	71.1%
ManiReg[Belkin et al., 2006]	59.5%	60.1%	70.7%

Table 4.2: This table shows the test classification accuracies of the semi-supervised learning experiments described in Section 4.4.1. The test sets consist of 1,000 data points. The GGP accuracies are averaged over 10 random restarts. The results for DCNN and MoNet are copied from [Monti et al., 2017] while the results for the other models are from [Kipf and Welling, 2016]. Please refer to Section 4.4.1 for discussions of the results.

well. The three benchmark data sets are citation networks with bag-of-words (BOW) features, and the prediction targets are the topics of the scientific papers in the citation networks.

The experimental results are presented in Table 4.2, and show that the predictive performance of the proposed GGP is competitive with the GCN and MoNet [Monti et al., 2017] (another deep learning model), and superior to the other baseline models. While the GCN outperforms the proposed model by small margins on the test sets with 1,000 data points, it is important to note that the GCN had access to 500 additional labelled data points for early stopping. As the GGP does not require early stopping, the additional labelled data points can instead be directly used to train the model to significantly improve the predictive performance. To demonstrate this

advantage, we report another set of results for a GGP trained using the 500 additional data points in Table 4.2, in the row labelled as '**GGP-X**'. The boost in the predictive performances shows that the GGP can better exploit the available labelled data to make predictions.

The GGP base kernel of choice is the 3rd degree polynomial kernel, which is known to work well with high-dimensional BOW features [Williams and Rasmussen, 2006]. We re-weighed the BOW features using the popular term frequency-inverse document frequency (TFIDF) technique [Sparck Jones, 1972]. The variational parameters and the hyper-parameters were jointly optimized using the ADAM optimizer [Kingma and Ba, 2014].

The baseline models that we compared to are the ones that were presented and compared to in [Kipf and Welling, 2016] and [Monti et al., 2017].

4.4.2 Active Learning on Graphs

Active learning is a domain that faces the same challenges as semi-supervised learning where labels are scarce and expensive to obtain [Zhu, 2005]. In active learning, a subset of unlabelled data points are selected sequentially to be queried according to an acquisition function, with the goal of maximizing the accuracy of the predictive model using significantly fewer labels than would be required if the labelled set were sampled uniformly at random [Balcan et al., 2010]. A motivating example of this problem scenario is in the medical setting where the time of human experts is precious, and the machines must aim to make the best use of the time. Therefore, having a data efficient predictive model that can generalize well with few labels is of critical importance in addition to having a good acquisition function.

In this section, we leverage GGP as the semi-supervised classification model of active learner in graph-based active learning problem [Zhu, 2005, Ma et al., 2013, Dasarathy et al., 2015, Jun and Nowak, 2016, Mac Aodha et al., 2014]. The GGP is paired with the proven Σ-optimal (SOPT) acquisition function to form an

active learner [Ma et al., 2013]. The SOPT acquisition function is a model agnostic acquisition function that leverages the spectral decomposition of the observed graph's Laplacian matrix and the indices of the labelled nodes to identify the next node to query, such that the predictive accuracy of the active learner is maximally increased. The main goal of the active learning experiments is to demonstrate that the GGP can learn better than both the GCN and the Label Propagation model (LP) [Zhu et al., 2003a] with very few labelled data points.

Starting with only 1 randomly selected labelled data point (i.e., node), the active learner identifies the next data point to be labelled using the acquisition function. Once the label of the said data point is acquired, the classification model is retrained and its test accuracy is evaluated on the remaining unlabelled data points. In our experiments, the process is repeated until 50 labels are acquired. The experiments are also repeated with 10 different initial labelled data points. In addition to the SOPT acquisition function, we show the results of the same models paired with the random acquisition function (RAND) for comparisons.

The test accuracies with different numbers of labelled data points are presented as learning curves in Figure 4.2. In addition, we summarize the results numerically using the Area under the Learning Curve (ALC) metric in Table 4.3. The ALC is normalized to have a maximum value of 1, which corresponds to a hypothetical learner that can achieve 100% test accuracy with only 1 label. The results show that the proposed GGP model is indeed more data efficient than the baselines and can outperform both the GCN and the LP models when labelled data are scarce.

The benchmark data sets for the active learning experiments are the Cora and Citeseer data sets. However, due to technical restriction imposed by the SOPT acquisition function, only the largest connected sub-graph of the data set is used. The restriction reduces the number of nodes in the Cora and Citeseer data sets to $2,485$ and $2,120$ respectively. Both of the data sets were also used as benchmark data sets in [Ma et al., 2013].

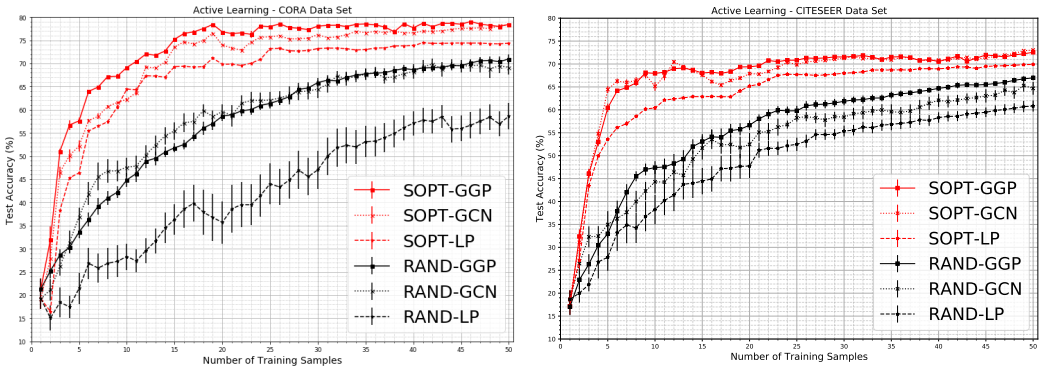

Figure 4.2: The sub-figures show the test accuracies from the active learning experiment (y-axis) for the Cora (left) and Citeseer (right) data sets with different number of labelled data points (x-axis). The results are averaged over 10 trials with different initial data points. SOPT and RAND refer to the acquisition functions described in Section 4.4.2. The smaller error bars of 'RAND-GGP' compared to those of 'RAND-GCN' demonstrate the relative robustness of the GGP models under random shuffling of data points in the training data set. The tiny error bars of the 'SOPT-*' results show that the 'SOPT' acquisition function is insensitive to the randomly selected initial labelled data point. Please also refer to Table 4.3 for numerical summaries of the results.

	Cora	**Citeseer**
SOPT-GGP	0.733 ± 0.001	0.678 ± 0.002
SOPT-GCN	0.706 ± 0.001	0.675 ± 0.002
SOPT-LP	0.672 ± 0.001	0.638 ± 0.001
RAND-GGP	0.575 ± 0.007	0.557 ± 0.008
RAND-GCN	0.584 ± 0.011	0.533 ± 0.008
RAND-LP	0.424 ± 0.020	0.490 ± 0.011

Table 4.3: This table shows the Area under the Learning Curve (ALC) scores for the active learning experiment. ALC refers to the area under the learning curves shown in Figure 4.2 normalised to have a maximum value of 1. The ALCs are computed by averaging over 10 different initial data points. The results show that the GGP is able to generalise better with fewer labels compared to the baselines. 'SOPT' and 'RAND' refer to the acquisition functions used. Please refer to Section 4.4.2 for discussions of the results.

We pre-process the BOW features with TFIDF and apply a linear kernel as the base kernel of the GGP. All parameters are jointly optimized using the ADAM optimizer. The GCN and LP models are trained using the settings recommended in [Kipf and Welling, 2016] and [Ma et al., 2013] respectively.

4.5 Discussions

A Gaussian process model that is data-efficient for semi-supervised learning problems on graphs is presented in this chapter. The experiments show that the proposed model is competitive with the state-of-the-art deep learning models, and outperforms when the number of labels is small. The proposed model is simple, effective and can leverage modern scalable variational inference algorithm for GP with minimal modification. In addition, the construction of our model is motivated by distribution regression using the empirical kernel mean embeddings, and can also be viewed under the framework of filtering in the graph spectrum. The spectral view offers a new potential research direction that can be explored in future work. Another interesting direction of research, as described in Section 4.2, is the investigation of non-naive ways to account for multi-hop neighbours in the graphs.

5

Edge Clustering Dynamic Network Model

In this chapter, I present a dynamic edge exchangeable network model that can capture sparse connections observed in real temporal networks, in contrast to existing models which are dense. The model achieved competitive link prediction accuracy on multiple data sets when compared to benchmark models, including a dynamic variant of the blockmodel and a dynamic latent feature model. It is also capable of extracting interpretable time-varying community structures from the data. In addition to sparsity, the model accounts for the effect of social influence on vertices' future behaviours. Compared to the dynamic blockmodels, the proposed model has a smaller latent space. The compact latent space requires a smaller number of parameters to be estimated in variational inference and results in a computationally friendly inference algorithm.

Many real-world events that involve collections of pair-wise interactions between entities and individuals can be considered as network data. The interactions between members of the networks often exhibit rich structures that evolve through time in a subtle but discernible pattern. Capturing the structure and pattern allows one to gain insights into the nature of the interactions and the identities of the members, as well as to predict future interactions. However, it can be challenging to distil useful information from dynamic networks because of the limited amount of interactions observed within each unit of time. These interactions are often also noisy and possibly non-stationary. Modelling such data sets requires practitioners to make

assumptions in order to borrow strength from interactions observed at different points in time and with different counter-parties.

Probabilistic dynamic network models that induce dependency across networks observed at different times with latent structures have been studied by many researchers. Many existing approaches are dynamic generalisations of the Stochastic Block Models (SBMs) [Nowicki and Snijders, 2001], Mixed-membership Stochastic Blockmodels (MMSBs) [Airoldi et al., 2008] and Latent Space Models [Hoff et al., 2002]. These dynamic generalisations include [Ho et al., 2011, Xu and Hero, 2014, Heaukulani and Ghahramani, 2013, Sarkar and Moore, 2006, Sewell et al., 2017]. A Gaussian process-based approach to dynamic network models was also explored in [Durante and Dunson, 2014]. While the existing models do indeed capture the connectivity patterns of networks and their evolution, they are not able to capture the sparse connections that are observed in many network data [Goldenberg et al., 2010].

Building on recent works on edge-exchangeable non-parametric Bayesian network models [Williamson, 2016, Cai et al., 2016, Crane and Dempsey, 2016], I propose a discrete-time dynamic network model designed to capture three important properties that are observed in real-world temporal networks. I discuss the three properties in Section 5.1, and introduce the proposed model in Section 5.2. In Section 5.4, I review some existing models for temporal networks and compare them to the proposed model. This is followed by further discussions on some related works in Section 5.5. Finally, I present some experimental results and discussions in Section 5.6.

5.1 Sparse Temporal Networks

We consider a temporal sequence of networks $\{\mathcal{G}^{(t)}\}_{t=1}^{T}$ indexed by $t \in \mathbb{Z}^{+}$, each containing a set of vertices and edges $\mathcal{G}^{(t)} \equiv \{\mathcal{V}^{(t)}, \mathcal{E}^{(t)}\}$. $\mathcal{E}^{(t)} = \{e_1^{(t)}, ..., e_{N^{(t)}}^{(t)}\}$ is the set of $N^{(t)}$ edges observed at t and $\mathcal{V}^{(t)}$ is the set of vertices that have participated in at least one edge up to t such that $\mathcal{V}^{(t-1)} \subseteq \mathcal{V}^{(t)}$. An edge $e_i^{(t)} = (v_i^{(t)}, v_i'^{(t)})$ is a tuple

of two interacting vertices, and may or may not be directed. We focus on undirected temporal networks in this paper and ignore the order of vertices in the tuple. Many events that involve pairwise interactions between entities and individuals can be considered as temporal networks. Real-world examples of temporal networks include e-mail communication networks, friendship networks, trading networks and many more.

Temporal networks exhibit statistical properties that are of practical interest. We focus on addressing three important properties in this paper: sparsity, community structure and social influence. Our proposed model differs from the existing models by taking into account all three properties simultaneously while being less computationally demanding compared to many existing ones. It also allows the set of vertices to grow over time, as opposed to forcing the set of vertices to remain constant.

5.1.1 Sparsity

The connections observed among the vertices in real-world networks are typically sparse, with only a small number of observed edges compared to all possible pairs of vertices [Barabási and Pósfai, 2016, Goldenberg et al., 2010]. Using a social network with thousands of members as an intuitive example, most, if not all members of the social network are connected to tens or hundreds of other members in the network, instead of thousands of other members. Therefore, the total number of connections in the social network is an order of magnitude smaller than the number of all possible pairs of members. Sparsity is an essential condition to maintain certain structural properties observed in real networks, such as the 'small world phenomena' and the power-law degree distributions, as these properties can only occur in sparse networks [Orbanz and Roy, 2015]. Formally, a network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{E}|$ edges and $|\mathcal{V}|$ vertices is sparse if $|\mathcal{E}| = o(|\mathcal{V}|^2)$ (i.e., $|\mathcal{E}|$ is asymptotically upper bounded by $c \cdot |\mathcal{V}|^2$ for $c > 0$).

In the context of temporal networks $\{\mathcal{G}^{(t)}\}_{t=1}^{T}$, each of the observed networks $\mathcal{G}^{(t)}$ may be sparse. We argue that in some cases, $\mathcal{G}^{(t)}$ is sparser than static networks

that are aggregates of temporal networks. Therefore, it is important that temporal network models are able to capture the sparse property of real observations. However, as we discuss in Section 5.4, most existing dynamic network models do not allow for sparse connections because of their underlying exchangeability assumption.

5.1.2 Community Structure

In many networks, the connections between vertices can be clustered into different categories, with overlapping subsets of vertices dominating each of the categories. In a social network, for example, edges can represent relationships between colleagues, college friends, family members and other types of social relationships. Within each type of relationship, a subset of vertices are over-represented compared to the others. Some vertices may also dominate multiple types of relationships. Vertices that participate in multiple types of connections are referred to as having mixed-memberships in the stochastic blockmodel literature [Airoldi et al., 2008]. In temporal networks, the number of edges belonging to each category can fluctuate through time and the dominating vertices of each category can also evolve [Xing et al., 2010, Xu and Hero, 2014, Ho et al., 2011]. The types of edges are not annotated in many network data sets, but can be inferred through statistical analysis. While dynamic variants of the mixed-membership stochastic blockmodel can model the mixed-membership community structure, they cannot account for sparse connections by construction [Orbanz and Roy, 2015]. As a consequence, these models do not allow for the presence of hubs and power-law degree distributions.

5.1.3 Social Influence

The presence of an edge connecting two vertices at time t implies that the two vertices had interacted in some ways during the period. The interaction may influence the states of both vertices at the subsequent time point, causing the two vertices to interact with the world similarly in the future. The two vertices may also interact with many other vertices at t, causing their respective future states to reflect their

own historical states and the social influence from their respective neighbours at t to various degrees. The causal nature of time allows us to draw potential causal links from the observed connections of vertices at t to their future states, and perform inference on future connections between vertices [Blundell et al., 2012, Farajtabar et al., 2015, Linderman and Adams, 2014]. The effect of social influence plays an important role in the evolution of temporal networks, and should be taken into account in building temporal network models [Heaukulani and Ghahramani, 2013].

5.2 Dynamic Edge Exchangeable Network Model

We propose a dynamic model for sparse temporal networks $\{\mathcal{G}^{(t)}\}_{t=1}^{T}$ that is built upon the edge exchangeable framework proposed in [Cai et al., 2016, Crane and Dempsey, 2016]. By enforcing the edge exchangeable assumption to the marginal distribution at each time point t, the model allows $\mathcal{G}^{(t)}$ to be sparse. In contrast, the existing dynamic models guarantee that $\mathcal{G}^{(t)}$ is either empty or dense (see Section 5.4). The edge exchangeable marginals for different time points are coupled by latent Gaussian Markov chains to model the social influence effects and evolution of the temporal networks. Additionally, we introduce a Poisson vertex birth mechanism to allow new vertices to join the networks at different times. In the following subsections, we first discuss the generative process for individual $\mathcal{G}^{(t)}$ and introduce the Markov dynamics that couple together the temporal sequence of networks. We then present a variational inference algorithm and discuss its computational complexity in Section 5.3.

5.2.1 Edge Exchangeable Sparse Networks

The edge exchangeable assumption applied to each of the networks $\mathcal{G}^{(t)}$ in the temporal sequence $\{\mathcal{G}^{(t)}\}_{t=1}^{T}$ dictates that the edges in the set $\mathcal{E}^{(t)} = \{e_1^{(t)}, ..., e_{N^{(t)}}^{(t)}\}$ are exchangeable and the probability distribution of $\mathcal{G}^{(t)}$ is invariant to the order of the edges in $\mathcal{E}^{(t)}$. Therefore, $e_1^{(t)}, ..., e_{N^{(t)}}^{(t)}$ are $i.i.d.$ samples of an edge distribution $P_t(e)$. Following our notations from Section 5.1, an undirected edge $e_i^{(t)}$ is a tuple of

two unordered participating vertices $(v_i^{(t)}, v_i'^{(t)})$. As a result, $P_t(e = (v_i^{(t)}, v_i'^{(t)}))$ can be factorised into a product of identical vertex distributions $P_t(v_i^{(t)})P_t(v_i'^{(t)})$. We present some simulation results in Section 5.6 to demonstrate that our edge exchangeable network construction can model sparsity. We refer the readers to [Janson, 2017] for a detailed general review of edge exchangeable networks.

5.2.2 Community Structure Mixture Model

To incorporate community structure, we adopted the edge clustering approach proposed for non-parametric static network model in [Williamson, 2016] to our model. We assume that the observed edges $\mathcal{E}^{(t)}$ are samples of a mixture of M edge distributions in Equation (5.1) (corresponding to M communities) where the per-edge latent mixture component indicators $c_i^{(t)} \in \{1, ..., M\}$ describe the types of connection that the edges belong to. We suppress the parameters in Equation (5.1) for compact presentation.

$$P_t(e = (v_i^{(t)}, v_i'^{(t)})) = \sum_{m=1}^{M} \left[P_t(c_i^{(t)} = m) P_t(v_i^{(t)} | c_i^{(t)} = m) P_t(v_i'^{(t)} | c_i^{(t)} = m) \right] \quad (5.1)$$

For undirected networks, $P_t(v_i^{(t)}|c_i^{(t)})$ and $P_t(v_i'^{(t)}|c_i^{(t)})$ are identical vertex distributions which we collectively denote as $P_t(v|c_i^{(t)})$ for notational convenience. $P_t(c_i^{(t)})$ and $\{P_t(v|c_i^{(t)} = m)\}_{m=1}^{M}$ are parameterised as logistic normal distributions with M-dimensional and $(|\mathcal{V}^{(t-1)}| + L^{(t)})$-dimensional support respectively. $|\mathcal{V}^{(t-1)}|$ is the number of vertices observed in the networks up to the previous time point (with $|\mathcal{V}^{(0)}| = 0$) and $L^{(t)}$ is the Poisson distributed number of potential new vertices that may join the network at t.

The relative sizes of the communities may grow and shrink over time and exhibit temporal dependency, as sudden large changes to the community sizes are rare. We construct the following latent Gaussian Markov chain for the parameters of

the logistic normal distributions $\mathbf{k}^{(t)} \in \mathbb{R}^{M\times 1}$ to capture the temporal dependency.

$$P_t(c_i^{(t)} = m|\mathbf{k}^{(t)}) \propto e^{k_m^{(t)}} \tag{5.2}$$

$$P(\mathbf{k}^{(1:T)}) = \mathcal{N}(\mathbf{k}^{(1)};\mathbf{m},\mathbf{L}\mathbf{L}^\intercal)\prod_{t=2}^{T}\mathcal{N}(\mathbf{k}^{(t)};\mathbf{A}\mathbf{k}^{(t-1)},\mathbf{L}\mathbf{L}^\intercal) \tag{5.3}$$

$\mathbf{m} \in \mathbb{R}^{M\times 1}$, $\mathbf{A} \in \mathbb{R}^{M\times M}$ and the lower-triangular matrix $\mathbf{L} \in \mathbb{R}^{M\times M}$ are the parameters of the Markov chain.

The m^{th} vertex distribution at t, $P_t(v|c_i^{(t)} = m)$ (abbreviated as $P_{t,m}(v)$ for compactness), encodes the relative dominance of the vertices in community m at time t. We endow each vertex v at time t with a Gaussian distributed latent state vector $\mathbf{h}_v^{(t)} \in \mathbb{R}^{M\times 1}$, such that

$$p_{t,m}(v = v_i) \propto e^{h_{v,m}^{(t)}}. \tag{5.4}$$

The normalising constant for Equation (5.4) is the sum over the exponentiated m^{th} hidden states of all the vertices in $\mathcal{V}^{(t-1)}$ and the $L^{(t)}$ potential new vertices at t.

5.2.3 Markov Dynamics with Social Influence

The latent state vectors of the $L^{(t)}$ potential new vertices are sampled from an initial Gaussian distribution parameterised by the mean vector $\boldsymbol{\mu} \in \mathbb{R}^{M\times 1}$ and the lower-triangular matrix $\mathbf{B} \in \mathbb{R}^{M\times M}$

$$p(\mathbf{h}) = \mathcal{N}(\mathbf{h};\boldsymbol{\mu},\mathbf{B}\mathbf{B}^\intercal). \tag{5.5}$$

If the potential new vertices are indeed sampled to form $\mathcal{G}^{(t)}$, they are added to the set $\mathcal{V}^{(t)}$ and together with the existing vertices that joined in the previous time steps, their respective latent state vector $\mathbf{h}_v^{(t)}$ evolves to the next time point according to the conditional Gaussian distribution in Equation (5.6). The new potential vertices at t that do not participate in $\mathcal{G}^{(t)}$ are dropped from the model and ignored in the next

time step.

$$p(\mathbf{h}_v^{(t+1)}|\mathcal{G}^{(t)},\{\mathbf{h}_i^{(t)}|i\in\mathcal{V}^{(t)}\}) = \mathcal{N}(\mathbf{h}_v^{(t+1)};\mathbf{f}(v,\mathcal{G}^{(t)},\{\mathbf{h}_i^{(t)}|i\in\mathcal{V}^{(t)}\}),\mathbf{B}\mathbf{B}^\intercal) \quad (5.6)$$

To model the varying degrees of social influence on the evolution of temporal networks, the mean of the conditional Gaussian in Equation (5.6) is parameterised as a M-dimensional vector function $\mathbf{f}_{v,t} = \mathbf{f}(v,\mathcal{G}^{(t)},\{\mathbf{h}_i^{(t)}|i\in\mathcal{V}^{(t)}\})$ defined as

$$\mathbf{f}_{v,t} = w_{vv}^{(t)}\mathbf{h}_v^{(t)} + \sum_{i\in ne(v,t)} w_{vi}^{(t)}\mathbf{h}_i^{(t)} \quad (5.7)$$

where $ne(v,t)$ is the set of neighbour vertices that vertex v formed an edge with in $\mathcal{G}^{(t)}$. Equation (5.7) is a weighted average of $\mathbf{h}_v^{(t)}$ and the previous latent state vectors of vertex v's neighbours at t. The parameterization of Equation 5.7 implies that the state of vertex v at time $t+1$ is influenced by the states of the vertices that it interacted with at t, capturing the effect of social influence among the vertices. The non-negative weights $w_{vi}^{(t)}$ is a dot-product based similarity measure between vertex v and i defined as

$$w_{vi}^{(t)} = \frac{e^{\mathbf{h}_v^{(t)}\cdot\mathbf{h}_i^{(t)}}}{e^{\mathbf{h}_v^{(t)}\cdot\mathbf{h}_v^{(t)}} + \sum_{j\in ne(v,t)} e^{\mathbf{h}_v^{(t)}\cdot\mathbf{h}_j^{(t)}}}. \quad (5.8)$$

Intuitively, the neighbours of the vertex v at t pull the latent state of v towards themselves at different degrees after they interacted at t. The neighbours that are more similar to vertex v in the latent space have higher influence on its future latent state. If the vertex v did not interact with any vertex, then $\mathbf{f}_{v,t} = \mathbf{h}_v^{(t)}$ and Equation (5.6) is simply a random walk.

The weighted average parameterisation of the conditional mean in Equation (5.7) is similar to the local context-based soft attention mechanism proposed in [Luong et al., 2015] for NLP neural networks in two ways. Firstly, it assigns higher weights to vertices that are more similar in the latent space. Similarly, the attention mechanism in [Luong et al., 2015] assigns higher weights to words that are

similar in context. Secondly, Equation (5.7) avoids the computationally expensive operation of summing over all existing vertices by looking only at the neighbours of vertex v at the previous time step. It is 'local' in the sense that it only sums over vertex v's immediate neighbours. The attention mechanism in [Luong et al., 2015] is 'local' in the sense that the attention mechanism only consider other words that surround the target word in a sentence instead of the whole corpus, leading to saving in computational costs. We refer to the latent process parameterized by Equation (5.7) as the Attention Augmented State-space (**ATTAS**). To the best of our knowledge, we are the first to propose an attention mechanism in probabilistic model for network data. We compare ATTAS to a simple random walk (**RW**) process as well as other models in the prediction experiments.

5.2.4 Poisson Vertex Birth Mechanism

The number of new vertices $L^{(t)}$ at time t is uncertain prior to observing $\mathcal{G}^{(t)}$. We seek to account for the uncertainty with a Poisson prior distribution with log-rate parameter $\lambda^{(t)}$

$$P(L^{(t)}|\lambda^{(t)}) = Poisson(e^{\lambda^{(t)}}). \tag{5.9}$$

We observed in many temporal networks that $L^{(1)}$ is typically large because no vertex existed in the networks prior to $t = 1$. The subsequent $L^{(t)}$ typically becomes smaller. Therefore, we propose to capture the temporal dynamics of $L^{(t)}$ with an auto-regressive Markov chain prior on the parameter sequence $\lambda^{(1:T)}$

$$p(\lambda^{(1:T)}) = \mathcal{N}(\lambda^{(1)}; \mu_\lambda; \sigma_\lambda^2) \prod_{t=2}^{T} \mathcal{N}(\lambda^{(t)}; a_\lambda \lambda^{(t-1)}, \sigma_\lambda^2). \tag{5.10}$$

In the scenario when $0 < a_\lambda < 1$, the long-run expectation of $L^{(t)}$ is 1 (as the long-run expectation of $\lambda^{(t)}$ is 0) despite the larger initial expectation of e^{μ_λ}. As previously described, the $L^{(t)}$ potential new vertices are assigned a latent state vector sampled from Equation (5.5) and are discarded from the model if they do not participate in $\mathcal{G}^{(t)}$.

5.2.5 Model Summary

We provide a generative summary of the dynamic edge-exchangeable network model described in the previous sub-sections, and depict the generative process for a sequence of $T = 3$ temporal networks with 2 communities in Figure 5.1. We assume that the number of edges sampled at each time point $N^{(1:T)}$ are directly specified. However, they can also be modeled with Poisson distributions or directly observed from the data sets in practice. The data generating process of the proposed model is described in Algorithm 3.

The data generating process is also presented pictorially in Figure 5.1. The individual vertices in Figure 5.1 are represented as colored balls in the data box, with pairs of vertices forming edges that compose the networks. The yellow vertex does not participate at $t = 2$ but remains in the network because it participated at $t = 1$ while the green vertex joins at $t = 2$. The community labels of the edges are sampled from the grayscale community distributions at the top, and annotated on the sampled edges. Conditioning on the community label, two vertices are sampled from the colored mixture distributions above the box to form an edge. The vertex state vectors $\mathbf{h}_v^{(t)}$ of the mixture distributions are represented as colored circles in the rectangles above the distributions, with their ATTAS evolution mechanism described in Section 5.2.3 depicted as directed arrows through time. $\mathbf{h}_v^{(t)}$ of new potential vertices are grouped in dotted rectangles, with those that do not immediately participate in the network discarded at every time step (e.g., black at $t = 1$). The number of new potential vertices $L^{(t)}$ is determined by Poisson distributions with log-rates $\lambda^{(t)}$ that evolve according to a Gaussian Markov chain.

Algorithm 3 Data generating process for the dynamic network model.

Input: $T, N^{(1:T)}, M, \theta = \{\mu_\lambda, \sigma_\lambda, a_\lambda, \boldsymbol{\mu}, \mathbf{B}, \mathbf{m}, \mathbf{A}, \mathbf{L}\}$
Output: A sequence of networks $[(\mathcal{V}^{(1)}, \mathcal{E}^{(1)}), \ldots, (\mathcal{V}^{(T)}, \mathcal{E}^{(T)})]$

$\mathcal{V}^{(0)} \leftarrow \{\}$
for $t \leftarrow 1$ `to` T **do**
  $\mathcal{V}^{(t)} \leftarrow \mathcal{V}^{(t-1)}, \mathcal{E}^{(t)} \leftarrow \{\}$
  `Sample` $\boldsymbol{\lambda}^{(t)}$ using Equation (5.10)
  `Sample` $L^{(t)}$ using Equation (5.9)
  `Sample` $\mathbf{k}^{(t)}$ using Equation (5.3)
  `Sample` $\mathbf{h}^{(t)}$ and $\mathbf{h}_z^{(t)}$ using Equation (5.6)
  for $i \leftarrow 1$ `to` N_t **do**
    `Sample` $c_i^{(t)}$ using Equation (5.2)
    `Sample` $e_i^{(t)} = (v_i^{(t)}, v_i'^{(t)})$ using Equation (5.4)
    $\mathcal{E}^{(t)} \leftarrow \{\mathcal{E}^{(t)}, e_i^{(t)}\}$
    $\mathcal{V}^{(t)} \leftarrow \mathcal{V}^{(t)} \cup \{v_i^{(t)}, v_i'^{(t)}\}$

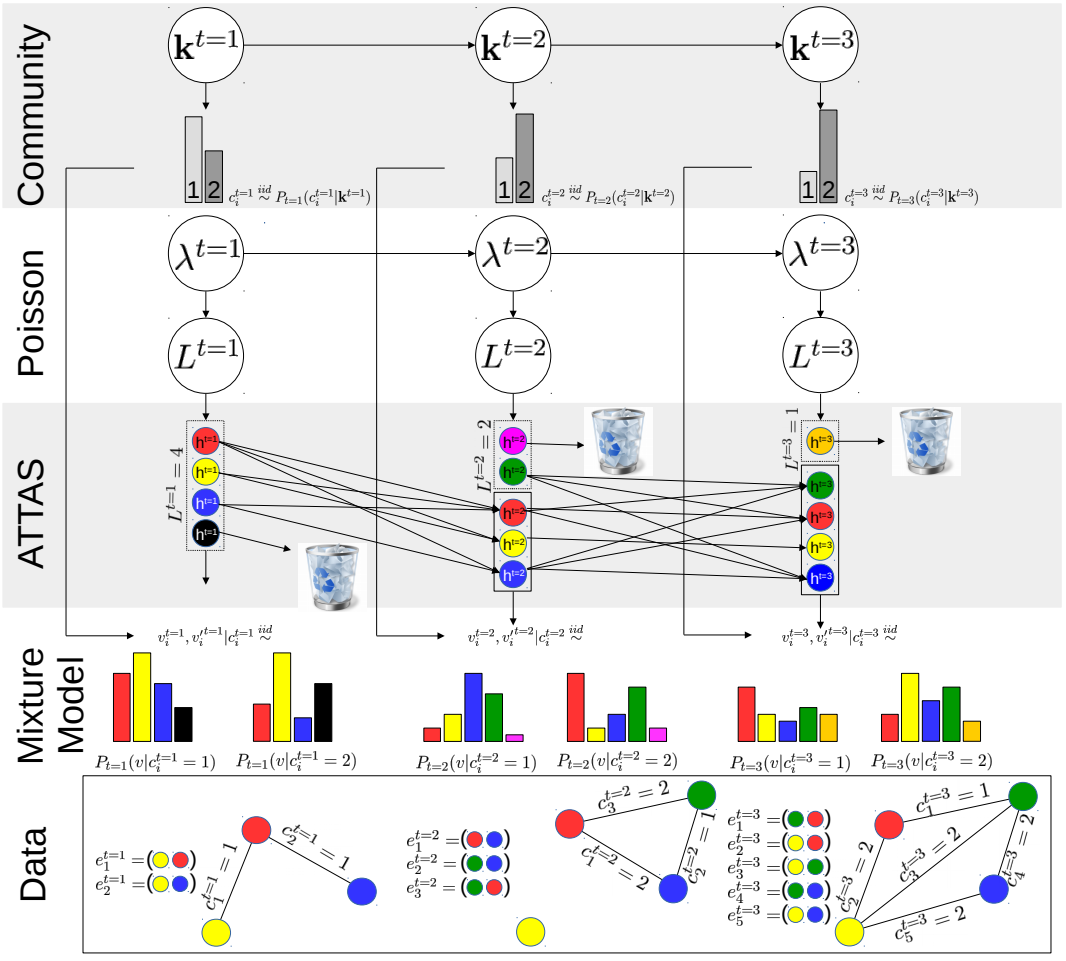

Figure 5.1: The figure shows the generative process for a sequence of 3 temporal networks with 2 communities. This figure is best understood together with the text in Section 5.2.5.

5.3 Variational Inference

We approximate the posterior distributions of the model's latent variables with a variational inference (VI) algorithm [Blei et al., 2017]. The latent variables of interest are $L^{(1:T)}$, $\boldsymbol{\lambda}^{(1:T)}$, $\mathbf{k}^{(1:T)}$, the per-edge community type latent variables $c_i^{(t)}$ and the per-vertex latent state vectors $\mathbf{h}_v^{(\tau_v:T)}$. τ_v is the time when vertex v first joined the networks.

The time dependency of temporal networks requires that the approximating variational distributions q for $\mathbf{h}_v^{(\tau_v:T)}$, $\mathbf{k}^{(1:T)}$ and $\boldsymbol{\lambda}^{(1:T)}$ to preserve their time dependency. To preserve the time dependency while allowing tractable variational distributions, we utilize the structured mean-field (SMF) family of variational distributions [Saul and Jordan, 1996]. The SMF family approximates $q(\mathbf{h}_v^{(\tau_v:T)})$ for $v \in \mathcal{V}^{(T)}$, $q(\mathbf{k}^{(1:T)})$ and $q(\boldsymbol{\lambda}^{(1:T)})$ as Gaussian Markov chains [Blei and Lafferty, 2006, Ghahramani and Jordan, 1997, Ng et al., 2016, Archer et al., 2015], and the remaining variational distributions as fully-factorized exponential family distributions.

The proposed ATTAS latent process introduce non-conjugate structures to the model through the $\mathbf{f}_{v,t}$ function in Equation (5.7). Additionally, the log-normalizing constants of the multivariate logistic normal distributions in Equation (5.2) and Equation (5.4) are also non-conjugate. The two sources of non-conjugacy render the evidence lower bound (ELBO) objective function of VI analytically intractable. We take a two-pronged approach to tackle the intractable ELBO. We first linearize the log-normalizing constants using Taylor's series approximation [Blei and Lafferty, 2006]. The linear approximation introduces an additional lower bound to the ELBO, but allows $\{\{q(c_i^{(t)})\}_{i=1}^{N^{(t)}}\}_{t=1}^{T}$ to be optimized analytically through fast conjugate updates. The conjugate updates are equivalent to optimizing the variational parameters with natural gradients, and lead to faster convergence. To tackle the more complex intractable terms introduced by ATTAS, we resort to optimizing the variational parameters of $q(\mathbf{h}_v^{(\tau_v:T)})$ with ADAM [Kingma and Ba, 2014] using unbiased Monte Carlo gradients computed with the reparameterization tricks [Kingma and Welling, 2013, Archer et al., 2015]. We alternate between performing the conjugate updates

and multiple steps of the stochastic gradient updates. We find that exploiting the linear approximations and conjugate updates lead to faster and better convergence when compared to a fully Monte Carlo approach. The details of the linear approximations and the conjugate updates are outlined in Appendix A.

We learn the model parameters θ by maximizing the same ELBO objective as VI. The model parameters updates are performed together with the stochastic gradient updates of VI.

We validated the goodness of the VI approximation using a simulated experiment to recover ground truth edge community labels $c_i^{(t)}$. The VI algorithm was able to recover 96% of the 1694 ground truth labels across 3 time steps at a normalized mutual information (NMI) score of 0.75. Derivations of the VI algorithm follows the standard ELBO maximization framework [Blei et al., 2017] and the conjugate updates follow from the method derived in [Hoffman et al., 2013].

5.3.1 Computational Complexity

The computational bottleneck of the variational inference algorithm lies in computing the $M \times T$ approximated expected log-normalizing constant terms and the corresponding gradients, contributed by the logistic normal distributions in Equation (5.4). Computing the approximated expectations requires summing over the expected exponentiated latent state vectors $\mathbf{h}_v^{(t)}$ for all vertices in $\mathcal{V}^{(T)}$. The sums are then used to update $\{\{q(c_i^{(t)})\}_{i=1}^{N^{(t)}}\}_{t=1}^{T}$. Therefore, the computational complexity of the VI algorithm is $O(E_{TOT}M + |\mathcal{V}^{(T)}|MT)$, where $E_{TOT} = \sum_{t=1}^{T} N^{(t)}$, $|\mathcal{V}^{(T)}|$ is the total number of vertices in the temporal networks, M is the number of communities and T is the number of time points. The computational complexity of VI for the proposed model is significantly lower than the dynamic variants of mixed-membership stochastic blockmodels, which have complexity of $O(M|\mathcal{V}^{(T)}|^2T)$ and beyond [Xing et al., 2010, Ho et al., 2011].

5.4 Comparisons to Existing Models

We compare and contrast the proposed dynamic edge exchangeable network model to some existing probabilistic models for temporal networks, focusing on how the proposed model handles the 3 key properties discussed in Section 5.1 differently, and the models' inference computational complexities.

One key property of the proposed model is its ability to model sparse connections in each networks $\mathcal{G}^{(t)}$ in the temporal network sequence by assuming the edges observed within each time points are exchangeable units of data. This is in contrast to the existing models that are dynamic variants of the mixed-membership stochastic blockmodel [Xing et al., 2010, Ho et al., 2011, Xu and Hero, 2014, Zreik et al., 2017], latent space model [Sarkar and Moore, 2006, Sewell et al., 2017] and latent feature model [Heaukulani and Ghahramani, 2013]. Marginally, these existing dynamic models make use of likelihood models that fall under the exchangeable random graph framework of Aldous-Hoover representation theorem [Aldous, 1981, Hoover, 1979]. It is well known that exchangeable random graphs are either empty or dense [Orbanz and Roy, 2015]. Therefore, under the existing models, $\mathcal{G}^{(t)}$ cannot be sparse. As we discussed in Section 5.1, certain structural properties of real networks are unique to sparse networks only. The limitation of the existing models in capturing sparsity is an important disadvantage.

The proposed model incorporates community structure by directly clustering edges with a mixture model, and interpret the per-edge latent mixture component indicators $c_i^{(t)}$ as the types of interactions between the two interacting vertices (e.g., work connections, college friends, family ties in social networks). The edge-clustering approach models the same type of community structure as the assortative mixed-membership stochastic blockmodel (aMMSB) [Gopalan et al., 2012] despite the differences in model construction. In aMMSB, each of the vertices assumes different interaction-specific latent roles when interacting with other vertices and two vertices form an edge with high probability only when both vertices assume the same latent roles (e.g., colleague-colleague interactions etc.). The types of

interactions encoded in aMMSB correspond to the interaction-specific latent roles of both interacting vertices, and is equivalent to the edge-clustering formulation. For example, a colleague-colleague interaction is equivalent to a work connection.

The direct edge-clustering approach of our model results in a significantly smaller set of latent variables compared to an equivalent aMMSB. To model a static network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with M communities, the proposed edge-clustering model requires $|\mathcal{E}|$ latent mixture component indicator $\{c_i\}_{i=1}^{|\mathcal{E}|}$ while aMMSB (or other types of MMSB) requires $2 \cdot |\mathcal{V}|^2$ latent role indicators. Generalizing to temporal networks $\{\mathcal{G}^{(t)}\}_{t=1}^{T}$, the proposed model requires only $\sum_{t=1}^{T} |\mathcal{E}^{(t)}|$ latent variables to capture the community structure compared to $2 \cdot T|\mathcal{V}^{(T)}|^2$ for dynamic MMSBs. The difference in the size of latent space is especially significant when the networks $\{\mathcal{G}^{(t)}\}_{t=1}^{T}$ are sparse. The compactness of our model results in fewer numbers of variational distributions to approximate in variational inference, and lead to computational gains as discussed in Section 5.3.1.

The dynamic latent feature propagation model proposed in [Heaukulani and Ghahramani, 2013] also accounts for the social influence of neighbours by assigning 'social influence weights' to the vertices. The model is restricted in that each vertex has equal influence on their neighbours. The ATTAS proposal bypassed the restriction with the attention mechanism, such that the vertices exert higher influence on their neighbours that are more similar. In addition, the ATTAS construction does not introduce any additional model parameters while the social influence weights of the dynamic latent feature propagation model need to be learned.

5.5 Related Work

In addition to the work on dynamic networks mentioned previously, the ATTAS latent process proposed in Section 5.2 also relates to continuous-time point processes models for reciprocating relationships such as [Blundell et al., 2012, Farajtabar et al., 2015, Linderman and Adams, 2014]. These models are typically applied to a fixed

number of nodes and correspond to dense generative models, like MMSBs, which provide no simple control on the number of sampled interactions within any particular time window: each interaction follows directly from node-level or node-cluster-level latent features, as opposed to edge-cluster-level features. Continuous-time models have advantages and disadvantages compared to discrete-time models that are well-studied. We favour discrete-time modelling due to the inferential ease by which we can enforce a bottleneck of event counts, and the flexibility of attention models as way of parameterising interactions using individual-level latent vectors.

There have been many advances on non-parametric sparse network models [Caron and Fox, 2014, Cai et al., 2016, Crane and Dempsey, 2016, Williamson, 2016], with efficient Markov chain Monte Carlo algorithms. However, existing works on relating these models to dynamic modelling is limited [Williamson, 2016]. Models based on non-parametric block models exist in the literature (e.g., [Ishiguro et al., 2010]). It is not our goal to fill in the gap between continuous-time dynamic and non-parametric edge-clustering models. In practice, we believe that social data is too non-stationary for the elegant machinery of Hawkes processes to really be effective, and we see our contribution as a practical framework for short term predictions and historical smoothing and clustering of observed interactions over which extensions can be built.

5.6 Experiments

We conducted 3 experiments with the following goals.

1. Investigate sparsity under various hyper-parameter settings.
2. Benchmark the model's link prediction powers.
3. Investigate the model's capacity to capture community structures.

5.6.1 Sparse Networks Simulations

The proposed dynamic network model consists of T logistic normal edge distributions that are coupled by latent Markov chains. Each edge distribution can capture sparse connections in $\mathcal{G}^{(t)}$. In this experiment, we investigate the sparsity of networks simulated from a logistic normal edge distribution (i.e., $T = 1$) with different hyper-parameter settings, and show that the networks simulated with certain hyper-parameter settings are sparse. To focus on sparsity, we set $M = 1$ and ignore the community structure.

To simulate the networks, we first sample the number of latent vertices $L \sim Poisson(10^6)$ and the vertices' latent states $h_i \sim \mathcal{N}(0, \sigma^2)$ for $i \in \{1, \ldots, L\}$. The edges are then sampled from the edge distribution $P(e = (i, j)|\sigma) = P(v = i|\sigma)P(v = j|\sigma)$ where $P(v = i|\sigma) = \frac{e^{h_i}}{\sum_{l=1}^{L} e^{h_l}}$. The hyper-parameter of interest is the standard deviation σ. The standard deviation fully determines the excess kurtosis of the transformed random variable e^{h_i}, which in turn governs the shape of the logistic normal $P(v|\sigma)$. A small σ value corresponds to a flat $P(v|\sigma)$ while a large σ corresponds to a multi-modal $P(v|\sigma)$ because of the resulted heavy-tailed distribution for e^{h_i}. With a flat $P(v|\sigma)$, the number of vertices sampled to join the network (i.e., active vertices) increases quickly with respect to the number of edges sampled as the probability mass is distributed more evenly among the potential vertices. With a multi-modal distribution the number of active vertices increases more slowly, leading to denser networks.

The quantity of our interest is the ratio of the log number of sampled edges $\log|\mathcal{E}|$ to the log number of active vertices $\log|\mathcal{V}|$. A network is sparse if the ratio is less than 2 [Cai et al., 2016]. We simulate networks with increasingly more edges at different σ values, and show that the sampled networks are sparse for small σ in Figure 5.2.

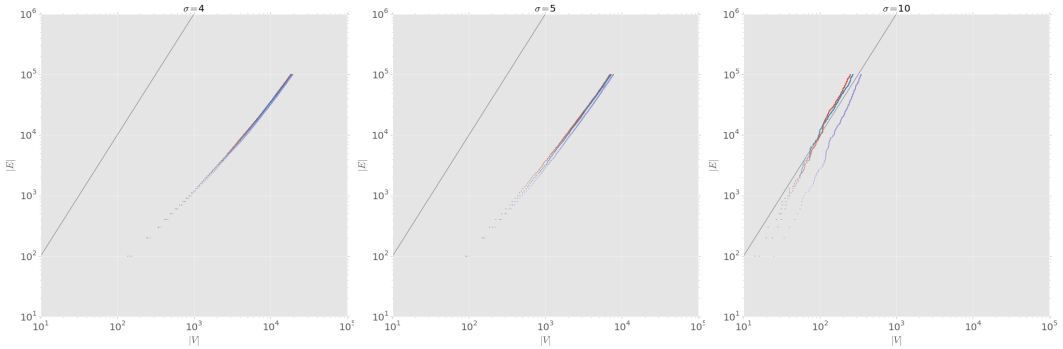

Figure 5.2: The log-log plots show the number of sampled edges v.s. the number of active vertices of the simulated networks. The σ hyper-parameter is set to $4, 5, 10$ (left to right), resulting in $\log|\mathcal{E}|/\log|\mathcal{V}|$ ratio of approximately $1.5, 1.7, 2.4$ respectively, as shown in the slopes of the colored dots. The different dot colors in each plot represent different random seed. The black solid lines have slopes equal to 2. The scales on the x and y axes are 10^1 to 10^5 and 10^6 respectively.

5.6.2 Link Predictions

We conducted 3-fold held-out link prediction experiments using three temporal binary network data sets. The experiments are motivated by the real-world scenario of predicting imminent interactions between pairs of vertices given partially observed data in the current time period, and the network's history. In the experiments, the edges observed in the last time slot were randomly split into three sets of similar sizes. Two of the sets were then combined with other edges from the previous time slots to form the training data set. The trained models were evaluated on the edges in the held-out fold. We called this experiment *Challenge 1*.

In addition, we compared the top-performing models from *Challenge 1* in a more difficult scenario of predicting edges between pairs of vertices that were not observed to have interacted in the network's history. We called this experiment *Challenge 2*. We dropped the COLLEGE data set in *Challenge 2* due to its extremely small held-out set after censoring the previously observed interactions.

In *Challenge 1*, the proposed model (ATTAS) is benchmarked against the dynamic mixture of mixed-membership stochastic blockmodel (dM^{3}SB) [Ho et al., 2011], dynamic latent feature propagation model (LFP) [Heaukulani and Ghahra-

mani, 2013], aMMSB with Poisson likelihood and two other baselines. We also compared the ATTAS to a variant of the proposed model that does not account for social influence, and instead model the evolution of vertex latent state vectors $\mathbf{h}_v^{(t)}$ with random walk Markov chains. This model is known as RW. The performance of the models are compared based on the ROC curves and the AUC metrics in Figure 5.3 and Table 5.1 respectively. The proposed ATTAS model significantly outperformed the dM^{3}SB which shares similar interpretable structure as our model. While the LFP model performed better than the ATTAS, it does not provide easily interpretable community structures to explain its predictions in the way that our model does. Furthermore, the computational complexity of LFP grows exponentially with respect to the number of time points T and the number of latent features K [Heaukulani and Ghahramani, 2013], rendering the model un-scalable. The Dirichlet-Multinomial baseline, which assigns predictive probability to a pair of vertices in proportion to the number of observed edges between the pair in the training data, appears to have high AUC scores. However, it cannot predict non-trivial edges (i.e. historically infrequent interactions) well as shown by its relatively flat ROC curves in Figure 5.3. For example, in the second fold of the ENRON data, the slope of Dirichlet-Multinomial's ROC curve at 0.2 False Positive Rate (x-axis, approximately where the blue and green/red curves crossed) is near zero while the slopes of ATTAS/RW ROCs are significantly higher.

The high AUCs of Dirichlet-Multinomial baseline indicates that there are many edges in the held-out data set that can be easily predicted by simply checking if a pair of vertices had previously interacted in the training data set. However, such trivial predictions are uninteresting and do not require sophisticated statistical models. This motivates *Challenge 2*, which requires the models to predict non-trivial edges. The maximum F1-scores and AUCs of the top-performing models from *Challenge 1* (LFP and ATTAS) in *Challenge 2* are reported in Table 5.2. The maximum F1 scores were computed by scanning over all possible precision/recall thresholds. In *Challenge 2*, the Dirichlet-Multinomial baseline has a F1-score of 0 and AUC of 0.5 by construction. The result indicates that in tasks where predicting non-obvious edges

are crucial, our models are preferable. This is especially true when interpretability of the predictions is important because of the built-in community detection mechanism in our proposed ATTAS model.

The results in Table 5.1 also show that modeling social influence is important and highly beneficial for link predictions. The ATTAS model that captures the social influence effect significantly outperformed the RW model with simple random walk latent process in the ENRON and COLLEGE data sets, while achieving essentially the same performance as RW in TRADING.

	ENRON	**TRADING**	**COLLEGE**
ATTAS	0.857±0.003	0.965±0.001	0.823±0.004
RW	0.829±0.012	0.970±0.001	0.811±0.017
dM^{3}SB	0.730±0.012	0.968±0.002	0.656±0.008
LFP	0.889±0.005	0.982±0.001	0.870±0.012
aMMSB	0.799±0.006	0.731±0.002	0.742±0.042
Dirich-Mult.	0.828±0.006	0.946±0.006	0.882±0.019
Equi-prob.	0.479±0.005	0.553±0.012	0.510±0.056

Table 5.1: 3-fold cross-validated AUCs for *Challenge 1*.

		ENRON	**TRADING**
F1	ATTAS	0.124±0.010	0.171±0.014
	LFP	0.102±0.014	0.160±0.012
AUC	ATTAS	0.806±0.012	0.881±0.011
	LFP	0.806±0.005	0.942±0.002

Table 5.2: 3-fold cross-validated F1 and AUCs for *Challenge 2*.

We describe the data sets used in the link prediction experiments in the following paragraphs.

ENRON [Shetty and Adibi, 2004]: 4 months of ENRON e-mail communication networks were used in the experiments. Two persons/vertices in the networks share an edge in a particular month if there is at least 1 e-mail communication between the pair in that month. There are 126 vertices in the first network, and the number of vertices increased to 138 by the end of the fourth month. We assumed the number of communities to be 3 in the predictive experiments.

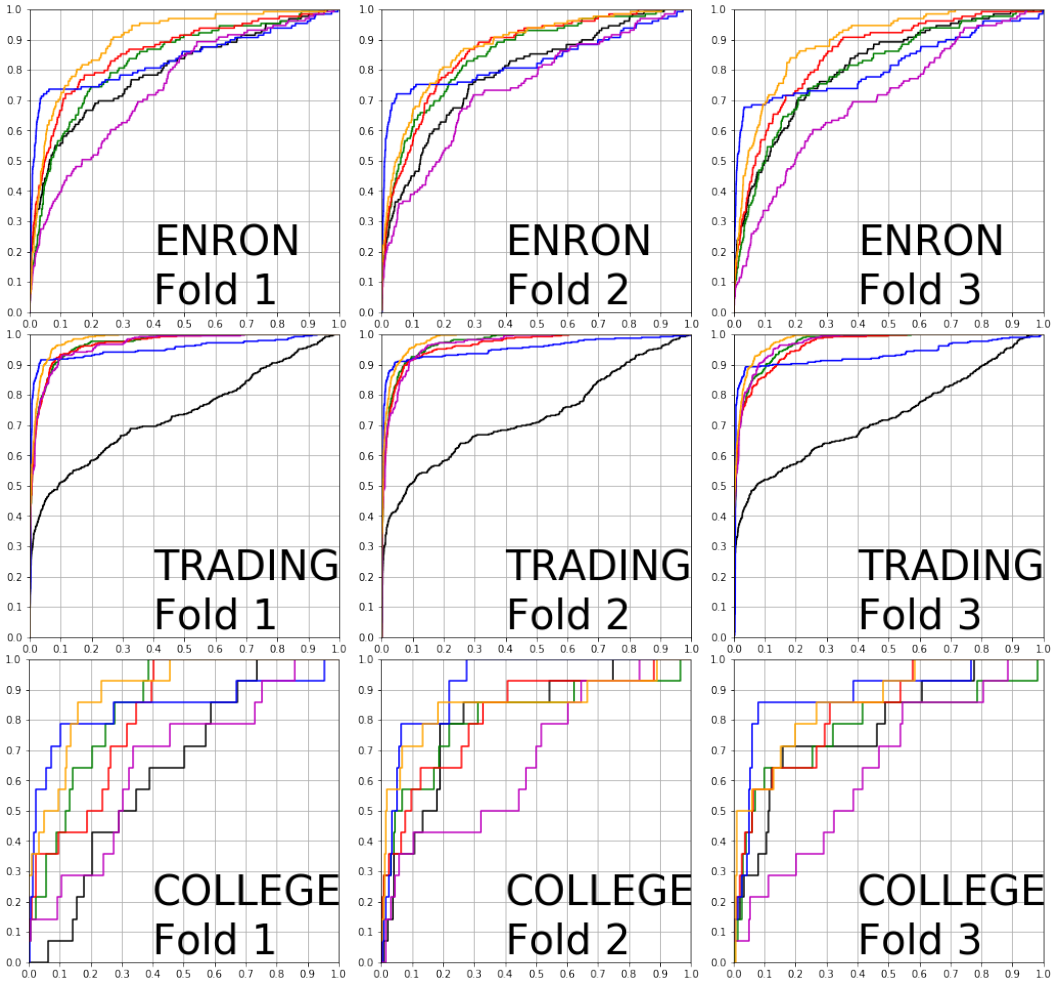

Figure 5.3: The sub-figures show the ROC curves of ATTAS (red), RW (green), dM^{3}SB (magenta), LFP (orange), aMMSB (black) and Dirichlet-Multinomial (blue) on different folds of the data.

TRADING [Economics Web Institute]: 4 years of the international trading networks from 1970 to 1973 were used in the experiments. Two countries/vertices in the networks share an edge in a particular year if the amount of trade between the two countries in the year is non-zero. There are 126 vertices in the network at 1970, and the number increased to 134 by the end of 1973. We assumed the number of communities to be 4 in the predictive experiments.

COLLEGE [Van de Bunt et al., 1999]: This data set consists of 7 snapshots of friendship networks between university freshmen in a Dutch university. The original data set consists of pair-wise friendliness scores between the freshmen surveyed, with

score of -1 indicating animosity, to +3 indicating a best friend. We pre-processed the 7 snapshots such that two freshmen/vertices share an edge only if both rated each other with a positive score in the same period. There are 4 vertices in the first network, and the number increased to 31 by the end of the seventh period. We assumed the number of communities to be 3 in the predictive experiments.

The following paragraphs describe the details of the models compared in the 3-fold cross-validated link prediction experiment.

ATTAS, RW: The proposed dynamic network model with the ATTAS/random process. The models were trained for $50,000$ iterations and given 5 random restarts per experiment. The predictive probability of seeing an edge between vertex i and j were computed using 500 Monte Carlo samples drawn from the fitted variational distributions.

dM^{3}SB: The dynamic mixture of mixed-membership stochastic blockmodel proposed in [Ho et al., 2011]. The model hyper-parameters were selected using the BIC grid search procedure proposed in [Ho et al., 2011]. The hyper-parameter grids for the number of mixture component and the number of community are $[2,3,4,5]$ and $[3,4,5,6]$ respectively. We performed 5 random restarts per configuration. The model was also modified to leave out the links in the hold-out set.

LFP: The dynamic latent feature propagation model proposed in [Heaukulani and Ghahramani, 2013]. The number of latent features K was set to 10 and the MCMC inference procedure was performed with the hyperparameters suggested in the original paper.

aMMSB: This is the assortative MMSB proposed in [Gopalan et al., 2012] with Poisson likelihood. All the edges observed in the training data set were aggregated and modelled as counts. The models were trained to convergence and given 5 random restarts per experiment. The predictive probability is the probability of observing at least one edge between two vertices conditioning on the training data.

Dirichlet-Mult.: The Dirichlet-multinomial distributions over edges is equivalent to an Infinite Relational Model [Kemp et al., 2006] where each pair of vertices is in its own cluster. Please refer to [Williamson, 2016] for details.

Equi-probable: Equi-probable links baseline [Williamson, 2016]. This baseline assumes the probability of observing an edge between two vertices is $\frac{1}{N\times(N-1)}$, where N is the number of vertices in the training data.

5.6.3 Community Detection

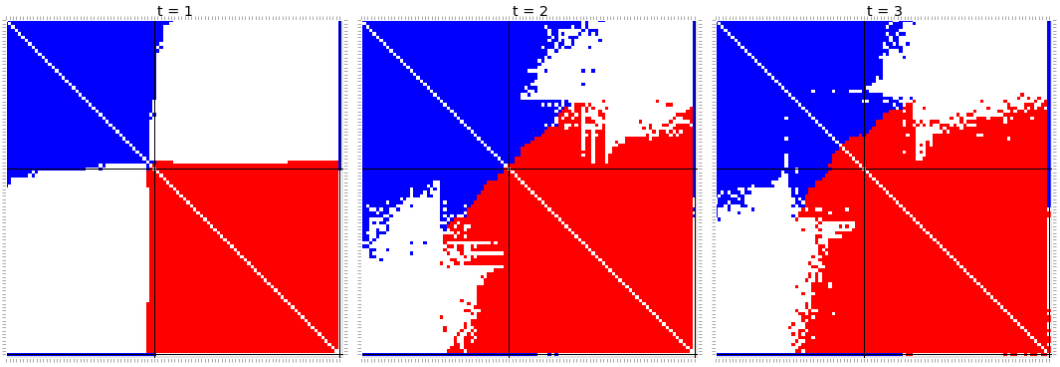

Figure 5.4: This figure shows the adjacency matrices of the US Congress data set with the edges coloured according to the inferred community types.

We demonstrate that the proposed ATTAS model can infer meaningful community structure in real temporal networks by fitting the model with $M = 2$ to a sequence of 3 temporal networks created from the first 9 months of the 109th US Congress voting records [US Congress]. The data set was divided into 3 three-month periods. Two senators share an edge within each of the periods if they casted the same votes for at least 50% of the bills voted on within the period.

Figure 5.4 shows the adjacency matrices of the temporal networks. The edges are colored according to their inferred types and the vertices (rows and columns) are sorted according to the senators' party affiliations, with the black lines separating the Democrats (left/top of the black lines) from the Republicans. A single unaffiliated senator is represented in the bottom row/right-most column. Within each party, the senators are sorted according to their relative frequencies of participating in each

edge type.

The homogeneous edge colors in the top-left and bottom-right quadrants of the adjacency matrices clearly show that ATTAS was able to infer meaning community structures that reflect the reality (i.e., party affiliations). We also observed that the sizes of the inferred communities changed over time, reflecting the changing positions of senators with different views from their fellow party members (e.g., Democrats with more conservative views) and the dominance of the Republican party during the Bush administration. It is interesting to notice that the voting patterns of some senators ed well with other senators from both parties. They are highly represented in both communities and are likely to form edges with both Republicans and Democrats, acting as hubs in the networks.

5.7 Discussions

In this chapter, I presented an edge exchangeable model for temporal networks that can model sparsity, community structures and social influences in the networks. The proposed model is also less computationally demanding compared to many existing models. The experiments show that the proposed ATTAS model performs well in link predictions and can recover meaningful community structures. The ATTAS model is particularly suitable for tasks where predicting non-obvious edges are crucial and when interpretability of the predictions is important, as its predictions can be explained by the built-in community detection mechanism. The work presented in this chapter can be extended in several different directions in future works. These future directions are discussed in details in Chapter 6.

6

Conclusion and Future Works

In this thesis, I presented and discussed three probabilistic models and scalable learning algorithms that account for temporal and network structured dependence in data sets. The proposed models and algorithms extended the existing literatures on probabilistic modelling by either allowing existing probabilistic models to scale up using novel variational inference algorithms, or by enriching the class of probabilistic models for dependent data through incorporating novel structures in the models.

In Chapter 3, I discussed the factorial hidden Markov model and the limitation of the existing inference algorithm in scaling up to very long sequential data. To address the scalability issue, a structured variational inference algorithm that leverages modern stochastic optimisation and deep learning techniques was proposed. The proposed algorithm allows factorial hidden Markov models with a large number of hidden Markov chains to be applied to very long sequences. The algorithm was then verified and validated through extensive experiments on real and simulated data sets. It may be possible to extend the proposed algorithms in two different directions in future works. Firstly, the proposed variational inference algorithm imposes independence assumption among the posterior Markov chains while they are not independent in the true posterior distribution. One interesting future direction to explore is to extend the scalable variational inference algorithm in order to capture the posterior dependence structure using flexible neural network density models. Secondly, it

will be useful to explore the development of scalable variational inference algorithm for factorial hidden Markov models with non-linear likelihood functions, such as the Gaussian likelihood function where the mean parameter is parameterised as a non-linear function of the factorial latent states. The non-linear factorial hidden Markov model can capture richer structure in the sequential data set than the existing model while retaining the the model's interpretability. However, the existing scalable inference algorithms, including the one proposed, are not readily applicable to the non-linear model. Therefore, it is useful to extend the existing scalable inference algorithms such that they can be applied to factorial hidden Markov models with non-linear parameters.

In Chapter 4, I presented a novel Gaussian process model for semi-supervised learning on data sets with network structure. The proposed probabilistic model accounts for information in the node-specific covariates and the dependence structure among the nodes as determined by the network structure, in order to predict the node labels. Experiments on benchmark data sets showed that the proposed model is competitive with the state-of-the-art deep learning models despite its simplicity, and is advantageous when the number of labelled nodes is small. As discussed in Chapter 4, it is possible to extend the proposed model in two interesting directions in future works. First of all, future research can investigate the extension of the covariance function of the proposed model through its spectral view. It may be possible to design richer covariance structure with better regularising property by imposing additional smoothness constraint on the spectral filters that correspond to the Gaussian process models. Another interesting direction is to explore the hierarchical extension of the proposed model, such that multi-hop long-range information of the network can be exploited.

In Chapter 5, I presented an edge exchangeable model for temporal networks that can model sparsity, community structures and social influences in temporal networks simultaneously. The work presented in the chapter can be extended in two directions in the future. Firstly, the computational complexity of the varia-

tional inference algorithm can be further improved through further approximations. The computational complexity of the proposed algorithm is bottle-necked by the computations of the normalising constants of the multivariate logistic normal likelihood functions. Computing the normalising constants requires summations over the exponentiated latent states of all the vertices present in the networks. The complexity for computing the normalising constants is therefore linear with respect to the number of vertices. The operation becomes expensive for networks with tens of thousands of vertices. In this regard, it is possible to explore the adaptation of techniques for approximate normalising constants in classifiers with large number of classes [Titsias, 2016, Botev et al., 2017, Bengio et al., 2003] as well as word embedding models for large corpus to the proposed dynamic network models [Mnih and Kavukcuoglu, 2013, Bamler and Mandt, 2017], such that the algorithm can scale to temporal network data sets with large number of vertices. Secondly, the proposed model can be extended to allow the 'states' of the vertices throughout their lifetime to be parsed. Temporal social networks are composed of heterogeneous sets of vertices, in which their participation in the networks evolve over time depending on the states of the vertices at the given time points. Inferring the states of the vertices through different time periods can reveal important insights to the overall networks, communities within the networks and individual vertices. One possible approach to infer the temporal trajectories of the vertices' states is the adaptation of the switching linear dynamical system state-space to the dynamic network model, such that the parameters for the vertices' transition distributions at different time points are selected from a dictionary based on their respective latent states at the corresponding time. The inferred latent states can be used to parse the vertices' properties and cluster them accordingly.

As discussed in various sections of this thesis, data sets in the real-world often exhibit rich dependent structures that can be exploited by probabilistic models. The research presented in this thesis explored the modelling of dependent structures in the temporal and network dimensions within the probabilistic modelling framework while proposing practical variational inference algorithms that enable the applica-

tions of the models to non-trivial data sets. One key limitation that often inhibits the exploration and application of probabilistic models to dependent data is the computational difficulty that is inherent to the models that can capture the dependency structure of the data sets. I hope that with the development of more efficient computational methods within the machine learning and statistics community, and the availability of cheaper computing power, the developments and applications of probabilistic models for dependent data will become more accessible.

Appendix A

Variational Inference Algorithm for the Dynamic Network Models

In this appendix, I present the derivation for the variational inference algorithm for the dynamic network models proposed in Chapter 5. The variational inference algorithm lower bounds the model's marginal log-likelihood with the evidence lower bound (ELBO) function in Equation A.1, and learn the variational parameters β and model parameters θ jointly by maximising the ELBO objective function with respect to these parameters.

$$
\begin{aligned}
\mathcal{L}(\beta,\theta) = & \sum_{t=1}^{T}\sum_{i=1}^{|\mathcal{E}^{(t)}|}[\mathbb{E}_q[\ln P(v_i^{(t)}|c_i^{(t)},\mathbf{h}^{(t)},\mathbf{h}_z^{(t)})]+\mathbb{E}_q[\ln P(v_i'^{(t)}|c_i^{(t)},\mathbf{h}^{(t)},\mathbf{h}_z^{(t)})]] \\
& + \sum_{t=1}^{T}[\mathbb{E}_q[\ln P(L_t|e^{\lambda_t})] + \sum_{i=1}^{|\mathcal{E}^{(t)}|}(\mathbb{E}_q[\ln P(c_i^{(t)}|\mathbf{k}^{(t)})]] + \mathbb{E}_q[\ln p_\theta(\mathbf{k}^{(1)})] \\
& + \mathbb{E}_q[\ln p_\theta(\lambda_1)] + \sum_{t=2}^{T}[\mathbb{E}_q[\ln p_\theta(\mathbf{k}^{(t)}|\mathbf{k}^{(t-1)})] + \mathbb{E}_q[\ln p_\theta(\lambda_t|\lambda_{t-1})]] \\
& + \sum_{i\in V^{(T)}}[\mathbb{E}_q[\ln p_\theta(h_i^{(\tau_i)})] + \sum_{t=\tau_i+1}^{T}\mathbb{E}_q[\ln p_\theta(h_i^{(t)}|\mathbf{h}^{(t-1)},\mathbf{h}_z^{(t-1)})]] - \mathbb{E}_q[\ln q_\beta].
\end{aligned}
\tag{A.1}
$$

The ELBO in Equation A.1 is analytically intractable due to the expectations with respect to the logistic normal log-normalizing constants in the equation. To allow tractable computations, we introduce a further approximation for the expectations over the logistic normal log-normalizing constant for the vertex log-likelihood. The exact same derivation applies to the edge cluster indicator log-likelihood.

The additional layer of approximation induces a further lower bound on the ELBO, and introduces additional variational parameters $\zeta_{m,t}$. The lower bound can be written as

$$\begin{aligned}
&\mathbb{E}_q[\ln P(v_i^{(t)}|c_i^{(t)}=m,\mathbf{h}^{(t)},\mathbf{h}_z^{(t)})] \\
&= \mathbb{E}_q[h_{j,m}^{(t)}] - \mathbb{E}_q[\ln \sum_{h\in\mathbf{h}_z^{(t)}} e^{h_m} + \sum_{h\in\mathbf{h}^{(t)}} e^{h_m}] \\
&\geq \mathbb{E}_q[h_{j,m}^{(t)}] - \frac{\mathbb{E}_q[\sum_{h\in\mathbf{h}_z^{(t)}} e^{h_m} + \sum_{h\in\mathbf{h}^{(t)}} e^{h_m}]}{\zeta_{m,t}} - \ln\zeta_{m,t} + 1. \qquad \text{(A.2)}
\end{aligned}$$

The optimal values for $\zeta_{m,t}$ can be solved for by setting the derivative of the lower bound in Equation A.2 w.r.t. $\zeta_{m,t}$ to 0:

$$\frac{\mathbb{E}_q[\sum_{h\in\mathbf{h}_z^{(t)}} e^{h_m} + \sum_{h\in\mathbf{h}^{(t)}} e^{h_m}]}{\zeta_{m,t}^2} - \frac{1}{\zeta_{m,t}} = 0 \qquad \text{(A.3)}$$

$$\zeta_{m,t,opt} = \mathbb{E}_q[\sum_{h\in\mathbf{h}_z^{(t)}} e^{h_m} + \sum_{h\in\mathbf{h}^{(t)}} e^{h_m}] \qquad \text{(A.4)}$$

With the approximation introduced above, it is possible to derive analytic coordinate ascent update formula for the variational parameters $c_i^{(t)}$. The coordinate ascent update equation for the variational parameters of $c_i^{(t)}$ is derived by singling

out all the terms involving the variational parameters related to $c_i^{(t)}$ in the variational lower bound, and set their derivatives to 0.

Denoting $\pi_i^{(t)} = [\pi_{i,1}^{(t)}, \ldots, \pi_{i,M}^{(t)}]^T$ as the categorical variational parameters corresponding to the latent variable $c_i^{(t)}$ and isolating all the terms in the lower bound that contains $\pi_{i,m}^{(t)}$ results in the following expression

$$\begin{aligned}
&\pi_{i,m}^{(t)}[\mathbb{E}_q[h^{(t)}_{v_i'^{(t)},m}] + \mathbb{E}_q[h^{(t)}_{v_i^{(t)},m}] - 2\frac{\mathbb{E}_q[\sum_{h\in\mathbf{h}_z^{(t)}} e^{h_m} + \sum_{h\in\mathbf{h}^{(t)}} e^{h_m}]}{\zeta_{m,t}} - 2\ln\zeta_{m,t} + 2] \\
&+ \pi_{i,m}^{(t)}[\mathbb{E}_q[[k_m^{(t)}] - \frac{\sum_l \mathbb{E}_q[e^{k_m^{(t)}}]}{\zeta_{k,t}} - \ln\zeta_{k,t} + 1] - \pi_{i,m}^{(t)}\ln\pi_{i,m}^{(t)}.
\end{aligned} \tag{A.5}$$

Setting the derivative of the expression above w.r.t $\pi_{i,m}^{(t)}$ and solve for the optimal $\pi_{i,m}^{(t)}$ gives

$$\begin{aligned}
\ln\pi_{i,m}^{(t)} \propto &\mathbb{E}_q[h^{(t)}_{v_i^{(t)},m}] - 2\frac{\mathbb{E}_q[\sum_{h\in\mathbf{h}_z^{(t)}} e^{h_m} + \sum_{h\in\mathbf{h}^{(t)}} e^{h_m}]}{\zeta_{m,t}} - 2\ln\zeta_{m,t} + 2] \\
&+ [\mathbb{E}_q[h^{(t)}_{v_i'^{(t)},m}] + [\mathbb{E}_q[[k_m^{(t)}] - \frac{\sum_l \mathbb{E}_q[e^{k_m^{(t)}}]}{\zeta_{k,t}} - \ln\zeta_{k,t} + 1] - 1.
\end{aligned} \tag{A.6}$$

The optimal $\pi_{i,m}^{(t)}$ is the normalized exponent of the expression in Equation A.6.

Bibliography

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

David J Aldous. Representations for partially exchangeable arrays of random variables. *Journal of Multivariate Analysis*, 11(4):581–598, 1981.

Evan Archer, Il Memming Park, Lars Buesing, John Cunningham, and Liam Paninski. Black box variational inference for state space models. *arXiv preprint arXiv:1511.07367*, 2015.

Andreas Argyriou, Mark Herbster, and Massimiliano Pontil. Combining graph laplacians for semi–supervised learning. In *Advances in Neural Information Processing Systems*, pages 67–74, 2006.

James Atwood and Don Towsley. Diffusion-convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1993–2001, 2016.

Maria-Florina Balcan, Steve Hanneke, and Jennifer Wortman Vaughan. The true sample complexity of active learning. *Machine learning*, 80(2-3):111–139, 2010.

Robert Bamler and Stephan Mandt. Dynamic word embeddings. In *International Conference on Machine Learning*, pages 380–389, 2017.

Albert-László Barabási and Márton Pósfai. *Network science*. Cambridge University Press, Cambridge, 2016. ISBN 9781107076266 1107076269. URL `http://barabasi.com/networksciencebook/`.

David Barber. *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

David Barber, A Taylan Cemgil, and Silvia Chiappa. *Bayesian time series models*. Cambridge University Press, 2011.

Matthias Bauer, Mark van der Wilk, and Carl Edward Rasmussen. Understanding probabilistic sparse gaussian process approximations. In *Advances in neural information processing systems*, pages 1533–1541, 2016.

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research*, 7(Nov):2399–2434, 2006.

Yoshua Bengio, Jean-Sébastien Senécal, et al. Quick training of probabilistic neural nets by importance sampling. In *AISTATS*, 2003.

David M Blei and John D Lafferty. Dynamic topic models. In *Proceedings of the 23rd international conference on Machine learning*, pages 113–120. ACM, 2006.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112 (518):859–877, 2017. doi: 10.1080/01621459.2017.1285773. URL `https://doi.org/10.1080/01621459.2017.1285773`.

Charles Blundell, Jeff Beck, and Katherine A Heller. Modelling reciprocating relationships with hawkes processes. In *Advances in Neural Information Processing Systems*, pages 2600–2608, 2012.

Aleksandar Botev, Bowen Zheng, and David Barber. Complementary sum sampling for likelihood approximation in large scale classification. In *Artificial Intelligence and Statistics*, pages 1030–1038, 2017.

Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

M Yu Byron, John P Cunningham, Gopal Santhanam, Stephen I Ryu, Krishna V Shenoy, and Maneesh Sahani. Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. In *Advances in neural information processing systems*, pages 1881–1888, 2009.

Diana Cai, Trevor Campbell, and Tamara Broderick. Edge-exchangeable graphs and sparsity. In *Advances in Neural Information Processing Systems*, pages 4242–4250, 2016.

François Caron and Emily B Fox. Sparse graphs using exchangeable random measures. *arXiv preprint arXiv:1401.1137*, 2014.

Jonathan Chang. lda: Collapsed gibbs sampling methods for topic models, 2011.

Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

Wei Chu, Vikas Sindhwani, Zoubin Ghahramani, and S Sathiya Keerthi. Relational learning with gaussian processes. In *Advances in Neural Information Processing Systems*, pages 289–296, 2007.

Fan R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

Harry Crane. *Probabilistic Foundations of Statistical Network Analysis*. Chapman and Hall/CRC, 2018.

Harry Crane and Walter Dempsey. Edge exchangeable models for network data. *arXiv preprint arXiv:1603.04571*, 2016.

Andreas Damianou and Neil Lawrence. Deep gaussian processes. In *Artificial Intelligence and Statistics*, pages 207–215, 2013.

Gautam Dasarathy, Robert Nowak, and Xiaojin Zhu. S2: An efficient graph based active learning algorithm with application to nonparametric classification. In *Conference on Learning Theory*, pages 503–522, 2015.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3844–3852, 2016.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

Adji B Dieng, Dustin Tran, Rajesh Ranganath, John Paisley, and David M Blei. The chidivergence for approximate inference. *arXiv preprint arXiv:1611.00328*, 2016.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159, 2011.

Daniele Durante and David Dunson. Bayesian logistic gaussian process models for dynamic networks. In *Artificial Intelligence and Statistics*, pages 194–201, 2014.

David Duvenaud. *Automatic Model Construction with Gaussian Processes*. PhD thesis, Computational and Biological Learning Laboratory, University of Cambridge, 2014.

David K Duvenaud, Hannes Nickisch, and Carl E Rasmussen. Additive gaussian processes. In *Advances in neural information processing systems*, pages 226–234, 2011.

Economics Web Institute. Economics web institute world trade. `http://www.economicswebinstitute.org/worldtrade.htm`, 2006. Accessed: 2017-05-18.

Gal Elidan. Copulas in machine learning. In Piotr Jaworski, Fabrizio Durante, and Wolfgang Karl Härdle, editors, *Copulae in Mathematical and Quantitative*

Finance, Lecture Notes in Statistics, pages 39–60. Springer Berlin Heidelberg, 2013.

Kai Fan, Chunyuan Li, and Katherine A. Heller. A unifying variational inference framework for hierarchical graph-coupled HMM with an application to influenza infection. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, pages 3828–3834, 2016.

Mehrdad Farajtabar, Yichen Wang, Manuel Gomez Rodriguez, Shuang Li, Hongyuan Zha, and Le Song. Coevolve: A joint point process model for information diffusion and network co-evolution. In *Advances in Neural Information Processing Systems*, pages 1954–1962, 2015.

Seth R Flaxman, Yu-Xiang Wang, and Alexander J Smola. Who supported Obama in 2012?: Ecological inference through distribution regression. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 289–298. ACM, 2015.

Nicholas Foti, Jason Xu, Dillon Laird, and Emily Fox. Stochastic variational inference for hidden markov models. In *Advances in Neural Information Processing Systems*, pages 3599–3607, 2014.

Emily Beth Fox. *Bayesian nonparametric learning of complex dynamical phenomena*. PhD thesis, Massachusetts Institute of Technology, 2009.

Xin Gao and Peter X-K Song. Composite likelihood em algorithm with applications to multivariate hidden markov model. *Statistica Sinica*, pages 165–185, 2011.

Samuel Gershman, Matt Hoffman, and David Blei. Nonparametric variational inference. *arXiv preprint arXiv:1206.4665*, 2012.

Samuel J Gershman and Noah D Goodman. Amortized inference in probabilistic reasoning. In *Proceedings of the 36th Annual Conference of the Cognitive Science Society*, 2014.

Zoubin Ghahramani and Michael I Jordan. Factorial hidden markov models. *Machine learning*, 29(2-3):245–273, 1997.

Mark Girolami and Simon Rogers. Variational bayesian multinomial probit regression with gaussian process priors. *Neural Computation*, 18(8):1790–1817, 2006.

Anna Goldenberg, Alice X Zheng, Stephen E Fienberg, Edoardo M Airoldi, et al. A survey of statistical network models. *Foundations and Trends® in Machine Learning*, 2(2):129–233, 2010.

Prem K Gopalan and David M Blei. Efficient discovery of overlapping communities in massive networks. *Proceedings of the National Academy of Sciences*, 110(36): 14534–14539, 2013.

Prem K Gopalan, Sean Gerrish, Michael Freedman, David M Blei, and David M Mimno. Scalable inference of overlapping communities. In *Advances in Neural Information Processing Systems*, pages 2249–2257, 2012.

Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 2, pages 729–734. IEEE, 2005.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864. ACM, 2016.

Shaobo Han, Xuejun Liao, David B Dunson, and Lawrence Carin. Variational gaussian copula inference. *arXiv preprint arXiv:1506.05860*, 2015.

Creighton Heaukulani and Zoubin Ghahramani. Dynamic probabilistic models for latent feature propagation in social networks. In *ICML (1)*, pages 275–283, 2013.

James Hensman, Alexander G. de G. Matthews, and Zoubin Ghahramani. Scalable variational gaussian process classification. In *Proceedings of the Eighteenth*

International Conference on Artificial Intelligence and Statistics, AISTATS 2015, San Diego, California, USA, May 9-12, 2015, 2015a. URL `http://jmlr.org/proceedings/papers/v38/hensman15.html`.

James Hensman, Alexander G Matthews, Maurizio Filippone, and Zoubin Ghahramani. Mcmc for variationally sparse gaussian processes. In *Advances in Neural Information Processing Systems*, pages 1648–1656, 2015b.

Daniel Hernández-Lobato, José M Hernández-Lobato, and Pierre Dupont. Robust multi-class gaussian process classification. In *Advances in neural information processing systems*, pages 280–288, 2011.

Geoffrey E Hinton, Peter Dayan, Brendan J Frey, and Radford M Neal. The" wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214): 1158–1161, 1995.

Qirong Ho, Le Song, and Eric Xing. Evolving cluster mixed-membership blockmodel for time-evolving networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 342–350, 2011.

Peter D Hoff, Adrian E Raftery, and Mark S Handcock. Latent space approaches to social network analysis. *Journal of the american Statistical association*, 97(460): 1090–1098, 2002.

Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

Douglas N Hoover. Relations on probability spaces and arrays of random variables. *Preprint, Institute for Advanced Study, Princeton, NJ*, 2, 1979.

Katsuhiko Ishiguro, Tomoharu Iwata, Naonori Ueda, and Joshua B Tenenbaum. Dynamic infinite relational model for time-varying relational data analysis. In *Advances in Neural Information Processing Systems*, pages 919–927, 2010.

T Jaakkola and M Jordan. A variational approach to bayesian logistic regression models and their extensions. In *Sixth International Workshop on Artificial Intelligence and Statistics*, volume 82, page 4, 1997.

Svante Janson. On edge exchangeable random graphs. *arXiv preprint arXiv:1702.06396*, 2017.

Matthew Johnson, David K Duvenaud, Alex Wiltschko, Ryan P Adams, and Sandeep R Datta. Composing graphical models with neural networks for structured representations and fast inference. In *Advances in neural information processing systems*, pages 2946–2954, 2016.

Kwang-Sung Jun and Robert Nowak. Graph-based active learning: A new look at expected error minimization. In *Signal and Information Processing (GlobalSIP), 2016 IEEE Global Conference on*, pages 1325–1329. IEEE, 2016.

Charles Kemp, Joshua B Tenenbaum, Thomas L Griffiths, Takeshi Yamada, and Naonori Ueda. Learning systems of concepts with an infinite relational model. In *AAAI*, volume 3, page 5, 2006.

Hyun-Chul Kim and Zoubin Ghahramani. Bayesian gaussian process classification with the em-ep algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):1948–1959, 2006.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Diederik P Kingma, Tim Salimans, and Max Welling. Improving variational inference with inverse autoregressive flow. *arXiv preprint arXiv:1606.04934*, 2016.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M Blei. Automatic differentiation variational inference. *arXiv preprint arXiv:1603.00788*, 2016.

Neil D Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *Advances in neural information processing systems*, pages 329–336, 2004.

Yingzhen Li and Richard E Turner. Rényi divergence variational inference. In *Advances in Neural Information Processing Systems*, pages 1073–1081, 2016.

Yingzhen Li, José Miguel Hernández-Lobato, and Richard E Turner. Stochastic expectation propagation. In *Advances in Neural Information Processing Systems*, pages 2323–2331, 2015a.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015b.

M. Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

Scott Linderman and Ryan Adams. Discovering latent network structure in point process data. In *International Conference on Machine Learning*, pages 1413–1421, 2014.

Scott W Linderman, Andrew C Miller, Ryan P Adams, David M Blei, Liam Paninski, and Matthew J Johnson. Recurrent switching linear dynamical systems. *arXiv preprint arXiv:1610.08466*, 2016.

James Lloyd, Peter Orbanz, Zoubin Ghahramani, and Daniel M Roy. Random function priors for exchangeable arrays with applications to graphs and relational data. In *Advances in Neural Information Processing Systems*, pages 998–1006, 2012.

László Lovász. *Large networks and graph limits*, volume 60. American Mathematical Society Providence, 2012.

Qing Lu and Lise Getoor. Link-based classification. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 496–503, 2003.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

Yifei Ma, Roman Garnett, and Jeff Schneider. σ-optimality for active learning on gaussian random fields. In *Advances in Neural Information Processing Systems*, pages 2751–2759, 2013.

Oisin Mac Aodha, Neill D.F. Campbell, Jan Kautz, and Gabriel J. Brostow. Hierarchical Subquery Evaluation for Active Learning on a Graph. In *CVPR*, 2014.

Dougal Maclaurin, David Duvenaud, Matthew Johnson, and Ryan P. Adams. Autograd: Reverse-mode differentiation of native Python, 2015. URL `http://github.com/HIPS/autograd`.

A Matthews. *Scalable Gaussian process inference using variational methods*. PhD thesis, PhD thesis. University of Cambridge, 2016.

Tom Minka et al. Divergence measures and message passing. Technical report, Technical report, Microsoft Research, 2005.

Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in neural information processing systems*, pages 2265–2273, 2013.

Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodolà, Jan Svoboda, and Michael M. Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5425–5434. IEEE Computer Society, 2017. ISBN 978-1-5386-0457-1. doi: 10.1109/CVPR.2017.576. URL `https://doi.org/10.1109/CVPR.2017.576`.

Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, Bernhard Schölkopf, et al. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends® in Machine Learning*, 10(1-2):1–141, 2017.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

Iain Murray, Ryan Prescott Adams, and David J. C. MacKay. Elliptical slice sampling. In *The Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, volume 9 of *JMLR: W&CP*, pages 541–548, 2010.

Roger B Nelsen. *An introduction to copulas*, volume 139. Springer Science & Business Media, 2013.

Yin Cheng Ng, Pawel M Chilinski, and Ricardo Silva. Scaling factorial hidden markov models: Stochastic variational inference without messages. In *Advances in Neural Information Processing Systems*, pages 4044–4052, 2016.

Krzysztof Nowicki and Tom A B Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455):1077–1087, 2001.

Peter Orbanz and Daniel M Roy. Bayesian models of graphs, arrays and other exchangeable random structures. *IEEE transactions on pattern analysis and machine intelligence*, 37(2):437–461, 2015.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.

Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Artificial Intelligence and Statistics*, pages 814–822, 2014.

Rajesh Ranganath, Dustin Tran, and David Blei. Hierarchical variational models. In *International Conference on Machine Learning*, pages 324–333, 2016.

Carl Edward Rasmussen and Christopher KI Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

Hugh Salimbeni and Marc Deisenroth. Doubly stochastic variational inference for deep gaussian processes. In *Advances in Neural Information Processing Systems*, pages 4588–4599, 2017.

Aliaksei Sandryhaila and Jose MF Moura. Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure. *IEEE Signal Processing Magazine*, 31(5):80–90, 2014.

Purnamrita Sarkar and Andrew W Moore. Dynamic social network analysis using latent space models. In *Advances in Neural Information Processing Systems*, pages 1145–1152, 2006.

Lawrence K Saul and Michael I Jordan. Exploiting tractable substructures in intractable networks. In *Advances in neural information processing systems*, pages 486–492, 1996.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

Daniel K Sewell, Yuguo Chen, et al. Latent space approaches to community detection in dynamic networks. *Bayesian Analysis*, 12(2):351–377, 2017.

Jitesh Shetty and Jafar Adibi. The enron email dataset database schema and brief statistical report. *Information sciences institute technical report, University of Southern California*, 4, 2004.

David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013.

Ricardo Silva, Wei Chu, and Zoubin Ghahramani. Hidden common cause relations in relational learning. In *Advances in neural information processing systems*, pages 1345–1352, 2008.

Alexander J Smola and Risi Kondor. Kernels and regularization on graphs. In *Learning theory and kernel machines*, pages 144–158. Springer, 2003.

Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, pages 1257–1264, 2006.

Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21, 1972.

Andreas Stuhlmüller, Jacob Taylor, and Noah Goodman. Learning stochastic inverses. In *Advances in neural information processing systems*, pages 3048–3056, 2013.

Zoltán Szabó, Bharath K Sriperumbudur, Barnabás Póczos, and Arthur Gretton. Learning theory for distribution regression. *The Journal of Machine Learning Research*, 17(1):5272–5311, 2016.

Y. W. Teh, D. Newman, and M. Welling. A collapsed variational Bayesian inference algorithm for latent Dirichlet allocation. In *Advances in Neural Information Processing Systems*, volume 19, 2007.

Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 4:2, 2012.

Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial Intelligence and Statistics*, pages 567–574, 2009.

Michalis Titsias. One-vs-each approximation to softmax for scalable estimation of probabilities. In *Advances in Neural Information Processing Systems*, pages 4161–4169, 2016.

Michalis Titsias and Miguel Lázaro-Gredilla. Doubly stochastic variational bayes for non-conjugate inference. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1971–1979, 2014.

Dustin Tran, David Blei, and Edo M Airoldi. Copula variational inference. In *Advances in Neural Information Processing Systems*, pages 3550–3558, 2015a.

Dustin Tran, Rajesh Ranganath, and David M Blei. The variational gaussian process. *arXiv preprint arXiv:1511.06499*, 2015b.

US Congress. 109th senate roll call data. `http://www.voteview.com/senate109.htm`, 2007. Accessed: 2017-04-22.

Gerhard G. Van de Bunt, Marijtje A. J. Van Duijn, and Tom A. B. Snijders. Friendship networks through time: An actor-oriented dynamic statistical network model. *Computational and Mathematical Organization Theory*, 5(2):167–192, 1999.

Mark van der Wilk, Carl Edward Rasmussen, and James Hensman. Convolutional gaussian processes. In *Advances in Neural Information Processing Systems*, pages 2845–2854, 2017.

S Vichy N Vishwanathan, Nicol N Schraudolph, Risi Kondor, and Karsten M Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 11(Apr):1201–1242, 2010.

Martin J Wainwright and Michael I Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2): 1–305, 2008.

Chong Wang and David M Blei. Variational inference in nonconjugate models. *Journal of Machine Learning Research*, 14(Apr):1005–1031, 2013.

Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*, pages 639–655. Springer, 2012.

Christopher KI Williams and David Barber. Bayesian classification with gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20 (12):1342–1351, 1998.

Christopher KI Williams and Carl Edward Rasmussen. Gaussian processes for machine learning. *the MIT Press*, 2(3):4, 2006.

Sinead A Williamson. Nonparametric network models for link prediction. *Journal of Machine Learning Research*, 17(202):1–21, 2016.

Andrew Gordon Wilson. *Covariance kernels for fast automatic pattern discovery and extrapolation with Gaussian processes*. PhD thesis, University of Cambridge, 2014.

Eric P Xing, Wenjie Fu, Le Song, et al. A state-space mixed membership blockmodel for dynamic network tomography. *The Annals of Applied Statistics*, 4(2):535–566, 2010.

Kevin S Xu and Alfred O Hero. Dynamic stochastic blockmodels for time-evolving social networks. *IEEE Journal of Selected Topics in Signal Processing*, 8(4): 552–562, 2014.

Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume*

48, ICML'16, pages 40–48. JMLR.org, 2016. URL http://dl.acm.org/citation.cfm?id=3045390.3045396.

Xiaojin Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005.

Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pages 912–919, 2003a.

Xiaojin Zhu, John D Lafferty, and Zoubin Ghahramani. Semi-supervised learning: From gaussian fields to gaussian processes. Technical report, Carnegie Mellon University, Computer Science Department, 2003b.

Rawya Zreik, Pierre Latouche, and Charles Bouveyron. The dynamic random subgraph model for the clustering of evolving networks. *Comput. Stat.*, 32(2): 501–533, June 2017. ISSN 0943-4062. doi: 10.1007/s00180-016-0655-5. URL https://doi.org/10.1007/s00180-016-0655-5.