FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Modeling and Analysis of Time-evolving Sparse Networks

PhD THESIS

submitted in partial fulfillment of the requirements for the degree of

Doctor of Technical Sciences

within the

Vienna PhD School of Informatics

by

Elahe Ghalebi
Registration Number 5478672024

to the Faculty of Informatics

at the TU Wien

Advisor: Radu Grosu, Technische Universität Wien (TU Wien)
Second advisor: Sinead A. Williamson, University of Texas at Austin (UT Austin)

External reviewers:
Amin Karbasi. Yale University (Yale), USA.
Trevor Campbell. University of British Columbia (UBC), CA.

Vienna, 1st June, 2020

Elahe Ghalebi

Radu Grosu

Declaration of Authorship

Elahe Ghalebi

I hereby declare that I have written this Doctoral Thesis independently, that I have completely specified the utilized sources and resources and that I have definitely marked all parts of the work - including tables, maps and figures - which belong to other works or to the internet, literally or extracted, by referencing the source as borrowed.

Vienna, 1st June, 2020

Elahe Ghalebi

To my love, Hamidreza.

Acknowledgements

I wish to express my deepest gratitude to my advisors, Radu Grosu and Sinead A. Williamson. It is whole-heartedly appreciated that your great advice for my study proved monumental towards the success of this study. This thesis would not have been possible without you and without the freedom and encouragement you have given me over the last three years. I want to thank Radu for all the advice and fully-funded conference trips. I would like to pay my special regards to Sinead, I can not thank you enough for everything you did for me since our collaboration began. You always knew where to give me a push in the right direction, but I never felt like you placed undue pressure on me. Even looking beyond research, these past two years have been some of the best years of my life and I know that a large part of that has been because you didn't just treat me as a student, but as a friend. So, thank you for all the advice, all the research and all the laughs. I won't forget it and I hope our collaboration continues forward.

I also want to thank all my collaborators over these years: Hamidreza Mahyar, Baharan Mirzasoleiman, Jure Leskovec, Graham W. Taylor, Manuel Gomez Rodriguez, Rouzbeh Hasheminezhad, Ali Nazemian, Ali Movaghar, and Hamid R. Rabiee –It is very enjoyable to work with all of you. I want to extend a special thank you to Baharan, for the hard-work and guidance during our collaboration.

I would also like to thank my committee members–Trevor Campbell, Amin Karbasi, Nobert Görtz, and Andreas Steininger–for providing me valuable feedback and a lot of encouragement. I know there are a million other responsibilities on your plates and I am truly honored that you all agreed to be on my committee. Your inputs and comments have been indispensable. Most importantly, thank you for letting me pass my defense, I literally could not have done without your backing.

I wish to thank all the people whose assistance was a milestone in the completion of this thesis. I am also grateful to the following university staff: Leo Mayerhorfer, Viktoria Vasalik, Barbara Wiesböck, Monika Di Angelo and Gerda Belkhofer. I would like to recognize the invaluable assistance that you all provided during my study.

I wish to acknowledge the unconditional support and great love of my friend, my love and my husband, Hamidreza, who has been a constant source of inspiration during the challenges of graduate school and life; I want to thank my parents, my in-law parents and my brother and sisters for supporting me spiritually throughout writing this thesis.

Kurzfassung

Ein wichtiges Problem, dass bei vielen Anwendungen in großen Netzwerken auftritt, ist beobachtete Daten zu verwenden, um auf die *Interaktionen* (z.B. Kanten) zwischen Individuen, welche zu einem komplexem gemeinsamen Verhalten führen, zu schließen. In unserer Arbeit spezialisieren wir uns auf sich *zeitlich-entwickelnde* Netzwerke, auf die grundlegende Annahmen über die Form oder Größe der zugrunde liegenden Netzwerktopologie nicht zutreffen. Mit zunehmender Notwendigkeit von Verfügbarkeit und Bedeutung zeitlicher Interaktionsdaten - wie der E-Mail-Kommunikation - wird es immer wichtiger, nicht nur ihre Datenstruktur zu analysieren, sondern auch ein Modell zur Erfassung ihrer Eigenschaften zu generieren. Bei der Analyse dieser Art von Netzwerken beobachten wir Interaktionen zwischen einer Reihe von Entitäten und möchten informative Darstellungen extrahieren, die nützlich sind, um Vorhersagen über die Entitäten und ihre Beziehungen zu treffen. Anschließend entwickeln wir *generische Modelle*, die durch Wahrscheinlichkeitsverteilungen erklärt werden, welche von den dynamischen Netzwerke beschrieben werden, passen solche Modelle an echte Netzwerke an und generieren daraus realistische Graphen.

Ein zentraler Schwerpunkt dieser Arbeit liegt auf Anwendungen, bei denen die Kanten ihres dynamischen Netzwerks möglicherweise nicht beschreiben sind. Stattdessen können wir die Dynamik von *stochastischen kaskadierenden Prozessen* (z. B. Informationstrübung, Virusausbreitung) beobachten, die über den nicht beschriebenen Teil des Netzwerkes auftreten. Weiters nehmen wir an, dass die abgeleiteten Kanten einen *Multigraph* konstruieren - in einem Multigraph kann es mehrere Kanten zwischen zwei Entitäten geben. Wir generieren ein Wahrscheinlichkeitsmodell für solche Daten, in dem eine Baum-Konstruktionsphase vorgeschlagen wird, um die wahrscheinlichsten Kanten des Netzwerks zu extrahieren. Die Verwendung eines solchen Modells ermöglicht es uns, das Netzwerk anhand der Beobachtungen aus dem stochastischen kaskadierenden Prozess abzuleiten.

Ein weiterer wichtiger Aspekt unserer Forschung ist die Untersuchung von *dünnbesetzten* realen Netzwerken. Die meisten realen Multigraphen, wie z. B. E-Mail Konversationsdatensätze, sind in der Praxis normalerweise schwachbesetzt und weisen einen modularen Aufbau mit den Verteilungen auf, die sich im Laufe der Zeit entwickeln. Wir nutzen den Vorteil von *nichtparametrischen* Modellen für Interaktions-Multigraphen, um die Dünnbesetzung von Kanten-austauschbaren Multigraphen mit den dynamischen Clustermustern, die dazu neigen die jüngsten Verhaltensmuster zu verstärken, zu kombinieren.

Abschließend wird das Problem der Identifizierung von *zentralen* Entitäten unter dem Gesichtspunkt der Information in einem Netzwerk behandelt, indem ein bekanntes Zentralitätsmaß in den Netzwerken berücksichtigt wird. Zusätzlich zu Algorithmen und theoretischen Analysen präsentieren wir eine umfangreiche empirische Bewertung unserer Ansätze anhand mehrerer synthetischer und realer Graphen.

Abstract

An important problem that arises in many applications of large networks is using observational data to infer the *interactions* (i.e., edges) between individuals (or vertices) which leads to complex collective behaviour. In our work, we focus on *time-evolving* networks where basic assumptions about the shape or size of the underlying network topology do not hold. As the availability and importance of temporal interaction data—such as email communication—increases, it becomes increasingly important to not only analyze their data structure but also to generate a model to capture their properties. When analyzing these kinds of networks, we observe interactions between a set of entities and we wish to extract informative representations that are useful for making predictions about the entities and their relationships. We then develop *generative models* that explain the probabilistic distributions governing the dynamic networks, fit such models to real networks, and use them to generate realistic graphs.

A central focus of this thesis is on applications where the edges of their dynamic network might not be observed, but instead we can observe the dynamics of *stochastic cascading processes* (e.g., information diffusion, virus propagation) occurring over the unobserved network. Further, we assume that the inferred edges construct a *multigraph*–there might be multiple edges between two entities in a multigraph. We generate a probabilistic model for such data, where a tree construction phase is proposed to extract the most probable edges of the network. Using such a model allows us to infer the network given observations from the stochastic cascading process.

Another important aspect of our research is the study of *sparse* real networks. most real-world multigraphs, such as email interaction datasets, are typically sparse in practice and they exhibit a modular structure with the distributions that evolve over time. We take the advantage of *nonparametric* models for interaction multigraphs which combines the sparsity of edge-exchangeable multigraphs coupled with the dynamic clustering patterns that tend to reinforce recent behavioral patterns.

Finally, the problem of identifying *central* entities from the information point of view in a network is addressed by considering a prominent centrality measure in the networks. In addition to providing algorithms and theoretical analyses, we present extensive empirical evaluation of our approaches on several synthetic and real-world graphs.

Contents

Kurzfassung **v**

Abstract **vii**

1 Introduction **1**

1.1 Motivation 1

1.2 Main contributions 5

1.2.1 Dynamic network model from partial observations 5

1.2.2 Bayesian nonparametric model for sparse temporal multigraphs . . 6

1.2.3 Compressive sensing framework for sparse recovery in networks . . 7

1.3 Summary of key contributions 8

1.4 Organization of this dissertation 8

1.5 Publications 8

2 Background and Survey **10**

2.1 Nonparametric Bayesian models 10

2.1.1 Bayesian modeling 10

2.1.2 Stochastic processes 11

Dirichlet process (DP) 11

Pitman-Yor process (PYP) 14

2.1.3 Latent class models 15

Chinese restaurant process (CRP) 16

Distance dependent Chinese restaurant process (ddCRP) 17
2.1.4 Inference 18
MCMC sampling 19
2.2 Statistical graph (network) models 20
2.2.1 Directed multigraphs 20
2.2.2 Undirected graphs 20
2.2.3 Bipartite graphs 20
2.3 Exchangeability 21
2.3.1 Exchangeability in networks 22
2.4 Graph modeling 23
2.4.1 Bayesian models for multigraphs 23
2.4.2 Models for dynamic graphs 28
2.4.3 Models for sparse graphs 30
2.4.4 Diffusion network inference 32
2.5 Network centrality 34
2.5.1 Network centrality measures 34
2.5.2 Detection of central vertices in networks 37
2.6 Applications 38
2.6.1 Link prediction 38
2.6.2 Influence maximization 38
2.6.3 Diffusion prediction 39

3 Dynamic Graph Model under Partial Observations 40

3.1 Nonparametric edge-exchangeable network model 42
3.2 Dynamic network inference model 43
3.2.1 Modeling observations from diffusion data 44
3.2.2 Updating latent model variables 45
3.2.3 On-line dynamic network inference 48
3.2.4 Modeling choices 48

3.3 Experiments . . . 49
3.4 Conclusion . . . 54

4 Nonparametric Model for Sparse Temporal Multigraphs 56
4.1 Mixture of Dirichlet Network Distributions . . . 57
4.2 Distance-dependent Chinese restaurant process . . . 58
4.3 Pitman-Yor process . . . 59
4.4 DNND: Dynamic Nonparametric Network Distribution . . . 60
4.4.1 Sparsity of the resulting graph . . . 61
Empirical evaluation of sparsity . . . 63
4.4.2 Discussion of modeling choices . . . 64
4.5 Inference . . . 64
4.6 Experiments . . . 66
4.6.1 Prediction of held-out edges . . . 69
4.6.2 Forecasting future interactions . . . 70
4.7 Conclusion . . . 71

5 Compressive Sensing Framework for Sparse Recovery in Networks 72
5.1 Problem definition . . . 73
5.2 Compressive sensing . . . 75
5.3 Compressive sensing over graphs . . . 76
5.4 CS-HiBet: Compressive Sensing of High Betweenness Centralities . . . 77
5.5 Complexity analysis . . . 81
5.6 Extension to closeness centrality . . . 83
5.7 Experiments . . . 84
5.7.1 Accuracy of CS-HiBet on identifying top-k central vertices . . . 85
5.7.2 Accuracy of CS-HiBet on rank prediction . . . 86
5.7.3 Correlation between our ego-closeness and global closeness . . . 88
5.8 Conclusion . . . 90

6 Conclusion and Future Research Directions **91**

6.1 Summary 92

6.2 Future research directions 92

List of Figures **97**

List of Tables **98**

List of Algorithms **99**

Bibliography **100**

CHAPTER 1

Introduction

1.1 Motivation

Many real-world systems can be structured and modeled by means of networks (or graphs), where entities of the system can act as vertices (or nodes), and the relations between these entities are demonstrated by links (or edges). The well-known examples of such networks include large communication systems (e.g., Internet, telephone network, world wide web), technological and transportation infrastructures (e.g., railroad and airline routes), biological systems (e.g., gene and/or protein interactions), information systems (e.g., network of citations between academic papers), and a variety of social interaction structures (e.g., online social networks) [SG05, KAS+13]. An important challenge in these systems is to understand and analyze the underlying structure that underpins these interactions.

Our main goal is to search for interesting measures that help us characterize and understand the underlying graph structure and the processes spreading over the networks. To this end, we develop models and algorithms that benefit from the identified structural network properties. The most interesting properties of large real-world networks are: community structure, sparsity, heterogeneity and dynamicity. Community[1] structure is commonly revealed in real-world networks, which implies the appearance of densely connected groups of vertices, with only sparser connections between these groups (e.g., based on user's work entitles, sport activities, etc.). In particular, entities are more likely

[1] In this thesis, community, group, cluster and module are used interchangeably.

to interact with the entities sharing the same interests than to entities affiliated with different communities. Such structural behaviour provides evidence for a modular view of the network's dynamics, with different clusters of interactions performing different functions with some degree of independence [HHJ03]. Another distinguishing property of networks is their degree of sparsity. Sparse networks only display interaction between a small fraction of vertex pairs, while dense networks have a number of interactions that is linear in the number of vertex pairs (or quadratic in the number of vertices). Moreover, the interactions in the networks typically express a type of heterogeneity, which we call asymmetric degree heterogeneity, meaning that some entities establish a large number of interactions to other entities, whereas they receive very few [New10]. Since many applications, ranging from neurological connectivity patterns [MHS$^+$14] to financial markets [NSRJ11] and social network analysis [ML10, AX09], contain massive sequences of multivariate time-stamped observations, analyzing time-dependent observations and modelling their underlying graph structure are essential. Extensively, in most applications, we observe that most real-world graphs exhibit dynamic processes that evolve over time, i.e., the number of vetrices and edges are growing (or shrinking) over time leading to structural changes in the graph [ARSM17]. Since dynamic network inference and modeling is a computationally expensive task, it is necessary to understand how the structure of these systems changes over a period of interest.

Development of generative probabilistic models has a rich history, and there exist several models that can generate models based on a priori assumption on the network structure [New18]. In order to capture the temporal dynamics of a graph structured data in the models, it is essential to understand the relationships between different entities and how these relationships evolve over time. Modelling complex distributions over dynamic graphs and then efficiently sampling from these distributions is challenging due to non-unique, high-dimensional nature of graphs and the complex, non-local dependencies that exists between edges in a given graph. Nonparametric models constitute an approach to model selection and adaptation, where the size of the models is allowed to grow as the data set grows. This is as opposed to parametric models where the number of parameters is fixed.

Most existing models over dynamic networks aim to model a fully observed network [HLL83, SN97, Wil16, CCB16] but in many real-world problems, the underlying network structure is not known. What is often known are partial observations of a stochastic cascading process that is spreading over the network. Consider for example modeling the spread of disease. We observe times at which vertices report infection by various

contagions; however not all vertices will report their infection. A fundamental problem, therefore, is to infer the underlying network structure from these partial observations. In recent years, there has been a body of research on inferring diffusion networks from vertex infection times. However, these efforts are mostly limited to inferring the set of most likely edges between infected vertices (assuming a fixed transmission model (see Section 3.3)), by formulating the problem as a submodular or convex optimization [ML10, GrS11, GRLK10, RS12]. More recently, there have been attempts to predict the transmission probabilities from infection times, either by learning vertex representations [BLG16], or by learning diffusion representations using the underlying network structure [BLG16, WZLC17, LMGM17]. However, it remains an open problem to provide a generative probabilistic model for the underlying network from partial observations. Thus, the following question becomes crucially important:

> Can evolving networks with an unbounded number of vertices be inferred and modeled without directly observing the interactions among their vertices?

We address this problem by providing a nonparametric dynamic network model—based on a mixture of coupled hierarchical Dirichlet processes–that can learn model structure based on partial observations (see Chapter 3).

Furthermore, many forms of interactions in real-world networks can be represented in terms of a multigraph—i.e., a graph where there can be multiple edges between two vertices. For example, vertices might correspond to individuals, with each edge representing an email between two individuals. In large-scale applications, such multigraphs are typically sparse, with the number of edges being small relative to the number of unconnected pairs of vertices. Edge-exchangeable models [CCB16, CD18] have been proposed as models for sparse multigraphs, and clustering-based edge-exchangeable models can capture the community structures of underlying graphs [Wil16]. Such models assume that we will see more edges and vertices in future, that makes them appropriate for time-evolving graphs. However, they assume that complex distributions over graphs are stationary; this assumption results in generation of graph models that are invariant to reordering the arrival times of edges. This is in contrast to the fact that most real-world multigraphs are dynamic, with the underlying distribution dynamically evolving. To this end, we wish to address the following question:

> Can we develop a generative model that can directly learn from graph-

> structural data, take the important properties of real networks (such as sparsity, heterogeneity, dynamicity and community structure) into account, and then paves a way to be useful in various applications, such as discovering new graph structures and completing evolving graphs?

We propose a dynamic nonparametric model for interaction multigraphs that combines the sparsity of edge-exchangeable multigraphs with dynamic clustering patterns that tend to reinforce recent behavioral patterns (see Chapter 4).

Lastly, we target the very important task of identifying *important vertices* of networks in many applications such as finding influencers in a social network [LRG13], and locating bottlenecked junctions/routers in a transportation network/the Internet [SG05]. This task can be tackled by analyzing the underlying network's behaviour using centrality measures [XW17, LZZS16, BK14]. Centrality refers to the identifiers that indicate the most important individuals (or vertices) in a network, and centrality measures quantify the role of vertices from different points of view. This concept has been around for decades and different kinds of measuring centrality have been proposed for a long time to quantitatively evaluate a vertex's importance from different perspectives [CRTVB07, Fre77, Sab66]. Some of them give consideration to local properties of the underlying network, while others, e.g., *betweenness* centrality and *closeness* centrality, reflect information about the global network structure. A good measure should usually include information from both global properties and local neighborhood, however many researchers consider these indicators together as a new one to identify central vertices in networks [CS09, HYL12]. In many cases, we are only interested in the k-highest centrality vertices rather than the centralities of all vertices in a network. This is reasonable since a vertex with a higher centrality is viewed as a more important vertex than a vertex with lower centrality. We will address the following question:

> How can we efficiently and accurately identify the top-k central vertices of a network without full knowledge of the network topological structure?

We tried to efficiently and accurately identify the top-k central vertices [KAS+13] by transforming this task to a sparse recovery problem and taking advantage of important centrality measures, i.e., betweenness centrality (see Chapter 5) .

1.2 Main contributions

In this thesis, we present a family of novel graph models and associated inference techniques, that allow us to model graphs that are sparse, have temporal variation and community structure, and may be partially observed. We also present a novel approach to identify important network components.

1.2.1 Dynamic network model from partial observations

This work, described in detail in Chapter 3, targets networks evolving in discrete time in which both vertices and edges can appear and disappear over time, such as dynamic networks of social interactions. Most existing dynamic network models assume a snapshot of the network at any particular time is conditionally independent from all previous snapshots given the current network states. Such an approach greatly simplifies the model and allows for tractable inference, but it may not be flexible enough to replicate certain observations from real network data, such as time durations' of edges, which are often inaccurately reproduced by models with hidden Markov dynamics. However, often times we can only observe the information diffusion over the network. While there have been efforts to infer networks based on such data, providing a *generative probabilistic model* that is able to identify the underlying time-varying network remains an open question.

Here we propose a novel online dynamic network inference framework, DYFERENCE, for providing non-parametric edge-exchangeable network models from partial observations. We build upon the non-parametric network model of [Wil16], namely MDND, that assumes that the network clusters into groups and then places a mixture of Dirichlet processes over the outgoing and incoming edges in each cluster while coupling the network using a shared discrete base measure. However, our framework is easily extended to arbitrary generative models replacing the MDND with other choices of latent representations, such as network models presented in [CCB16, CD16, HSM16]. Given a set of cascades spreading over the network, we process observations in time intervals. For each time interval we first find a probability distribution over the cascade diffusion trees that may have been involved in each cascade. We then calculate the marginal probabilities for all the edges involved in the diffusion trees. Finally, we sample a set of edges from this distribution and provide the sampled edges to a Gibbs sampler to update the model variables. In the next iteration, we use the updated edge probabilities provided by the model to update the probability distributions over edges supported by each cascade.

We continue the above iterative process until the model does not change considerably. Extensive experiments on synthetic and real-world networks show that DYFERENCE is able to track changes in the structure of dynamic networks and provides accurate online estimates of the time-varying edge probabilities for different network topologies. We also apply DYFERENCE for diffusion prediction and predicting the most influential vertices in Twitter and MemeTracker datasets, as well as bankruptcy prediction in a financial transaction network.

1.2.2 Bayesian nonparametric model for sparse temporal multigraphs

Many forms of social interaction can be represented in terms of a multigraph, where each individual interaction corresponds to an edge in the graph, and repeated interactions may occur between two individuals. For example, we might have multigraphs where the value of an edge corresponds to the number of emails between two individuals, or the number of packages sent between two computers. Recently, the class of edge exchangeable graphs [CCB16, CD18, Wil16] have been proposed for modeling networks as exchangeable sequences of edges. These models are able to capture many properties of large-scale social networks, such as sparsity, community structure, and power-law degree distribution. Being explicit models for sequences of edges, the edge-exchangeable models are appropriate for networks that grow over time: we can add more edges by expanding the sequence, and their nonparametric nature means that we expect to introduce previously unseen vertices as the network expands. However, their exchangeable nature precludes graphs whose properties change over time. In practice, the dynamics of social interactions tend to vary over time. In particular, in models that aim to capture community dynamics, the popularity of a given community can wax and wane over time.

We propose a new model for dynamic multigraphs, the Dynamic Nonparametric Network Distribution (DNND). The DNND uses a hierarchical clustering mechanism to capture interaction patterns within the multigraph, and uses distance-dependent Chinese Restaurant Processes (ddCRP) [BF11] to incorporate temporal dynamics by preferentially assigning edges to clusters and vertices that have been recently active. This dynamic mechanism, in combination with a Pitman-Yor process-distributed base measure, ensures that our model is appropriate for both sparse and dense multigraphs, with the degree of sparsity controlled by a single parameter. The increased flexibility allowed by our model leads to improved performance over both its exchangeable counterpart and a range of state-of-the-art dynamic network models.

1.2.3 Compressive sensing framework for sparse recovery in networks

In network analysis and mining, detection of important vertices from an *information flow* point of view has been a challenging problem in analysis of structural organization of networks [XW17, LZZS16, BK14]. *Betweenness centrality* [Fre77] is one of the prominent centrality measures expressing importance of a vertex within a network, in terms of the fraction of shortest paths passing through that vertex. Vertices with high betweenness centrality have significant impacts on the spread of influence and idea in social networks, the user activity in mobile phone networks, the contagion process in biological networks, and the congestion in communication networks. *Closeness centrality* [SGI17] is another fundamental measure for evaluating a vertex's importance and influence within a given network, based on its accessibility in the network. In many cases, we are only interested in the k-highest centrality vertices rather than the centralities of all vertices in a network. This is reasonable since a vertex with a higher centrality is viewed as a more important vertex than a vertex with lower centrality, and we are only interested in the most important vertices in many applications such as finding influencers in a social network [LRG13], and locating bottlenecked junctions/routers in a transportation network/the Internet [SG05]. In addition, community detection as one of the most well-known applications of the network centrality, utilizes the highest centrality vertices only [LCC16]. Suri [SN08] has defined the problem of finding top-k central vertices as; "for any given positive integer k, the top-k vertices in a network, based on a certain measure appropriate for the network".

First, we introduce ego-centric (local) metrics with very low computational complexity, that correlate well with the global centrality measures (*i.e.* betweenness and closeness centralities). Then, we propose a compressive sensing (CS) framework that can accurately and efficiently recover top-k central vertices in the network using the proposed local metrics. Computations of our ego-centric metric and the aggregation procedure are both carried out effectively in a distributed manner, using only local interactions between neighboring vertices.

The performance of the proposed method is evaluated by extensive simulations under several configurations on various types of synthetic and real-world networks. The experimental results demonstrate that the proposed local metrics correlate well with the global centrality measures and our CS-based framework outperforms the state-of-the-art methods for sparse recovery of every notion of centrality, with notable improvements.

1.3 Summary of key contributions

Inferring dynamic graphs is challenging primarily because it is not obvious how to simultaneously estimate both the underlying graph itself and the change in its structure over time. We first consider the case where we do not observe the underlying structure of the graphs (i.e., the edges connecting the vertices) and proposed a framework to iteratively extract the observations and generate a probabilistic model for the underlying network structure. Second, we focus on the fact that in most applications we are faced with time-dependent sequences of interactions. We propose a model for those interactions which construct a multigraph and are typically sparse, and then use the generative model of the corresponding temporal sparse multigraph for prediction tasks. Finally, we identify the top-k central vertices in a network enforcing the problem of network centrality where via sparse recovery solutions (i.e., compressive sensing).

1.4 Organization of this dissertation

In Chapter (1), we will first present an introduction of the thesis and summarize our main contributions related to this thesis. We will then briefly discuss the relevant terminologies as well as a comprehensive review of existing works on graphs models, diffusion network inference and network centrality (Chapter 2). In Section 2.6, we discuss concrete applications of proposed models. Later, the main contributions are widely described in parts (3), (4), and (5). Finally Chapter 6 summarizes the main results of the thesis and discusses future work.

1.5 Publications

Publications Covered in this Dissertation This dissertation covers material primarily from the following publications.

- Elahe Ghalebi, Hamidreza Mahyar, Radu Grosu, Graham W. Taylor, Sinead A. Williamson, “A Nonparametric Bayesian Model for Sparse Temporal Multigraphs”, submitted to NeurIPS 2020.
- Elahe Ghalebi, Hamidreza Mahyar, Radu Grosu, Graham W. Taylor, Sinead A. Williamson. “Sequential Edge-Clustering in Temporal Multigraphs”, Graph Representation Workshop of NeurIPS 2019, Vancouver, Canada.

- Elahe Ghalebi, Baharan Mirzasoleiman, Radu Grosu, Jure Leskovec, "Dynamic network model from partial observations", Advances in Neural Information Processing Systems, 9862-9872, NeurIPS 2018, Montreal, Canada.

- Hamidreza Mahyar, Rouzbeh Hasheminezhad, Elahe Ghalebi, Radu Grosu, H E. Stanley, "A Compressive Sensing Framework for Distributed Detection of High Closeness Centrality vertices in Networks", International Conference on Complex Networks and their Applications, 91-103, 2018.

- Hamidreza Mahyar, Rouzbeh Hasheminezhad, Elahe Ghalebi, Ali Nazemian, Radu Grosu, Ali Movaghar, Hamid R. Rabiee, "Compressive sensing of high betweenness centrality vertices in networks", Physica A: Statistical Mechanics and its Applications 497, 166-184, 2018.

- Elahe Ghalebi, Hamidreza Mahyar, Radu Grosu, Hamid R. Rabiee, "Compressive sampling for sparse recovery in networks", Proc of the 23rd ACM SIGKDD conference on Knowledge Discovery and Data mining (KDD), 13th International Workshop on Mining and Learning with Graphs, Halifax, Nova Scotia, Canada, 2017.

CHAPTER 2

Background and Survey

In this chapter, we provide background on nonparametric Bayesian learning, as well as latent class/feature models of nonparametric priors and the terminology used in this thesis. We also review related work that is relevant to each part of this thesis. We also discuss potential applications of analyzing and modelling graphs.

2.1 Nonparametric Bayesian models

To draw inferences about unobserved properties of the underlying network structure given some observations, it is beneficial to introduce probabilities into the model. The advantage of using probabilistic models in machine learning is to prevent overfitting and provide uncertainty estimates. Probabilistic modeling offers a simple and coherent framework which can be used as a basis to build general purpose tools for modelling and inference over graphs.

2.1.1 Bayesian modeling

Bayesian learning is a popular framework in statistics and machine learning due to its particular ability to embrace uncertainty, encode prior knowledge, and endow interpretability. Bayesian approaches have distinct advantages over non-Bayesian approaches, such as: (1) embracing uncertainty by explicitly representing, manipulating and mitigating it based on a solid theoretical foundation–probability–which makes it more robust facing real-world systems [Jor10], (2) encoding the prior knowledge of a problem while learn-

ing from observations, which makes it immune to overfitting–priors can be treated as regularization of the observed data–, and (3) admitting clear interpretability because of its clear and meaningful probabilistic structure. In particular, Bayesian nonparametric learning (BNL) has become known due to its modelling flexibility and representation power. The data in parametric models is assumed to be represented by a fixed, finite number of parameters and the parameters can not grow with sample size. Conversely, the number of parameters in a nonparametric model is a random variable which can grow with the sample size.

Definition 1. *(Bayesian nonparametric model [OT10]) A Bayesian nonparametric model (BNP) is a Bayesian model parameterized by an infinite number of parameters.*

In the context of this thesis, we consider Bayesian nonparametric models with a countable infinity of parameters. Such models can be characterized in terms of discrete measures on some parameter space.

2.1.2 Stochastic processes

Bayesian nonparametric processes use priors that take the form of stochastic processes. A stochastic process is a set of random indexed variables, where indices are derived from an index/parameter space and the variables are derived from a state space [XLZ19]. In this section, we review some important stochastic process priors used in this thesis.

Dirichlet process (DP)

The Dirichlet process (DP) [Fer73, Teh10] is the pioneer and foundation of BNL. The DP is a distribution over infinite-dimensional probability measures, which can in turn be used as priors in mixture models. The DP can be described first by introducing the Dirichlet distribution.

Definition 2. *(Dirichlet distribution) The Dirichlet distribution is a distribution over the $(K-1)$-dimensional simplex, i.e., it is a distribution over the relative values of K components, with the constraint that the components should be positive and that they sum up to 1, which is parameterized by a K-dimensional vector $(\alpha_1, \ldots, \alpha_K)$ such that $\alpha_k \geq 0, \forall k$ and $\sum_k \alpha_k > 0$,*

$$\pi = (\pi_1, \ldots, \pi_K) \sim Dir(\alpha_1, \ldots, \alpha_K) \sim \frac{\prod_{k=1}^{K} \Gamma(\alpha_k)}{\Gamma\left(\sum_{k=1}^{K} \alpha_k\right)} \prod_{k=1}^{K} \pi_k^{\alpha_k - 1} \tag{2.1}$$

where $\pi_k \geq 0$ *for all* k *and* $\sum_{k=1}^{K} \pi_k = 1$. *The expectation of this distribution is as follows:*

$$\mathbb{E}[(\pi_1, \ldots, \pi_K)] = \frac{(\alpha_1, \ldots, \alpha_K)}{\sum_k \alpha_k} \tag{2.2}$$

The Dirichlet distribution is conjugate to the multinomial distribution. Thus, if $\pi \sim \text{Dir}(\alpha_1, \ldots, \alpha_K)$ and x_n are *iid* samples from π, the posterior distribution becomes

$$\begin{aligned} p(\pi|x_1, \ldots, x_n) &\propto p(x_1, \ldots, x_n|\pi)p(\pi) \\ &= \left(\frac{\prod_{k=1}^{K} \Gamma(\alpha_k)}{\Gamma(\sum_{k=1}^{K} \alpha_k)}\right)\left(\frac{n!}{m_1! \ldots m_K!} \pi_1^{m_1} \ldots \pi_1^{m_K}\right) \\ &\sim \text{Dir}(\alpha_1 + m_1, \ldots, \alpha_K + m_K) \end{aligned} \tag{2.3}$$

where m_k is the counts of instances of $x_n = k$ in the data set. A sample from a Gaussian is a real-valued number whereas a sample from a Dirichlet distribution is a probability vector.

In a clustering model, we can use the Dirichlet distribution as part of a mixture model with K components (clusters). We say that a distribution f is a mixture of K component distributions $f_1, \ldots, f_K$, if

$$f(x) = \sum_{k=1}^{K} \lambda_k f_k(x) \tag{2.4}$$

where λ_k are the mixing weights and $\lambda_k > 0, \sum_k \lambda_k = 1$. The density function for a K-component finite mixture is given by

$$f(y|x; \theta_1, \ldots, \theta_K; \pi_1, \ldots, \pi_K) = \sum_{k=1}^{K} \pi_k f_k(y|x; \theta_k) \tag{2.5}$$

where $0 < \pi_k < 1$ and $\sum_{k=1}^{K} \pi_k = 1$. Here, $f_k(y|x; \theta_k)$ is a likelihood distribution–for example, a Gaussian. However, in most real-world applications, K is usually not known a priori and thus we want an infinite number of clusters a priori, i.e., DP. A DP mixture model can either assign the data points to a previously seen cluster, or can start a new cluster with the number of clusters being a random variable.

Now, let Ω denote the set of parameters of the likelihood model and let $\mathcal{F}_\Omega$ be a σ-algebra (i.e. set of events) on that set. If $A \in \mathcal{F}_\Omega$, A is a set of parameters. Let $\delta_\theta : \mathcal{F}_\Omega \to \{0, 1\}$ denote a Dirac delta, defined as follows:

$$\delta_\theta = \begin{cases} 1 & \theta \in A \\ 0 & o.w. \end{cases}$$

According to Komolmogorov consistency theorem, in order to guarantee the existence of a stochastic process on a probability space $(\Omega, \mathcal{F}(\Omega))$, it is enough to provide a consistent definition of what the marginals of this stochastic process are. As the name suggest, in the case of a Dirichlet process, the marginals are Dirichlet distributions. The constructive definition, known as the stick breaking or GEM process goes as follows:

Definition 3. *(Stick-breaking construction [Set94]) Let $\beta_k \sim Beta(1, \alpha_0)$ be independent. Define the stick lengths $\pi \sim GEM(\alpha_0)$ as:*

$$G = \sum_{k=1}^{\infty} \pi_k \delta_{\theta_k}$$
$$\pi_1 = \beta_1$$
$$\pi_k \coloneqq \beta_k \prod_{j<k} (1 - \beta_j)$$

π can be viewed as a partition of the interval $[0, 1]$ into countably many pieces, just as the Dirichlet distribution can be thought of a partition of the interval into finitely many pieces. The GEM(α) distribution can be viewed as a recursive stick-breaking procedure. The first break point $\beta_1 \sim \text{Beta}(1, \alpha)$ is drawn and we have $\pi_1 = \beta_1$. We then break off a piece of proportion β_1 from our stick of length 1, and keep the remaining piece of length $1 - \beta_1$. Then, we can draw $\beta_2 \sim \text{Beta}(1, \alpha)$ and break off a β_2 proportion of the remaining stick which results in $\pi_2 = \beta_2(1 - \beta_1)$.

If we generate independently an infinite list of stick locations $\theta_k \sim G_0$, then we will have a Dirichlet process, G, that obeys the definition of the stick-breaking process [Set94]. We say that $G : \mathcal{F}_{\Omega'} \to (\mathcal{F}_{\Omega} \to \{0, 1\})$ is distributed according to the Dirichlet process distribution, denoted by $G \sim \text{DP}(\alpha_0, G_0)$, if for all measurable partitions of $\Omega, (A_1, \ldots, A_K)$(this means that A_k are events, $A_K \in \mathcal{F}$, that they are disjoint, and that their union is equal to Ω) [BM73], we have:

$$(G(A_1), G(A_2), \ldots, G(A_K)) \sim \text{Dir}(\alpha_0 G_0(A_1), \alpha_0 G_0(A_2), \ldots, \alpha_0 G_0(A_K)) \tag{2.6}$$

Extending the DP to groups of clustering problems, we want to know how the clusters should be shared among the groups, that is, if we have grouped data, we want each group to be clustered but we want parameters to be shared across groups. This problem can be resolved by taking a hierarchical Bayesian approach.

Hierarchical Dirichlet process (HDP). Given the problem of clustering data points within each group, we consider a set of random measures G_j, one for each group j, where G_j is distributed according to a group-specific Dirichlet process $\mathrm{DP}(\alpha_{0j}, G_{0j})$. To have a link between the set of clusters of multiple groups, we link the group-specific DPs via a shared measure H. Integrating over this shared measure H induces dependencies among the DPs. Although clusters arise within each group via the discreteness of draws from a DP, if H is a continuous measure then the atoms associated with the different groups are different and there is no sharing of clusters between groups.

To incorporate shared clusters, consider a nonparametric hierarchical model in which H is itself a draw from a Dirichlet process $\mathrm{DP}(\alpha, \Theta)$. Note that while Θ can be continuous or discrete, H is always a discrete measure, which ensures that the G_j have common support. We define the hierarchical model by:

$$\begin{aligned} H|\alpha, \Theta &\sim \mathrm{DP}(\alpha, \Theta) \\ G_j|\tau, H &\sim \mathrm{DP}(\tau, H), \quad \text{for each } j, \end{aligned} \tag{2.7}$$

This model is known as a *hierarchical Dirichlet process* [TJBB05]. Dirichlet process models are not ideal for modeling graphs, as they do not capture the power-law behavior often found in graph structured data.

Since we are concerned with large scale graphs that exhibit heavy tailed degree distribution, in chapter 4, we will take advantage of the Pitman-Yor process (PYP), described next. PYP is a two-parameter extension to the DP that allows heavier-tailed distributions over partitions.

Pitman-Yor process (PYP)

An interesting extension of DP is the Pitman-Yor process (PYP) [PY+97]. This two-parameter generalization of the DP can also be nested within a hierarchical structure [Teh06] and allows for very flexible control of the clustering behaviour [DBFL+13].

Definition 4. *(Pitman-Yor process [PY+97]) The Pitman-Yor process is defined by $PYP(d, \alpha, G_0)$ which is a distribution over discrete probability measures, having two parameters d and α, where d is called a discount parameter–also known as a concentration parameter–and α is called a strength parameter. The strength parameter $\alpha > -d$ controls the degree to which the new draw is assigned to among those which are not appeared in the past, while a discount parameter $0 \leq d < 1$ specifies the degrees to which the new*

draw resembles the base distribution G_0.

$$\begin{aligned} G &= \sum_{k=1}^{\infty} \pi_k \delta_{\theta_k} \\ \phi_k &= \pi_k \prod_{j=1}^{k-1} (1 - \pi_j) \\ \pi_j &\sim Beta(1 - d, \alpha + jd) \end{aligned} \tag{2.8}$$

When $0 < d < 1$, the PYP exhibits power-law behaviour. When $d = 0$, we recover the stick-breaking construction for the Dirichlet process given in Equation 4. The more data points have been assigned to a draw from G_0, the more likely subsequent data points will be assigned to the draw (the richer-gets-richer property), while the more we draw from G_0, the more likely a new data point will be assigned to a new draw from G_0. As the discount parameter increases, we get increasingly heavy-tailed distributions over the atom sizes in the resulting probability measure.

2.1.3 Latent class models

In the *latent class models*, each data point x_n belongs to a latent class $c_n = k, k = 1, \ldots, K$, where K is the number of possible classes. If $K = \infty$, the model has an infinite number of classes and thus has an infinite number of parameters, or in other words the model is nonparametric. Such class models are interested on a distribution, G, with parameter μ_k^* over class assignments.

One main property of real-world networks separates the observations into groups, and yet allow the groups to remain linked. To consider problems involving groups of data, we say each observation within a group is a draw from a mixture model, and we put shared mixture components between the groups. To achieve so, we consider a hierarchical modeling where parameters are shared among groups and the randomness of the parameters induces dependencies among the groups. Within each group we wish to capture latent structure in the data such as clustering. The number of mixture components and the clusters within them are unknown and are to be inferred from the data. We model a set of observations $\{x_1, \ldots, x_n\}$ using a set of latent parameters $\{\theta_1, \ldots, \theta_n\}$.

Each θ_i is drawn i.i.d. from G, while each x_i has distribution $F(\theta_i)$ parameterized by θ_i:

$$\begin{aligned} x_i|\theta_i &\sim F(\theta_i) \\ \theta_i|G &\sim G \\ G|\alpha, H &\sim \mathrm{DP}(\alpha, H) \end{aligned} \tag{2.9}$$

Because G is discrete, multiple θ_i's can take on the same value simultaneously, and the above model can be seen as a mixture model, where x_i's with the same value of θ_i belong to the same cluster. The mixture perspective can be made more in agreement with the usual representation of mixture models using the stick-breaking construction. Let z_i be a cluster assignment variable, which takes on value k with probability π_k. Then Equation 2.9 can be equivalently expressed as

$$\begin{aligned} \pi|\alpha &\sim \mathrm{GEM}(\alpha) & \phi_k|H &\sim H \\ z_i|\pi &\sim \mathrm{Multinomial}(\pi) & x_i|z_i, \{\phi_k\} &\sim F(\phi_{z_i}) \end{aligned}$$

with $G = \sum_{k=1}^{\infty} \pi_k \delta_{\phi_k}$ and $\theta_i = \phi_{z_i}$. In mixture modeling terminology, π is the mixing proportion, ϕ_k are the cluster parameters, $F(\phi_k)$ is the distribution over data in cluster k, and H the prior over cluster parameters. The CRP describes (up to a reordering) the distribution over clustering, if the DP is marginalized out [BM73].

Chinese restaurant process (CRP)

The CRP is a stochastic process that generates an exchangeable (see Section 2.3) partition of data points, and can be described using a metaphor of a Chinese restaurant with an unbounded number of tables. Due to its flexibility and extendability, CRP is widely used in DP-based models. In other words, CRP is a distribution on partitions of data points, that can be described in terms of a process by which N customers sit down in a Chinese restaurant with an infinite number of tables each with infinite capacity. The first customer sits at the first table. The n-th customer sits at a table drawn from the following distribution

$$p(c_n = k|c_{1:n-1}) \propto \begin{cases} m_k & \text{if } k \text{ is previous occupied} \\ \alpha & \text{if } k \text{ is a new table} \end{cases} \tag{2.10}$$

where m_k is the number of previous customers sitting at table k and α is a positive scalar. Each table is endowed with a parameter vector θ drawn from a base measure G_0. Each

customer is associated with the parameter vector at the table at which she sits,

$$\theta_i|\theta_{1:i-1}, G_0, \alpha \sim \sum_k \frac{m_k}{i-1+\alpha}\delta(\phi_k) + \frac{\alpha}{i-1+\alpha}G_0 \tag{2.11}$$

where ϕ_k is the parameter of the dish served at table k. The resulting distribution on a sequences of parameter values is referred to as P*ó*lya urn model [Spr78]. If the parameters of the table refer to a family of probability distributions (e.g., multivariate Gaussian), the P*ó*lya urn distribution can induce a probabilistic clustering on the generating data, since customers sitting around each table share the same parameter vector. Although the customers are described as entering the restaurant one by one in order, any permutation of their ordering has the same joint probability distribution which points out the exchangeability property of CRP mixture model (see Section 2.3).

Distance dependent Chinese restaurant process (ddCRP)

Distance-based methods leverage upon a distance metric between data points and do not in general require a generative probability model of the data, while model-based methods rely on discrete mixture models, which model the data in different clusters as arising from kernels having different parameter values. To introduce dependency among random variables, an extension mixture model of CRP was introduced called *Distance Dependent Chinese Restaurant Process* (ddCRP) [BF11]. In the ddCRP, the partitions formed by assigning the customers to tables in the CRP, are replaced with the relationships between one customer and other customers. Similarly to the metaphor of CRP, in the ddCRP model, each customer decides who she/he wants to sit with among the already seated customers in the restaurant. Given the relations among the customers, inferring which customers will sit at the same table is easy where the tables are a byproduct of the customer assignments. Let f denote the decay function, $d_{i,j}$ the distance measure between two data points (customers) i and j, α the scaling parameter and c_i the assignment of i-th customer. c_i denoted that i-th customer selects the customer c_i to sit with,

$$p(c_i = j) \propto \begin{cases} f(d_{i,j}) & i \neq j \\ \alpha & i = j \end{cases} \tag{2.12}$$

where the distance measure $d_{i,j}$ evaluates the difference between the two observations i and j and the decay function $f(.)$ mediates how the distance affects the probability of the relation between two customers. Therefore, the farther the distance, the smaller their relationship probability. The traditional CRP can be induced by considering the decay function to be a constant, i.e., $f(d_{i,j}) = 1$.

2.1.4 Inference

The stochastic part of a Bayesian nonparametric model, such as the DP, has infinite number of dimensions where computing the posterior using a Bayes' equation is not feasible; the discrete, infinite-dimensional models do not admit a density. Instead, we can make use of conjugacy that allows practical inference. Conjugate posterior refers to the fact that the posterior belongs to the same model family as the prior, and the posterior parameters can be computed as a function of the prior hyperparameters and the observed data.

As an example of conjugacy in a parametric model, consider a Dirichlet distribution prior and a multinomial likelihood. The Dirichlet multinomial can be in the following form,

$$p_1, \ldots, p_K \sim \text{Dirichlet}(\alpha_1, \ldots, \alpha_K) \tag{2.13}$$

$$\theta_1, \ldots, \theta_K \sim \text{Multinomial}(p_1, \ldots, p_K) \tag{2.14}$$

We see that the posterior distribution over θ is a Dirichlet distribution with updated parameters. This relationship means we can evaluate the posterior distribution and the posterior predictive without explicitly using Bayes Law. If we want to infer a new data point y with the distribution $f(y|\theta)$, the posterior predictive can be defined as:

$$f(y|X) = \int f(y, \theta|X)d\theta = \int f(y|\theta)f(\theta|X)d\theta \tag{2.15}$$

where

$$\begin{aligned} f(\theta|X) = \frac{f(\theta, X)}{f(X)} &\propto f(\theta, X) \\ &= f(p_1, \ldots, p_K|\alpha_1, \ldots, \alpha_K) \prod_{y_i \in X} f(y_i|p_1, \ldots, p_K) \\ &\propto \prod_{j=1}^{K} p_j^{\alpha_j - 1} \prod_{y_i \in X} \prod_{j=1}^{K} p_j^{y_i^{(j)}} \\ &= \prod_{j=1}^{K} p_j^{\alpha_j - 1 + \sum y_i^{(j)}} \end{aligned}$$

We can make use of this conjugacy in the Dirichlet process, even though Bayes' Law does not apply. Referring to Equation 2.6, if we see an observation in the J-th partition, then

$$(G(A_1), \ldots, G(A_j), \ldots, G(A_K)|\Theta_1 \in A_j) \sim \text{Dir}(\alpha_0 G_0(A_1), \ldots, \alpha_0 G_0(A_j)+1, \ldots, \alpha_0 G_0(A_K)) \tag{2.16}$$

Since this must be true for all possible partitions, this is only possible if the posterior for G is given by:

$$G|\Theta_1 = \theta \sim \mathrm{DP}(\alpha + 1, \frac{\alpha_0 G_0 + \delta_\theta}{\alpha_0 + 1}) \tag{2.17}$$

Thus in general, if $G \sim \mathrm{DP}(\alpha, H)$, the posterior of G under observations $\theta_1, \ldots, \theta_n$ is again a Dirichlet process, $\mathrm{DP}(\alpha + n, \frac{1}{\alpha+n}(\alpha H + \sum \delta_{\theta_i}))$.

The Dirichlet process is therefore conjugate to the multinomial, which makes inference in the Dirichlet/multinomial model easy.

MCMC sampling

In practice, nonparametric Bayesian models are too complicated to directly obtain the posterior. Monte Carlo methods consist of a wide variety of algorithms that enable us to sample from complicated probability distribution functions without explicitly generating the probabilistic density function.

The high-dimensional probability distribution $f(X)$ makes this integral hard to compute. Markov chain Monte Carlo (MCMC) methods [ADFDJ03] provide a class of algorithms that produce estimates of this integral based on iterative sampling, combining Monte Carlo integration with samples from a specially constructed Markov chain. MCMC algorithms use $f(\theta|X)$ for proposal distribution, instead of $f(X)$. This process therefore generates a Markov chain from samples $\theta_1, \theta_2, \ldots$.

To construct the Markov chain, we use Gibbs sampling. Gibbs sampling samples sequentially from the conditional distributions of each parameter. If our component distributions are conjugate, then we can do this easily. Each random variable θ_i is sampled given the rest of random variables denoted by θ_{-i}:

1. Picks an index $i \in \{1, \ldots, m\}$ either via round-robin or uniformly at random
2. Sets $\theta_j^{(t+1)} = \theta_j^{(t)}$, for $j \neq k$, i.e., $\theta_{-i}^{(t+1)} = \theta_{-i}^{(t)}$
3. Generates $\theta_{-i}^{(t+1)} \sim f(\theta_i|\theta_{-i}^{(t)})$

where only one component of θ is updated at a time.

2.2 Statistical graph (network) models

Our primary focus is on multigraph models, but our model construction implicitly covers directed/undirected graphs. By forming an undirected edge in the directed multigraph where any directed edge between two vertices exist, the undirected graph is produced.

2.2.1 Directed multigraphs

Let $V = (\theta_1, \theta_2, \ldots)$ be a (countably) infinite set of vertices where $\theta_i \in \mathbb{R}_+$. The directed multigraph can be indicated using an atomic measure on $\mathbb{R}_+^2$

$$D = \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} m_{i,j} \delta_{\theta_i, \theta_j} \tag{2.18}$$

where $m_{i,j}$ is the number of directed edges from vertex θ_i to vertex θ_j. To model D, each vertex is associated with a sociability parameter $w_i > 0$ that can be defined via the atomic random measure

$$W = \sum_{i=1}^{\infty} w_i \delta_{\theta_i} \tag{2.19}$$

which can be distributed according to any stochastic process defined in Section 2.1.2.

2.2.2 Undirected graphs

Correspondingly, an undirected graph is defined as:

$$Z = \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} z_{i,j} \delta_{\theta_i, \theta_j} \tag{2.20}$$

where $z_{i,j} = z_{j,i} \in \{0, 1\}$. In the graph, $z_{i,j} = z_{j,i} = 1$ indicated an undirected edge between vertices θ_i and θ_j. Given the directed multigraph, we can transform to an undirected graph by setting $z_{i,j} = z_{j,i} = 1$ if $m_{i,j} + m_{j,i} > 0$ and $z_{i,j} = z_{j,i} = 0$ otherwise. In other words, an undirected edge is placed between vertices θ_i and θ_j if and only if there is at least one directed edge between them.

2.2.3 Bipartite graphs

Our construction of the graph models can be extended to bipartite graphs. Let $V = (\theta_1, \theta_2, \ldots)$ and $V' = (\theta'_1, \theta'_2, \ldots)$ be two (countably) infinite set of vertices where $\theta_i, \theta'_i \in \mathbb{R}_+$. In bipartite graphs, the interactions are only allowed between different set of vertices.

The directed bipartite multigraph can be represented as:

$$D = \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} m_{i,j} \delta_{\theta_i, \theta'_j} \tag{2.21}$$

where $m_{i,j}$ is the number of directed edges from vertex θ_i to vertex θ'_j. Similarly, the bipartite graph is represented by:

$$Z = \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} z_{i,j} \delta_{\theta_i, \theta'_j} \tag{2.22}$$

where $z_{i,j} = 1$ if there is a connection from vertex θ_i to vertex θ'_j.

2.3 Exchangeability

Since in many scenarios, the order in which the data points are observed is insignificant [BC09, Hof09], the assumption of a notion of exchangeability becomes inevitable. The intuition behind this notion is when the order of observing the data is not important, we can benefit from such distributions where a model whose distribution depends on the order in which we see our data is not justified.

Definition 5. *Exchangeable sequence. We call a finite sequence* $(X_1, \ldots, X_N)$ *exchangeable, if its distribution is invariant under any permutation* σ *of* $\{1, \ldots, N\}$*, meaning*

$$X_1, \ldots, X_N \stackrel{d}{=} X_{\sigma(1)}, \ldots, X_{\sigma(N)}$$

An infinite sequence $X_1, X_2, \ldots$ *is identified as infinitely exchangeable if all of its finite subsequences are exchangeable.*

Consider the array of random variables $(X_{i,j}), 1 \leq i, j < \infty$, such that permutations of rows or of columns do not alter the distribution of the array. Let R_i denote the i-th row, that is, $R_i = (X_{i,j}; j \geq 1)$. X is *row-exchangeable* if the sequence $(R_i)_{i \geq 1}$ of random vectors is exchangeable. Let $C_j = (X_{i,j}; i \geq 1)$ denote the j-th column, X is called *column-exchangeable* if $(C_j)_{j \geq 1}$ is exchangeable. In this case, a statistical model is a family $\mathcal{P} = \{P_\theta | \theta \in \mathbf{T}\}$ on distributions on $\mathbb{X}$, indexed by elements θ of some parameter space $\mathbf{T}$. If the sequence X is exchangeable, de Finetti's theorem tells us that there is some model $\mathcal{P}$ and some distribution ν on $\mathbf{T}$ such that the joint distribution of X is

$$\mathbb{P}(X \in .) = \int_{\mathbf{T}} P_\theta^{\infty}(.) \nu(d\theta) \tag{2.23}$$

Which implies that the sequence can be generated by first generating a random parameter value $\Theta \sim \nu$, and then sampling $X_1, X_2, \ldots | \Theta \sim_{iid} P_\Theta$. In particular, the elements of the sequence are conditionally *iid* given Θ.

Theorem 2.3.1. *(de Finetti)[OR14]. Let $(X_1, X_2, \ldots)$ be an infinite sequence of random variables with values in a space $\mathbf{X}$. The sequence $X_1, X_2, \ldots$ is exchangeable if and only if there is a random probability measure Θ on $\mathbf{X}$ such that the X_i are conditionally iid given Θ and*

$$\mathbb{P}(X_1 \in A_1, X_2 \in A_2, \ldots) = \int_{M(X)} \prod_{i=1}^{\infty} \theta(A_i) \nu(d\theta) \tag{2.24}$$

where ν is the distribution of Θ.

2.3.1 Exchangeability in networks

We call a random (infinite) directed graph G on Ω, *vertex exchangeable* if its joint distribution is invariant under all permutations π of the vertices,

$$(G_{ij})_{i,j \in \Omega} \stackrel{d}{=} (G_{\pi(i)\pi(j)})_{i,j \in \Omega} \tag{2.25}$$

We call this G *separately exchangeable*, when the distribution is preserved under permutation of each index separately, for instance $(G_{ij}) \stackrel{d}{=} (G_{\pi(i)\sigma(j)})_{i,j \in \Omega}$ for any permutation π and σ which arises for adjacency matrices of bipartite graphs.

The Aldous-Hoover theorem [Ald81] proves that these graphs are either dense or empty with probability one [OR14]. It follows from this theorem that any vertex-exchangeable graph is a mixture of sampling procedures from graphons. A graphon is a symmetric, measurable function $W : [0,1]^2 \to [0,1]$ [DHJ08]. Given a graphon W, we can define an adjacency matrix $(G_{ij})_{i,j \in \mathbb{N}}$, where $G = (\mathbb{N}, W)$ is a countably infinite exchangeable graph associated to W,

$$\begin{aligned} U_i &\stackrel{\text{iid}}{\sim} \text{Uniform}[0,1] \quad \text{for } i \in \mathbb{N} \\ G_{i,j} | U_i, U_j &\stackrel{\text{ind}}{\sim} \text{Bernoulli}(W(U_i, U_j)) \quad \text{for } i < j. \end{aligned} \tag{2.26}$$

with $G_{ij} = G_{ji}$ for $i < j$ and $G_{ii} = 0$. As a result, any graph sampled from a graphon is almost surely dense or empty [OR14]. Most real-world networks however tend to be sparse. Many popular network models (see, e.g., Lloyd et al. [LOGR12] for an extensive list) share the undesirable scaling property that they yield dense sequences of graphs with probability one. Thus, vertex-exchangeable random graph models are misspecified models for sparse network datasets, as they generate dense graphs.

Edge exchangeability was first introduced by Crane and Dempsey [CD15a, CD15b], and Williamson [Wil16]. *edge-exchangeability* uses a single infinite latent set of vertices to generate a finite but growing set of vertices [CCB16, Jan18]. Any graph in an edge-exchangeable graph sequence is invariant to the arrival order of edges–rather than the order of the vertices. Crane and Dempsey [CD15a] established a similar notion of edge exchangeability in the context of a larger statistical modeling framework. Crane and Dempsey [CD15a, CD15b] provided sparsity and power law results for the case where the weights $(w_i)_i$ are generated from a Pitman-Yor process and power law degree distribution simulations. Williamson [Wil16] described a similar notion of edge exchangeability and developed an edge-exchangeable model where the weights $(w_i)_i$ are generated from a DP, a mixture model extension, and an efficient Bayesian inference procedure. Crane and Dempsey [CD16] re-examined edge exchangeability, provided a representation theorem, and studied sparsity and power laws for the same model based on Pitman-Yor weights.

2.4 Graph modeling

Learning with graph structured data, such as molecules, social, biological, and financial networks, requires effective modeling of their underlying structure, peculiarly where the data evolves over time. Nonparametric Bayesian models present an attractive opportunity to model the stationary and dynamic evolution of entities and interactions, where the number of entities can be unbounded. However, modelling explicitly the future trajectory of the entities in the underlying graph has not been completely covered by existing dynamic methods. In this section, we first shortly introduce the statistical network models, and then review the previous models for sparse graphs, dynamic graphs and multigraphs.

2.4.1 Bayesian models for multigraphs

Bayesian networks are a combination of graph theory and probability theory. While learning the probability distributions one also needs to learn the network topology in a Learning algorithm over Bayesian networks. Bayesian models for multigraphs can loosely be divided into three camps. Most common are models where the value of an edge between vertices u and v is a random variable parameterized by the value of some function $\theta_{u,v}$ which might be a Poisson distribution. The resulting multigraphs is referred as jointly vertex-exchangeable [Ald81, Hoo79, OR14], since the distribution over the adjacency

matrix is invariant to jointly permuting the row and column indices. For data represented by exchangeable sequences, Bayesian nonparametrics has developed into a flexible model. Its modeling primitives—Dirichlet processes, Gaussian processes, etc.—are widely applied and well-understood, and can be used as components in hierarchical models or dependent models to address a wide variety of data analysis problems.

This class includes the stochastic blockmodel (**SB**) [SN97, KN11] which is a generative model for blocks, groups, or communities in networks. The simplest SB assigns each vertex to one of K groups, and undirected edges are placed independently between vertex pairs with probabilities that are a function only of the group memberships of the vertices. Define a $K \times K$ matrix ψ of probabilities such that the matrix element $\psi_{g_i g_j}$ is the independent probability of an edge between vertices i and j. In this work, a minor extension of the classic SB, heterogeneity in the degrees of vertices is incorporated which results in a better model in practice, giving significantly improved fits to network data.

The infinite relational model (**IRM**) [KTG$^+$06] focuses on different types of relations between pairs and try to partition each type into clusters. For instance, there are several social relations defined on domain *people*×*people*: *likes*(.,.), *admires*(.,.), *respects*(.,.), and *hates*(.,.). With the IRM, the relationships between pairs can be predicted by their cluster assignments. For example, if demographic attributes for the people is included, the clustering, social relationship prediction, and the demographic attributes can be done simultaneously. Suppose the observed data are m relations involving n types, and let R^i be the i-th relation, T^j be the j-th type and z^j be a vector of cluster assignments of T^j. To infer the cluster assignments, the posterior distribution $P(z^1, \ldots, z^n | R^1, \ldots, R^m)$ is estimated via a generative model for relations and cluster assignments:

$$P(R^1, \ldots, R^m, z^1, \ldots, z^n) = \prod_{i=1}^{m} P(R^i | z^1, \ldots, z^n) \prod_{j=1}^{n} P(z^j) \tag{2.27}$$

where the relations are conditionally independent given their cluster assignments. The prior $P(z^j)$ assigns some probability mass to all possible partitions of type T which is a CRP. IRM starts with a single cluster containing a single object, then adds objects until all the objects belong to clusters. Under the CRP, assignment to a cluster is proportion to the cluster's size,

$$P(z_i = a | z_{1:i-1}) = \begin{cases} \frac{n_a}{i-1+\alpha} & n_a > 0 \\ \frac{\alpha}{i-1+\alpha} & a \text{ is new cluster} \end{cases} \tag{2.28}$$

where n_a is the number of objects in cluster a and α is a concentration parameter that controls the number of clusters. Here, the distribution on z induced by the CRP is exchangeable, therefore the order of assigning objects to clusters can be permuted without changing the resulting clustering distribution. To generate relations from cluster, IRM assumes that the relations R are binary:

$$\begin{aligned} z|\alpha &\sim \text{CRP}(\alpha) \\ \eta_{a,b}|\beta &\sim \text{Beta}(\beta,\beta) \\ R_{i,j}|z,\eta &\sim \text{Bernoulli}(\eta_{z_i,z_j}) \end{aligned} \tag{2.29}$$

where $a, b \in \mathcal{N}$. The parameter $\eta_{a,b}$ identifies the probability that an edge exists between any given pair (i, j) where i belongs to cluster a and j belongs to cluster b.

The mixed-membership stochastic blockmodel (**MMSB**) [ABFX08] combines global parameters that instantiate dense patches of connectivity (SB) with local parameters that instantiate vertex-specific variability in the relations (mixed membership). The MMSB assigns each observation to multiple clusters rather than one cluster and defines a probability-like vector. This relates to the fact that each object can have multiple latent roles or cluster-memberships that influence their relationships to others. For instance, what a protein or a social actor interacts with different partners, different functional or social contexts applies and thus the protein or the actor may play different latent roles. The MMSB also relaxes the assumption of independency of relations given their cluster-memberships. Let $G(N, Y)$ be a graph of observed relational data, $Y(u, v)$ maps pairs of vertices to values, edge weights, and let $\pi_{i,g}$ be the probability of relation i belonging to group g. The probabilities of interactions between different groups are defined by a matrix of Bernoulli rates $B_{(K\times K)}$, where $B(g, h)$ represents the probability of a link between from group g to group h. The graph is drawn from a generative model,

$$\begin{aligned} \pi_u &\sim \text{Dirichlet}(\alpha) \\ z_{u\to v} &\sim \text{Multinomial}(\pi_u) \\ z_{v\to u} &\sim \text{Multinomial}(\pi_v) \\ Y(u,v) &\sim \text{Bernoulli}(z_{v\to u} B z_{u\to v}) \end{aligned} \tag{2.30}$$

The MMSB also considers the sparsity of relational data, by introducing a parameter $\rho \in [0, 1]$ in the model to characterize the source of non-interactions. Then, instead of sampling a relation $Y(u, v)$ from Bernoulli with parameter, the probability of successful interaction down-weight to $(1-\rho)z_{v\to u} B z_{u\to v}$. As larger as the ρ value be, the interactions in the matrix will have higher weighted more than non-interactions.

All of these three cluster-based representations struggle with the issue of determining how many latent clusters there are for a given problem. IRM tackle this issue with allowing the number of clusters to be determined at inference time. MMSB does the same as well as considering each entities to be belonging to more than one clusters. The Nonparametric Latent Feature Relational Model (**NRM**) [MJG09] infers the number of latent features there are simultaneously with inferring what features each entity has and how those features influence the observations. Given the directed relational links between a set of N entities, let $Y_{N\times N}$ be the binary matrix containing these links, where $y_{ij} = Y(i,j)$ equals to 1 if there is link from entity i to entity j, and 0 otherwise. The goal is predict unobserved links from the observed links. If there are K features, and Z is the $N \times K$ binary matrix of features where $Z_{ik} \equiv Z(i,k) = 1$ if the i-th entry has feature k and $z_{i,k} = 0$ otherwise. Let $W_{K\times K}$ be the real-valued weight matrix of links, then $w_{k\acute{k}}$ is the weight that affects the probability of link from entity i to entity j if both entity i has feature k and entity j has feature $\acute{k}$. Given the feature matrix Z and weight matrix W, the probability of being a link from i to j is

$$P(y_{ij} = 1|Z,W) = \sigma\Big(Z_i W Z_j^T\Big)\sigma\Big(\sum_{k,\acute{k}} z_{ik} z_{j\acute{k}} w_{k\acute{k}}\Big) \tag{2.31}$$

where $\sigma(.)$ is a function which transforms values on $(-\infty,\infty)$ to $(0,1)$ such that sigmoid function $\sigma(x) = \frac{1}{1+\exp(-x)}$. Using an IBP prior on Z, the generative latent feature relational model become

$$\begin{aligned} Z &\sim \text{IBP}(\alpha) \\ w_{k\acute{k}} &\sim N(0,\sigma_w^2) \qquad \text{for all } k,\acute{k} \text{ for which features } k \text{ and } \acute{k} \text{ are non-zero} \\ y_{ij} &\sim \sigma\Big(Z_i W Z_j^T\Big) \qquad\qquad\qquad\qquad \text{for each observation} \end{aligned} \tag{2.32}$$

While these models are able to capture interesting community structure, the resulting graphs are dense almost surely, [Ald81, Hoo79]. This makes them a poor choice for large real-world multigraphs, which are typically sparse. Further, they assume zero-valued edges are explicitly observed, making them poorly suited for many prediction tasks.

Second, we have multigraphs where the edges are distributed according to a Poisson process (PP) on the space of potential edges. A canonical example of such a model is the sparse exchangeable multigraph of [CF17], where the edges are sampled according to a PP whose rate measure is distributed according to a generalized gamma process. In order to leverage some properties of generative exchangeable modeling while producing

sparse graphs, the exchangeablity of a continuous-space representation of networks base on a point process on $\mathcal{R}_+^2$ is considered,

$$Z = \sum_{i,j} z_{ij} \delta_{(\theta_i, \theta_j)} \tag{2.33}$$

where $z_{ij} = 1$ if there is a link between vertices θ_i and θ_j, and is 0 otherwise. So, the jointly exchangeable random measure theorem [Kal06] is used. Such a model has been extended to have community structure [LJCC18] on inhomogeneous random graphs, where a class of sparse graph models with overlapping community structure and well-specified asymptotic degree distributions is proposed. Let $(G_n)_{n\geq 1}$ be a sequence of simple random graphs of size n, the probability of being a link between vertices i and j in the graph G_n is given by

$$p_{ij}^{(n)} = 1 - \exp\left(- \frac{w_i w_j}{s^{(n)}} \right) \tag{2.34}$$

where $s^{(n)} = \sum_{i=1}^{n} w_i$ and w_i is a positive parameter for degree heterogeneity in the graph (sociability of vertex i), and is *iid* sample from some distribution F with $\mathbb{E}(w_1) < \infty$. The larger this parameter, the more likely vertex i is to connect to other vertices.

These models yield sparse graphs with power-law degree distributions, properties that are common in large social networks. However, like the vertex-exchangeable graphs, they assume a fully observed graph where no new edges will be seen between existing vertices, making them poorly suited to prediction tasks.

Third, we have multigraphs constructed using an exchangeable sequence of edges [CCB16, CD18, Wil16]. Here, we assume the edges are generated by sequentially sampling pairs of vertices. Unlike the vertex-exchangeable and Poisson process based models, edge exchangeable multigraphs can grow over time by adding new edges, either between new or previously seen vertices. For this reason, we focus on edge-exchangeable graphs in our model development. These pairs of vertices are *iid* given some nonparametric prior. This prior could be a Dirichlet process, such ad MDND model [Wil16] which is described in detail in section 3.1.

[CCB16] considers a normalized generalized gamma process as the prior. They propose an alternative notion of exchangeability, *edge-exchangeability*, which means that the distribution of a graph sequence is invariant to the order of the edges. Such models are stationary across steps of the graph sequence and can produce sparse graphs. Let

$\mathcal{W}$ be a Poisson process on $[0,1]$ with a nonatomic, σ-finite rate measure ν such that $\nu([0,1]) = \infty, \int_0^1 w\nu(dw) < \infty$.

$$\mathcal{W} := \sum_{\{i,j\}:i,j\in\mathbb{N}} w_{\{i,j\}}\delta_{\theta_{\{i,j\}}} \tag{2.35}$$

where $(w_{\{i,j\}})$ are the edge rates (or frequencies) in $\mathcal{W}$. To construct edge-exchangeable graphs one can apply *single edge per step* or *multiple edges per step* approach. To use single-edge-per-step approach the rates need to be normalized, $\sum_{\{i,j\}:i,j\in\mathbb{N}} w_{\{i,j\}} = 1$, in which $(w_{\{i,j\}})$ is a distribution over all possible pairs. At each step, an edge e's values is drawn from $\mathcal{W}$ and the edge set is recursively constructed, $E_{n+1} = E_n \cup \{e\}$. Since the edge at every step is drawn *iid* from $\mathcal{W}$, the result graph is edge-exchangeable. The multiple-edges-per-step approach does not consider the rates to be normalized, and defines a distribution over non-negative integers m given some rate $w \in \mathbb{R}$, $f(m|w)$. Start from $F = \emptyset$ at the n-th step. For every possible edge $e = \{i,j\}$ draw the multiplicity of the edge $e$in this step as $m_e \overset{\text{ind}}{\sim} f(.|w_e)$ and add m_e copies of edge e to F, and then set $E_{n+1} = E_n \cup F$. The graph after n steps is given by

$$\mathcal{W} \sim \text{PPP}(\nu) \qquad M_{\{i,j\}} \overset{\text{ind}}{\sim} \text{Binom}(n, w_i, w_j) \quad \text{for } i < j \in \mathbb{N} \tag{2.36}$$

where $M_{\{i,j\}}$, the multiplicity of edge $e = \{i,j\}$ after n steps has a binomial distribution.

There is another model where a PYP (see Section 2.1.2) is considered as a prior [CD18]. Under certain conditions on this prior, the resulting multigraphs will be sparse, [CCB16, CD18].

2.4.2 Models for dynamic graphs

There has been significant research attention on dynamic network modelling. A common approach relies on the extensions of *stationary* network models to a *dynamic* framework. These models can be categorized into Bayesian models and extensions of non-Bayesian models. We start this section by first reviewing briefly Bayesian models, and then discussing other non-Bayesian models for dynamic graphs.

Most dynamic Bayesian networks such as the exponential random graph model [GHFX07] and matrix and tensor factorization-based methods [DKA11] extend jointly vertex-exchangeable graphs. For example, [XH14] extends the stochastic blockmodel using an extended Kalman filter based algorithm; [DD14] allows parameters to evolve according to stochastic processes; and [Xu15] relaxes a hidden Markov assumption on the edge-level

dynamics, allowing the presence or absence of edges to directly influence future edge probabilities. Several methods have also been used to incorporate temporal dynamics into the mixed membership stochastic blockmodel framework [FSX09, XFS+10, HSX11], the infinite relational model [NS17] and the latent feature relational model [FDA+11, HG13, KL13].

Most recently, several models have extended Poisson factor analysis. The dynamic gamma process Poisson factorization (**DGPPF**) [AGZ15] introduces dependency by incorporating a Markov chain of marginally gamma random variables into the latent representation. The dynamic Poisson gamma model (**DPGM**) [YK18b] extends a bilinear form of Poisson factor analysis [Zho15] in a similar manner. The dynamic relational gamma process model (**DRGPM**) [YK18a] also incorporates a temporally dependent thinning process.

Much less work has been carried out on dynamic extensions of the sparse graphs generated using Poisson processes or via a sequence of exchangeable edges. In the Poisson process-based space, [PCT16] uses a time-dependent base measure, and assume edges have a geometric lifespan. In the edge exchangeable case, [NS17] incorporates temporal dynamics into the MDND by introducing a latent Gaussian Markov chain, and a Poisson vertex birth mechanism. [GMGL18] extends the MDND to partially observed data and uses a temporally informed inference algorithm, but the underlying model is stationary.

One group of graph models, try to learn latent representations of vertices in a network. These representations encode interactions in a continuous vector space, which can be simply exploited by statistical models. **DeepWalk** Transforms a graph structure into a sample collection of linear sequences consisting of vertices using uniform sampling. These simulation random walks can be used to learn d-dimensional feature representations, local neighborhood of vertices. The sampling strategy can be also inferred by using vertex2vec and setting $p = 1$ and $q = 1$. Although empirically effective, it lacks a clear objective function to articulate how to preserve the network structure. It is prone to preserving only the second-order proximity. DeepWalk works well in language modeling and unsupervised feature learning from sequences of words to graphs. The learned social representations, meaning latent features of the vertices that capture neighborhood similarity and community membership, encode social relations in a continuous vector space with a relatively small number of dimensions. DeepWalk evaluates its performance on multi-label classification tasks. They also demonstrate the scalability of DeepWalk by building representations of web-scale graphs (i.e., YouTube) using a parallel imple-

mentation. Given a partially labeled social network $G_L = (V, E, X, Y)$, with attributes $X \in \mathbb{R}^{|V| \times S}$, where S is the size of the feature space for each attribute vector, and $Y \in \mathbb{R}^{|V| \times |\mathcal{Y}|}$, $\mathcal{Y}$ is the set of labels. The goal is to learn $X_E \in \mathbb{R}^{|V| \times d}$, where d is small number of latent dimensions. Each dimension contributes to a subset of social concepts expressed by the space and each social phenomena is expressed by a subset of dimensions.

For relational classification problem (or the collective classification problem), DeepWalk proposes an unsupervised method to capture the network topology by learning features that capture the graph structure independent of the labels' distribution instead of using iterative approximate inference algorithm (such as iterative classification algorithm, Gibbs Sampling, or label relaxation). Therefore, this separation between structural representation and the labeling task avoids cascading errors of iterative methods. While learning social representations, DeepWalk seeks 4 important characteristics of real-world data; (1) Adaptability: in order to capture evolving networks, and not requiring repeating the learning process all over again. (2) Community aware: the distance between latent dimensions represent a metric for evaluating social similarity. (3) Low dimensional: low-dimensional models generalize well when labeled data is scarce and makes them converge faster. (4) Continuous: to model partial community memberships in a continuous space, a continuous representation is required with smooth decision boundaries between communities.

When applied to language modeling, DeepWalk considers a set of short truncated random walks its own corpus, and the graph vertices as its own vocabulary ($\mathcal{V} = V$). Therefor, DeepWalk algorithm consists of two main parts: random walk generator, and update procedure. The random walk generator takes a graph and samples uniformly a random vertex v_i as the root of random walk $\mathcal{W}_{v_i}$. Then, a walk samples uniformly from the neighbors of the last vertex visited until the maximum fixed length is reached. The results shows no privilege of using restart walks. To update the representations using these walks, SkipGram iterates over all possible collocations in random walk that appear within the window ω. Each vertex v_i is mapped to its current representation vector $\Phi(v_i) \in \mathbb{R}^d$ to maximize the probability of its neighbors in the walk.

2.4.3 Models for sparse graphs

Many real-world structural networks are sparse, where the number of non-zero entities grows as $O(M)$, where M is the number of entities. Since the number of interactions an entity makes does not grow with the size of the network, the adjacency matrices encoding

pairwise interactions are often sparse, and they contain many zeros or non-interactions (i.e., they may be the result of the pairs that rarely interact in general, or they are due to the fact that the pair of relevant groups rarely interact). In applications of social networks, for instance, vertices may represent people and groups of people may represent social communities. We expect that a large portion of the non-interactions in this example is because of limited opportunities of contact between people rather than the structure of the underlying graph. It is shown that several real-world networks have a very high probability of exhibiting this form of sparsity [CF17].

A lot of recent models tried to achieve sparsity by changing the fundamental model of vertex exchangeability. One type of these models employ an additional parameter to uniformly decrease the probabilities of edges as the network grows (e.g., Bollobás et al.[BJR07], Borgs et al. [BCCG15], Wolfe and Olhede [WO13]). However, these models generate non-projective graph sequences, since there is not any strict subgraph relationship between earlier graphs and later graphs in the sequence. The pioneers on modeling sparse, projective graph sequences are Caron and Fox [CF17] who consider a notion of graph exchangeablity called *Kallenberg-style exchangeability*. The generative mechanism in this idea allows a new countable infinity of latent vertices at every step in the graph sequence which is suitable for non-stationary domain and is bounded with continuous-valued labels of the vertices. Whereas, we are interested in the models in which the model complexity grows with the size of dataset and analyzing the graph sequences based solely on its topology. Some models combine classic Bayesian nonparametric models–e.g., the Chinese restaurant process (CRP) and Indian buffet process (IBP)–with hidden Markov models or other explicit non-stationary mechanisms to generate a finite but growing set of vertices.

In contrary to vertex exchangeability, edge exchangeability admits sparsity in random graph sequences. There is a class of sparse, edge-exchangeable multigraph sequences introduced by [CCB16] in which $\mathcal{W}$ is a *Poisson point process* (PPP). It is said that $(w_i)_i$ is distributed according to a Poisson point process parameterized by rate measure ν, $(w_i)_i \sim \text{PPP}(\nu)$, if (a) $\{i : w_i \in A\} \sim \text{Poisson}(\nu(A))$ for any set A with finite measure $\nu(A)$ and (b) $\{i : w_i \in A\}$ are independent random variables across any finite collection of disjoint sets $(A_j)_{j=1}^J$. Let $\mathcal{W}$ be a PP on $[0, 1]$ with a nonatomic, σ-finite rate measure ν satisfying two conditions $\nu[0, 1] = \infty$ and $\int_0^1 w\nu(dw) < \infty$, which guarantees that $\mathcal{W}$ be a countably infinite set of rates in $[0, 1]$ and that $\Sigma_{w\in\mathcal{W}} w < \infty$ almost surely. To construct a set of rates, if $i \neq j$, $w_{\{i,j\}} = w_i w_j$, and $w_{\{i,j\}} = 0$ which avoids self-loops.

For the multiple-edges-per step setting, let $f(.|w) \sim \text{Bernoulli}(w)$. Each edge $\{i, j\}$ is added in each step with probability $w_i w_j$, which makes its multiplicty $M_{\{i,j\}}$ become a binomial distribution after n step with parameters $n, w_i w_j$. Therefore, the graph after n steps is described by:

$$\mathcal{W} \sim \text{PPP}(\nu) \qquad M_{i,j} \overset{\text{ind}}{\sim} \text{Binomial}(n, w_i w_j) \quad \text{for } i < j \in \mathbb{N}. \tag{2.37}$$

This generative model yields edge-exchangeable graph, with the number of vertices and edges defined as:

$$|\bar{V}_n| = |V_n| = \sum_i \mathbb{1}\left(\sum_{j \neq i} M_{\{i,j\}} > 0\right), |E_n| = \frac{1}{2}\sum_{i \neq j} M_{\{i,j\}}, |\bar{E}_n| = \frac{1}{2}\sum_{i \neq j} \mathbb{1}(M_{\{i,j\}} > 0). \tag{2.38}$$

and the expected number of vertices and edges are,

$$\mathbb{E}(|\bar{V}_n|) = \mathbb{E}(V_n) = \int \left[1 - \exp\left(-\int (1 - (1 - wv)^n)\nu(dv)\right)\right]\nu(dw), \tag{2.39}$$

$$\mathbb{E}(|E_n|) = \frac{n}{2}\int\int wv\nu(dw)\nu(dv), \tag{2.40}$$

$$\mathbb{E}(|\bar{E}_n|) = \frac{1}{2}\int\int (1 - (1 - wv)^n)\nu(dv)\nu(dv). \tag{2.41}$$

2.4.4 Diffusion network inference

There is a body of work on inferring diffusion network from partial observations. In order to study network diffusion, we should address two fundamental challenges; 1) tracking cascading processes taking place in a network, and 2) identifying to trace the contagion as it is diffusing through the network. NetInf [GRLK10] and MultiTree [RS12] formulate the problem as submodular optimization. They show that the algorithm recovers the structure of simulated data and it appears to work well with real data. News topics and "memes" can be also be tracked on the web to characterize a news cycle. NetInf addresses the aforementioned challenges by inferring the underlying unknown network over which contagions are propagating. They assume the underlying network is static and does not change over time with the diffusion ocurring at equal rates across different edges. For each contagion, the vertices and their infection times are observed. Given these various contagions, the edges of the underlying network are inferred. On a fixed hypothetical network, there may be many possible ways of the directed tree that a contagion has been propagated on. By considering only the most likely propagation tree,

the computations are reduced to cubic time. Then, they proposed a method to find G that maximizes:

$$G^* = \arg\max_{|G|\leq k} F_C(G) \tag{2.42}$$

where the maximization is over all graphs G of at most k edges, and $F_C(G)$ is the most likely propagation tree of contagion C. MultiTree finds a solution with provable sub-optimality guarantees by exploiting a natural diminishing returns property of the network inference problem: submodularity.

NetRate [GrS11] and CoNNIe [ML10] further infer the transmission rates using convex optimization.Later, [GrS11] modeled diffusion process as a continuous temporal process occurring at different rates, and proposed NetRate to infer the time-varying transmission rates via stochastic convex optimization. CoNNIe includes a l_1-like penalty term that controls sparsity while NetRate provides a unique sparse solution by allowing different transmission rates across edges. Consider edge (i, j), CoNNIe infers a prior probability $\beta_{i,j}$ but NetRate infers a transmission rate $\alpha_{i,j}$. These methods are more expensive computationally than NetInf.

InfoPath [GRLS13] considers inferring varying transmission rates in an online manner using stochastic convex optimization. The above methods assume that diffusion rates are derived from a predefined parametric probability distribution, such as Exponential, Power-law, or Rayleigh. This method can infer temporally heterogeneous interactions within a network and formulates the inference into a MLE problem and utilize a stochastic gradient method to solve it. Despite successful aspects of InfoPath, their method is constrained by specific contagion transmission models. EmbeddedIC [BLG16] embeds vertices in a latent space based on Independent Cascade model, and infer diffusion probabilities based on the relative vertex positions in the latent space. EmbeddedIC considers each inactive (not infected) vertex to be activated by the active vertices. It differentiates two kinds of roles for the vertices; an active vertex as a sender, and an inactive vertex as a receiver. For each of these roles, a vector of vertex's embedding is learned. An activation at time stamp t is enabled between an inactive vertex and all the existing active vertices by t, regardless of the network structure. Then an activation based on the closeness between an inactive vertex's receiver embedding vector and the active vertices' sender embedding vectors is modeled. Note that the embedding of each sender (i.e., active vertex) is learned without the diffusion topologies.

More recently, there have been tries to predict the transmission probabilities from

infection times, either by learning vertex representations [BLG16], or by learning diffusion representations using the underlying network structure [BLG16, WZLC17, LMGM17]. DeepCas [LMGM17] and TopoLSTM [WZLC17] use the network structure to learn diffusion representations and predict diffusion probabilities. DeepCas predicts the future cascade size by modelling the cascade at each time step t with an induced subgraph over the active vertices. The model decomposes the subgraph into some random walks, and uses Gated Recurrent Unit (GRU) [GL16] to learn an embedding vector of the subgraph. Then, it predicts the cascade size based on this subgraph embedding vector. However, the resulting embedding is only partially aware of the graph structure because first the induced subgraph at each time step does not capture how the diffusion spreads and secondly each subgraph is further decomposed into paths and embedded independently.

TopoLSTM proposes a diffusion model, which can estimate the activation probability for an inactive vertex in a cascade. They have the same approach as EmbeddedIC with this difference that their model considers the dynamicity of cascades. Cascades can grow over time, and this makes the embedding vectors of active vertices to change over time. However, since the inactive vertices have not participated in the cascade so far, the embedding approach over receiver vertices (inactive vertices as a receiver) remains the same. TopoLSTM takes the dynamic DAGs (directed acyclic graphs) of each diffusion as inputs and generates a topology-aware embedding for each vertex in the DAGs as outputs. The embeddings of each active vertex is learned by taking all embeddings and the path of diffusion into account.

2.5 Network centrality

The problem of detecting central vertices (or vertices) in networks is addressed in this thesis. In this section, we first review the previous work on exact and approximate computation of several centrality measures. Then, we investigate the work on detection of central vertices (instead of computing all centralities) in a distributed way without requiring knowledge of the full network topology.

2.5.1 Network centrality measures

The idea that the structural position of a vertex in a network may be correlated with the relative influence or importance of that vertex was first postulated by Bavelas [Bav50]. Since then, many notions of what it means to be important or central in a network have

been proposed and applied with great success in a variety of different contexts.

degree centrality refers to the problem of identifying the k vertices with the largest degree in the undirected graph $G = (V, E)$ with N vertices and $|E|$ edges. The problem may seem trivial if one assume that the network structure and the relation between entities are known. However, even then finding the top-k vertices in the graph G takes $O(N)$ time which is not tractable for very large networks. Furthermore, the data of many networks (e.g. social networks) is typically available only to managers of the networks. There exist many sampling approaches that use random walk (RW) to tackle this problem. In [CRS12], Weighted random walk method (WRW) is used to detect high degree vertices. For each edge $(u, v) \in E$, WRW sets a positive weight $w_{u,v} = (\deg(u)\deg(v))^{\beta}$, which indicates that at each step a walk needs to obtain degrees of current sampled vertex's neighbors. Expansion sampling (XS) is proposed in [MBW10] to detect high degree vertices via modifying the random walk to select the next vertex relative to the number of neighbors. Currently, there is a rapidly growing interest in algorithms that only use local information about the degree of a vertex and its neighbors to give a good approximation [CRS12, ALST14, BK10, KLMT08]. In [ALST14], a random walk containing restart that uses only the information on the degree of a currently visited vertex was suggested. A local algorithm for power law networks is proposed in [BK10] that finds a vertex with degree smaller than the maximum value of factor c. Another algorithm for efficiently discovering the correct set of web pages with largest incoming degrees in a fixed network (specially world wide web) is proposed in [KLMT08].

Between centrality measures the portion of shortest paths in the network that go through a specific vertex and can be defined as [KAS+13]:

$$C_B(u) = \sum_{v,w,v \neq w} \frac{\sigma_{vw}(u)}{\sigma_{vw}} \tag{2.43}$$

where σ_{vw} is the total number of shortest paths between every $v, w \subset V, v \neq w$ and $\sigma_{vw}(u)$ is the number of such paths that pass through vertex u. The computation of betweenness centrality can be accordingly by exact [Bra01] or approximate algorithms [BP07]. The former explores the network using well-known algorithms (i.e., BFS, Dijkstra) and then efficiently aggregates the discovered paths circumventing the high complexity of the involved all-pairs shortest-path problem. The later seek to provide accurate sampling methods that approximate the betweenness of each vertex based on number of the single source shortest paths originating from certain reference vertices or even restrict the considered vertex pairs to those that lie at most k hops away from each other [BE06]. On

the contraty, methods that rely on locally available information are increasingly desirable as they are better suited to the fact that network topology is not accessible. Besides, the distributed-fashion approaches that leverage random walk strategies to reveal vertices with high betweenness centrality [LMR+11, New05] or central vertices in a more general sense [KLMST11], do not qualify for the task because they require information gathering from the whole or part of the network topology, respectively. [KAS+13] introduces k-path betweenness centrality that considers simple random walks which are not necessarily shortest paths. A generalization of betweenness centrality is presented in [DEP10] by considering routing policies in the network. A new distance function between vertex pairs is also proposed that penalizes paths with the high number of hops in weighted networks [OAS10]. All of these algorithms are not efficient for large online social networks. The fastest known exact algorithm to compute the betweenness centrality of vertices in a network is Brandes [Bra01] that requires $O(N + |E|)$ space and a running time of $O(N|E|)$ on unweighted and $O(N|E| + N^2 \log(N))$ on weighted networks. This method computes the shortest path to every other vertex for each vertex u and then traverses these paths backwards to efficiently compute the contribution of the shortest paths from u to the betweenness to other vertices. Moreover, Random sampling is a natural approach to speed up the computation of betweenness [Wan06, JKL+05, BP07], where the contributions of some vertices are sampled uniformly at random instead of computing the contribution of all vertices to the betweenness of the others.

Closeness centrality measures how close a vertex is to all other vertices in the network, and can be defined as [OCL08]:

$$C_C(u) = \frac{N-1}{\sum_{u \neq v \in V} d(u,v)} \tag{2.44}$$

where $d(u, v)$ is the shortest distance between vertices u and v. Locating public facilities over a transportation network such that they are easily accessible to everyone, or identifying people with ideal social network location for the purpose of information dissemination or network influence can be mentioned as scenarios in which identifying high closeness centralities is of great interest. The simple way to compute this centrality measure is solving all-pairs shortest-paths problem that can be computed in $O(N|E| + N^2 \log(N))$ or $O(N^3)$ in worst case. Faster specialized algorithms are proposed for special graphs, such as interval graphs [ACL95], and chordal graphs [HSS97]. [ACIM99] proposed an algorithm with computation time $O(N^{2.5}\sqrt{\log(N)})$ with an additive error of 2 for unweighted graphs. [Wan06] developed an approximation algorithm with $O(\frac{\log(N)}{\epsilon^2}(N \log(N) + |E|))$ time. These centrality measures that have been developed to evaluate the importance of

a vertex are still costly in large real-world networks. Therefore, it is important to propose a new approach to efficiently identify the top-k central vertices with high precision, low computational cost, and without full knowledge of the network topological structure.

2.5.2 Detection of central vertices in networks

There exist several previous methods to assess network centrality in a distributed way without requiring knowledge of the full network topology. The sampling-based methods [LMR+11, MBW10] are among them, which require two different steps: (1) sampling a subset of vertices in the network, (2) estimating the characteristics of interesting vertices in the induced sub-graph containing the sampled vertices. [LK04] proposed a framework to compute shortest-path based centralities (such as betweenness and closeness centralities) in a decentralized way, but their method is still computationally expensive for real networks. A new centrality metric, called Localized bridging centrality (LBC), was introduced in [NK08] to provide a specialized centrality which is targeted at locating bridges (i.e., edges whose removal disconnects the network). Second order centrality (SOC) was introduced in [KLMST11] to assess network centrality without full knowledge of the network topology by perpetual random walk which not only requires a potentially long and undetermined convergence time, but also has a high message passing complexity. Principal component centrality (PCC) [ISLR13] was proposed only for friendship graphs. DANCE method [WGZ11] assigns a volume-based centrality to each vertex, using beyond-localized information with relatively high message passing. A deterministic algorithm for computation of centrality measures in directed graphs is proposed in [YTQ16] that converges in finite time but its usage is only limited to oriented tree graphs. [BBC+19] presented a new algorithm for computing the top-k vertices ranking based on the closeness centrality in unweighted networks by reducing the number of traversed links in breadth-first search. [ALN+10] introduced a Monte Carlo method and [LMR+11] presented a parsimonious sampling method to estimate the k most central vertices in a network. [KAS+13] introduced a new metric, called k-path centrality, and a randomized algorithm for estimating it. Two major drawbacks of these methods are that they require beyond-local-scope information about the network topological structure, and they also need direct measurement of each individual vertex in the network, which restrict their applicability in large real-world networks.

2.6 Applications

2.6.1 Link prediction

Since the availability and importance of relational data has been significantly increased in recent years, it becomes a crucial task to have good models for such data. For this task, what we observed are the interactions between a set of entities and what we want to achieve is to extract informative representations that are useful for making predictions about the entities and their relations. One basic challenge is then link prediction, where we observe the interactions (or in case of partially observable datasets, we extract the interactions) between some pairs of entities in a graph (or network) and we try to predict unobserved interactions (considering unseen entities). For instance, in a social network, we might only know some subset of people are friends and some are not, and seek to predict which other people are likely to get along.
The kinds of latent structure that have been proposed to use in link prediction in such networks have been very limited. Particularly, the machine learning community has focused on latent class models, adapting Bayesian nonparametric methods to jointly infer the latent classes while learning which entities belong to each class. Our main goal is to improve the expressiveness and performance of generative models based on latent structure extraction that represents the properties of individuals from the observations. We pursue the Bayesian nonparametric approach with a richer kind of latent features to simultaneously infer the features at the same time we learn which interactions have each feature. Our model combines these inferred features with known covariates (e.g., time) in order to perform link prediction.

2.6.2 Influence maximization

Social networks–such as Linkedin and Twitter–has brought together small and disconnected relations between people which made them becoming a huge marketing platform. These platforms allow information and ideas (posts) to affect a large population in a short period of time. One challenging problem in viral marketing is to effectively find a set of influential users. By activating this set of users, and sending advertising messages to this set, one can reach out the largest area of the network. Online social networks are appropriate choices to achieve this goal. A question therefore arises here that is how to effectively identify the most influential users to place them into the seed set. This problem is known as *influence maximization* problem, where given a graph, the objective

of the influence maximization problem is to identify a small set of k seed vertices, which when activated, will likely result in the activation of the maximum number of vertices in the input graph, based on a given diffusion model for influence. Therefore, finding the top influential entities in a large-scale social network becomes a very promising task. It is known that the influence of a vertex is highly dependent on the network structure and its location in the network, but how to calculate the real centrality of a vertex is nontrivial. By analysing social networks, and then modelling their latent structure, we can identify top k influential users in both stationary and dynamic networks.

2.6.3 Diffusion prediction

Models of contagion dynamics have proven relevant to the study of information, news, and opinions in online social systems. Modelling the latent structure over which the diffusion processes propagate, and predicting viral information cascades are important problems in the graph modelling. Therefore, the problem of diffusion prediction can be resolved by estimating the probability of an inactive vertex to be activated next in a cascade. The information diffusion has many applications, such as helping to predict which user is an opinion leader [WLY+14], how much a cascade will grow [CAD+14], who are the diffusion sources [GRLK10], and so on. Given a graph $G = (V, E)$, and a set of cascade sequences over V, $C = \{(v_1, t_1), \ldots, (v_{|C|}, t_{|C|})\}$, we wish to predict a vertex to activate next.

CHAPTER 3

Dynamic Graph Model under Partial Observations

In many applications, the underlying network over which contagions spread is unknown, and we only observe the adoption of pieces of information, or spread of infections in the network. Many real-world interacting networks, such as information propagation [KLR17], social networks [KKT03], viral marketing [AW12], and epidemiology [HBG04], can be observed in terms of signals propagating over their underlying graph structure. Uncovering and inferring the structure of such network is possible by using the propagation processes. These propagation processes appear as a *contagion* at some vertex of a network and then spread like an epidemic from vertex to vertex over the edges of the network. For instance, a contagion can correspond to a type of behavior or action, e.g., purchasing a new product, or the propagation of a contagion disease over social network of individuals. Epidemiologists can observe when a person becomes ill but they cannot tell who infected her or how many exposures and how much time was necessary for the infection to take hold. In information propagation, we observe when a blog mentions a piece of information. However if, as is often the case, the blogger does not link to her source, we do not know where she acquired the information or how long it took her to post it.

For the problem of inferring diffusion networks, instead of acquiring the whole structure of the underlying network (access to the vertices and edges of its corresponding graph), we only observe the contagions over the network, i.e., tracking the times when vertices reproduce a piece of information, get infected by a virus, or buy a product. In all of these

scenarios, we observe where and when but not how or why information propagates through the underlying network. The problem of dynamically inferring the underlying network structure given a set of cascades (partial observations) can be defined by considering a hidden directed dynamic network where vertices and edges may appear or disappear over time. At each time step t, the network $G^t = (V^t, E^t)$ consists of a set of vertices V^t, and a set of edges E^t. We assume that $V^1 \subseteq V^2 \cdots \subseteq V$. However, we consider a vertex as disappeared if it doesn't have any edges. A set C of s contagions spread over edges of the network from infected to non-infected vertices. For each contagion $c \in C$, we observe a cascade $t^c := (t_1^c, \cdots, t_{|V|}^c)$, recording the times when each vertex got infected by the contagion c. If a vertex u is not infected by the contagion c, we set $t_u^c = \infty$. For each cascade, we only observe the time t_u^c when a vertex u got infected, but not who infected the vertex. Our goal is to infer a model $\mathcal{M}^t$ to capture the latent structure of the dynamic network G^t over which cascades propagated, using partial observations. Such a model, in particular, can provide us with the probabilities of the $M^t = |V^t|^2$ potential edges between all vertices $u, v \in V^t$.

While there have been efforts to infer the edges over which information propagated from infection times, providing a Bayesian-inspired dynamic algorithm which is able to reproduce the underlying time-varying network has remains an open question. This allows us, for instance, to estimate the time constants of network evolution or infer community structure from temporal network data using cues embedded both in the probabilities over time that vertex pairs are connected by edges and in the characteristic dynamics of edge appearance and disappearance. Here, we consider the problem of developing a dynamic network inference from partial observations, i.e. vertex infection times. In particular, we propose a novel framework for providing a nonparametric network model—based on a mixture of coupled hierarchical Dirichlet processes—from diffusion data. This allows us, for instance, to infer the evolving community structure, or to obtain an explicit predictive distribution over the edges—including those that were not involved in transmission of any contagion, or are likely to appear in the future. We show the effectiveness of our approach using extensive experiments on synthetic as well as real-world networks.

To this end, we propose a novel online network inference framework, DYFERENCE, for providing nonparametric edge-exchangeable network models from partial observations. We build upon the nonparametric network model of [Wil16], namely MDND, that assumes that the network clusters into groups and then places a mixture of Dirichlet processes over the outgoing and incoming edges in each cluster while coupling the network using

a shared discrete base measure (see Section 3.2.4). However, our framework is easily extended to arbitrary generative models replacing the MDND with other choices of latent representations, such as network models presented in [CCB16, CD16, HSM16]. Given a set of cascades spreading over the network, we process observations in time intervals. For each time interval we first find a probability distribution over the cascade diffusion trees that may have been involved in each cascade. We then calculate the marginal probabilities for all the edges involved in the diffusion trees. Finally, we sample a set of edges from this distribution and provide the sampled edges to a Gibbs sampler to update the model variables. In the next iteration, we use the updated edge probabilities provided by the model to update the probability distributions over edges supported by each cascade. We continue the above iterative process until the model does not change considerably. Extensive experiments on synthetic and real-world networks show that DYFERENCE is able to track changes in the structure of dynamic networks and provides accurate online estimates of the time-varying edge probabilities for different network topologies. We also apply DYFERENCE for diffusion prediction and predicting the most influential vertices in Twitter and MemeTracker datasets, as well as bankruptcy prediction in a financial transaction network.

3.1 Nonparametric edge-exchangeable network model

In order to have a flexible model that allows the network to grow over time, we adopt the Bayesian nonparametrics model of [Wil16]. Here, the network is modeled as an exchangeable sequence of directed binary links. More specifically, each community in the network is modeled by a mixture of Dirichlet network distributions (MDND). For every community k, two Dirichlet distribution A_k, and B_k are associated to the outlinks and inlinks in community k. To allow links between the communities associated with each mixture component, the network is coupled using a shared, discrete base measure

H. The model $\mathcal{M}$ can be described as:

$$D \coloneqq (d_k, k \in \mathbb{N}) \sim \text{GEM}(\alpha) \qquad c_{uv} \sim D \qquad u, v \in V \tag{3.1}$$
$$H \coloneqq \sum_{i=1}^{\infty} h_i \delta_i \sim \text{GEM}(\gamma) \qquad u \sim A_{c_{uv}}$$
$$A_k \coloneqq \sum_{i=1}^{\infty} a_{k,i} \delta_i \sim \text{DP}(\tau, H) \qquad v \sim B_{c_{uv}}$$
$$B_k \coloneqq \sum_{i=1}^{\infty} b_{k,i} \delta_i \sim \text{DP}(\tau, H) \qquad z_{ij} = \sum_{u,v \in V} \mathbb{I}(u = i, v = j)$$

where GEM(α) is a distribution over the real-valued integers, with weights assigned according to the stick-breaking construction of a Dirichlet process with concentration parameter α that controls the number of clusters. The concentration parameter τ controls the overlap between clusters, and γ controls the total number of vertices in the network.

3.2 Dynamic network inference model

In this section, we describe our algorithm, DYFERENCE, for inferring the latent structure of the underlying dynamic network from diffusion data. We assume that contagion proceeds according to latent edges which are generated according to a MDND model. A vertex v becomes infected when it forms an edge with an infected vertex u, with probability proportional to a function of the time that has elapsed since that vertex became infected $f(\Delta_{uv}^c)$. Concretely,

$$P(e_{uv}^t | \Delta_{uv}^c, \theta, \mathcal{M}) \propto P(e | \mathcal{M}, \theta)\ f(\Delta_{uv}^c), \quad \text{for } u, v \in V^t \tag{3.2}$$

Where $P(e|\mathcal{M}, \theta)$ is the edge probability under the MDND, and θ are the parameters associated with that model, and

$$f(\Delta) = 1 + \exp(-\Delta) \tag{3.3}$$

Our task is to infer the underlying network using the probability distribution over the edges that can be formed by the contagion. The space of all possible contagion networks grows exponentially with the number of vertices, making the inference challenging. For this reason, rather than performing a full Bayesian analysis and sampling over networks, we use an approximate inference strategy. DYFERENCE works based on the following iterative idea:

1. At each time step t, we first find the set of all possible edges in each cascade c_i, denoted by $E^t_{c_i} = \{e_{uv} | t^{c_i}_u < t^{c_i}_v < \infty\}$. Then, we select the set of edges with highest probability, conditioned on the latent parameters θ and infection time differences Δ^c. This set of edges for cascade c_i have to construct a tree. This constraint allows to perform inference in a scalable manner.

2. We then provide the set of edges associated with the most probable trees as observations to a Gibbs sampler that updates the posterior distribution of the latent variables θ of our nonparametric network model $\mathcal{M}$.

3. Using MCMC generative model, we get updated probabilities, say d_{uv}, for every possible edge e_{uv} in the network.

4. We update the probability distribution over edges of each E_{c_i} using updated d_{uv}s.

5. We update the latent variables with the new set of edges.

In the following, we describe these steps in more detail.

3.2.1 Modeling observations from diffusion data

Without loss of generality, we assume that every infected vertex in a cascade c gets infected through one of its neighbors, and therefore the diffusion process propagates as a directed tree T. For a cascade c, each possible way in which a diffusion process can spread over the underlying network G creates a tree. Evidently, it is possible to have multiple infection sources, and once it's infected by the first source, it might not be affected by other former sources. However, considering multiple sources —and allowing non-tree contagion networks—may improve modeling performance. We did not explore this here for computational reasons. For any cascade c, the probability of the cascade given a diffusion tree T in a network model $\mathcal{M}$ with parameter vector θ can be written as:

$$p(t^c | T, \theta, \mathcal{M}) \propto \prod_{e_{uv} \in E_T} p(e_{uv} | \theta, \mathcal{M}), \tag{3.4}$$

where E_T is the set of edges in tree T. Note that for a given cascade, all diffusion trees have the same $|E_T| = |V_T| - 1$ number of edges, where $|V_T|$ is the number of infected vertices in the cascade.

Initially, without any prior knowledge about the structure of the underlying network, we define $\Delta^c_{uv} = t^c_v - t^c_u$ for all $t^c_u < t^c_v < \infty$, a function $f(\Delta)$, and initialize $p(e_{uv} | \theta, \mathcal{M}) \propto$

Algorithm 1 COLLECT_OBSERVATIONS

Input: Set of infection times $\{t^{c_1}, \cdots, t^{c_s}\}$.
Output: List of observations X.

```
X ← {}
for c ∈ C do
    T_c ← {}
    for all 0 < t^c < ∞ do
        u = {arg max_{i:t_i^c<t_v^c<∞} p(e_iv|θ, M)}
        T_c ← T_c ∪ {e_uv}
    end for
    X ← {X, T^c}
end for
```

$\sum_{c \in C} f(\Delta_{uv}^c)$ for all the potential edges between infected vertices in different cascades. For the rest of the edges e_{uv}, for which $t_u^c = \infty$ or $t_v^c = \infty$, in all cascades, we initialize $p(e_{uv}|\theta, \mathcal{M}) = 0$.

To build our list of observations, we only consider the most probable directed tree $\hat{T}_c$ supported by every cascade $c \in C$, i.e.,

$$\hat{T}_c = \arg\max_{T \in \mathcal{T}_c} p(t^c | T, \theta, \mathcal{M}), \tag{3.5}$$

where $\mathcal{T}_c$ is the set of all the diffusion trees supported by cascade c. Our final observation list X consists of all the edges in the most probable trees supported by the set of cascades $c \in C$.

$$X = \{E_{\hat{T}_{c_1}}, \cdots, E_{\hat{T}_{c_s}}\} \tag{3.6}$$

Note that each edge can be observed multiple times. The number of observations n_{uv} correspond to e_{uv} is the number of the most probable trees containing edge e_{uv}, i.e.,

$$n_{uv} = \sum_{c \in C} \mathbb{I}[e_{uv} \in \hat{T}_c] \tag{3.7}$$

The pseudocode for collecting the list observations from diffusion cascades is shown in Algorithm 1.

3.2.2 Updating latent model variables

To update the posterior distribution of the latent model variables conditioned on the observations, we need to resort to approximate inference. Here, we use a collapsed Gibbs sampler [TJBB05]–a Markov chain Monte Carlo (MCMC) algorithm–to approximate

Algorithm 2 UPDATE_NETWORK_MODEL

Input: Parameter vector $c_{1:M}, p_{1:M}, \beta_{1:N}$, Set of infection times $\{t^{c_1}, \cdots, t^{c_s}\}$.
Output: Updated parameter vector $c_{1:M}, p_{1:M}, \beta_{1:N}$

for $i = 1, 2, \cdots$ until convergence **do**
  $X \leftarrow$ Collect_Observations($\{t^{c_1}, \cdots, t^{c_s}\}$)
  for $j = 1, 2, \cdots$ until convergence **do**
    Select e_{uv} randomly from X
    Sample c from the conditional distribution $p(c_{uv} = k | u, v, c_{1:M}^{\neg e_{uv}}, \beta_{1:N})$ ▷ Equation 3.8
    Sample e from the conditional distribution $p(e_{uv} = e_{ij} | c_{1:M}, e_{1:M}^{\neg e_{uv}}, \beta_{1:N})$ ▷ Equation 3.9
    Sample ρ from the conditional distribution $p(\rho_{u.}^{(k)} = \rho | c_{1:M}, \rho_{u.}^{(k)\neg e_{uv}}, \beta_{1:N})$ ▷ Equation 3.10
    Sample $(\beta_1, \cdots, \beta_{|V|}, \beta_u) \sim \mathrm{Dir}(\rho_1^{(.)}, \cdots, \rho_{|V|}^{(.)}, \gamma)$ ▷ Equation 3.11
  end for
end for

the posterior distribution of the latent variable in our hierarchical mixture model (see Section 2.1.4).

Sampling *c*. Following [Wil16], we model the posterior probability for an edge e_{uv} to belong to cluster k as a function of the importance of the cluster in the network, and the importance of u as a source and v as a destination in cluster k, as well as the importance of u, v in the network. To this end, we measure the importance of a cluster by the total number of its edges, i.e., $\eta_k = \sum_{u,v \in V} [\mathbb{I}_{c_{uv}} = k]$. Similarly, the importance of u as a source, and the importance of v as a destication in cluster k is measured by the number of outlinks of u associated with cluster k, i.e. $l_{u.}^{(k)}$, as well as inlinks of v associated with cluster k, i.e. $l_{.v}^{(k)}$. Finally, the importance of a vertex i in the network is determined by the probability mass of the edges associated with it, i.e. $\beta_i = \sum h_{i.} + \sum h_{.i}$. The distribution over the cluster assignment c_{uv} of an edge e_{uv}, given the end vertices u, v, the cluster assignments for all other edges, and β is given by:

$$p(c_{uv} = k | u, v, c_{1:M}^{\neg e_{uv}}, \beta_{1:N}) \propto \begin{cases} \eta_k^{\neg e_{uv}} (l_{u.}^{(k)\neg e_{uv}} + \tau\beta_u)(l_{.,v}^{(k)\neg e_{uv}} + \tau\beta_v) & \text{if } \eta_k^{\neg e_{uv}} > 0 \\ \alpha\tau^2\beta_u\beta_v & \text{if } \eta_k^{\neg e_{uv}} = 0 \end{cases} \tag{3.8}$$

where $\neg e_{uv}$ is used to exclude the current observation. As discussed in Section 3.2, α, τ, and γ controls the number of clusters, cluster overlaps, and the number of vertices in the network. Moreover, N, M are the number of vertices and edges in the network.

Sampling e. Due to the edge-exchangeability, we can treat e_{uv} as the last variable being sampled. The conditional posterior for e_{uv} given the rest of the variables can be calculated as:

$$p(e_{uv}=e_{ij}|c_{1:M},e_{1:M}^{\neg e_{uv}},\beta_{1:N}) = \begin{cases} \sum_{k=1}^{K+} \frac{\eta_k}{M+\alpha} \frac{l_{u.}^{(k)\neg e_{uv}}+\tau\beta_u}{\eta_k+\tau} \frac{l_{.,v}^{(k)\neg e_{uv}}+\tau\beta_v}{\eta_k+\tau} + \frac{\alpha}{M+\alpha}\beta_u\beta_v & \text{if } i,j\in V \\ \sum_{k=1}^{K+} \frac{\eta_k}{M+\alpha} \frac{l_{u.}^{(k)\neg e_{uv}}+\tau\beta_u}{\eta_k+\tau}\beta_n + \frac{\alpha}{M+\alpha}\beta_u\beta_n & \text{if } i\in V, j\notin V \\ \sum_{k=1}^{K+} \frac{\eta_k}{M+\alpha} \frac{l_{.,v}^{(k)\neg e_{uv}}+\tau\beta_u}{\eta_k+\tau}\beta_n + \frac{\alpha}{M+\alpha}\beta_n\beta_v & \text{if } i\notin V, j\in V \\ \beta_n^2 & \text{if } i,j\notin V \end{cases} \tag{3.9}$$

where $\beta_n = \sum_{i=N+1}^{\infty} h_i$ is the probability mass for all the edges that may appear in the network in the future. We observe that an edge may appear between existing vertices in the network, or because one or two vertices has appeared in the network. Note that the predictive distribution for a new link to appear in the network can be calculated similarly using Equation 3.9.

Sampling ρ. The probability mass on the outlinks and inlinks of a vertex i associated with cluster k are modeled by variables $\rho_{i.}^{(k)}$ and $\rho_{.i}^{(k)}$. The posterior distribution of $\rho_{u.}^{(k)}$ (similarly $\rho_{.v}^{(k)}$), can be calculated using:

$$p(\rho_{u.}^{(k)}=\rho|c_{1:M},\rho_{u.}^{(k)\neg e_{uv}},\beta_{1:N}) = \frac{\Gamma(\tau\beta_u)}{\Gamma(\tau\beta_u+l_{u.}^{(k)})} s(l_{u.}^{(k)},\rho)(\tau\beta_u)^{\rho}, \tag{3.10}$$

where $s(l_{u.}^{(k)},\rho)$ are unsigned Stirling numbers of the first kind. I.e., $s(0,0)=s(1,1)=1, s(n,0)=0$ for $n>0$ and $s(n,m)=0$ for $m>n$. Other entries can be computed as $s(n+1,m)=s(n,m-1)+ns(n,m)$. However, for large $l_{u.}^{(k)}$, it is often more efficient to sample $\rho_{k,i}$ by simulating the table assignments of the Chinese restaurant according to Equation 3.9.

Sampling β. Finally, the distribution of edges of the vertices in the network is modeled by a Dirichlet distribution, i.e.,

$$(\beta_1,\cdots,\beta_N,\beta_n) \sim \mathrm{Dir}(\rho_1^{(.)},\cdots,\rho_N^{(.)},\gamma), \tag{3.11}$$

where $\rho_i^{(.)} = \sum_k \rho_{i.}^{(k)} + \rho_{.i}^{(k)}$.

The pseudocode for inferring the latent network variables from diffusion data is given in Algorithm 4.

Algorithm 3 DYNAMIC_NETWORK_INFERENCE (DYFERENCE)

Input: Set of infection times $\{t^{c_1}, \cdots, t^{c_s}\}$, interval length w.
Output: Updated network model $\mathcal{M}^t$ at any time $t = iw$.

1: $t = w$, initialize $\mathcal{M}^0$ randomly.
2: **while** $t <$ last infection time **do**
3: **for all** $c \in C$ **do**
4: $t^{cw} \leftarrow t_u^c \in [t - w, t)$
5: $Y^t \leftarrow \{Y^t, t^{cw}\}$
6: **end for**
7: $\mathcal{M}^t \leftarrow$ Update_Network_Model$(\mathcal{M}^{t-w}, Y^t)$
8: $t = t + w$.
9: **end while**

3.2.3 On-line dynamic network inference

In order to capture the dynamics of the underlying network and keep the model updated over time, we consider time intervals of length w. For the i-th interval, we only consider the infection times $t^c \in [(i-1)w, iw)$ for all $c \in C$, and update the model conditioned on the observations in the current time interval. There is a trade-off between the speed of tracking changes in the underlying network structure, and the accuracy of our algorithm. For smaller intervals, we incorporate new observations more rapidly, and therefore we are able to track changes faster. On the other hand, for larger intervals we have a higher number of observations from each cascade. Hence, the most probable tree supported by each cascade is larger, and can provide more information about the network structure.

Note that we don't infer a new model for the network based on infection times in each time interval. Instead, we use new observations to *update* the latent variables from the previous time interval. Updating the model with observations in the current interval results in a higher probability for the observed edges, and a lower probability for the edges that have not been observed and the size of their neighborhood –in terms of the degree of the two end vertices– hasn't been increased. Therefore, we don't need to consider an aging factor to take into account the older cascades. The pseudocode of our dynamic inference method is given in Algorithm 3.

3.2.4 Modeling choices

We note that our proposed framework is not restricted to MDND and can be used along with any non-parametric network model to capture the underlying dynamic network structure from partial observations. A weakness of our modeling approach is that the

MDND does not guarantee trees. Effectively, we are using the MDND as a prior on contagion networks, but assigning zero likelihood to networks that are not trees. The state-of-the-art methods treat contagions variously. NetInf [GRLK10] considers only the most propagation tree to achieve high efficiency. MulTtree [GRLS13] considers all propagation trees supported by each cascade. NETRATE [GrS11] and CONNIE [ML10] also consider all possible directed trees supported by each cascade. In practice, we found the MDND-based approach worked well, despite not being a tree-based prior.

Infections tend to spread through communities [LSK06], since members of communities interact frequently, motivating the choice of MDND, which models the network in terms of community structure. Communities happen in many networked systems such as social, biological, information, and technological systems. Detecting communities and their boundaries in networks is an essential task to understand the structure, function and evolution in various areas for complex networks specifically social networks. Vertices in the same community commonly share similar properties and/or play similar roles throughout the network.

Our proposed model is not identifiable, which certainly limits the ability to interpret the latent structure; however the lack of identifiability should not impact predictive performance.

3.3 Experiments

In this section, we address the following questions: (1) What is the predictive performance of DYFERENCE in static and dynamic networks and how does it compare to the existing network inference algorithms? (2) How does predictive performance of DYFERENCE change with the number of cascades? (3) How does running time of DYFERENCE compare to the baselines? And, (4) How does DYFERENCE perform for the task of predicting diffusion and influential vertices?

Baselines. We compare the performance of DYFERENCE to NETINF [GRLK10], NETRATE [GrS11], TOPOLSTM [BLG16], DEEPCAS [LMGM17], EMBEDDEDIC [BLG16] and INFOPATH [GRLS13]. INFOPATH is the only method able to infer dynamic networks, hence we can only compare the performance of DYFERENCE on dynamic networks with INFOPATH (see Section 3.2.1 for the description of the baselines).

Evaluation Metrics. For performance comparison, we use Precision, Recall, F1 score,

Map@k and Hit@k. Precision is the fraction of edges in the inferred network present in the true network, Recall is the fraction of edges of the true network present in the inferred network, and F1 score is 2×(precision×recall)/(precision+recall). MAP@k is the classical mean average precision measure and Hits@k is the rate of the top-k ranked vertices containing the next infected vertex.

In all the experiments we use a sample size of $q = |E_c| - 1$ for all the cascades $c \in C$. We further consider a window of length $w = 1$ day in our dynamic network inference experiments in Fig 3.1 and $w = 2$-years in Table 3.3.

Synthetic Experiments. We generated synthetic networks consist of 1024 vertices and about 2500 edges using Kronecker graph model [LCK$^+$10]: core-periphery network (CP) (parameters [0.9,0.5;0.5,0.3]), hierarchical community network (HC) (parameters [0.9,0.1;0.1,0.9]), and the Forest Fire model [LKF07]: with forward and backward burning probability 0.2 and 0.17.

To simulate several cascades over synthetic networks in static settings, we use the generative approach of [GrS11]. In this method one can generate a cascade set based on three transmission models (i.e. Exponential, Power-law, or Rayleigh). There are two parameters in the cascade setting. α is the transmission rate of a transmission model, and β controls the cascade length. We use the same setting as [GrS11] and set $\alpha = [0.01, 2]$ and β=0.4 (for core-periphery β=0.1). We pick the starting vertices uniformly at random and generate 500 cascades for each network. In the experiment in which we want to show the effect of number of edges on the accuracy of the model, we generate as many cascades as needed to cover 95% of the edges in the underlying network. In dynamic settings, we generate 500 cascades per time slot and considering one type of edge transmission rate for each network that we are generating the cascades over. For dynamic networks, we assign a pattern to each edge uniformly at random from a set of five edge evolution patterns: Slab, and Hump (to model outlinks of vertices that temporarily become popular), Square, and Chainsaw (to model inlinks of vertices that update periodically), and Constant (to model long term interactions) [GRLS13]. Transmission rates are generated for each edge according to its evolution pattern for 100 time steps. We then generate 500 cascades per time step (1 day) on the network with a random initiator [GrS11].

Figures 3.1a, and 3.1b compare precision, recall and F1 score of DYFERENCE to INFOPATH for online dynamic network inference on CP-Kronecker network with exponential edge transmission model, and HC-Kronecker network with Rayleigh edge transmission model.

It can be seen that DYFERENCE outperforms INFOPATH in terms of F1 score as well as precision and recall on different network topologies in different transmission models. Figures 3.1c, 3.1d, 3.1e compare F1 score of DYFERENCE compared to INFOPATH and NETRATE for static network inference for varying number of cascades over CP-Kronecker network with Rayleigh and Exponential edge transmission model, and Forest Fire network with Power-law edge transmission model. We observe that DYFERENCE consistently outperforms the baselines in terms of accuracy and is robust to varying number of cascades. This setting ensures that NETINF performs its best. As depicted in the figure, the NPNM method yields a very good results for accurately inferring the diffusion network for all types of network which emphasizes on its non-parametric nature. NETINF performs better than the other competing methods, while INFOPATH is the worst one in terms of accuracy of estimating the static diffusion network. NETRATE works well for the networks with the power-law transmission model, however its accuracy degrades in the networks with the exponential transmission model. As we expected, NETINF as the best greedy algorithm performs well, but the interesting point is that the NPNM as a non-parametric model can infer the diffusion network with similar accuracy.

Now, we evaluate the performance of the proposed method against dynamic diffusion data. Since INFOPATH is the only method among the baselines that can estimate dynamic networks, we compare the NPNM with this method in terms of Precision-Recall and accuracy in Figure 3.1. As illustrated in the figure, the proposed method almost always performs better than INFOPATH in different time slots. In Figure 3.1(a,b), the NPNM as a non-parametric network model has almost constant Precision-Recall for different time slots and also for different network models with different transmission rates. However, the Precision-Recall of INFOPATH in HC Kronecker network with Rayleigh transmission model is higher than that of CP Kronecker network with Exponential transmission model. This observation is similar for Figure 3.1(c,d) in which the accuracy of NPNM is around 0.7 for both network models with different transmission rates, however the accuracy of INFOPATH in HC network is higher than that of CP network. Moreover, the accuracy of INFOPATH has a decreasing trend over time.

Real-world Experiments. Figure 3.1(g-j) shows the Accuracy, Precision and Recall over time for the dynamic network for April 2011. We set the transmission model of the edges to exponential in sake of INFOPATH setting. We observe daily periodicity and the overall encouraging performance of around 0.4 for all three performance metrics. The results show that NPNM is more accurate than INFOPATH. Moreover, our model

Table 3.1: Performance of DYFERENCE for diffusion prediction compared to DEEPCAS, TOPOLSTM,and EMBEDDEDIC on Twitter and Memes datasets (TOPOLSTM requires the underlying network).

	Twitter			Memes		
MAP@k	@10	@50	@100	@10	@50	@100
DEEPCAS	9.3	9.8	9.8	18.2	19.4	19.6
TOPOLSTM	20.5	20.8	20.8	29.0	29.9	30.0
EMBEDDED-IC	12.0	12.4	12.5	18.3	19.3	19.4
DYFERENCE	**20.6**	**20.8**	**20.9**	**29.4**	**31.5**	**32.4**
Hits@k	@10	@50	@100	@10	@50	@100
DEEPCAS	25.7	31.1	33.2	43.9	60.5	70.0
TOPOLSTM	28.3	33.1	34.9	**50.8**	69.5	76.8
EMBEDDEDIC	25.1	33.5	36.6	35.1	56.0	65.0
DYFERENCE	**30.0**	**34.3**	**36.7**	47.4	**71.0**	**84.0**

Table 3.2: Top 10 predicted influential websites of Memes (Linkedin) on 30-06-2011. The correct predictions are indicated in bold.

DYFERENCE	INFOPATH
pressrelated.com	podrobnosti.ua
arsipberita.com	scribd.com
news.yahoo.com	derstandard.at
in.news.yahoo.com	heraldonline.com
podrobnosti.ua	startribune.com
article.wn.com	canadaeast.com
ctv.ca	news.yahoo.com
fair-news.de	proceso.com.mx
fanfiction.net	**article.wn.com**
bbc.co.uk	prnewswire.com

performs similar in all these topics because of its non-parametric nature. Furthermore, INFOPATH has higher Precision but a very low recall. Nevertheless, our method has almost similar Precision and Recall. Overall, one can find that NPNM has a better performance in comparison with INFOPATH to infer the real diffusion network.

We applied DYFERENCE to three real wold datasets, (1) Twitter [HL14] contains the diffusion of URLs on Twitter during 2010 and the follower graph of users. The network consists of 6,126 vertices and 12,045 edges with 5106 cascades of length of 17 on average,

Table 3.3: Performance of DYFERENCE for dynamic bankruptcy prediction compared to INFOPATH on financial transaction network from 2010 to 2016. In 2010, a financial crisis hit the network.

	2012			2014			2016		
MAP@k	@10	@20	@30	@10	@20	@30	@10	@20	@30
INFOPATH	4.0	5.3	6.6	35.0	34.5	30.0	54.7	65.0	65.0
DYFERENCE	**17.6**	**19.1**	**20.6**	**62.0**	**51.9**	**38.1**	**69.6**	**85.7**	**85.7**
Hits@k	@10	@20	@30	@10	@20	@30	@10	@20	@30
INFOPATH	20.0	25.0	26.6	50.0	55.0	50.0	80.0	65.0	65.0
DYFERENCE	**40.0**	**45.0**	**46.6**	**70.0**	**65.0**	**50.0**	**80.0**	**70.0**	**70.0**

(2) Memes [LBK09] contains the diffusion of memes from March 2011 to February 2012 over online news websites; The real diffusion network is constructed by the temporal dynamics of hyperlinks created between news sites. The network consists of 5,000 vertices and 313,669 edges with 54,847 cascades of length of 14 on average, and (3) a European county's financial transaction network. The data is collected from the entire country's transaction log for all transactions larger than 50K Euros over 10 years from 2007 to 2017, and includes 1,197,116 transactions between 103,497 companies. 2,765 companies are labeled as bankrupted with corresponding timestamps. In 2010, a financial crisis hit the network. For every 2 years from 2010, we built a diffusion of bankruptcy with average length of 85 between 200 bankrupted vertices that had the highest amount of transactions each year.

Figures 3.1g, 3.1h, 3.1i, 3.1j compare the F1 score of DYFERENCE to INFOPATH for online dynamic network inference on the time-varying hyperlink network with four different topics over time from March 2011 to July 2011. As we observe, DYFERENCE outperforms INFOPATH in terms of the prediction accuracy in all the networks. Figure 3.1f compares the running time of DYFERENCE to that of INFOPATH. We can see that DYFERENCE has a running time that is comparable to INFOPATH, while consistently outperforms it in terms of the prediction accuracy.

Diffusion Prediction. Table 3.1 compares Map@k and Hits@k for DYFERENCE vs. TOPOLSTM, DEEPCAS, and EMBEDDEDIC. We use the infection times in the first 80% of the total time interval for training, and the remaining 20% for the test. It can be seen that DYFERENCE has a very good performance for the task of diffusion prediction. Note that TOPOLSTM needs complete information about the underlying network structure for predicting transmission probabilities, and INFOPATH relies on predefined parametric

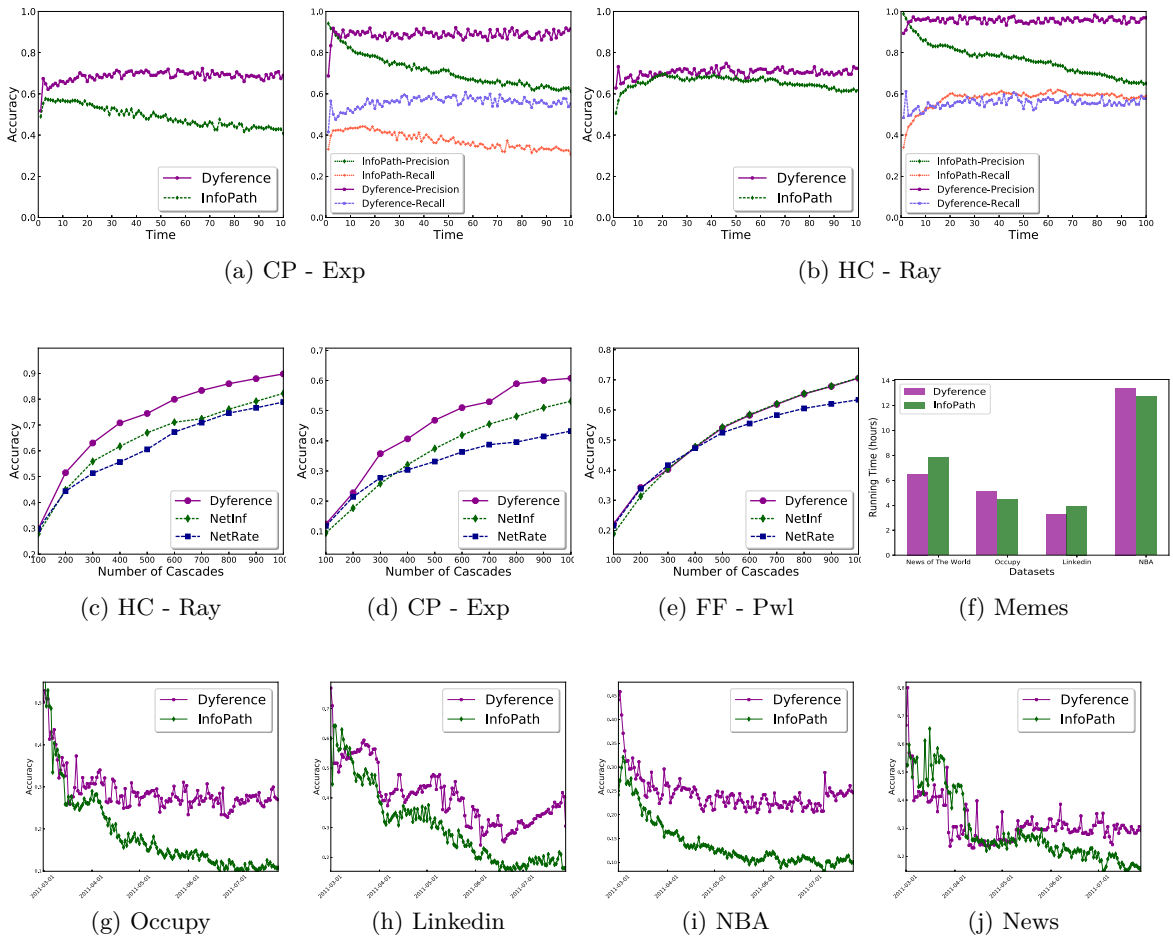
probability distributions for transmission rates. On the other hand, DYFERENCE does not need any information about network structure or transmission rates.

Table 3.3 compares Map@k and Hits@k for DYFERENCE vs. INFOPATH for dynamic bankruptcy prediction on the financial transaction network since the crisis hit the network at 2010. We used windows of length $w = 2$ years to build cascades between bankrupted vertices and predict the companies among the neighbors of the bankrupted vertices that are going to get bankrupted the next year. It can be seen that DYFERENCE significantly outperforms INFOPATH for bankruptcy prediction.

Influence Prediction. Table 3.2 shows the set of influential websites found based on the predicted dynamic Memes network by DYFERENCE vs INFOPATH. The dynamic Memes network for Linkedin is predicted till 30-06-2011, and the influential websites are found using the method of [KKT03]. We observe that using the predicted network by DYFERENCE we could predict the influential vertices with a good accuracy.

3.4 Conclusion

We considered the problem of developing generative dynamic network models from partial observations, i.e. diffusion data. We proposed a novel framework, DYFERENCE, for providing a nonparametric edge-exchangeable network model based on a mixture of coupled hierarchical Dirichlet processes (MDND). However, our proposed framework is not restricted to MDND and can be used along with any generative network models to capture the underlying dynamic network structure from partial observations. DYFERENCE provides online time-varying estimates of probabilities for all the potential edges in the underlying network, and track the evolution of the underlying community structure over time. We showed the effectiveness of our approach using extensive experiments on synthetic as well as real-world networks.



(a) CP - Exp (b) HC - Ray

(c) HC - Ray (d) CP - Exp (e) FF - Pwl (f) Memes

(g) Occupy (h) Linkedin (i) NBA (j) News

Figure 3.1: Precision, Recall and F1 score of DYFERENCE. **(a)** Compared to INFOPATH for dynamic network inference over time on Core-Periphery (CP) Kronecker network with exponential transmission model, and **(b)** Hierarchical (HC) Kronecker network with Rayleigh transmission model. **(c)** accuracy of DYFERENCE compared to INFOPATH and NETRATE for static network inference for varying number of cascades over CP-Kronecker network with Rayleigh, and **(d)** Exponential transmission model, and **(e)** on Forest Fire network with Power-law transmission model. **(f)** compares the running time of DYFERENCE with INFOPATH for online dynamic network inference on the time-varying hyperlink network with four different topics Occupy with 1,875 sites and 655,183 memes, Linkedin with 1,035 sites and 155,755 memes, NBA with 1,875 sites and 655,183 memes, and News with 1,035 sites and 101,836 memes. **(g), (h), (i), (j)** compare the accuracy of DYFERENCE to INFOPATH for online dynamic network inference on the same dataset and four topics from March 2011 to July 2011.

CHAPTER 4

Nonparametric Model for Sparse Temporal Multigraphs

Many forms of social interaction can be represented in terms of a multigraph—*i.e.* a graph where there can be multiple edges between two vertices. For example, vertices might correspond to individuals, with each edge representing an email between two individuals. In a networking context, vertices might correspond to hosts on a network, with edges representing packets of data sent or the number of packets sent between them. In large-scale applications, such multigraphs are typically sparse, with the number of edges being small relative to the number of unconnected pairs of vertices.

Edge-exchangeable models [CCB16, CD18] have been proposed as models for sparse multigraphs, and clustering-based edge-exchangeable models allow the incorporation of community-type structure, [Wil16]. Such models assume that we will see more edges and vertices in future, making them appropriate for growing graphs. However they assume that the distribution is stationary, and that the resulting multigraph is invariant to reordering the arrival times of the edges. In practice, most real-world multigraphs will be dynamic, with the underlying distribution dynamically evolving. For example, topics of communication may wax and wane in popularity, and the set of individuals interested in a given topic may evolve over time.

We propose a new model for dynamic multigraphs, the Dynamic Nonparametric Network Distribution (DNND). The DNND uses a hierarchical clustering mechanism to capture interaction patterns within the multigraph, and uses distance-dependent Chinese Restau-

rant Processes (ddCRPs) [BF11] to incorporate temporal dynamics by preferentially assigning edges to clusters and vertices that have been recently active. This dynamic mechanism, in combination with a Pitman-Yor process-distributed base measure ensures our model is appropriate for both sparse and dense multigraphs, with the degree of sparsity controlled by a single parameter. The increased flexibility allowed by our model leads to improved performance over both its exchangeable counterpart and a range of state-of-the-art dynamic network models.

Although this model is previously described in Section 3.1, we review this model in next Section in more detail.

4.1 Mixture of Dirichlet Network Distributions

While edge-exchangeable models based on a single distribution have desirable sparsity and degree distribution properties, and are appropriate for growing graphs, they lack community-type structure. The mixture of Dirichlet network distributions (MDND) [Wil16] uses a mixture of edge-exchangeable models, with shared infinite-dimensional support. Each edge i is associated with a cluster z_i, governing which edge-exchangeable model it is generated from. The z_i are distributed according to a Chinese restaurant process (CRP), which describes the distribution over partitions in a DP mixture model, allowing an unbounded number of clusters.
Within the k-th cluster, each edge is associated with two "tables", $b_i^{(s)}$ and $b_i^{(r)}$, sampled from two separate CRPs.[1] These tables are associated with the sender and recipient of the edge, respectively. Each table $b_i^{(s)}$ (or $b_i^{(r)}$) in cluster k is linked with a vertex $\phi_{k,b_i}^{(s)}$ (or $\phi_{k,b_i}^{(r)}$, so that, for example, all edges i where $z_i = k$ and $b_i^{(s)} = c$ have the same sender $\phi_{k,c}^{(s)}$. To ensure that the cluster-specific multigraphs have common support, the vertices for each table are sampled from a global, DP-distributed probability measure H, that is shared across all clusters. The overall distribution over the ith directed edge (s_i, r_i) can

[1] While the original MDND paper describes this distribution in terms of Dirichlet processes, we will provide an equivalent construction in terms of CRPs, to provide consistency with later contributions.

be written as

$$
\begin{aligned}
H &\sim \mathrm{DP}(\gamma, \Theta) \\
P(z_i = k | z_{<i}) &\propto \begin{cases} m_k & k \leq K_+ \\ \alpha & k = K_+ + 1 \end{cases} \\
P(b_i^{(s)} = c | z_i = k, b_{<i}^{(s)}) &\propto \begin{cases} \rho_{k,c}^{(s)} & c \leq C_{k+}^{(s)} \\ \tau & k = C_{k+}^{(s)} + 1 \end{cases} \\
P(b_i^{(r)} = c | z_i = k, b_{<i}^{(r)}) &\propto \begin{cases} \rho_{k,c}^{(r)} & c \leq C_{k+}^{(r)} \\ \tau & k = C_{k+}^{(r)} + 1 \end{cases} \\
\phi_{k,c}^{(s)} &\sim H \qquad s_i = \phi_{z_i,b_i}^{(s)} \\
\phi_{k,c}^{(r)} &\sim H \qquad r_i = \phi_{z_i,b_i}^{(r)}
\end{aligned}
\tag{4.1}
$$

where m_k is the number of times we have seen cluster k; $z_{<i} = (z_1, \ldots, z_{i-1})$; $s_i(r_i)$ is the sender(recipient) of edge i; $\rho_{k,c}^{(s)}$ is the number of senders sat at table c in cluster k (with $\rho_{k,c}^{(r)}$ defined analogously); K_+ is the number of previously seen clusters; and $C_{k+}^{(s)}$ and $C_{k+}^{(r)}$ are the number of previously seen sender and recipient tables, respectively, in cluster k. Here, $\alpha > 0$ controls the number of clusters, $\gamma > 0$ controls the overall number of vertices, τ controls similarity between clusters, and Θ is some diffuse measure on the space of vertices.

The resulting model exhibits structure due to the clustering of edges. Both the distribution over clusters, and the distribution over edges within a cluster, are exchangeable. While the use of CRPs (as opposed to heavier-tailed distributions such as Pitman-Yor processes and normalized generalized gamma processes) allows for straightforward inference, they mean that the MDND does not yield sparse graphs.

4.2 Distance-dependent Chinese restaurant process

Under the MDND both the distribution over clusters, and the distributions over sender and recipient vertices within each cluster, are exchangeable. A natural way to incorporate temporal dependence in such a model is to replace the associated CRPs with temporally varying clustering mechanisms. Several models add temporal dynamics to the DP and/or CRP, *e.g.* [Mac00, LGF10, RDC08]. For our purposes, we choose to use the distance-dependent CRP (ddCRP) [BF11].

Under the ddCRP with concentration parameter α and non-negative, non-increasing decay function f such that $f(\infty) = 0$, the probability of an observation x_i joining a cluster k is,

$$P(z_i = k|z_{<i}) \propto \begin{cases} \sum_{j:z_j=k} f(d_{i,j}) & k \leq K_+ \\ \alpha & k = K_+ + 1 \end{cases}$$

where K_+ is the number of previously seen clusters;

$$d_{i,j} = \begin{cases} t_i - t_j & t_i \geq t_j \\ \infty & \text{otherwise} \end{cases} \tag{4.2}$$

captures how much time has elapsed between x_i and x_j; and the concentration parameter α controls the expected number of clusters.

4.3 Pitman-Yor process

The DP distribution over H and the CRPs used to assign edges to tables do not have sufficiently heavy tails to yield sparse multigraphs (see [CCB16] and [CD18]). The Pitman-Yor process (see Section 2.1.2), is an alternative distribution over probability distributions that has heavier tails than the DP. The Pitman-Yor process has previously been used in the construction of sparse, edge-exchangeable multigraphs, albeit without the structure of the MDND [CD18].

The exchangeable distribution over partitions implied by the Pitman-Yor process can be described using a two-parameter CRP. With discount parameter $0 \leq \sigma < 1$ and concentration parameter $\gamma > -\sigma$, the clustering behavior of the two-parameter CRP is given by

$$P(z_i = k|z_{<i}) \propto \begin{cases} m_k - \sigma & k \leq K_+ \\ \gamma + \sigma K_+ & k = K_+ + 1 \end{cases} \tag{4.3}$$

where K_+ is the number of existing clusters and m_k is number of times we have seen cluster k. If $\sigma = 0$, this reduces to the standard CRP with concentration parameter γ; as σ increases, the distribution has increasingly heavy tails.

4.4 DNND: Dynamic Nonparametric Network Distribution

The MDND, described in Section 4.1, assumes that the observed edges are exchangeable—*i.e.*, that the probability of the graph is invariant to reordering of the edge arrival times. This obviously limits their application to dynamically growing graphs, where the underlying mechanism is non-stationary. Further, since it is based on DPs rather than heavier-tailed distributions such as the normalized generalized gamma process or the Pitman-Yor process, it does not concentrate on sparse graphs for any parameter settings.

Inspired by the MDND, we propose a new model, Dynamic Nonparametric Network Distribution (DNND), for dynamic multigraphs with community structure. DNND replaces the Dirichlet process-distributed random measures in Equation (4.1) with distributions that are either temporally varying, or have heavy tails. As we show in this section, the resulting time-evolving models can capture both sparse and dense multigraphs, depending on the hyperparameters.

Recall that the MDND uses DPs or CRPs in three parts of its construction (see Equation (4.1)): $H \sim \mathrm{DP}(\gamma, \Theta)$ controls the number of vertices, and their overall popularity within the graph. CRPs parametrized by τ determine the tables at which senders and recipients sit, in turn controlling the cluster-specific distributions over the "sender" and "recipient" of edges in the graph. Finally, a CRP parametrized by α governs the clustering structure of the edges.

We replace the top-level DP with a Pitman-Yor process, which increases the probability of adding previously unseen vertices. We replace the CRP controlling the overall clustering with a ddCRP with decay function f_1, and the CRPs controlling the cluster-specific table allocations with ddCRPs with decay function f_2.

The resulting temporal dependency means that both the cluster probabilities, and the cluster-specific distributions over vertices, can evolve over time. The resulting generative

process over directed edges (s_i, r_i) takes the form

$$
\begin{aligned}
H &:= \sum_{i=1}^{\infty} h_i \delta_{\theta_i} \sim \mathrm{PY}(\gamma, \sigma, \Theta) \\
P(z_i = z | z_{<i}) &\propto \begin{cases} \sum_{j:z_j=k} f_1(d_{i,j}) & k \leq K_+ + 1 \\ \alpha & k = K_+ \end{cases} \\
s_i | z_i, s_{<i}, H &\begin{cases} = s & \text{w.p.} \ \propto \sum_{j:z_j=z_i, s_j=s} f_2(d_{i,j}) \\ \sim H & \text{w.p.} \ \propto \tau \end{cases} \\
r_i | z_i, r_{<i}, H &\begin{cases} = r & \text{w.p.} \ \propto \sum_{j:z_j=z_i, r_j=r} f_2(d_{i,j}) \\ \sim H & \text{w.p.} \ \propto \tau \end{cases}
\end{aligned}
\tag{4.4}
$$

where K_+ is the number of previously seen clusters. The hyperparameters α, γ and τ influence the number of clusters and components in a manner analogous to Equation 4.1; the parameter $\sigma \in [0, 1)$ controls the sparsity of the multigraph, as we will explore in Section 4.4.1. To reduce the expected number of loops (edges with repeat vertices), we specify our distance as

$$
d_{i,j} = \begin{cases} t_i - t_j & t_i \geq t_j, i \neq j \\ \infty & \text{otherwise,} \end{cases}
\tag{4.5}
$$

4.4.1 Sparsity of the resulting graph

In this section, we show that, under mild conditions, if $\sigma > 0.5$ the model described by Equation (4.4) yields sparse multigraphs. Following [CCB16], we define a multigraph as sparse if the number of edges grows sub-quadratically with the number of vertices. Throughout, we make the following assumption on decay functions used in ddCRP.

Assumption 1. *The decay function f_2 in Equation 4.4 satisfies $\sum_{i=1}^{j} f_2(d_{i,j}) \leq D$ for some $D < \infty$ and all j.*

Remark. *The condition $\sum_{i=1}^{j} f_2(d_{i,j}) \leq D$ is easily satisfied provided the rate of arrival of edges is bounded. For example, if f_2 is a window function of size w, then D is the maximum number of edges arriving in a period of length w. If f_2 is an exponential function, $f_2(d) = e^{-d/\lambda}$, and m is the maximum number of edges arriving per unit time,*

then

$$\sum_{i<j} f_2(d_{i,j}) \leq \sum_{i<j} e^{-\lfloor d_{i,j} \rfloor/\lambda} \leq \sum_{\ell=0}^{\infty} m e^{-\ell/\lambda} = \frac{m e^{-\lambda}}{e^{-\lambda}-1}$$

where the final inequality is due to the fact that there are at most m observations with $\lfloor d_{i,j} \rfloor = \ell$ for $\ell = 1, \ldots, \infty$. If f is a logistic function, $f(d) = e^{-d+\lambda}/(1+e^{-d+\lambda})$, then $\sum_{i<j} f(d_{i,j})$ is bounded above by $\sum_{\ell=0}^{\infty} e^{\lambda} e^{-\ell} = e^{\lambda+1}/(e-1)$.

Theorem 4.4.1. *If f_2 satisfies Assumption 1, and if $\sigma > 0.5$, the model described by Equation (4.4) is sparse.*

Before proving Theorem 4.4.1, we introduce some lemmas that are required for our main result.

Lemma 4.4.2. *If the decay function of a ddCRP satisfies Assumption 1, the number of clusters grows linearly in the number of observations, n.*

Proof. The probability of the i-th observation starting a new cluster is $\frac{\tau}{\tau+\sum_{j<i} f(d_{i,j})}$, therefore the expected number of clusters in n observations is

$$\mathbb{E}[K_n] = \sum_{i=1}^{n} \frac{\tau}{\tau + \sum_{j<i} f(d_{i,j})} \geq \frac{n\tau}{\tau + b}$$

□

Next, define a hierarchical Pitman-Yor ddCRP as a time-dependent clustering model for grouped data. Let $x^1, \ldots x^K$ represent K data groups, where $x^k = (x_1^k, \ldots, x_{n_k}^k)$ contains n_k observations. Associate with each observation x_i^k a table indicator b_i^k, and associate with the c-th table in group k a dish $\phi_{k,c}$, according to

$$\begin{aligned} H \sim& \mathrm{PY}(\gamma, \sigma, \Theta) \\ P(b_i^k = c | z_{<i}^k, \tau) \propto& \begin{cases} = \sum_{j:b_j^k=c} f_2(d_{i,j}) & c \leq C_+^k \\ \tau & c = C_+^k + 1 \end{cases} \\ \phi_{k,c} \sim& H, \quad \theta_i^k = \phi_{k,b_i^k}. \end{aligned} \tag{4.6}$$

observations with the same value of θ_i^k belong to the same cluster. This formulation is a slight variation on the spatial ddCRP of [GUSB11], with a Pitman-Yor process replacing of the top-level DP.

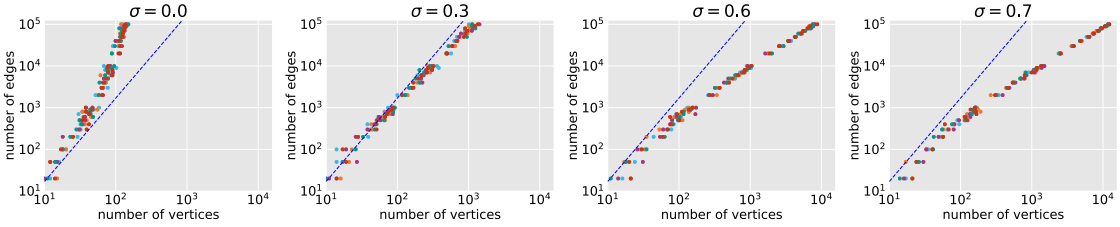


Figure 4.1: Relationship between the number of edges and the number of vertices in DNND multigraphs generated according to Equation (4.4), for various values of σ. Plots are shown on a log-log scale. Different colors correspond to different random seeds. The blue dashed line has a slope of 2, indicating a quadratic relationship. We see the multigraphs become increasingly sparse as σ increases.

Lemma 4.4.3. *If the decay function satisfies the condition in Lemma 4.4.2, then the number of clusters in the hierarchical Pitman-Yor ddCRP grows as $O(n^\sigma)$.*

Proof. Following Lemma 4.4.2, the number of tables associated with each group x^k grows linearly in the number n_k of observations in the group, so the total number of tables across all groups grows linearly in the total number $n = \sum_k n_k$ of observations. Tables are then assigned dishes according to a Pitman-Yor CRP (Equation (4.3)). Under this distribution, the expected number of dishes in k is [P+02],

$$\frac{\Gamma(\gamma + k + \sigma)\Gamma(\gamma + 1)}{\sigma\Gamma(\gamma + k)\Gamma(\gamma + \sigma)} - \frac{\gamma}{\sigma} \simeq \frac{\Gamma(\gamma + 1)}{\sigma\Gamma(\gamma + \sigma)} k^\sigma .$$

By the law of iterated expectations, the expected number of dishes sampled by n customers grows as $O(n^\sigma)$ for $\sigma > 0$. □

Note that the above proof does not depend on how observations are assigned to groups, and therefore the lemma holds if the groups are assigned via a ddCRP (or other clustering mechanism).

Proof of Theorem 1. Following from Lemma 4.4.3, the expected number of distinct vertices grows as $O(n^\sigma)$, where n is the number of edges. The resulting random multigraph is therefore sparse provided $\sigma > 0.5$. □

Empirical evaluation of sparsity

In Figure (4.1), we empirically investigate the effect of σ on the sparsity of multigraphs generated according to Equation (4.4). Recall that a model is considered sparse if the

number of edges grows subquadratically with the number of vertices. Figure (4.1) plots the number of edges and number of vertices (on a log-log scale) for multigraphs with $\alpha = 1$, $\gamma = 1$, $\tau = 0.2$, and various values of σ. Each dot represents a single sampled multigraph, and the blue dashed line has slope 2—providing the boundary between sparse and dense graphs. We see that, for $\sigma > 0.5$, the number of edges grows subquadratically with the number of vertices, as expected according to Theorem 4.4.1. As we decrease σ, the graphs become denser. With $\sigma = 0.3$, the number of edges is approximately quadratic in the number of vertices. With $\sigma = 0$, corresponding to a DP-distributed base measure, the number of edges is superquadratic in the number of vertices.

4.4.2 Discussion of modeling choices

A number of dynamic extensions to the Dirichlet process have been proposed, that could be used to introduce temporal dependency into the MDND. We choose the ddCRP for three reasons. First, it allows us to guarantee sparsity, as described in Section 4.4.1. Second, it captures behavior that we are likely to see in a dynamic network context: recent communications are likely to be more influential than more distant communications. Finally, its construction easily lends itself to an easy-to-implement sampler, as described in Section 4.5; by contrast, many other dependent Dirichlet processes have much more complicated inference algorithms, which would limit scalability.

A known limitation of the ddCRP is that it assumes that all data has been observed up to the current time point; the distribution is not invariant to adding edges at previously observed time points. This is not a concern in our setting, since we are typically able to observe past instances of the full graph, and are interested in predicting future edges.

We note that the hierarchical structure over the cluster-specific distributions is similar to the spatial ddCRP proposed by [GUSB11], where a collection of ddCRPs are coupled using a shared Dirichlet process. We choose to use a Pitman-Yor process in place of the Dirichlet process to ensure sparsity. Alternative heavy-tailed nonparametric distributions such as the normalized generalized gamma process could be used in place of the Pitman-Yor process; we chose the Pitman Yor process for ease of inference.

4.5 Inference

Conditioned on the hyperparameters, we can directly generate samples from the posterior by sequentially sampling cluster assignments. Let b_i^s and b_i^r be the table assignments

Algorithm 4 Dynamic Nonparametric Network Distribution Inference

Input: Sequence $E = (e_1, e_2, \ldots)$, $\alpha_0, \gamma_0, \tau_0, \sigma_0$

```
Compute distances {d_{i,j}}                                   ▷ Equation (4.5)
for epoch= 1, 2, ...,max_epoch do
    for i = 1, ..., n do
        Sample z_i|z_{<i}, s_i, r_i ...                       ▷ Equation (4.7)
        Sample b_i^s|s_i, z_i, ...                            ▷ Equation (4.9)
        Sample b_i^r|r_i, z_i, ...                            ▷ Equation (4.9)
    end for
    Sample α, γ, τ, σ via random walk Metropolis Hastings
end for
```

of the i-th sender and recipient, respectively. As we proceed, let K_+ indicate the total number of clusters seen so far; V_+ be the total number of vertices seen so far; and ρ_v be the total number of tables so far associated with vertex v. We can sample the i-th cluster assignment, given the previous assignments, as

$$P(z_i = k|s_i, r_i) \propto P(s_i|z_i = k)P(r_i|z_i = k)P(z_i = k|z_{<i}) \tag{4.7}$$

where

$$P(s_i = v|z_i = k) = \begin{cases} \frac{\sum_{j:j<i, z_j=k, s_j=v} g(d_{i,j})}{\tau + \sum_{j:j<i, z_j=k} g(d_{i,j})} & \\ + \frac{\tau}{\tau + \sum_{j:j<i, z_j=k} g(d_{i,j})} \frac{\rho_v - \sigma}{\sum_v \rho_v + \gamma} & v \leq V_+ \\ \frac{\tau}{\tau + \sum_{j:j<i, z_j=k} g(d_{i,j})} \frac{\gamma + V\sigma}{\sum_v \rho_v + \gamma} & v = V_+ + 1 \end{cases}$$

$$P(z_i = k|z_{<i}) \propto \begin{cases} \sum_{j:z_j=k} f_1(d_{i,j}) & k \leq K_+ \\ \alpha & k = K_+ + 1 \end{cases} \tag{4.8}$$

and $P(r_i = v|z_i = k)$ is defined analogously to $P(s_i = v|z_i = k)$. Conditioned on the cluster assignment, we sample a table b_i^s for the sender as

$$P(b_i^s = c|s_i, z_i) \propto \begin{cases} \sum_{\substack{j:z_j=z_i, \\ s_j=s_i, b_j^s=c}} f_2(d_{i,j}) & \text{existing table} \\ \tau \frac{\rho_{s_i} - \sigma}{\sum_v \rho_v + \gamma} & \text{new table} \end{cases}. \tag{4.9}$$

Once we have sampled the complete set of assignments, we can sample the hyperparameters using Metropolis Hastings steps (see [BF11] for details on evaluating the likelihood

for the ddCRP). We alternate between these two steps until convergence. The procedure is summarized in Algorithm 4.

Sampling H Inspired by augmented representation schemes for the hierarchical Dirichlet process [TJBB05], we represent the infinite measure H using a finite-dimensional vector $(h_1, h_2, \ldots, h_{V_n}, h_{V_{n+1}})$, where h_i for $i \leq V_{n+1}$ is the probability associated with node i, and $h_{V_{n+1}}$ is the probability associated with previously unseen nodes. Note that, in practice, V_n is constant during train time. Let

$$\eta_v = \sum_i I(b_i^s = i) + I(b_i^r = i)$$

—note that this corresponds to the number of tables in a Chinese restaurant franchise representation [TJBB05]. Following Corollary 20 of [Pit96], we can sample

$$(h_1, \ldots, h_V, h_+) \sim \text{Dirichlet}(\eta_1 - \sigma, \ldots, \eta_V - \sigma, \gamma + V\sigma)$$

Sampling b_i^s and b_i^r Conditioned on the clusters, we can Gibbs sample b_i^s (and similarly, b_i^r) based on the conditional distribution

$$P(b_j^s | \mathbf{c}, \mathbf{s}, \mathbf{r}) \propto \begin{cases} f_2(d_{ij}) & z_i = z_j, j < i, s_i = s_j \\ \tau h_{s_i} & i = j \\ 0 & z_i \neq z_j \\ 0 & s_i \neq s_j. \end{cases} \tag{4.10}$$

4.6 Experiments

In this section, we address the following questions: (1) How well does DNND capture the underlying multigraph behavior to predict unseen held-out edges? and (2) How accurate is DNND in terms of forecasting future interactions, compared to state-of-the-art dynamic interaction graph models?

Datasets. We evaluated our model on four real-world temporal multigraphs:

Algorithm 5 DNND-Inference

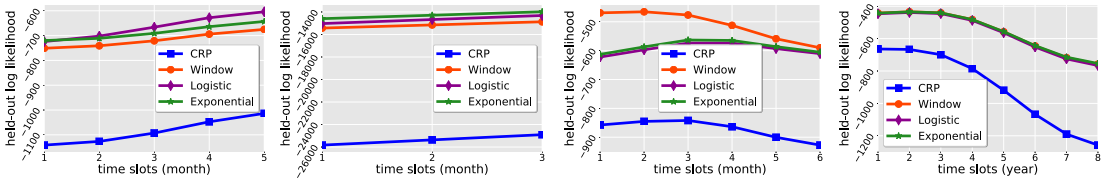
initialize c_i uniformly at random from $\{1, \dots, N\}$ for each edge i
initialize link_i to previous edge in its cluster for each edge i
compute $f_{d_{ij}}$ for each edge i where $j = \{1, \dots, i-1\}$ ▷ For the priors
for $t \in \{1, \dots, T\}$ **do**
  sample $H \sim \text{Dir}(\eta_1 - \sigma, \dots, \eta_V - \sigma)$
  sample hyper-parameters $\alpha, \gamma, \tau, \sigma$ and windows of functions f_1, f_2
  for $i \in \{1, \dots, N\}$ **do**
    sample $z_i \propto \begin{cases} f_1(d_{ij}) - \sigma & j < i \\ \alpha + K\sigma & j = i \end{cases}$
    $P(c_i) = \log p(z_i)$
    if $z_i < K$ **then**
      sample $b_i^{(s/r)} | z_i \propto H = \begin{cases} \sum_{j:z_j=z_i} \frac{f_2(d_{ij})}{\sum_{j:z_j=z_i} f_1(d_{ij})+\tau} \\ +\left(\frac{\tau}{\sum_{j:z_j=z_i} f_1(d_{ij})+\tau} \frac{t_i^{(s/r)}}{\sum_l t_l^{(s)}+t_l^{(r)}+\gamma}\right) & t_i^{(s/r)} > 0 \\ \frac{\tau}{\sum_{j:z_j=z_i} f_1(d_{ij})+\tau} \frac{\gamma}{\sum_l t_l^{(s)}+t_l^{(r)}+\gamma} & t^{(s/r)} = 0 \end{cases}$
    else
      sample $b_i^{(s/r)} | z_i \propto H = \begin{cases} \frac{t_i^{(s/r)}}{\sum_l t_l^{(s)}+t_l^{(r)}+\gamma} & t_i^{(s/r)} > 0 \\ \frac{\gamma}{\sum_l t_l^{(s)}+t_l^{(r)}+\gamma} & t^{(s/r)} = 0 \end{cases}$
    end if
    $L(c_i|X) = \log p(b_i^{(s)}) + \log p(b_i^{(r)})$
    sample link_i from L and set $c_i = c_{\text{link}_i}$
  end for
end for

- *CollegeMsg network* [2] [POC09] records private messages in an online social multi-graph at the University of California, Irvine, with 1,899 vertices and 20,296 interactions over 193 days. The average monthly sparsity[3] is 0.013.
- *Email-Eu-core temporal network*[4] [PBL17] consists of all incoming and outgoing emails in a large European research institution. 986 individuals exchange 332,334 separate e-mails over 803 days. We considered a subset of the first 3 months, with 49,282 edges and 702 vertices, with an average monthly sparsity of 0.028.
- *Social Evolution network (SocialEv)*[5] [MCM+11] tracks the everyday life of 70

[2] http://snap.stanford.edu/data/CollegeMsg.html
[3] The sparsity of a network at time slot (month) t, with V_t being its number of vertices, and E_t being its number of edges is computed as: $\frac{E_t}{V_t(V_t-1)}$. The average monthly sparsity is then averaged of the sparsity over time slots.
[4] http://snap.stanford.edu/data/email-Eu-core-temporal.html
[5] http://realitycommons.media.mit.edu/socialevolution.html

Table 4.1: Predictive log likelihood of held-out edges on four real-world datasets (mean ± standard deviation over time slots).

	MDND (CRP)	DNND-Window	DNND-Logistic	DNND-Exponential
CollegeMsg	-1084.37±18.73	-716.80±9.72	**-665.45±10.78**	-686.21±11.04
Email-Eu	-25364.11±355.17	-15151.65±206.36	-14672.37±279.81	**-14289.64±223.70**
SocialEv	-872.73±53.41	**-577.06±42.54**	-595.81±41.64	-586.57±41.97
DBLP	-906.00±37.27	-559.83±23.35	-565.58±23.46	**-558.53±23.21**



(a) CollegeMsg (b) Email-Eu (c) SocialEv (d) DBLP

Figure 4.2: Predictive log likelihood vs. time slots. Each evaluation is the average value over 20 samples.

students within a dormatory, based on mobile phone data. We consider Bluetooth connections, calls and SMSs as interactions, yielding a multigraph high clustering coefficient and about $1M$ events over six months. We sampled a subset of edges, which resulted in $6,140$ edges and 323 vertices, with average monthly sparsity of 0.414.

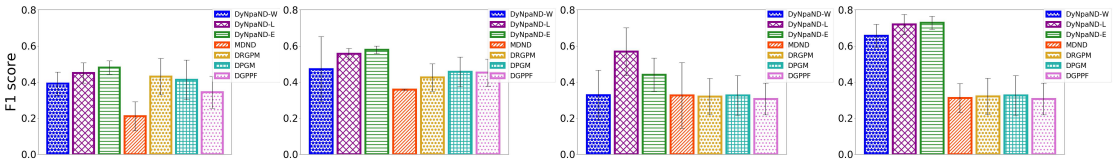
- *DBLP* [APU09] contains coauthorship information for more than 800,000 computer science publications among 958 authors over ten years (1997-2006). We consider the 324 most connected authors, yielding a multigraph with $11,154$ edges and average annual sparsity of 0.022.

Experimental setting. For DNND, we considered three decay functions: (1) Window decay: $f(d) = 1[d < \lambda]$ only considers dependency with edges that are distant at most λ from the current edge, (2) Exponential decay: $f(d) = e^{-d/\lambda}$ decays exponentially with time, and (3) Logistic decay: $f(d) = \frac{e^{-d+\lambda}}{1+e^{-d+\lambda}}$ is a smooth version of the window decay. We used the same decay function for f_1 and f_2. We explored several values for λ and found that $\lambda = 5$ for window decay and $\lambda = 1.5$ for other decays work well on all datasets. We ran all algorithms for 100 iterations for each dataset. We used Gamma$(5, 1)$ priors for α, β, and τ, and a Beta$(1, 1)$ prior for σ.

Baselines. For the held-out edge prediction task, we compared against the (stationary) MDND, described in Section 4.1. We implemented MDND using the inference algorithm in Section 4.5, with $\sigma = 0$ and $d_{i,j} = 1$ for all $i \geq j$; we found that this algorithm gave

Table 4.2: MAP@k for the future interaction prediction task (mean value over time slots).

	DNND			MDND			DRGPM			DPGM			DGPPF		
MAP@k	@10	@50	@100	@10	@50	@100	@10	@50	@100	@10	@50	@100	@10	@50	@100
CollegeMsg	**0.466**	**0.504**	**0.516**	0.267	0.295	0.321	0.118	0.091	0.085	0.037	0.071	0.067	0.052	0.085	0.065
Email-Eu	**0.390**	0.452	**0.585**	0.238	0.227	0.351	0.330	**0.464**	0.494	0.308	0.451	0.490	0.199	0.380	0.451
SocialEv	**0.586**	**0.515**	**0.423**	0.179	0.162	0.253	0.001	0.008	0.009	0.033	0.011	0.024	0.008	0.014	0.021
DBLP	**0.493**	0.496	0.492	0.205	0.118	0.241	0.388	0.459	0.451	0.451	**0.516**	**0.498**	0.374	0.485	0.476



(a) CollegeMsg (b) Email-Eu (c) SocialEv (d) DBLP

Figure 4.3: F1 score for future interaction prediction. Decay functions for DNND are Window (W), Exponential (E), and Logistic (L) (averaged over time slots).

comparable results to the implementation of [Wil16].

For the graph forecasting task, we also compared against three recent Bayesian dynamic network models, introduced in Section 2.4.2: the dynamic relational gamma process model (DRGPM) [YK18a], the dynamic Poisson gamma model (DPGM) [YK18b], and the dynamic gamma process Poisson factorization (DGPPF) [AGZ15]. These models are not applicable to the held-out edge prediction task, since they assume all vertices are observed. However, they can be used to predict the entire graph at the next time slot over seen vertices. We modify this distribution to be appropriate to our forecasting task by predicting the n_T edges with the highest probability at time slot T.

4.6.1 Prediction of held-out edges

A common method to evaluate a Bayesian model is to infer a distribution over parameters given an incomplete training set, and look at the probability of the unseen held-out set given that distribution. The higher the likelihood of the held-out set values, the better the model performance.

The held-out log likelihood allows us to evaluate whether our model is a good fit for the data; we use this to evaluate whether incorporating time-dependence and sparsity allows us to better capture variation in the data compared with the MDND. Estimating test set log likelihood can be tricky in models where we have latent variables for each data point, since the likelihood depends heavily on the assignments of those latent variables, and the state space of assignments is too large to explore exhaustively.

To evaluate DNND on this task, we split each data set into time slots (one month for CollegeMsg, Email-Eu-core, and SocialEv; one year for DBLP), and train on 85% of interactions in each time slot. We estimate the log predictive likelihood of the remaining interactions using a "Left-to-Right" evaluation algorithm [WMSM09].

Table (4.1) shows the predictive log likelihood computed by DNND using three different decays (*i.e.* Window, Exponential and Logistic) in comparison with the CRP decay function used in [Wil16] on four real multigraphs. We see that in each case, all three dynamic DNND multigraphs outperform the stationary MDND. This behaviour is expected as the theoretical results showed that considering time dependent mixture model with a sparsity parameter, provide a better fit than the exchangeable CRP mixture. In Figure (4.2), we illustrate the predictive log likelihood per time slot. Again, we see that DNND outperforms the stationary MDND model across all time slots. This demonstrates that considering important properties observed in real-world data (*i.e.* time dependency and sparsity) results in a better log likelihood. On the CollegeMsg, DBLP and Email-Eu datasets, all the decays are performing almost the same. On the SocialEv dataset, a window decay with a window size 5 performs best.

4.6.2 Forecasting future interactions

While log likelihood allows us to compare models, it does not give us a quantitative measure of how much better a model will perform on a concrete prediction task for future observations. To assess this, we consider two metrics for evaluating future predictions when we have all observations before time T, and want to predict the edges arriving at time T. To allow comparison with vertex-exchangeable models, which predict the entire adjacency matrix for a single time step, we set the time stamp for all test set edges to the time of the first test set edge. We assume we know the total number, N_{test}, of edges in the test set, allowing us to return a predicted set of appropriate size, along with their probability of appearance. For comparison methods, we selected the N_{test} edges with highest probability of appearing.

For performance comparison, we look at the F1 score and MAP@k. F1 score is 2×(precision×recall)/(precision+recall). Precision is the fraction of edges in the predicted graph present in the true graph, and recall is the fraction of edges of the true graph present in the predicted graph. MAP@k is the classical mean average precision measure. We calculated values based on 10 posterior samples.

Table (4.2) shows the MAP@k for DNND and the four comparison methods. For space reasons, the numbers reported for DNND represent the best result of the three decay functions. Figure (4.3) summarizes the corresponding F1 scores.The results show that DNND performs comparably or better on both metrics across all datasets. We hypothesize that this is due to several reasons. First, DNND is explicitly designed in terms of a predictive distribution over edges, making it well-suited to predicting future edges. Second, DNND is able to increase the number of vertices over time, and is likely better able to capture natural multigraph growth. Conversely, the other methods assume the number of vertices is fixed—and explicitly incorporate the absence of edges at earlier time points into the likelihood. Third, unlike the other methods, DNND allows us to capture sparsity; if the multigraph is sparse, then this should lead to a more accurate model.

4.7 Conclusion

We have presented a new distribution for temporally varying, structured multigraphs, that allows us to represent both sparse and dense networks. Since our model explicitly describes a sequence of edges, it is well-suited to predict future edges. As we saw in Section 4.6, these properties translate into impressive predictive performance compared with state-of-the-art Bayesian models.

In this chapter, we incorporate dynamics using a ddCRP model, which encourages edges to belong to clusters that have been recently active. An interesting avenue for future research would be to explore alternative forms of dependency, and incorporate mechanisms that can capture link reciprocity [BBH12].

CHAPTER 5

Compressive Sensing Framework for Sparse Recovery in Networks

In the last two chapters, our focus was mainly on Bayesian modeling of graphs, considering time and sparsity as the two major properties of real-world networks. In this chapter, we explore the use of non-Bayesian methods in a graph context. More specifically, we address a problem in a graph, either dense or sparse, which requires analyzing sparse data. We focus on the problem of how to distinguish the importance of each individual vertex in the graph, which is a key question in structural analysis of real-world networks. Centrality refers to identifiers that indicate the most important vertices in a network, and centrality measures quantify the role of vertices from different points of view. The most popular centrality measure from the information flow standpoint is the *betweenness centrality*.

Betweenness centrality (see Section 2.5) measures the proportion of shortest paths in the network passing through a specific vertex. This is the fraction of times that a vertex acts as a bridge in transferring any valuable information between any pair of vertices along their shortest paths. High betweenness centrality individuals play significant roles in the spread of propaganda, ideologies, or gossips in social networks. As an example, in the advertisement industry, in order to improve the effectiveness of word-of-mouth advertising and increase the recommendation-based product adoption, companies are interested to give away free product samples to central individuals instead of random users. For example, CNET stated that Samsung gives iPhone users a trial run with new

Galaxy smartphones [Gee17]. To this end, Samsung tried to identify dissatisfied iPhone owners who are the most important users in terms of communicating with other users, thus they are the most influential in spreading word-of-mouth recommendation, and then offered free Galaxy phones to some of them.

In this chapter, we introduce CS-HiBet, a new method to efficiently detect top-k central vertices in networks, using compressive sensing. We first formalize this problem and introduce the notion of compressive sensing and its application in graph-structured data. Then, we describe our proposed method for recovering the most important vertices of the network, and analyze its complexity. Finally, we experimentally evaluate the performance of our method compared to the state-of-the-art methods under various configurations.

5.1 Problem definition

Suppose every vertex $i \in V$ in the network $G = (V, E)$ has a real value x_i (*e.g.* betweenness centrality value over vertex i), and vector $x = (x_1, x_2, ..., x_N)$ is associated with the set V. If $\|x\|_0 = k$, x is a k-sparse vector, namely x has only k non-zero elements in its support, where $\|.\|_0$ is the ℓ_0-norm of a vector. Thus, the sparsity of the vector $x \in \mathbb{R}^N$ is k. When a vector has just a few large coefficients and many small coefficients, it can be well-approximated by a k-sparse vector. It is worth noting that an N-dimensional vector x containing the betweenness centrality of all vertices in a real-world network has the sparsity property, because the number of top-k betweenness centrality vertices is much smaller than the total number of all vertices in the networks ($k \ll N$). For example, [New10] showed that the betweenness centrality follows a power law on most networks. This power law distribution suggests that there exist a few vertices with very high betweenness centrality in comparison with the rest of the vertices in the network, which indeed satisfies the sparsity property. As another example, Narayanan [Nar05] investigated several generated genome-wide protein interaction networks for many organisms including Saccharomyces cerevisiae (baker's yeast), Caenorhabditis elegans (worm) and Drosophila melanogaster (fruit fly), and he observed that the distribution of the vertex betweenness centrality in these networks tends to be a power law. Moreover, Lammer et al. [LGH06] studied the German road network and obtained very broad distributions of betweenness centrality with a power law exponent in the range [1.279, 1.486]. In addition, [PTT+16] analysed a seismic data set measured in the central zone of Chile before and after the large earthquake of Illapel 2015 considering it as a spatial complex network, and they found a

power law betweenness centrality distribution in it. Furthermore, Lee [Lee06] uncovered that the betweenness follows a power law distribution irrespective of the type of networks, and he examined this characteristic in terms of the conditional probability distribution of the betweenness, given the degree. The conditional distribution also exhibits a power law behavior independent of the degree which explains partially, if not whole, the origin of the power law distribution of the betweenness. He then validated this observation on three real networks: the collaboration network, the protein interaction network of D. melanogaster, and the neural network of C. elegans. Besides, the authors in [KHP+07] showed that vertices of both fractal and non-fractal scale-free networks have power law betweenness centrality distribution $P(B) \sim B^{-\delta}$, such that for non-fractal scale-free networks $\delta = 2$ and for fractal scale-free networks $\delta = 2 - 1/d$, where d is the dimension of the fractal network. They also supported these results by explicit calculations on four real networks: pharmaceutical firms, yeast, world wide web (WWW), and a sample of the Internet network at autonomous system (AS) level. As a result, the number of top-k betweenness centrality vertices is much smaller than the set of all vertices in a wide range of real-world networks. Therefore, identifying high betweenness centrality vertices in the networks can be framed as recovering sparse high-dimensional data from a much smaller number of measurements, widely known as the *sparse signal recovery* problem.

The fundamental constraint to make our problem similar to the sparse recovery problem is the sparsity property of the desired solution set. The number of top-k betweenness centrality nodes is much smaller than the set of all network nodes. The breakthrough in solving the sparse signal recovery problem is compressive sensing (CS) described in Section 5.2. One of the most restricting challenges in CS is the construction of measurement matrix that needs to be feasible based on two fundamental constraints: (1) A measurement matrix in networks must contain only non-negative integer elements, which is more restrictive in comparison with random Gaussian measurement matrices usually used in the CS literature. (2) Measurements in a network are limited by network topological constraint; in other words, only nodes that induce a connected sub-graph can be aggregated together in the same measurement. This is contrary to the assumption of most existing CS results that any subset of vector entries can be aggregated together in the same measurement.

5.2 Compressive sensing

The breakthrough in solving the sparse signal recovery problem is *compressive sensing* [DDEK12, Don06] which allows for efficiently acquiring and reconstructing a signal, by finding solutions to under-determined linear systems. To be more precise, an N-dimensional sparse signal x can be simultaneously sampled and compressed to a linear sketch y of the original signal, through a measurement matrix $A \in \mathbb{R}^m \times \mathbb{R}^N$, where $m \ll N$. Although the dimension of $y \in \mathbb{R}^m$ is typically much smaller than that of $x \in \mathbb{N}$, the sketch y contains plenty of useful information about the data vector x. In this case, the sparsity property is used to uniquely identify and recover the underlying signal x given the measurement vector y and the measurement matrix A. One can formulate this problem by the following linear system with more unknowns than equations which is equivalently under-determined, as:

$$y = A\ x \tag{5.1}$$

In sparse signal recovery, especially compressive sensing, the set of sparse solutions to the above system are of interest. The solutions can be obtained by solving the following optimization problem [Don06]:

$$\min_x \|x\|_0 \quad \text{s.t} \quad y = Ax \tag{5.2}$$

It is proved that solving the above optimization problem is NP-hard. Therefore, the sparsity inducing ℓ_1-regularization is considered as a convex relaxation of Equation (5.2):

$$\min_x \|x\|_1 \quad \text{s.t} \quad y = Ax \tag{5.3}$$

The above objective function is known as *Basis Pursuit*. Note that the strict condition $y = Ax$ within the Basis Pursuit formulation is very sensitive to noise, imperfect sparsity or truncated values in the measurement matrix A, and the sketch y. The following formulation addresses this by removing the exact constraint and penalizing its violation:

$$\min_x \lambda\|x\|_1 + \|Ax - y\|_2^2 \tag{5.4}$$

This objective function is also known as LASSO [Tib94] and is used in this part of thesis for the optimization step. λ is a tuning parameter that controls the amount of regularization. It is noteworthy that sparse recovery over networks using compressive sensing has a closely related field called graph constrained group testing [CKMS12].

Group testing and compressive sensing over networks have the same requirements for measurement matrix and the differences are only in: (1) x is a logical vector in group testing, instead of real vector for the compressive sensing problem; (2) the operations used in each group testing measurement are the logical "AND"and "OR", in contrary to the additive linear mixing of the vector x over real numbers in compressive sensing. The authors in [WXMT12] stated that compressive sensing can perform better than group testing based on the required number of measurements, thus, we use compressive sensing for sparse recovery in networks.

5.3 Compressive sensing over graphs

Based on the compressive sensing framework, we would like to efficiently recover k highest centrality vertices from m indirect end-to-end measurements, in a way that $m \ll N$. In the linear system $y = A\ x$, let A be an $m \times N$ measurement matrix, where its i-th row corresponds to the i-th feasible measurement. For $i = 1, ..., m$ and $j = 1, ..., N$, $A_{ij} = 1$ if and only if vertex j is visited by the i-th measurement, otherwise $A_{ij} = 0$. Let x be an $N \times 1$ non-negative vector whose j-th entry is the value of a certain type of network characteristic (*e.g.* betweenness or closeness centrality) over vertex $j \in V$, and $y \in \mathbb{R}^m$ denotes the measurements vector whose i-th entry represents the additive aggregation values of network vertices in the i-th row of the measurement matrix A that induces a *connected sub-graph* over G.

To understand how the additive aggregation over connected induced sub-graphs is motivated for each measurement in practice, we mention an example from [WXMT12]. Consider a network where the vertices represent sensors and the edges represent communications between sensors. For the set T of active vertices within an arbitrary feasible measurement that induce a connected sub-graph, a vertex $u \in T$ monitors the total values corresponding to vertices in T. Every vertex in T obtains values from its children, if any, and aggregates them with its own value on the spanning tree rooted at u, then sends the sum to its parent. After that, the fusion center can obtain the sum of values corresponding to all the vertices in T by only communicating with u. The explained paradigm in data acquisition and aggregation is highly utilized within the wireless sensor network literature for applications such as air quality monitoring, volcanic activity detection and object localization [MCN16]. Some recent work has applied a similar acquisition and aggregation paradigm in network tomography [MRH13, MRHS13, GKMGR17], community detection [MRM+15a] and identification of key actors/connections in social

networks [Mah15, MRM+15b].

5.4 CS-HiBet: Compressive Sensing of High Betweenness Centralities

In this section, we introduce CS-HiBet, a compressive sensing based approach for efficiently identifying k-highest betweenness centrality vertices in a network. In CS-HiBet, we aim to construct a measurement matrix A with m independent measurements, each of which is a connected sub-graph over the network. Each measurement additively aggregates values of network vertices and it tends to incorporate high betweenness centrality vertices. In a distributed fashion, each vertex in the network calculates a local betweenness value that can be obtained from the one-hop adjacency matrix of that vertex. The intuition behind this local metric is that a vertex with high local betweenness acts as a bridge in transferring any valuable information between any pair of its non-adjacent neighbors along their geodesics of length 2 which passes through that vertex.

Calculating betweenness centrality with one-hop matrix. Since we want to recover the top-k betweenness centrality vertices, we try to visit these vertices more than the other vertices by our measurements. To achieve this goal, we select the best next vertex relative to $P_{selection}(v)$. This is for measuring vertex importance from an information flow standpoint. For the computation of this score, consider vertex v in the neighbor set $\mathcal{N}(S)$ as ego vertex, thus every pair of non-adjacent alters must have a geodesic distance of length 2 which passes through ego vertex v. We only need to consider these geodesics and geodesics of length 1 do not contribute to betweenness centrality. Let $H(v)$ be the one-hope adjacency matrix of ego vertex v with the dimensions of $|\{v\} \cup \mathcal{N}(v)| \times |\{v\} \cup \mathcal{N}(v)|$, then $H^2_{i,j}(v)$ contains the number of walks of length 2 connecting i and j when $i \neq j$. Therefore, we only need to count the number of walks of length 2 for non-adjacent alters since these will be the geodesics contributing to the local betweenness [EB05]. It follows that $\left[H^2(v)\,(1 - H(v))\,\right]_{i,j}$ gives the number of geodesics of length 2 joining i to j, where 1 is a matrix of all 1's. The sum of reciprocals of the entries gives the local betweenness of the ego vertex v.

Our method includes 6 steps:

(i) For each vertex $v \in V$ in the network, its one-hop ego adjacency matrix with the dimensions of $|\{v\} \cup \mathcal{N}(v)| \times |\{v\} \cup \mathcal{N}(v)|$ is constructed. Next, the sum of the

reciprocals of the entries in $H^2(v)\,(1-H(v))$ is calculated as the local score for vertex v. Finally, the probability of selecting a specific vertex v is also computed as $P_{selection}(v)$, in lines (5)-(9). This pre-processing step can be performed in a parallel or distributed manner by each vertex independently from the others.

(*ii*) The first vertex is selected randomly from the set of all vertices $v_{first} \in V$ in line (13). The first vertex v_{first} is added to the visited set S and all of its neighbors are added to the neighbor set $\mathcal{N}(S)$ in lines (14)-(15).

(*iii*) The next vertex is selected relative to $P_{selection}(v)$ for each vertex $v \in \mathcal{N}(S)$ in line (17). The selected next vertex is added to the visited set S and it is removed from the neighbor set $\mathcal{N}(S)$, then its neighbors are added to the neighbor set $\mathcal{N}(S)$, in lines (18)-(20).

(*iv*) The step (*iii*) is fulfilled 'l' times which is the length of a measurement, to generate a new row for the matrix A and the vector y in lines (16)-(21).

(*v*) The steps (*ii*)-(*iv*) are repeated 'm' times to construct a feasible measurement matrix A with 'm' measurements (in parallel) and the corresponding measurement vector y, in lines (10)-(24).

(*vi*) In order to find the sparse approximation $\hat{x}$ of x, we optimize the LASSO objective function subject to the linear sketch of $y = Ax$, in line (25), based on Equation (5.4).

The pseudo code of the proposed method is depicted in Algorithm 6.

An example. Consider the network shown in Figure 5.1a, the one-hop adjacency matrix for the ego vertex v_1 is:

$$H(v_1) = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_7 \\ v_8 \end{array} \begin{array}{c} \begin{array}{ccccc} v_1 & v_2 & v_3 & v_7 & v_8 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & \boxed{0} & 1 \\ 1 & 1 & 0 & \boxed{0} & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{pmatrix} \end{array} \tag{5.5}$$

Since the matrix is symmetric we only need to consider the zero entries above the leading diagonal and calculates $\left[H^2(v)\,(1-H(v))\right]_{i,j}$ for those entries. These entries are shown

Algorithm 6 The Proposed Method: CS-HiBet

Input: V, m, l

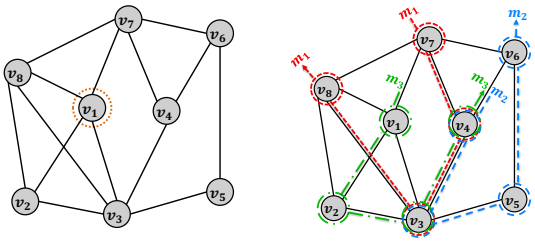
V: set of network vertices
m: number of measurements
l: measurement length
A = NULL ▷ Initializing measurement matrix
Foreach $v \in V$ **do** ▷ In a parallel or distributed manner
    $H(v)$ = The one-hop adjacency matrix of ego vertex v with the dimensions of $|\{v\} \cup \mathcal{N}(v)|^2$
    $Score(v)$ = The sum of the reciprocals of the entries in $H^2(v)\,(1 - H(v))$
    $P_{selection}(v) = \frac{Score(v)}{\sum_v Score(v)}$
end for
for $i = 1 \rightarrow m$ **do** ▷ In a parallel manner
    S = NULL ▷ Visited set
    $\mathcal{N}(S)$ = NULL ▷ Neighbor set of visited vertices
    v_{first} = Select start vertex randomly from the vertex set V
    $S = \{v_{first}\}$ ▷ Add v_{first} to the visited set S
    $\mathcal{N}(S) = \{u : u \in \mathcal{N}(v_{first})\}$ ▷ Add all neighbors of v_{first} to the $\mathcal{N}(S)$
    for $j = 1 \rightarrow l$ **do**
        v_{next} = Select next vertex relative to $P_{selection}(v)$ for $v \in \mathcal{N}(S)$
        Add v_{next} to the visited set S
        Remove v_{next} from the neighbor set $\mathcal{N}(S)$
        Add all neighbors of v_{next} to the neighbor set $\mathcal{N}(S)$
    end for
    $A[i,:]$ = Add visited vertices in S to the measurement matrix A as row
    $y[i,:]$ = Add the accumulative sum of vertex values in S to the vector y
end for
$\hat{x} = \min_x \|x\|_1 + \|Ax - y\|_2^2$ ▷ See Equation (5.4)

Output: sparse approximation $\hat{x}$

by the red box and the calculation gives:

$$H^2(v_1)\,(1 - H(v_1)) = \begin{pmatrix} * & * & * & * & * \\ * & * & * & 2 & * \\ * & * & * & 2 & * \\ * & * & * & * & * \\ * & * & * & * & * \end{pmatrix} \tag{5.6}$$

The local betweenness of the ego vertex v_1 is then simply the sum of reciprocals of these entries, that is 1, as the $Score(v_1)$. The ego-centric betweenness metric [EB05] is computationally more tractable than the traditional global betweenness centrality. It can be calculated locally in a parallel or distributed manner, by letting each vertex communicate only with its immediate neighbors. Then, the probability $P_{selection}(v)$ of selecting a specific vertex $v \in V$ is calculated by the normalized scores, as in our



(a) (b)

Figure 5.1: A network G with 8 vertices and 14 edges. (a) The ego vertex v_1 has 4 vertices in its neighbor set $\mathcal{N}(v_1) = \{v_2, v_3, v_7, v_8\}$. (b) Three measurements m_1, m_2, and m_3 over graph G constructed from the CS-HiBet method.

Table 5.1: Centrality measures for the sample network in Figure 5.1a. We used the iGraph package in Python to calculate the global betweenness centrality based on Equation (2.43). The ego-centric betweenness is calculated based on [EB05], which is described in this section.

Centrality Measure	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
Global betweenness	1.08	0	**6**	**3.5**	1.08	1.08	0.75	1.5
Ego-centric betweenness	1	0	**7**	**4**	1	2	1	2

pre-processing step. The global and local betweenness centrality for all vertices in the sample network in Figure 5.1a are given in Table 5.1. This table shows that vertices v_3 and v_4 are central in this network because they have the largest centrality measure. This observation is valid because the number of top-k betweenness centrality vertices is usually much smaller than the total number of all vertices in the networks. Hence, one may safely conclude that these vertices are the most important in the network.

We want to recover such central vertices without full knowledge of the network topological structure via indirect aggregated measurements. To this end, CS-HiBet constructs a feasible measurement matrix A with non-negative integer entries by using m measurements with the step size of l. Every measurement in A goes through a connected subgraph which guarantees the feasibility of A considering the network topological constraints. As an example, in the sample network G in Figure 5.1a, we set the number of measurements to 3 ($m = 3$) and the measurement length to 4 ($l = 4$) in the Algorithm 6, then the constructed measurements m_1, m_2, and m_3 are depicted in Figure 5.1b and the

corresponding measurement matrix A is as follows:

$$A = \begin{array}{c} \\ m_1 \\ m_2 \\ m_3 \end{array} \begin{array}{c} \begin{array}{cccccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 & v_8 \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array} \tag{5.7}$$

For the visited vertices in each measurement, the accumulative sum of their scores (ego-centric betweenness) is added to the corresponding element of the measurement vector y. According to the above measurement matrix A for the sample network Figure 5.1b, the constructed measurement vector is: $y = [14\ 14\ 12]$. To be more precise, the first measurement m_1 is started from v_7 and the walker passes through vertices v_4 and v_3, and it ends with v_8, so the visited vertex set for m_1 is $S = \{v_3,\ v_4,\ v_7,\ v_8\}$ and its corresponding entry in y is the sum of their scores (ego-centric values in Table 5.1) which is $y_1 = 7 + 4 + 1 + 2 = 14$. This is similar for m_2 and m_3. After generation of A and y, we form the linear system of $y = A\ x$. Finally, we find the sparse approximation $\hat{x}$ for this system by optimizing the LASSO objective function, based on Equation (5.4). For the constructed A and y in the network Figure 5.1b, the sparse approximation $\hat{x} = [0.38\ 0.38\ \mathbf{6.61}\ \mathbf{6.61}\ 0.13\ 0.13\ 0.13\ 0.13]$ is the output of CS-HiBet. Thus, one can easily conclude that CS-HiBet detected v_3 and v_4 as the top-2 central vertices in the sample network, and it is completely true based on the actual global betweenness centralities in Table 5.1. We will explore the performance of CS-HiBet via the extensive experimental evaluations in Section 5.7.

5.5 Complexity analysis

Consider the network $G = (V, E)$ with the assumption that any arbitrary vertex $v \in V$ has only local view to its immediate neighbors. The CS-HiBet method can be executed and analyzed in its three main steps according to Algorithm 6.

First, the one-hop adjacency matrix of ego vertex v, namely $H(v)$, and also $1 - H(v)$ can be computed in $O(deg(v)^2)$, where $deg(v)$ is the degree of vertex v. Using the famous Strassen's Algorithm, $H^2(v)$ can be obtained in $O(deg(v)^{\log_2^7 + o(1)}) \approx O(deg(v)^{2.8074})$. Finally, at the end of the first step, $Score(v)$ can be locally computed for any arbitrary vertex $v \in V$ in $O(deg(v)^{\log_2^7 + o(1)})$ time. Hence, the scores can be locally computed at each vertex in at most $O(\Delta^{\log_2^7 + o(1)})$ time, where Δ is the largest degree of a vertex in the network G. Afterwards, all transition probabilities at each vertex can also be locally

obtained in at most $O(\Delta)$ time.

Second, we can generate m random numbers between 1 and n in $O(m)$ time, to choose the seeds for starting a measurement. m is the number of measurements that correspond to the rows of the measurement matrix A.

Third, we can begin to construct m measurements locally in a distributed manner. For each measurement pre-calculated transition probabilities have been already obtained in the previous steps. They are locally and independently accessible in $O(1)$ time. In the proposed algorithm CS-HiBet, the next vertex can be determined by using the weighted selection algorithm relative to the probabilities $P_{selection}(u)$ for every $u \in \mathcal{N}(S)$. The selected vertex u is removed from the $\mathcal{N}(S)$ and added to the set S, then its neighbors are added to the $\mathcal{N}(S)$. The accumulative search for construction of a measurement costs at most $O(N\log(N))$ time, by utilizing the binary search. It is noteworthy that at step i from the l total steps corresponding to a measurement, the neighbor set $\mathcal{N}(S)$ contains at most $(i\Delta - i)$ vertices. In this case, each addition and deletion is taking only $O(1)$ time per operation in an array and overall $\min(l\Delta, N)$. The time complexity for the weighted selection search will be at most $O(N\log(N))$ which should be considered in two different cases. One for the case that $l \leq \left\lfloor \frac{N}{\Delta-1} \right\rfloor$ and the other is for $l > \left\lfloor \frac{N}{\Delta-1} \right\rfloor$. The time complexity in each case is computed as follows:

▷ Case when $l \geq \left\lfloor \frac{n}{\Delta-1} \right\rfloor$ then:

$$
\begin{aligned}
Search\_Cost &= \sum_{i=1}^{\lfloor \frac{N}{\Delta-1} \rfloor} \log\left(i\Delta - i\right) + \left(N - \left\lfloor \tfrac{N}{\Delta-1} \right\rfloor\right)\log(N) \\
&= \log\left(\left\lfloor \tfrac{N}{\Delta-1} \right\rfloor!\right) + \left\lfloor \tfrac{N}{\Delta-1} \right\rfloor \log\left(\Delta - 1\right) + \left(N - \left\lfloor \tfrac{N}{\Delta-1} \right\rfloor\right)\log(N) \\
&\leq \left\lfloor \tfrac{N}{\Delta-1} \right\rfloor \log\left(\left\lfloor \tfrac{N}{\Delta-1} \right\rfloor\right) + \left\lfloor \tfrac{N}{\Delta-1} \right\rfloor \log\left(\Delta - 1\right) + \left(N - \left\lfloor \tfrac{N}{\Delta-1} \right\rfloor\right)\log(N) \\
&\leq \left\lfloor \tfrac{N}{\Delta-1} \right\rfloor \left(\log\left(\tfrac{N}{\Delta-1}\right) + \log\left(\Delta - 1\right)\right) + \left(N - \left\lfloor \tfrac{N}{\Delta-1} \right\rfloor\right)\log(N) \\
&= N\log(N)
\end{aligned}
$$

▷ Case when $l < \left\lfloor \frac{N}{\Delta-1} \right\rfloor$ then:

$$
\begin{aligned}
Search\_Cost &= \sum_{i=1}^{l} \log\left(i\Delta - i\right) = \log\left(l!\right) + l\log\left(\Delta - 1\right) \\
&\leq l\log(l) + l\log(\Delta) \leq 2N\log(N)
\end{aligned}
$$

Therefore, the aggregated total time complexity of the CS-HiBet method will be $O(\Delta^{\log_2^7 + o(1)} + N\log(N) + m + \min(l\Delta, N))$. As previously mentioned that $m \ll N$ and $\min(l\Delta, N) \leq N$, hence by exploiting the ability of the proposed algorithm to perform locally with only local view at each vertex, the total time complexity will be

$O(\Delta^{\log_2^7 + o(1)} + N \log(N))$.
The required space storage for any arbitrary vertex $v \in V$ is $O(deg(v)^2)$ for the ego adjacency matrix, scores, and the transition probabilities. Thus, the local storage at each vertex for space complexity of our method will be at most $O(\Delta^2)$. Moreover, the space complexity $O(m)$ is needed to store the randomly chosen initiative seeds and the final measurement vector. Also at most the space $\min(l\Delta, N)$ is needed for the weighted selection algorithm in each measurement.

It is worth noting that in most real-world networks, in particular social networks, vertices are connected to a very small portion of the whole network's vertices, which means Δ is very small. For example, the maximum number of connections allowed on Twitter [Twi17b] and Facebook [Fac13] is about 5,000 which is much smaller than their network size. Consequently, CS-HiBet can be practically scalable for efficient detection of k-highest betweenness centrality vertices in large real networks, in terms of time and space complexity.

5.6 Extension to closeness centrality

In this section, we introduce a new h-hop ego-centric (local) closeness centrality of vertex v as:

$$egoC_h(v) = \sum_{\tau=1}^{h} |B_\tau(v)|/\tau \tag{5.8}$$

where $B_\tau(v)$ indicates the set of vertices that have an exact shortest distance of length τ from vertex v. The intuition behind this metric is that, the farther vertices from v have lower effect in dissemination of goods (*e.g.* information) emerged from it. The computation of the sets $B_\tau(v)$ for $\tau \leq h, v \in V$ can be done by executing a breadth-first search (BFS) process at each vertex in parallel, with exploration radius set to h. This will require computational cost of at most $O(\Delta^h)$ where Δ is the maximum degree of the network. The required memory storage at each vertex is also $O(\Delta^h)$. The computed sets can be utilized to evaluate ego closeness centrality at each vertex in a distributed and decentralized manner, with $O(1)$ computational and storage cost per vertex. One can easily extend the CS-Hibet method for detection of top-k closeness centrality vertices, by replacing step (i) of Algorithm 6 with the proposed local closeness metric as scores of network vertices.

5.7 Experiments

In this section, we experimentally evaluate the performance of the CS-HiBet, in both real and synthetic networks, under various configurations.

Datasets. We considered both synthetic and real-world networks for the evaluations. The data from well-known real-world networks were: (1) Facebook-like social network [OP09] with 1,899 vertices and 20,296 edges; (2) Twitter's mentions and retweets of the twitter network [Twi17a] with 3,656 vertices and 188,712 edges; (3) Wikipedia vote (WikiVote) network [LHK10] with 7,115 vertices and 103,689 edges; (4) Youtube video-sharing network [YL15] with 1,134,890 vertices and 2,987,624 edges; (5) Road network of Pennsylvania (RoadNet) [LLDM09] with 1,088,092 vertices and 1,541,898 edges; (6) Pokec social network [TZ12] with 1,632,803 vertices and 30,622,564 edges. In case of disconnected networks, we always extracted the largest (strongly) connected component first. In addition, we used three kinds of synthetic networks: (7) Scale-free network based on the Barabási-Albert (BA) model [BA99] with 500 vertices and 2,979 edges, where edges created by each new vertex were 6; (8) Random network based on the Erdös-Rényi (ER) model [ER60] with 500 vertices, 4,000 edges, and average degree of 16; (9) Small-world network based on the Watts-Strogatz (SW) model [WS98a] with 500 vertices and 4,466 edges, where the rewiring probability was 0.2 and the number of initial closest neighbor was 9.

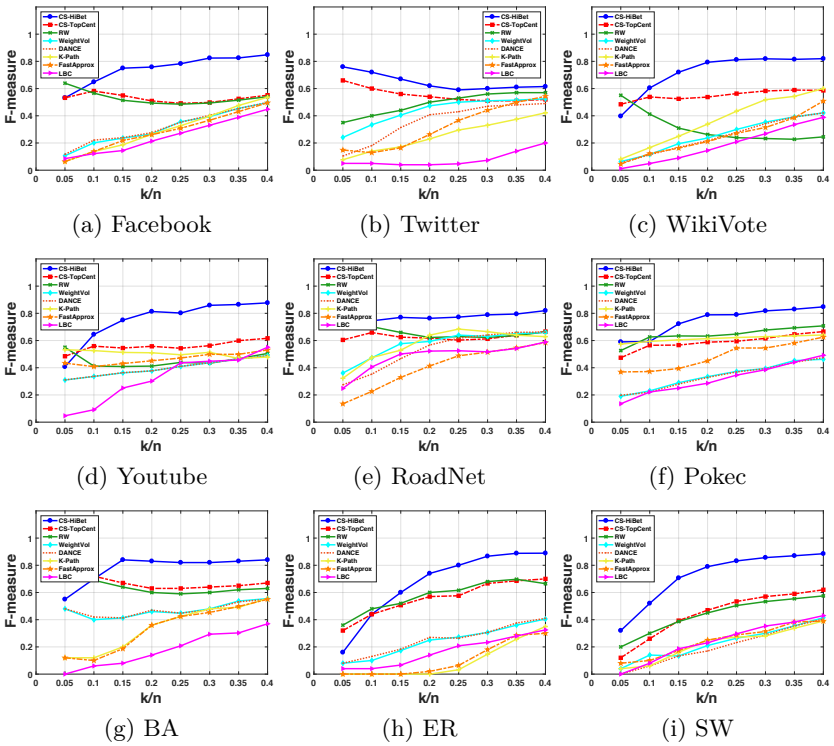
Baselines. We refer to our algorithm as CS-HiBet. The baseline methods that we compared our performance to were: (1) *DANCE:* In this framework [WGZ11], an estimate for a certain centrality metric is approximated based on the *h-neighborhood* of each vertex using a local function. (2) *WeightVol:* This method [KY12] is a model for selecting a set of k vertices as the initial influenced vertices so that they can effectively disseminate the information to the rest of the network. (3) *LBC:* This method [NK08] proposed a new centrality metric, called localized bridging centrality, which combines the egocentric betweenness centrality with the locally computable bridging coefficient. (4) *FastApprox:* The authors of [RK16] proposed an efficient randomized algorithm for betweenness centrality estimation, employing the random sampling of shortest paths, which offers probabilistic guarantees on the quality of the approximation. (5) *K-Path:* [KAS+13] introduced a new centrality measure, called k-path, and a randomized algorithm for its estimation. They showed that the vertices with high k-path centrality value have high vertex betweenness. (6) *CS-TopCent:* To address the disadvantages of the sampling-

based approaches, [Mah15] proposed a compressive sensing approach for detection of central vertices in networks without full knowledge of the network topology via indirect end-to-end measurements. (7) *RW:* Motivated by network tomography problem, this method [XMT11] introduced a compressive sensing (CS)-based framework to recover a sparse unknown vector that represents certain features of the elements over the network via collective additive measurements.
Although the choice of h is application-specific, the literature (*e.g.* [WGZ11, NK08]) showed that small values, typically $h = 1$ or $h = 2$, yield good results in distributively assessing network centralities for different kinds of complex networks. So, to have a fair comparison with our method in the experiments, we set $h = 1$ for the methods.

Experimental setting. To evaluate the accuracy of the proposed approach, we measured the *precision* and *recall* of the methods, and to consider both measures, we used the F-measure metric which represents the harmonic mean of both precision and recall, as $F\text{-}measure = 2 \times \frac{Precision \times Recall}{Precision + Recall}$.
The CS-HiBet, CS-TopCent, and RW methods possess a source of randomness, hence, in each of their respective experiments we did 10 repetitions. The denoted points in the figures represent the mean value of these repetitions. However, the other implemented competing methods are deterministic and there is no need for repetitions.

5.7.1 Accuracy of CS-HiBet on identifying top-k central vertices

Figure 5.2 shows the accuracy evaluation of our approach in comparison with the competing methods, in terms of F-measure for varying sparsity percentage to identify k-highest betweenness centrality vertices. Each point in the horizontal axis is proportional to the number of top-k vertices divided by the number of all vertices in the network (*i.e.* $\frac{k}{N}$). In this experiment, for the CS-based methods (*i.e.* CS-HiBet, CS-TopCent, RW), we performed a set of $m = 0.2N$ measurements of length $l = 0.4N$, in each network. We almost observed an increasing trend for F-measure as we increase the bracket size of top-k vertices. As clearly depicted in all test cases, CS-HiBet performs better than the competing methods in terms of having higher F-measure, even on the lower sparsity (higher value of $\frac{k}{N}$). For higher values of F-measure, one can observe more correlation between the vertices lists identified by the methods and by the global betweenness centrality. Thus, the results show that the CS-HiBet and the betweenness centrality correlate well with regard to the number of correctly identified top-k central vertices.
One of the main reasons for the superiority of the CS-based methods in the accurate



(a) Facebook (b) Twitter (c) WikiVote

(d) Youtube (e) RoadNet (f) Pokec

(g) BA (h) ER (i) SW

Figure 5.2: Comparison of accuracy between CS-HiBet and the baselines for the number of correctly identified top-k betweenness centrality vertices in networks for varying sparsity percentage.

detection of top-k betweenness centrality vertices over the h-neighborhood-based competing methods is the fact that: the vertices with a similar h-neighborhood structure will be assigned the same score in each of the latter approaches, however the former approaches may visit those vertices with different rates based on their global positions in the network. As an example, almost all vertices in a line graph have exactly the same 1-hop neighborhood structure resulting in the same assigned scores by each of the latter methods. But, the vertices in the middle of the line graph, which have a higher global betweenness centrality, will have a higher rate of being visited by the measurements (walks) performed in the CS-based approaches. Therefore, these vertices have a higher chance of being recovered as the top-k central vertices.

5.7.2 Accuracy of CS-HiBet on rank prediction

The evaluation described till now focuses on the number of correctly identified vertices in top-k set assembled according to CS-HiBet and the baselines. In a more fine-grained

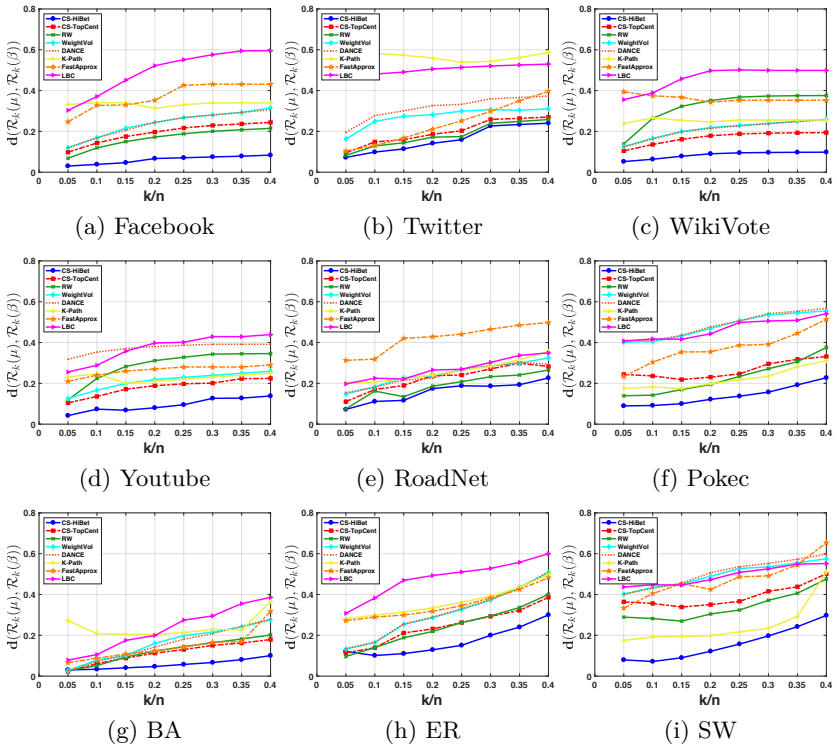


Figure 5.3: Comparison of accuracy for the distance between top k ranks assigned by CS-HiBet and the competing methods ($\mathcal{R}_k(\mu)$), and those determined by the global betweenness ($\mathcal{R}_k(\beta)$).

analysis, we are also interested in quantifying the accuracy of ranks assigned by these methods. To this end, we compute the difference between ranks assigned by CS-HiBet and the competing methods, and those determined by the global betweenness centrality. Furthermore, the significance of correct ranking of high-ranked vertices is more important than low-ranked vertices. To address this goal, we consider a distance metric to compare the relevance of two ordered lists. We denote the list of vertices with length k in descending order of importance in the aforementioned methods by $\mathcal{R}_k(\mu)$, and the list of vertices with length k in descending order of the global betweenness centrality by $\mathcal{R}_k(\beta)$. The distance is normalized in the range $[0, 1]$, where 0 corresponds to the perfect match between two given ordered lists, and vice-versa. The normalized weighted distance metric $\mathbf{d} \in [0, 1]$ between these two ordered lists is defined as [ISLR13]:

$$\mathbf{d}(\mathcal{R}_k(\mu), \mathcal{R}_k(\beta)) = \frac{\sum_{i \in \mathcal{R}_k(\beta)} \left[\frac{w_i \left| \mathcal{R}_k(\mu_i) - \mathcal{R}_k(\beta_i) \right|}{N - 2i + 1} \right]}{\sum_{i \in \mathcal{R}_k(\beta)} w_i} \tag{5.9}$$

where w_i is the betweenness of user i and N is the number of vertices in the network.

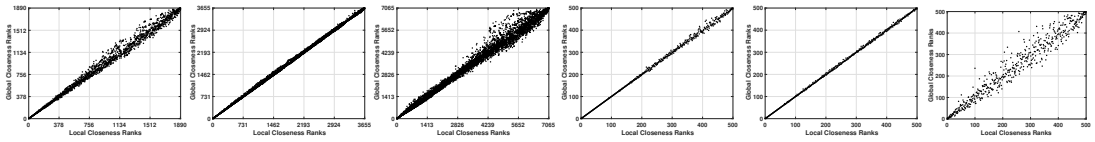
Figure 5.3 depicts the accuracy evaluation of the proposed framework CS-HiBet in comparison with the other methods, in terms of the distance between two ordered lists of ranks assigned by them and the global betweenness centrality. Each point in the horizontal axis is proportional to the sparsity level $\frac{k}{N}$. In this experiment, for the CS-based methods, we performed a set of $m = 0.2N$ measurements of length $l = 0.4N$, in each network. It is clear that in all test cases, CS-HiBet performs better than the competing methods in terms of having lower rank distance with the global betweenness centrality for all sparsity percentages. In addition, we can easily observe an increasing trend for $\mathbf{d}(\mathcal{R}_k(\mu), \mathcal{R}_k(\beta))$ when we increase k. For all datasets at $k = 20\%$ of N, the distance $\mathbf{d}$ for CS-HiBet is lower than 0.17 ($d < 0.17$) and also $d < 0.3$ at $k = 40\%$ of N. Hence, the results show that the CS-HiBet correlates well with the regular betweenness centrality, compared to the baselines, in terms of estimated ranks for top-k central vertices.

5.7.3 Correlation between our ego-closeness and global closeness

We experimentally analyzed the correlation between the proposed ego-centric (local) centrality metric and the global closeness centrality over several synthetic and real-world networks. To compare these two centrality metrics, we used Pearson product moment correlation coefficient (ρ), which in fact measures the strength of a linear association between two variables. The Pearson coefficient ρ can take a value in range $[-1, +1]$. A value of 0 shows that there is not any association, a value greater than 0 indicates a positive association, and a value less than 0 indicates a negative association. For comparison, we considered several well-known h-neighborhood local metrics that tend to be correlated well with the closeness centrality: (1) Dist-Exact [YTQ16], (2) DACCER [WZ12], and (3) Weight-Vol [KY12]. It is worth noting that for $h = 1$, any local metric would be the same as the degree centrality. According to problem addressed in this part, we want to identify top-k central vertices for $k \ll N$, so the results show that in this case choosing $h = 2$ is sufficient, in terms of having a good trade-off between computational complexity and accuracy. Table 5.2 shows the Pearson correlation coefficients between the competing local metrics and our proposed ego-centric centrality with $h = 2$ versus the global closeness centrality on synthetic and real-world networks. In this experiment, we analyze the correlation coefficients of all local metrics for high sparsity levels $k = \{0.1N, 0.2N, 0.3N, 0.4N\}$. The results show that Dist-Exact for $h = 2$ has linear correlation, but negative association with the closeness centrality in networks. One can easily observe that our proposed metric has almost always the best correlation

Table 5.2: Pearson correlation coefficients between the existing local metrics with $h = 2$ and the global closeness centrality on various networks, where $k \ll N$.

k/N	Local Metric	Facebook	Twitter	wikiVote	BA	ER	SW
0.1	Dist-Exact	-0.93	-0.41	-0.94	-0.94	-0.99	-0.93
	DACCER	0.91	0.47	0.83	0.97	0.96	0.95
	Weight-Vol	0.97	0.80	0.97	0.99	0.97	0.93
	Our Metric	**1.00**	**1.00**	**0.99**	**1.00**	**1.00**	**0.96**
0.2	Dist-Exact	-0.93	-0.61	-0.94	-0.93	-0.99	-0.91
	DACCER	0.92	0.58	0.89	0.97	0.98	0.94
	Weight-Vol	0.97	0.77	0.98	0.99	0.99	0.94
	Our Metric	**1.00**	**1.00**	**0.99**	**1.00**	**1.00**	**0.96**
0.3	Dist-Exact	-0.93	-0.73	-0.94	-0.92	-0.99	-0.93
	DACCER	0.93	0.63	0.92	0.97	0.99	0.95
	Weight-Vol	0.97	0.75	0.98	0.99	0.99	**0.96**
	Our Metric	**1.00**	**1.00**	**0.99**	**1.00**	**1.00**	**0.96**
0.4	Dist-Exact	-0.92	-0.79	-0.94	-0.91	-0.99	-0.94
	DACCER	0.95	0.67	0.95	0.97	0.99	0.95
	Weight-Vol	0.98	0.75	0.98	0.99	0.99	**0.96**
	Our Metric	**1.00**	**1.00**	**0.99**	**1.00**	**1.00**	**0.96**



(a) Facebook (b) Twitter (c) WikiVote (d) BA (e) ER (f) SW

Figure 5.4: Correlations between the vertices' ranks provided by the proposed local metric and the global closeness centrality. These two metrics correlate very well.

coefficient compared to the other metrics.

To have more analysis of the correlation between the proposed ego-centric (local) metric and the global closeness centrality, Figure 5.4 shows the scatter plots of all vertices' ranks provided by one versus the other, on various networks. Each point in the figure corresponds to a vertex's rank using these two metrics. As discussed before, we calculated our local measure for $h = 2$ to have low computational complexity, yet high accuracy. One can observe the linear correlation and positive association (as the rank with respect to the local metric increases, so does the rank with respect to the global metric), especially for the top-k vertices' ranks which is the target of thesis.

5.8 Conclusion

Betweenness centrality has been widely used as a fundamental metric for quantitatively measuring the relative importance of vertices in a network. It is highly correlated to the impact of a specific vertex on the spread of influence in social networks, the user activity in mobile phone networks, and the contagion process in biological networks. Thus, identification of k-highest betweenness centrality vertices is of great interest. Although many exact and approximation schemes have been proposed for this problem, the vast majority of these algorithms fail to scale on real-world networks because of their high time and space complexity. On the other hand, some of them tend to assume full knowledge of the network topological structure which is not often the case in real networks. Another fact is that direct measurement of each individual vertex in networks may impose remarkable overhead. To overcome these shortcomings, we proposed CS-HiBet, a novel approach for efficiently identifying top-k central vertices in networks, using compressive sensing with end-to-end measurements. We assumed that each vertex has only localized information about its neighbors allowing our approach to perform as a distributed algorithm. Extensive experimental evaluations on synthetic and real datasets demonstrated that the CS-HiBet and global betweenness centrality correlate very well with regard to the number of correctly identified central vertices and their estimated rank in networks.

CHAPTER 6

Conclusion and Future Research Directions

To study networks in biology, engineering, and social sciences, we need to understand how such systems work, know their structure and understand the model of the underlying structure of the inter-connected entities of such systems. This thesis concentrates on the fundamental task of modeling and generating graphs.

A key challenge in this area is developing methods that can capture the dynamicity of the underlying graph. Most real-world graphs are dynamic, with the underlying distribution dynamically evolving. For example, the edges of a network can appear and disappear over time, and the communities of the interactions may wax and wane in popularity, and the set of entities interested in a given topic may evolve over time. In addition, oftentimes we do not have access to the full topology of the graphs and may not observe the all of the sequence of the interactions. In such systems, we can only observe stochastic processes propagating through the network.

In part 3 we described how we extracted the observations from these partially information cascades and then how to generate a graph that can predict future interactions. Further, part 4 takes advantage of the time-dependent interactions in a network to model a multigraph where the sparsity property of the network is delivered by considering the PYP on top of a hierarchical mixture model. Finally, the intuition behind the sparsity property of the networks, inspired us to propose an algorithm for the sparse recovery problem in order to find top-k central nodes in the network in part 5. In this chapter,

we will summarize the contributions related to this thesis, and then thoroughly explore potential future research directions in Section 6.2.

6.1 Summary

In this thesis, we considered the problem of developing generative dynamic network models from partial observations (i.e. diffusion data) as well as temporal observations (i.e., time-dependent sequence of interactions). We proposed a novel framework, DYFERENCE, for providing a non-parametric edge-exchangeable network model based on a generative network model to capture the underlying dynamic network structure from partial observations. Furthermore, our DYFERENCE algorithm provides online time-varying estimates of probabilities for all the potential edges in the underlying network, and track the evolution of the underlying community structure over time.

We have presented a new distribution for temporally varying, structured multigraphs, that allows us to represent both sparse and dense networks. Since our model explicitly describes a sequence of edges, it is well-suited to predict future edges. We showed the effectiveness of our approaches using extensive experiments on synthetic as well as real-world networks.

Finally, a compressive sensing based framework for identifying central nodes in social networks was proposed that uses only $O(k \log(\frac{n}{k}))$ indirect end-to-end measurements to highly detect top-k central nodes, with provable recovery guarantees, in social networks with n nodes. It is worth noting that in our analysis, we take the advantage of the fact that the number of top-k central nodes of interest is sparse.

6.2 Future research directions

A tremendous amount of unstructured data is generated every second, and demand fast analysis and efficient modelling for prediction tasks. Graphs are fundamental data structures which concisely capture relational unstructured data in many important real-world domains, such as knowledge graphs, physical and social interactions, language, and chemistry. Our long-term research goal is to have a scalable model for such amount of data, which is a trivial problem in nowadays data analysis. Building large-scale machine learning methods for predictive as well as prescriptive analytics have many applications,

including chemistry [GSR+17], medicine [KPF+17], or computer vision [SK17]. In order to achieve this long-term goal, we define the following intermediate steps and directions:

- Developing scalable deep generative models for graphs.
- Building more efficient inference algorithm for time-evolving graphs.

To make progress on these long-term goals, we pursue the following research directions.

Developing scalable deep generative models for graphs. The ability to generate graphs has many applications such as; a generative model of molecular graph structures employed for drug design [GBWD+18, LZL18, LVD+18, YLY+18], graph generative models for model architecture search [XKGH19], graph generative models in network science [WS98b, LCK+10, AB02], and also deep graph generative models [LLS+19]. Deep learning on graphs has very recently become a popular research topic in these applications. Previous works concentrate on learning graph embeddings, i.e., encoding a graph into a vector representation [LLS+19]. However, the recent development of fast-paced generative models for image and text inspires us to model more efficient graph generators [RAY+16]. The problem of building graph generative models using neural networks has attracted increasing attention. Compared to traditional random graph models, and statistical graph models, these deep generative models have a greater capacity to learn structural information from data and can model graphs with complicated topology and constrained structural properties. However, none of these models consider the sparsity property of networks. We aim to benefit from this feature of real-world graphs in order to reduce the effort of considering all possible node permutations to generate larger graphs.

More recent graph generative models eschew the simple parametric form of models such as the stochastic blockmodel, instead using neural networks to model the distribution over edge probabilities. For example, the GraphRNN model [YYR+18] uses a recurrent neural network to obtain the distribution over the ith row of the lower triangle of the adjacency matrix, conditioned on the previous rows. However, despite the added modeling flexibility provided by the neural network, such models still concentrate on dense graphs: we sample the value for each edge from some distribution which has non-zero expectation. While edge-exchangeable models allow us to capture graph sparsity (in addition to properties such as power law degree distribution), the resulting graphs have limited structure. By construction, the graphs are exchangeable, meaning that their distribution is invariant to re-ordering the nodes. Further, all structure in the model is governed by a single

distribution, meaning we do not see meaningful clustering coefficients or community interaction. While such issues have been partially addressed by using mixture models [Wil16] and replacing the nonparametric priors with temporally dependent distributions [GMG+19], the resulting models still lack the flexibility of modern neural-network-based graph models.

We aim to construct deep generative models for temporally growing multigraphs. Inspired by the Bayesian nonparametric edge-exchangeable graphs, these models allow us to capture graph sparsity and generate graphs that preserve that sparsity. Unlike their edge-exchangeable counterparts, our model will be able to capture complex graph dynamics and interactions.

Building more efficient inference algorithm for time-evolving graphs. While BNL brings powerful representations and highly flexible models to the learning table, model inference still faces great challenges. Despite of the pace of advancement in statistical inference techniques, such as sampling-based inference algorithms (e.g., slice sampling and Hamiltonian Monto Carlo), optimization-based inference algorithms (e.g., variational inference), and hybrid inference algorithms (e.g., stochastic gradient Markov Chain Monto Carlo), applying efficient inference algorithm for the problem under study still remains a challenging problem.

In this thesis, we used MCMC methodologies for the task of posterior inference. While MCMC algorithms are robust or universal, this robustness may however induce a slow convergence behaviour in that the exploration of the relevant space–meaning the part of the space supporting the distribution that has a significant probability mass under that distribution–may take a long while. In particular, with a finite number of samples, Gibbs sampling is very often myopic in that it provides a good illumination of a local area, while remaining unaware of the global support of the distribution. More generally, an MCMC algorithm may require a large number of iterations to escape the attraction of its starting point θ_0 and to reach stationary point, to the extent that some versions of such algorithms fail to converge in the time available (i.e., in practice if not in theory). It thus makes sense to seek ways of accelerating (a) the convergence of a given MCMC algorithm to its stationary distribution, (b) the convergence of a given MCMC estimate to its expectation, and/or (c) the exploration of a given MCMC algorithm of the support of the target distribution. In the other hand, due to the exponentially increase in the amount of data, MCMC algorithm

One commonly accepted solution is truncation method [Fox09, FSJW09], which sets the component number so large that the given data would only adopt a subset of them, but it does introduce an approximation error. Another successful technique to resolve this issue is data/variable augmentation [TW+10], also referred as slice sampling [DWW99, N+03]. The slice sampling algorithm for DP [KGW11] introduces an auxiliary variable $u_i \sim N(0, \phi_{k_i})$, which functions as an adaptive truncation for the data i, and then only needs to sample $\pi > u_i$ for data i.

We aim to extend our current inference algorithms into parallel versions for multiple processors/machines. Parallel MCMC [SWA09] uses an asynchronous method that makes it easy to incorporate new data and processors and it is extremely fault-tolerant. The disadvantage of this method is that it includes additional approximation. Further, parallel MCMC for DP and PYP was proposed to overcome this problem [LAM12, DWPX14, WDX13] based on the inverse-superposition of DPs where each processor or machine handles one supercluster. Although these methods can marginalize representations of DP to avoid truncation, a slice-sampling-based parallel MCMC [GCWG15] has been developed to explicitly sample the weights of DP in an elegant way. We will use these approaches as starting points.

In addition to scalable MCMC methods, we will also explore the use of other inference techniques. An alternative approach for inference in BNL is variational inference [BKM17, BJ+06]. This method uses a set of variational distributions to approximate the real posterior distribution and transforms this posterior distribution into a high-dimensional optimization problem that can be solved with help of gradients. Thus, choosing an optimization method that boosts the inference efficiency together with setting variational distributions which can significantly reduce the additional approximation error. The form of variational inference is based on a stick-breaking representation where the latent variables include stick weights, atom parameters and data assignment indices. *Ordinary Variational Inference* adopts a truncation method to ensure that the infinite number of atoms produced by DP reduced to only a finite number of atoms to be approximated [KWV07]. *Collapsed Variational Inference* of DP [KWT07] marginalized out the stick weights which makes it more accurate and efficient algorithm.

Some other inference methods that have been applied to BNL are: *Sequential Monto Carlo*, also known as particle filtering that approximated the posterior distribution through a large collection of particles (or samples) that are propagated over time and updated by sequential importance sampling. It has been applied to DP mixutre with

time-varying mixtures [CDD12], beta-binomial DP mixtures [MCL99], general conjugate DP mixtures [Fea04], and nonparametric Bayesian matrix factorization [WG07], *Power Expectation Propagation* [Min01] generalizes expectation propagation and variational inference using a flexible α-divergence.

List of Figures

3.1 Precision, Recall and F1 score of DYFERENCE. **(a)** Compared to INFOPATH for dynamic network inference over time on Core-Periphery (CP) Kronecker network with exponential transmission model, and **(b)** Hierarchical (HC) Kronecker network with Rayleigh transmission model. **(c)** accuracy of DYFERENCE compared to INFOPATH and NETRATE for static network inference for varying number of cascades over CP-Kronecker network with Rayleigh, and **(d)** Exponential transmission model, and **(e)** on Forest Fire network with Power-law transmission model. **(f)** compares the running time of DYFERENCE with INFOPATH for online dynamic network inference on the time-varying hyperlink network with four different topics Occupy with 1,875 sites and 655,183 memes, Linkedin with 1,035 sites and 155,755 memes, NBA with 1,875 sites and 655,183 memes, and News with 1,035 sites and 101,836 memes. **(g), (h), (i), (j)** compare the accuracy of DYFERENCE to INFOPATH for online dynamic network inference on the same dataset and four topics from March 2011 to July 2011. 55

4.1 Relationship between the number of edges and the number of vertices in DNND multigraphs generated according to Equation (4.4), for various values of σ. Plots are shown on a log-log scale. Different colors correspond to different random seeds. The blue dashed line has a slope of 2, indicating a quadratic relationship. We see the multigraphs become increasingly sparse as σ increases. 63

4.2 Predictive log likelihood vs. time slots. Each evaluation is the average value over 20 samples. 68

4.3 F1 score for future interaction prediction. Decay functions for DNND are WINDOW (W), EXPONENTIAL (E), and LOGISTIC (L) (averaged over time slots). 69

5.1 A network G with 8 vertices and 14 edges. (a) The ego vertex v_1 has 4 vertices in its neighbor set $\mathcal{N}(v_1) = \{v_2, v_3, v_7, v_8\}$. (b) Three measurements m_1, m_2, and m_3 over graph G constructed from the CS-HiBet method. 80

5.2 Comparison of accuracy between CS-HiBet and the baselines for the number of correctly identified top-k betweenness centrality vertices in networks for varying sparsity percentage. 86
5.3 Comparison of accuracy for the distance between top k ranks assigned by CS-HiBet and the competing methods ($\mathcal{R}_k(\mu)$), and those determined by the global betweenness ($\mathcal{R}_k(\beta)$). 87
5.4 Correlations between the vertices' ranks provided by the proposed local metric and the global closeness centrality. These two metrics correlate very well. 89

List of Tables

3.1 Performance of DYFERENCE for diffusion prediction compared to DEEPCAS, TOPOLSTM,and EMBEDDEDIC on Twitter and Memes datasets (TOPOLSTM requires the underlying network). 52
3.2 Top 10 predicted influential websites of Memes (Linkedin) on 30-06-2011. The correct predictions are indicated in bold. 52
3.3 Performance of DYFERENCE for dynamic bankruptcy prediction compared to INFOPATH on financial transaction network from 2010 to 2016. In 2010, a financial crisis hit the network. 53

4.1 Predictive log likelihood of held-out edges on four real-world datasets (mean ± standard deviation over time slots). 68
4.2 MAP@k for the future interaction prediction task (mean value over time slots). 69

5.1 Centrality measures for the sample network in Figure 5.1a. We used the iGraph package in Python to calculate the global betweenness centrality based on Equation (2.43). The ego-centric betweenness is calculated based on [EB05], which is described in this section. 80
5.2 Pearson correlation coefficients between the existing local metrics with $h = 2$ and the global closeness centrality on various networks, where $k \ll N$. 89

List of Algorithms

1 Collect_Observations 45
2 Update_Network_Model 46
3 Dynamic_Network_Inference (Dyference) 48
4 Dynamic Nonparametric Network Distribution Inference 65
5 DNND-Inference 67
6 The Proposed Method: CS-HiBet 79

Bibliography

[AB02] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47, 2002.

[ABFX08] 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.

[ACIM99] Donald Aingworth, Chandra Chekuri, Piotr Indyk, and Rajeev Motwani. Fast estimation of diameter and shortest paths (without matrix multiplication). *SIAM Journal on Computing*, 28(4):1167–1181, 1999.

[ACL95] Mikhail J. Atallah, Danny Z. Chen, and DT Lee. An optimal algorithm for shortest paths on weighted interval and circular-arc graphs, with applications. *Algorithmica*, 14(5):429–441, 1995.

[ADFDJ03] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. An introduction to mcmc for machine learning. *Machine learning*, 50(1-2):5–43, 2003.

[AGZ15] Ayan Acharya, Joydeep Ghosh, and Mingyuan Zhou. Nonparametric bayesian factor analysis for dynamic count matrices. *arXiv preprint arXiv:1512.08996*, 2015.

[Ald81] David J Aldous. Representations for partially exchangeable arrays of random variables. *Journal of Multivariate Analysis*, 11(4):581–598, 1981.

[ALN+10] Konstantin Avrachenkov, Nelly Litvak, Danil A Nemirovsky, Elena Smirnova, and Marina Sokol. Monte carlo methods for top-k personalized pagerank lists and name disambiguation. *arXiv preprint arXiv:1008.3775*, 2010.

[ALST14] Konstantin Avrachenkov, Nelly Litvak, Marina Sokol, and Don Towsley. Quick detection of nodes with large degrees. *Internet Mathematics*, 10(1-2):1–19, 2014.

[APU09] Sitaram Asur, Srinivasan Parthasarathy, and Duygu Ucar. An event-based framework for characterizing the evolutionary behavior of interaction graphs. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 3(4):16, 2009.

[ARSM17] Oriol Artime, José J Ramasco, and Maxi San Miguel. Dynamics on networks: competition of temporal and topological correlations. *Scientific reports*, 7:41627, 2017.

[AW12] Sinan Aral and Dylan Walker. Identifying influential and susceptible members of social networks. *Science*, 337(6092):337–341, 2012.

[AX09] Amr Ahmed and Eric P Xing. Recovering time-varying networks of dependencies in social and biological studies. *Proceedings of the National Academy of Sciences*, 106(29):11878–11883, 2009.

[BA99] A. L. Barabasi and R. Albert. Emregence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[Bav50] A. Bavelas. Communication patterns in task-oriented groups. *J. Acoustical Soc. of Am.*, 22:725–730, 1950.

[BBC+19] Elisabetta Bergamini, Michele Borassi, Pierluigi Crescenzi, Andrea Marino, and Henning Meyerhenke. Computing top-k closeness centrality faster in unweighted graphs. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 13(5):53, 2019.

[BBH12] 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.

[BC09] Peter J Bickel and Aiyou Chen. A nonparametric view of network models and newman–girvan and other modularities. *Proceedings of the National Academy of Sciences*, 106(50):21068–21073, 2009.

[BCCG15] Christian Borgs, Jennifer T Chayes, Henry Cohn, and Shirshendu Ganguly. Consistent nonparametric estimation for heavy-tailed sparse graphs. *arXiv preprint arXiv:1508.06675*, 2015.

[BE06] Stephen P Borgatti and Martin G Everett. A graph-theoretic perspective on centrality. *Social networks*, 28(4):466–484, 2006.

[BF11] David M Blei and Peter I Frazier. Distance dependent chinese restaurant processes. *Journal of Machine Learning Research*, 12(Aug):2461–2488, 2011.

[BJ+06] David M Blei, Michael I Jordan, et al. Variational inference for dirichlet process mixtures. *Bayesian analysis*, 1(1):121–143, 2006.

[BJR07] Béla Bollobás, Svante Janson, and Oliver Riordan. The phase transition in inhomogeneous random graphs. *Random Structures & Algorithms*, 31(1):3–122, 2007.

[BK10] Michael Brautbar and Michael J Kearns. Local algorithms for finding interesting individuals in large networks. 2010.

[BK14] Joonhyun Bae and Sangwook Kim. Identifying and ranking influential spreaders in complex networks by neighborhood coreness. *Physica A: Statistical Mechanics and its Applications*, 395:549–559, 2014.

[BKM17] 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.

[BLG16] Simon Bourigault, Sylvain Lamprier, and Patrick Gallinari. Representation learning for information diffusion through social networks: an embedded cascade model. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 573–582. ACM, 2016.

[BM73] D. Blackwell and J. MacQueen. Ferguson distributions via Polya urn schemes. *The Annals of Statistics*, 1:353–355, 1973.

[BP07] Ulrik Brandes and Christian Pich. Centrality estimation in large networks. *International Journal of Bifurcation and Chaos*, 17(07):2303–2318, 2007.

[Bra01] Ulrik Brandes. A faster algorithm for betweenness centrality. *Journal of mathematical sociology*, 25(2):163–177, 2001.

[CAD+14] Justin Cheng, Lada Adamic, P Alex Dow, Jon Michael Kleinberg, and Jure Leskovec. Can cascades be predicted? In *Proceedings of the 23rd international conference on World wide web*, pages 925–936, 2014.

[CCB16] Diana Cai, Trevor Campbell, and Tamara Broderick. Edge-exchangeable graphs and sparsity. In *Advances in Neural Information Processing Systems*, pages 4249–4257, 2016.

[CD15a] Harry Crane and Walter Dempsey. A framework for statistical network modeling. *arXiv preprint arXiv:1509.08185*, 2015.

[CD15b] Harry Crane and Walter Dempsey. A typical scaling behavior persists in real world interaction networks. *arXiv preprint arXiv:1509.08184*, 2015.

[CD16] Harry Crane and Walter Dempsey. Edge exchangeable models for network data. *arXiv preprint arXiv:1603.04571*, 2016.

[CD18] Harry Crane and Walter Dempsey. Edge exchangeable models for interaction networks. *Journal of the American Statistical Association*, 113(523):1311–1326, 2018.

[CDD12] François Caron, Manuel Davy, and Arnaud Doucet. Generalized polya urn for time-varying dirichlet process mixtures. *arXiv preprint arXiv:1206.5254*, 2012.

[CF17] François Caron and Emily B Fox. Sparse graphs using exchangeable random measures. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(5):1295–1366, 2017.

[CKMS12] M. Cheraghchi, A. Karbasi, S. Mohajer, and V. Saligrama. Graph constrained group testing. *IEEE Trans. Inf. Theory*, 58(1):248–262, Jan. 2012.

[CRS12] Colin Cooper, Tomasz Radzik, and Yiannis Siantos. A fast algorithm to find all high degree vertices in power law graphs. In *Proceedings of the 21st International Conference on World Wide Web*, pages 1007–1016. ACM, 2012.

[CRTVB07] L da F Costa, Francisco A Rodrigues, Gonzalo Travieso, and Paulino Ribeiro Villas Boas. Characterization of complex networks: A survey of measurements. *Advances in physics*, 56(1):167–242, 2007.

[CS09] Jing Chen and Linfu Sun. Evaluation of node importance in complex networks [j]. *Journal of Southwest Jiaotong University*, 44(3):426–429, 2009.

[DBFL+13] Pierpaolo De Blasi, Stefano Favaro, Antonio Lijoi, Ramsés H Mena, Igor Prünster, and Matteo Ruggiero. Are gibbs-type priors the most natural generalization of the dirichlet process? *IEEE transactions on pattern analysis and machine intelligence*, 37(2):212–229, 2013.

[DD14] Daniele Durante and David B Dunson. Nonparametric bayes dynamic modelling of relational data. *Biometrika*, 101(4):883–898, 2014.

[DDEK12] M Davenport, M Duarte, Y Eldar, and G Kutyniok. Introduction to compressed sensing, chapter in compressed sensing: Theory and applications, 2012.

[DEP10] Shlomi Dolev, Yuval Elovici, and Rami Puzis. Routing betweenness centrality. *Journal of the ACM (JACM)*, 57(4):25, 2010.

[DHJ08] Persi Diaconis, Susan Holmes, and Svante Janson. Threshold graph limits and random threshold graphs. *Internet Mathematics*, 5(3):267–320, 2008.

[DKA11] Daniel M Dunlavy, Tamara G Kolda, and Evrim Acar. Temporal link prediction using matrix and tensor factorizations. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5(2):10, 2011.

[Don06] D. Donoho. Compressed sensing. *IEEE Trans. Inf. Theory*, 52(4):1289–1306, Apr. 2006.

[DWPX14] Avinava Dubey, Sinead Williamson, and Eric P Xing. Parallel markov chain monte carlo for pitman-yor mixture models. 2014.

[DWW99] P Damlen, John Wakefield, and Stephen Walker. Gibbs sampling for bayesian non-conjugate and hierarchical models by using auxiliary variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(2):331–344, 1999.

[EB05] Martin Everett and Stephen P Borgatti. Ego network betweenness. *Social networks*, 27(1):31–38, 2005.

[ER60] P. Erdos and A. Renyi. On the evolution of random graphs. In *Publication of the Mathematical Institute of the Hungarian Academy of Science*, pages 17–61, 1960.

[Fac13] Facebook Connections Limit. *https://www.facebook.com/help/community/question/?id=492434414172691*, 2013.

[FDA+11] James Foulds, Christopher DuBois, Arthur Asuncion, Carter Butts, and Padhraic Smyth. A dynamic relational infinite feature model for longitudinal social networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 287–295, 2011.

[Fea04] Paul Fearnhead. Particle filters for mixture models with an unknown number of components. *Statistics and Computing*, 14(1):11–21, 2004.

[Fer73] Thomas S Ferguson. A bayesian analysis of some nonparametric problems. *The annals of statistics*, pages 209–230, 1973.

[Fox09] Emily Beth Fox. *Bayesian nonparametric learning of complex dynamical phenomena*. PhD thesis, Massachusetts Institute of Technology, 2009.

[Fre77] Linton C Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[FSJW09] Emily Fox, Erik B Sudderth, Michael I Jordan, and Alan S Willsky. Nonparametric bayesian learning of switching linear dynamical systems. In *Advances in Neural Information Processing Systems*, pages 457–464, 2009.

[FSX09] Wenjie Fu, Le Song, and Eric P Xing. Dynamic mixed membership blockmodel for evolving networks. In *Proceedings of the 26th annual international conference on machine learning*, pages 329–336, 2009.

[GBWD+18] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

[GCWG15] Hong Ge, Yutian Chen, Moquan Wan, and Zoubin Ghahramani. Distributed inference for dirichlet process mixture models. In *International Conference on Machine Learning*, pages 2276–2284, 2015.

[Gee17] D. Geere. *Samsung offers free phones to frustrated iPhone users, http://www.cnn.com/2010/TECH/mobile/07/24/ samsung. replacing.iphones, CNN Tech.*, 2017.

[GHFX07] Fan Guo, Steve Hanneke, Wenjie Fu, and Eric P Xing. Recovering temporally rewiring networks: A model-based approach. In *Proceedings of the 24th international conference on Machine learning*, pages 321–328. ACM, 2007.

[GKMGR17] E. Ghalebi K., H. Mahyar, R. Grosu, and H. R. Rabiee. Compressive Sampling for Sparse Recovery in Networks. In *The 23rd ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), 13th MLG Workshop, Halifax, Nova Scotia, Canada*, August 2017.

[GL16] 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.

[GMG+19] Elahe Ghalebi, Hamidreza Mahyar, Radu Grosu, Graham W Taylor, and Sinead A Williamson. A nonparametric bayesian model for sparse temporal multigraphs. *arXiv preprint arXiv:1910.05098*, 2019.

[GMGL18] Elahe Ghalebi, Baharan Mirzasoleiman, Radu Grosu, and Jure Leskovec. Dynamic network model from partial observations. In *Advances in Neural Information Processing Systems*, pages 9862–9872, 2018.

[GRLK10] Manuel Gomez Rodriguez, Jure Leskovec, and Andreas Krause. Inferring networks of diffusion and influence. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1019–1028. ACM, 2010.

[GRLS13] Manuel Gomez Rodriguez, Jure Leskovec, and Bernhard Schölkopf. Structure and dynamics of information pathways in online media. In *Proceedings*

of the sixth ACM international conference on Web search and data mining, pages 23–32. ACM, 2013.

[GrS11] Manuel Gomez-rodriguez and David Balduzzi Bernhard Schölkopf. Uncovering the temporal dynamics of diffusion networks. In *in Proc. of the 28th Int. Conf. on Machine Learning (ICMLâ11*. Citeseer, 2011.

[GSR+17] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1263–1272. JMLR. org, 2017.

[GUSB11] Soumya Ghosh, Andrei B Ungureanu, Erik B Sudderth, and David M Blei. Spatial distance dependent Chinese restaurant processes for image segmentation. In *Advances in Neural Information Processing Systems*, pages 1476–1484, 2011.

[HBG04] Lars Hufnagel, Dirk Brockmann, and Theo Geisel. Forecast and control of epidemics in a globalized world. *Proceedings of the National Academy of Sciences*, 101(42):15124–15129, 2004.

[HG13] Creighton Heaukulani and Zoubin Ghahramani. Dynamic probabilistic models for latent feature propagation in social networks. In *International Conference on Machine Learning*, pages 275–283, 2013.

[HHJ03] Petter Holme, Mikael Huss, and Hawoong Jeong. Subnetwork hierarchies of biochemical pathways. *Bioinformatics*, 19(4):532–538, 2003.

[HL14] Nathan O Hodas and Kristina Lerman. The simple rules of social contagion. *Scientific reports*, 4:4343, 2014.

[HLL83] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

[Hof09] Peter D Hoff. Multiplicative latent factor models for description and prediction of social networks. *Computational and mathematical organization theory*, 15(4):261, 2009.

[Hoo79] D.N. Hoover. Relations on probability spaces and arrays of random variables. *Preprint. Institute for Advanced Study, Princeton.*, 1979.

[HSM16] Tue Herlau, Mikkel N Schmidt, and Morten Mørup. Completely random measures for modelling block-structured sparse networks. In *Advances in Neural Information Processing Systems*, pages 4260–4268, 2016.

[HSS97] K Han, Chandra N Sekharan, and R Sridhar. Unified all-pairs shortest path algorithms in the chordal hierarchy. *Discrete Applied Mathematics*, 77(1):59–71, 1997.

[HSX11] 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.

[HYL12] Bonan Hou, Yiping Yao, and Dongsheng Liao. Identifying all-around nodes for spreading dynamics in complex networks. *Physica A: Statistical Mechanics and its Applications*, 391(15):4012–4017, 2012.

[ISLR13] Muhammad U Ilyas, M Zubair Shafiq, Alex X Liu, and Hayder Radha. A distributed algorithm for identifying information hubs in social networks. *IEEE journal on selected areas in communications*, 31(9):629–640, 2013.

[Jan18] Svante Janson. On edge exchangeable random graphs. *Journal of statistical physics*, 173(3-4):448–484, 2018.

[JKL+05] Riko Jacob, Dirk Koschützki, Katharina Anna Lehmann, Leon Peeters, and Dagmar Tenfelde-Podehl. Algorithms for centrality indices. In *Network Analysis*, pages 62–82. Springer, 2005.

[Jor10] Michael I Jordan. Bayesian nonparametric learning: Expressive priors for intelligent systems. *Heuristics, probability and causality: A tribute to Judea Pearl*, 11:167–185, 2010.

[Kal06] Olav Kallenberg. *Probabilistic symmetries and invariance principles.* Springer Science & Business Media, 2006.

[KAS+13] Nicolas Kourtellis, Tharaka Alahakoon, Ramanuja Simha, Adriana Iamnitchi, and Rahul Tripathi. Identifying high betweenness centrality nodes in large social networks. *Social Network Analysis and Mining*, 3(4):899–914, 2013.

[KGW11] Maria Kalli, Jim E Griffin, and Stephen G Walker. Slice sampling mixture models. *Statistics and computing*, 21(1):93–105, 2011.

[KHP+07] Maksim Kitsak, Shlomo Havlin, Gerald Paul, Massimo Riccaboni, Fabio Pammolli, and H Eugene Stanley. Betweenness centrality of fractal and nonfractal scale-free model networks and tests on real networks. *Physical Review E*, 75(5):056115, 2007.

[KKT03] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[KL13] Myunghwan Kim and Jure Leskovec. Nonparametric multi-group membership model for dynamic networks. In *Advances in neural information processing systems*, pages 1385–1393, 2013.

[KLMST11] Anne-Marie Kermarrec, Erwan Le Merrer, Bruno Sericola, and Gilles Trédan. Second order centrality: Distributed assessment of nodes criticity in complex networks. *Computer Communications*, 34(5):619–628, 2011.

[KLMT08] Ravi Kumar, Kevin Lang, Cameron Marlow, and Andrew Tomkins. Efficient discovery of authoritative resources. In *2008 IEEE 24th International Conference on Data Engineering*, pages 1495–1497. IEEE, 2008.

[KLR17] Elihu Katz, Paul F Lazarsfeld, and Elmo Roper. *Personal influence: The part played by people in the flow of mass communications*. Routledge, 2017.

[KN11] B. Karrer and M.E.J. Newman. Stochastic blockmodels and community structure in networks. *Physical Review E*, 83(1):016107, 2011.

[KPF+17] Sofia Ira Ktena, Sarah Parisot, Enzo Ferrante, Martin Rajchl, Matthew Lee, Ben Glocker, and Daniel Rueckert. Distance metric learning using graph convolutional networks: Application to functional brain networks. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 469–477. Springer, 2017.

[KTG+06] C. Kemp, J.B. Tenenbaum, T.L. Griffiths, T. Yamada, and N. Ueda. Learning systems of concepts with an infinite relational model. In *National Conference on Artificial Intelligence (AAAI)*, pages 381–388, 2006.

[KWT07] Kenichi Kurihara, Max Welling, and Yee Whye Teh. Collapsed variational dirichlet process mixture models. In *IJCAI*, volume 7, pages 2796–2801, 2007.

[KWV07] Kenichi Kurihara, Max Welling, and Nikos Vlassis. Accelerated variational dirichlet process mixtures. In *Advances in neural information processing systems*, pages 761–768, 2007.

[KY12] Hyoungshick Kim and Eiko Yoneki. Influential neighbours selection for information diffusion in online social networks. In *2012 21st international conference on computer communications and networks (ICCCN)*, pages 1–7. IEEE, 2012.

[LAM12] Dan Lovell, Ryan P Adams, and VK Mansingka. Parallel markov chain monte carlo for dirichlet process mixtures. In *Workshop on Big Learning, NIPS*, 2012.

[LBK09] Jure Leskovec, Lars Backstrom, and Jon Kleinberg. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 497–506. ACM, 2009.

[LCC16] Min-Joong Lee, Sunghee Choi, and Chin-Wan Chung. Efficient algorithms for updating betweenness centrality in fully dynamic graphs. *Information Sciences*, 326:278–296, 2016.

[LCK+10] Jure Leskovec, Deepayan Chakrabarti, Jon Kleinberg, Christos Faloutsos, and Zoubin Ghahramani. Kronecker graphs: An approach to modeling networks. *Journal of Machine Learning Research*, 11(Feb):985–1042, 2010.

[Lee06] Chang-Yong Lee. Correlations among centrality measures in complex networks. *arXiv preprint physics/0605220, 2006*, 2006.

[LGF10] Dahua Lin, Eric Grimson, and John W Fisher. Construction of dependent dirichlet processes based on poisson processes. In *Advances in neural information processing systems*, pages 1396–1404, 2010.

[LGH06] Stefan Lämmer, Björn Gehlsen, and Dirk Helbing. Scaling laws in the spatial structure of urban road networks. *Physica A: Statistical Mechanics and its Applications*, 363(1):89–95, 2006.

[LHK10] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. Predicting positive and negative links in online social networks. In *Proceedings of the 19th international conference on World wide web*, pages 641–650. ACM, 2010.

[LJCC18] Juho Lee, Lancelot F James, Seungjin Choi, and François Caron. A Bayesian model for sparse graphs with flexible degree distribution and overlapping community structure. *arXiv preprint arXiv:1810.01778*, 2018.

[LK04] Katharina A Lehmann and Michael Kaufmann. *Decentralized algorithms for evaluating centrality in complex networks*. Universität Tübingen, 2004.

[LKF07] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Trans. Knowl. Discov. Data*, 1(1), March 2007.

[LLDM09] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, 6(1):29–123, 2009.

[LLS+19] Renjie Liao, Yujia Li, Yang Song, Shenlong Wang, Charlie Nash, William L. Hamilton, David Duvenaud, Raquel Urtasun, and Richard Zemel. Efficient graph generation with graph recurrent attention networks. In *NeurIPS*, 2019.

[LMGM17] Cheng Li, Jiaqi Ma, Xiaoxiao Guo, and Qiaozhu Mei. Deepcas: An end-to-end predictor of information cascades. In *Proceedings of the 26th International Conference on World Wide Web*, pages 577–586. International World Wide Web Conferences Steering Committee, 2017.

[LMR+11] Yeon-sup Lim, Daniel S Menasché, Bruno Ribeiro, Don Towsley, and Prithwish Basu. Online estimating the k central nodes of a network. In *2011 IEEE Network Science Workshop*, pages 118–122. IEEE, 2011.

[LOGR12] James Lloyd, Peter Orbanz, Zoubin Ghahramani, and Daniel M Roy. Random function priors for exchangeable arrays with applications to graphs and relational data. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 998–1006. Curran Associates, Inc., 2012.

[LRG13] Jian-Guo Liu, Zhuo-Ming Ren, and Qiang Guo. Ranking the spreading influence in complex networks. *Physica A: Statistical Mechanics and its Applications*, 392(18):4154–4159, 2013.

[LSK06] Jure Leskovec, Ajit Singh, and Jon Kleinberg. Patterns of influence in a recommendation network. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 380–389. Springer, 2006.

[LVD+18] Yujia Li, Oriol Vinyals, Chris Dyer, Razvan Pascanu, and Peter Battaglia. Learning deep generative models of graphs. *arXiv preprint arXiv:1803.03324*, 2018.

[LZL18] Yibo Li, Liangren Zhang, and Zhenming Liu. Multi-objective de novo drug design with conditional graph generative model. *Journal of cheminformatics*, 10(1):33, 2018.

[LZZS16] Linyuan Lü, Tao Zhou, Qian-Ming Zhang, and H Eugene Stanley. The h-index of a network node and its relation to degree and coreness. *Nature communications*, 7:10168, 2016.

[Mac00] Steven N MacEachern. Dependent dirichlet processes. *Unpublished manuscript, Department of Statistics, The Ohio State University*, pages 1–40, 2000.

[Mah15] Hamidreza Mahyar. Detection of top-k central nodes in social networks: a compressive sensing approach. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 902–909. ACM, 2015.

[MBW10] Arun S Maiya and Tanya Y Berger-Wolf. Online sampling of high centrality individuals in social networks. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 91–98. Springer, 2010.

[MCL99] Steven N MacEachern, Merlise Clyde, and Jun S Liu. Sequential importance sampling for nonparametric bayes models: The next generation. *Canadian Journal of Statistics*, 27(2):251–267, 1999.

[MCM+11] Anmol Madan, Manuel Cebrian, Sai Moturu, Katayoun Farrahi, et al. Sensing the" health state" of a community. *IEEE Pervasive Computing*, 11(4):36–45, 2011.

[MCN16] R. Middya, N. Chakravarty, and M. K. Naskar. Compressive sensing in wireless sensor networks-a survey. *IETE Technical Review, 2016*, 2016.

[MHS+14] Ricardo Pio Monti, Peter Hellyer, David Sharp, Robert Leech, Christoforos Anagnostopoulos, and Giovanni Montana. Estimating time-varying brain connectivity networks from functional mri time series. *NeuroImage*, 103:427–443, 2014.

[Min01] Thomas P Minka. Expectation propagation for approximate bayesian inference. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 362–369. Morgan Kaufmann Publishers Inc., 2001.

[MJG09] Kurt Miller, Michael I Jordan, and Thomas L Griffiths. Nonparametric latent feature models for link prediction. In *Advances in neural information processing systems*, pages 1276–1284, 2009.

[ML10] Seth Myers and Jure Leskovec. On the convexity of latent social network inference. In *Advances in neural information processing systems*, pages 1741–1749, 2010.

[MRH13] Hamidreza Mahyar, Hamid R Rabiee, and Zakieh S Hashemifar. Ucs-nt: an unbiased compressive sensing framework for network tomography. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4534–4538. IEEE, 2013.

[MRHS13] H. Mahyar, H. R. Rabiee, Z. S. Hashemifar, and P. Siyari. UCS-WN: An Unbiased Compressive Sensing Framework for Weighted Networks. In *Conference on Information Sciences and Systems (CISS), Baltimore, USA*, pages 1–6, Mar. 2013.

[MRM+15a] H. Mahyar, H. R. Rabiee, A. Movaghar, E. Ghalebi, and A. Nazemian. CS-ComDet: A compressive sensing approach for inter-community detection in social networks. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), Paris, France*, pages 89–96, Aug. 2015.

[MRM+15b] H. Mahyar, H. R. Rabiee, A. Movaghar, R. Hasheminezhad, E. Ghalebi, and A. Nazemian. A low-cost sparse recovery framework for weighted

networks under compressive sensing. In *IEEE International Conference on Social Computing and Networking (SocialCom), Chengdu, China*, pages 183–190, Dec. 2015.

[N+03] Radford M Neal et al. Slice sampling. *The annals of statistics*, 31(3):705–767, 2003.

[Nar05] S. Narayanan. *The Betweenness Centrality of Biological Networks*. University Libraries, Virginia Polytechnic Institute and State University, 2005.

[New05] Mark EJ Newman. A measure of betweenness centrality based on random walks. *Social networks*, 27(1):39–54, 2005.

[New10] S. Newman. Networks: An introduction. *Oxford University Press*, pages 168–234, 2010.

[New18] Mark Newman. *Networks*. Oxford university press, 2018.

[NK08] Soumendra Nanda and David Kotz. Localized bridging centrality for distributed network analysis. In *2008 Proceedings of 17th International Conference on Computer Communications and Networks*, pages 1–6. IEEE, 2008.

[NS17] Yin Cheng Ng and Ricardo Silva. A dynamic edge exchangeable model for sparse temporal networks. *arXiv:1710.04008*, 2017.

[NSRJ11] A Namaki, AH Shirazi, R Raei, and GR Jafari. Network analysis of a financial market based on genuine correlation and threshold method. *Physica A: Statistical Mechanics and its Applications*, 390(21-22):3835–3841, 2011.

[OAS10] Tore Opsahl, Filip Agneessens, and John Skvoretz. Node centrality in weighted networks: Generalizing degree and shortest paths. *Social networks*, 32(3):245–251, 2010.

[OCL08] Kazuya Okamoto, Wei Chen, and Xiang-Yang Li. Ranking of closeness centrality for large-scale social networks. In *International Workshop on Frontiers in Algorithmics*, pages 186–195. Springer, 2008.

[OP09] Tore Opsahl and Pietro Panzarasa. Clustering in weighted networks. *Social networks*, 31(2):155–163, 2009.

[OR14] 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, 2014.

[OT10] Peter Orbanz and Yee Whye Teh. Bayesian nonparametric models. *Encyclopedia of Machine Learning*, pages 81–89, 2010.

[P+02] Jim Pitman et al. Combinatorial stochastic processes. Technical report, Technical Report 621, Dept. Statistics, UC Berkeley, 2002. Lecture notes for . . . , 2002.

[PBL17] Ashwin Paranjape, Austin R Benson, and Jure Leskovec. Motifs in temporal networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 601–610. ACM, 2017.

[PCT16] Konstantina Palla, Francois Caron, and Yee Whye Teh. Bayesian nonparametrics for sparse dynamic networks. *arXiv preprint arXiv:1607.01624*, 2016.

[Pit96] Jim Pitman. Some developments of the Blackwell-MacQueen urn scheme. *Lecture Notes-Monograph Series*, pages 245–267, 1996.

[POC09] Pietro Panzarasa, Tore Opsahl, and Kathleen M Carley. Patterns and dynamics of users' behavior and interaction: Network analysis of an online community. *Journal of the American Society for Information Science and Technology*, 60(5):911–932, 2009.

[PTT+16] D. Pastén, F. Torres, B. Toledo, V. Muñoz, J. Rogan, and J. A. Valdivia. Time-based network analysis before and after the m_w 8.3 illapel earthquake 2015 chile. *Pure and Applied Geophysics*, 173(7):2267–2275, 2016.

[PY+97] Jim Pitman, Marc Yor, et al. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, 25(2):855–900, 1997.

[RAY+16] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016.

[RDC08] Lu Ren, David B. Dunson, and Lawrence Carin. The dynamic hierarchical dirichlet process. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 824–831, New York, NY, USA, 2008.

[RK16] Matteo Riondato and Evgenios M Kornaropoulos. Fast approximation of betweenness centrality through sampling. *Data Mining and Knowledge Discovery*, 30(2):438–475, 2016.

[RS12] Manuel Gomez Rodriguez and Bernhard Schölkopf. Submodular inference of diffusion networks from multiple trees. *arXiv preprint arXiv:1205.1671*, 2012.

[Sab66] Gert Sabidussi. The centrality index of a graph. *Psychometrika*, 31(4):581–603, 1966.

[Set94] Jayaram Sethuraman. A constructive definition of dirichlet priors. *Statistica sinica*, pages 639–650, 1994.

[SG05] Brajendra K Singh and Neelima Gupte. Congestion and decongestion in a communication network. *Physical Review E*, 71(5):055103, 2005.

[SGI17] Akrati Saxena, Ralucca Gera, and SRS Iyengar. Fast estimation of closeness centrality ranking. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 80–85, 2017.

[SK17] Martin Simonovsky and Nikos Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3693–3702, 2017.

[SN97] T.A.B. Snijders and T. Nowicki. Estimation and prediction for stochastic blockmodels for graphs with latent block structure. *Journal of Classification*, 14(1):75–100, 1997.

[SN08] N Rama Suri and Yadati Narahari. Determining the top-k nodes in social networks using the shapley value. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 3*, pages 1509–1512. International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[Spr78] DA Sprott. Urn models and their application—an approach to modern discrete probability theory, 1978.

[SWA09] Padhraic Smyth, Max Welling, and Arthur U Asuncion. Asynchronous distributed learning of topic models. In *Advances in Neural Information Processing Systems*, pages 81–88, 2009.

[Teh06] Yee Whye Teh. A hierarchical bayesian language model based on pitman-yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 985–992. Association for Computational Linguistics, 2006.

[Teh10] Yee Whye Teh. Dirichlet process. *Encyclopedia of machine learning*, pages 280–287, 2010.

[Tib94] R. Tibshirani. Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society B*, 58:267–288, 1994.

[TJBB05] Yee W Teh, Michael I Jordan, Matthew J Beal, and David M Blei. Sharing clusters among related groups: Hierarchical dirichlet processes. In *Advances in neural information processing systems*, pages 1385–1392, 2005.

[TW+10] Martin A Tanner, Wing H Wong, et al. From em to data augmentation: the emergence of mcmc bayesian computation in the 1980s. *Statistical science*, 25(4):506–516, 2010.

[Twi17a] Twitter. Gephi platform for interactive visualization and exploration of graphs, 2017.

[Twi17b] Twitter Connections Limit. *https://support.twitter.com/articles/66885*, 2017.

[TZ12] L. Takac and M. Zabovsky. Data analysis in public social networks. In *International Scientific Conference and International Workshop Present Day Trends of Innovations*, volume 1, 2012.

[Wan06] David Eppstein Joseph Wang. Fast approximation of centrality. *Graph Algorithms and Applications*, 5(5):39, 2006.

[WDX13] Sinead Williamson, Avinava Dubey, and Eric Xing. Parallel markov chain monte carlo for nonparametric mixture models. In *International Conference on Machine Learning*, pages 98–106, 2013.

[WG07] Frank Wood and Thomas L Griffiths. Particle filtering for nonparametric bayesian matrix factorization. In *Advances in neural information processing systems*, pages 1513–1520, 2007.

[WGZ11] Klaus Wehmuth, Antonio Tadeu A Gomes, and Artur Ziviani. Dance: a framework for the distributed assessment of network centralities. *arXiv preprint arXiv:1108.1067*, 2011.

[Wil16] Sinead A Williamson. Nonparametric network models for link prediction. *The Journal of Machine Learning Research*, 17(1):7102–7121, 2016.

[WLY+14] Yingcai Wu, Shixia Liu, Kai Yan, Mengchen Liu, and Fangzhao Wu. Opinionflow: Visual analysis of opinion diffusion on social media. *IEEE transactions on visualization and computer graphics*, 20(12):1763–1772, 2014.

[WMSM09] Hanna M. Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. Evaluation methods for topic models. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1105–1112, 2009.

[WO13] Patrick J Wolfe and Sofia C Olhede. Nonparametric graphon estimation. *arXiv preprint arXiv:1309.5936*, 2013.

[WS98a] D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393(6684):440–442, 1998.

[WS98b] Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world'networks. *nature*, 393(6684):440, 1998.

[WXMT12] M. Wang, W. Xu, E. Mallada, and A.k Tang. Sparse recovery with graph constraints: Fundamental limits and measurement construction. In *IEEE INFOCOM*, pages 1871–1879, Mar. 2012.

[WZ12] Klaus Wehmuth and Artur Ziviani. Distributed assessment of the closeness centrality ranking in complex networks. In *Simp. Comp. Net. for Pract.*, 2012.

[WZLC17] Jia Wang, Vincent W Zheng, Zemin Liu, and Kevin Chen-Chuan Chang. Topological recurrent neural network for diffusion prediction. *arXiv preprint arXiv:1711.10162*, 2017.

[XFS+10] 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.

[XH14] 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.

[XKGH19] Saining Xie, Alexander Kirillov, Ross Girshick, and Kaiming He. Exploring randomly wired neural networks for image recognition. *arXiv preprint arXiv:1904.01569*, 2019.

[XLZ19] Junyu Xuan, Jie Lu, and Guangquan Zhang. A survey on bayesian nonparametric learning. *ACM Computing Surveys (CSUR)*, 52(1):13, 2019.

[XMT11] Weiyu Xu, Enrique Mallada, and Ao Tang. Compressive sensing over graphs. In *2011 Proceedings IEEE INFOCOM*, pages 2087–2095. IEEE, 2011.

[Xu15] Kevin Xu. Stochastic block transition models for dynamic networks. In *International Conference on Artificial Intelligence and Statistics*, pages 1079–1087, 2015.

[XW17] Shuang Xu and Pei Wang. Identifying important nodes by adaptive leaderrank. *Physica A: Statistical Mechanics and its Applications*, 469:654–664, 2017.

[YK18a] Sikun Yang and Heinz Koeppl. Dependent relational gamma process models for longitudinal networks. In *International Conference on Machine Learning*, pages 5547–5556, 2018.

[YK18b] Sikun Yang and Heinz Koeppl. A poisson gamma probabilistic model for latent node-group memberships in dynamic networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[YL15] J. Yang and J. Leskovec. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems*, 42(1):181–213, 2015.

[YLY+18] Jiaxuan You, Bowen Liu, Zhitao Ying, Vijay Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. In *Advances in Neural Information Processing Systems*, pages 6410–6421, 2018.

[YTQ16] Keyou You, Roberto Tempo, and Li Qiu. Distributed algorithms for computation of centrality measures in complex networks. *IEEE Transactions on Automatic Control*, 62(5):2080–2094, 2016.

[YYR+18] Jiaxuan You, Rex Ying, Xiang Ren, William L. Hamilton, and Jure Leskovec. GraphRNN: Generating realistic graphs with deep auto-regressive models. In *International Conference on Machine Learning*, pages 5694–5703, 2018.

[Zho15] Mingyuan Zhou. Infinite edge partition models for overlapping community detection and link prediction. In *Artificial Intelligence and Statistics*, pages 1135–1143, 2015.