Learning with Attributed Networks: Algorithms and Applications

by

Jundong Li

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved June 2019 by the
Graduate Supervisory Committee:

Huan Liu, Chair
Christos Faloutsos
Jingrui He
Guoliang Xue

ARIZONA STATE UNIVERSITY

August 2019

ABSTRACT

Attributes - that delineating the properties of data, and connections - that describing the dependencies of data, are two essential components to characterize most real-world phenomena. The synergy between these two principal elements renders a unique data representation - the attributed networks. In many cases, people are inundated with vast amounts of data that can be structured into attributed networks, and their use has been attractive to researchers and practitioners in different disciplines. For example, in social media, users interact with each other and also post personalized content; in scientific collaboration, researchers cooperate and are distinct from peers by their unique research interests; in complex diseases studies, rich gene expression complements to the gene-regulatory networks. Clearly, attributed networks are ubiquitous and form a critical component of modern information infrastructure. To gain deep insights from such networks, it requires a fundamental understanding of their unique characteristics and be aware of the related computational challenges.

My dissertation research aims to develop a suite of novel learning algorithms to understand, characterize, and gain actionable insights from attributed networks, to benefit high-impact real-world applications. In the first part of this dissertation, I mainly focus on developing learning algorithms for attributed networks in a static environment at two different levels: (i) attribute level - by designing feature selection algorithms to find high-quality features that are tightly correlated with the network topology; and (ii) node level - by presenting network embedding algorithms to learn discriminative node embeddings by preserving node proximity w.r.t. network topology structure and node attribute similarity. As changes are essential components of attributed networks and the results of learning algorithms will become stale over time, in the second part of this dissertation, I propose a family of online algorithms for attributed networks in a dynamic environment to continuously update the learn-

ing results on the fly. In fact, developing application-aware learning algorithms is more desired with a clear understanding of the application domains and their unique intents. As such, in the third part of this dissertation, I am also committed to advancing real-world applications on attributed networks by incorporating the objectives of external tasks into the learning process.

To my beloved parents and all my family.

ACKNOWLEDGMENTS

First and foremost, I am greatly indebted to my advisor, Dr. Huan Liu, for his consistent guidance, encouragement, inspiration, and support. I still remembered that he interrupted my first presentation in the group meeting, and taught me how to find good research problems and how to give intriguing presentations. Looking back, his critical feedback motivated me to set a high standard for myself in my early academic career, which I will benefit all my life. During the past five years, Dr. Liu gives me a lot of freedom to work on the research problems I am interested in, but meanwhile, he is always there when I need a second opinion, and helps me see the grand vision. He also tirelessly shapes me into an independent researcher by providing me a lot of opportunities engaged in a comprehensive set of academic activities. Dr. Liu is not only a great advisor but also a life mentor and an easygoing friend. I have benefited tremendously from his knowledge and experience, and his philosophies about work and life. Working with him for the past five years is definitely my lifelong assets.

I would like to thank my committee members Dr. Christos Faloutsos, Dr. Jingrui He, and Dr. Guoliang Xue, for their insightful comments and helpful suggestions on my research, as well as continued support with my job search. Dr. Christos Faloutsos has been a role model since I stepped into the data mining field. I really enjoyed the thought-provoking discussions with him, his sharp insights helped me rethink my research and inspired novel ideas. I took the statistical machine learning course from Dr. Jingrui He, which helped me lay down a solid foundation for my later research. I also took the optimization and game theory courses from Dr. Guoliang Xue, which provided me a new angle to look at optimization and applied mathematics. I also want to thank my master advisor Dr. Osmar R. Zaïane who introduced me into this exciting field and my intern mentor Dr. Yi Chang who helped me see real industry problems at Yahoo! Research.

I really appreciated the time working with Data Mining and Machine Learning lab members. I would like to thank Ali Abbasi, Ghazaleh Beigi, Kewei Cheng, Lu Cheng, Harsh Dani, Matthew Davis, Kaize Ding, Huiji Gao, Ruocheng Guo, Philippe Christophe Faucon, Pritam Gundecha, Xia (Ben) Hu, Isaac Jones, Nur Shazwani Kamrudin, Shamanth Kumar, Yunzhong Liu, Vineeth Rakesh Mohan, Raha Moraffah, Fred Morstatter, Tahora Hossein Nazer, Suhas Ranganath, Justin Sampson, Kai Shu, Jiliang Tang, Robert Trevino, Suhang Wang, Liang Wu, and Reza Zafarani for the invaluable interactions. In particular, Ben helped me a lot in my early stage and is always there for me. Jiliang passed the torch of feature selection to me and helped me finish my first paper at ASU. Ruocheng and Liang are not only reliable collaborators but also good friends to whom I can pour out my worries and stress.

The Ph.D. life will be lonely and miserable without the company and support of friends. I am very fortunate to have a group of wonderful friends who make the journey much more fun and enjoyable. I would like to thank Harsh Dani, Kaize Ding, Ruocheng Guo, Ling Jian, Liangyue Li, Xing Liang, Liang Wu, Yao Zhou, Dawei Zhou, and many other good friends. I would like to especially thank my dearest girlfriend Chen Chen for her unwavering love, support, and patience, and for making future career decisions together with me. I am very lucky to have you in my life.

Last but not least, I would like to express my deepest gratitude to my parents Feng Li and Xiaoxuan Wang for their endless love all these years, and they have been extremely supportive for every decision I made. Also, many thanks to my maternal grandparents Xinnan Wang and Longzhu Zhang who raised me up and always loved me; and my paternal parents Ruhe Li and Jiazhou Zhou who are always ready to hear my news and share my happiness. Unfortunately, my grandfather Ruhe Li passed away one and a half years ago. I am sure he will be very proud to see my graduation and future success. This dissertation is dedicated to all of them.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 Motivation and Overview

In data mining and machine learning, we often use attributes[1] as measurable properties to characterize phenomena of real-world data (Bishop, 2006). One of the most common assumptions in conventional data analytical tasks is that the feature representations of different data samples are independent and identically distributed (*i.i.d.*). However, this assumption is often untenable in the real world as different data samples are often explicitly or implicitly correlated with complex dependencies (Getoor and Taskar, 2007; Sen *et al.*, 2008). The synergy between the feature representations of individual data samples and the dependencies among different data samples yields a unique data representation - referred as attributed networks (Akoglu *et al.*, 2012; Huang *et al.*, 2017a,b; Li *et al.*, 2017b,c; Perozzi *et al.*, 2014a; Perozzi and Akoglu, 2016, 2018; Pfeiffer III *et al.*, 2014; Rezaei *et al.*, 2017; Robles *et al.*, 2016). Attributed networks are ubiquitous in myriad of high-impact domains, including social media platforms, citation networks, biology networks, and critical infrastructure systems, to name a few. As such, mining attributed networks has attracted a surge of research interests and has been attractive to researchers and practitioners from different disciplines, such as computer science, social science, network science, biology, and other related interdisciplinary research subjects. In order to harness the power of attributed networks, we propose a suite of novel learning algorithms to help better

[1]In this dissertation, we use *attributes* and *features* interchangeably. Meanwhile, we also use *networks* and *graphs* interchangeably.

understand, characterize, and gain actionable insights from such networks, to benefit high-impact real-world applications from different domains.

1.1.1 What Are Attributed Networks?

Networks[1] are widely used to represent various types of information systems where nodes represent entities such as users, web pages, and genes; while edges represent interactions between entities such as friendships, hyperlinks, and gene interactions. In contrast to conventional plain networks where only pairwise node dependencies are observed, nodes on attributed networks are often affiliated with a rich set of attributes delineating their properties. For example, the popularity of social media services not only allow users to interact with each other at a low cost but also enables them to generate and share rich content information (e.g., user profile and user-generated posts) (Mislove *et al.*, 2010); in the course of scientific collaboration, the collaborations among scholars form a co-authorship network and scholars are also distinct from their peers through their unique research profile (e.g., research interests and publications) (Yang *et al.*, 2013); to unravel the cellular organizations and functionalities in biology studies, gene ontology (GO) annotation information is often complementary to the raw protein-protein interaction topological structure (Zhang *et al.*, 2013).

Before presenting the detailed definitions of attributed networks, we will first summarize the notations used in this dissertation. Following the commonly used notations, we use bold uppercase characters for matrices (e.g., $\mathbf{A}$), bold lowercase characters for vectors (e.g., $\mathbf{a}$), normal lowercase characters for scalars (e.g., a), and calligraphic characters for sets (e.g., $\mathcal{F}$). We represent the i-th row of matrix $\mathbf{A}$ as $\mathbf{A}(i,:)$ or $\mathbf{A}_{i*}$, the j-th column as $\mathbf{A}(:,j)$ or $\mathbf{A}_{*j}$, the (i,j)-th entry as $\mathbf{A}(i,j)$ or $\mathbf{A}_{ij}$, the i-th element of vector $\mathbf{a}$ as a_i, transpose of $\mathbf{A}$ and $\mathbf{a}$ as $\mathbf{A}'$ and $\mathbf{a}'$ respectively, and trace of $\mathbf{A}$ as $tr(\mathbf{A})$ if it is a square matrix. We use diag($\mathbf{a}$) to denote the

diagonalization of vector $\mathbf{a}$. $\mathbf{1}$ denotes a column vector whose elements are all 1 and $\mathbf{I}$ denotes the identity matrix. For any matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$, its Frobenius norm is defined as $\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{d} \mathbf{A}(i,j)^2}$, its $\ell_{2,1}$-norm is $\|\mathbf{A}\|_{2,1} = \sum_{i=1}^{n} \sqrt{\sum_{j=1}^{d} \mathbf{A}(i,j)^2}$, and its ℓ_0-norm counts the number of nonzero elements in the matrix. The ℓ_2-norm of a vector $\mathbf{a} \in \mathbb{R}^d$ is $\|\mathbf{a}\|_2 = \sqrt{\mathbf{a}^T \mathbf{a}}$. The ℓ_1-norm of $\mathbf{a} \in \mathbb{R}^d$ is $\|\mathbf{a}\|_1 = \sum_{i=1}^{d} |\mathbf{a}_i|$. $\mathbf{1}(.)$ denotes an indicator function. Meanwhile, we use the subscript to denote the matrices, vectors, or sets at a specific time stamp (e.g., $\mathbf{A}^t$, $\mathbf{a}^t$, and $\mathcal{S}^t$).

Based on the above notations, the formal definition of attributed network and its data representation are as follows.

Definition 1. *(**Attributed Networks**): An attributed network $G = (\mathcal{V}, \mathcal{E}, \mathcal{F})$ consists of three important components: (1) $\mathcal{V}$: a set of nodes; (2) $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$: a set of edges showing the dependencies among the nodes in $\mathcal{V}$; and (3) $\mathcal{F}$: a set of attributes[2] delineating the properties of nodes; and (4) $\mathcal{X}$: a set of node-attribute pairs such that $\mathcal{X} \subseteq \mathcal{V} \times \mathcal{F}$.*

Definition 2. *(**Data Representation of Attributed Networks**): Let G be the given attributed network, where $\mathcal{V} = \{v_1, ..., v_n\}$ is the node set, $\mathcal{E} = \{e_1, ..., e_m\}$ is the edge set, and $\mathcal{F} = \{f_1, ..., f_d\}$ is the attribute set. We use matrix $\mathbf{A} \in \mathbb{R}_{\geq 0}^{n \times n}$ to denote the adjacency matrix of the network, where $\mathbf{A}_{ij} > 0$ is a positive number denoting the edge weight between v_i and v_j ($\mathbf{A}_{ij} = 0$ if no connection). Meanwhile, we use the matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n]' \in \mathbb{R}^{n \times d}$ to denote the feature representation of these n data samples (e.g., the values of the node-attribute pairs $\mathcal{X}$), where $\mathbf{x}_i \in \mathbb{R}^d$ is the attribute information of node v_i.*

The illustration of a typical attributed network is shown in Figure 1.1. As can

[2]We assume the feature values are numerical. The categorical features can also be considered here with one-hot encoding.

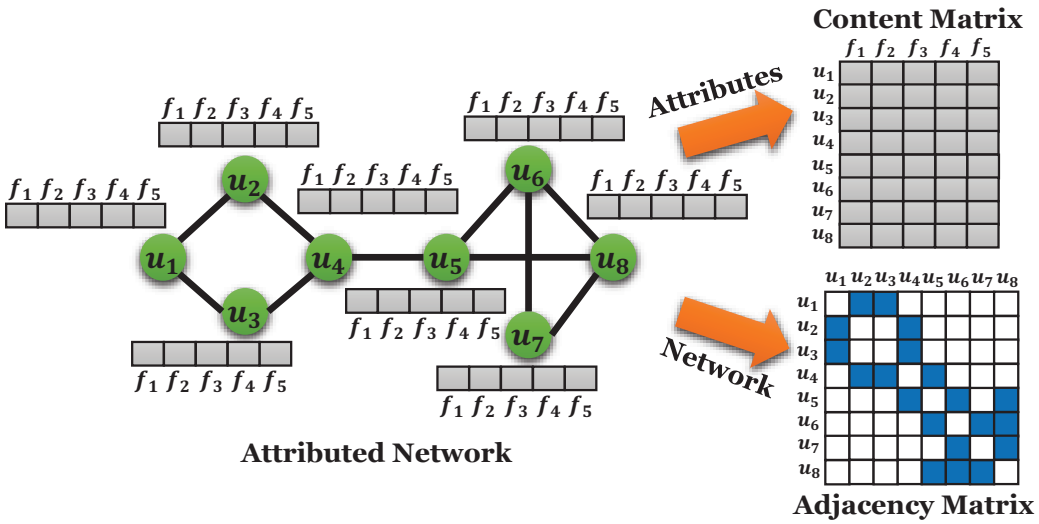


Figure 1.1: An Illustration of the Attributed Network.

be observed from the figure, an attributed network consists of two different data modalities: (i) the content matrix describing the properties of data samples; and (ii) the adjacency matrix characterizing the dependencies among different data samples.

1.1.2 Why Study Attributed Networks?

Apparently, to distill patterns or values from attributed networks, on one hand we can only look at the attribute information of different data samples, and then apply off-the-shelf machine learning and data mining tools; on the other hand, we can also work on the network information and then take advantage of advanced graph mining techniques. For example, to infer the political polarizations of users in a social network, we can either employ the user content information (e.g., user profile and user posts) and formulate the problem as a conventional classification problem, or take advantage of relationships among users (e.g., friendships or following/follower relations) and tackle the problem within the statistical relational learning paradigm (Getoor and Taskar, 2007; Taskar *et al.*, 2001). These approaches, however, inevitably ignore

the inherent correlations among two different data modalities of attributed networks. In fact, these two data modalities are often complementary with each other, and recent studies (Jensen *et al.*, 2004; La Fond and Neville, 2010) verified the existence of statistical dependencies (a.k.a. *autocorrelations*) between the attributes of linked nodes. For example, in social media, the political views (reflected in the user posts) of connected users are often more similar than those of a randomly selected user pair. The root cause of the correlations can be attributed to various social phenomena, including social influence (Marsden and Friedkin, 1993), homophily effect (McPherson *et al.*, 2001), and diffusion process (Doreian, 1989), among others. Hence, one natural question to ask is *will the fusion of these seemingly irrelevant information sources bring new insights?*

Meanwhile, there are a wealth of fascinating research questions that we can study on attributed networks. How to seamlessly fuse friendship relations and user-generated content to boost friend recommendation in context-rich social networks? How to identify anomalies that lead to system failures in a critical infrastructure network by making use of the observed information of infrastructures (e.g., electricity capacity of a power plant)? How to track the evolutionary patterns of academic communities in a time-evolving collaboration network? Whether the attributed networks can be used to solve other research problems with implicit network structure?

1.2 Research Challenges

To answer these above questions, it requires us to have a fundamental understanding of the unique characteristics of attributed networks, and be aware of the related computational challenges. Here, we summarize the main challenges that we often encounter when we are dealing with attributed networks:

1. *C1: Content Challenge* – We are now in the era of big data, where huge amounts

of high-dimensional data become ubiquitous in various domains. For example, social media platforms are swarming with low-quality user-generated content, and conventional text representations (e.g., Bag-of-Words and TF-IDF) often lead to a high-dimensional noisy representation of users. Another example is the high-dimensional gene expression of proteins in protein-protein interaction (PPI) networks. In these cases, a critical issue known as *the curse of dimensionality* arises. It refers to the phenomenon that data becomes sparser in the high-dimensional space, adversely affecting the learning algorithms designed for low-dimensional data (Friedman *et al.*, 2001; Guyon and Elisseeff, 2003; Keogh and Mueen, 2011; Li *et al.*, 2017a). Also, with a large number of features, learning models tend to overfit which may cause performance degradation on unseen data; and it significantly increases the memory storage requirements and computational costs for data analytics. As a summary, noisy and high-dimensional feature representations of nodes make conventional learning algorithms unequipped to handle attributed networks.

2. *C2: Structure Challenge* – Undoubtedly, in addition to the noisy node features, the network structure which encodes the dependencies among different data samples could also be noisy due to the imprecise data collection process and other potential factors (Abufouda and Zweig, 2017; Gu *et al.*, 2013; Namata *et al.*, 2010). Specifically, many attributed networks suffer from noisy links, and the underlying reason is that these links do not encode the real data dependencies and hence decreases the quality of the network. For example, in biology studies, the PPI network is often obtained based on high-throughput screening data analysis. In practice, the whole process is often erroneous, introducing noisy links to the constructed PPI networks. Meanwhile, the observed

network structure may only yield us a coarse indication of relations, while in many cases, a more nuanced representation of tie strength information among different nodes is required. The aforementioned issues bring challenges to accurate analysis of attributed networks as well as downstream applications.

3. *C3: Fusion Challenge* – We have shown that there are inherent correlations among the two different data modalities of attributed networks, i.e., the formation of one depends on and also influences the other one. Hence, effective data fusion plays a central role when developing principled learning algorithms on attributed networks. Despite its importance, the incompatibility issues may naturally appear when we are trying to leverage these two data sources in a collective manner (Singh and Gordon, 2008; Zhu *et al.*, 2007). Firstly, the illusion that all attributes are complementary to the network structure information will break as connected nodes may not be similar in the original node feature space due to the existence of noisy and irrelevant features. Secondly, as the network structure itself is very noisy, the node attribute similarity may not be aligned with the network topology structure. As such, a synergistic data fusion over attributed networks necessitates a consensus feature space in which the autocorrelation is maximally preserved.

4. *C4: Evolution Challenge* – A fundamental assumption behind existing learning algorithms on attributed networks is that networks and the node attributes are static and given a prior. However, this assumption is often untenable in practice as most real-world attributed networks are intrinsically dynamic, characterized by frequent structure and content changes. On one hand, the network structure often evolves a wide range of ways with different evolutionary semantics (Aggarwal and Subbian, 2014). On the other hand, node attributes also change

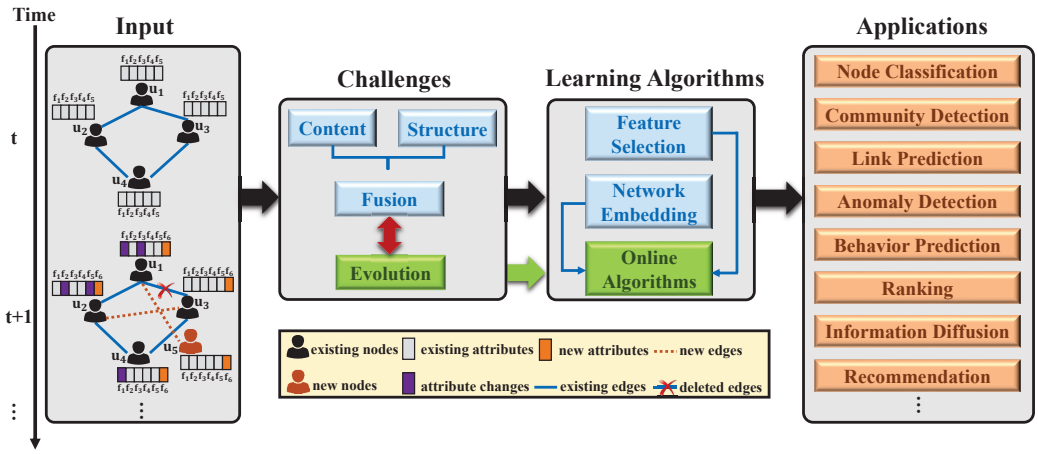


Figure 1.2: An Overview of My Research Contributions in This Dissertation.

dynamically such that new content patterns may emerge and outdated patterns will fade. Examples include emerging topics and slang words in social media after disasters. We refer such networks with both structure and content changes as *dynamic attributed networks*. In such cases, learning models are required to support online updates to maintain the freshness of end results for real-time insights.

1.3 Research Contributions

My research work boils down to developing principled learning algorithms to tackle the computational challenges (C1, C2, and C3) for static attributed networks, and with an additional challenge (C4) for attributed networks in a dynamic environment. The contributions of this dissertation are summarized as follows.

- *Leverage feature selection for learning in a static environment*: We study static attributed networks at the attribute level by designing novel feature selection algorithms to facilitate learning. The target is to find a subset of high-quality features that show strong dependencies with the network structure, which are

taken as input later by various off-the-shelf machine learning and data mining algorithms for actionable insights.

- *Leverage network embedding for learning in a static environment*: We study static attributed networks at the attributed level by presenting novel network embedding algorithms. The goal is to map each node on the attributed network into a new discriminative feature space in which the node proximity w.r.t. two different data modalities are well preserved. Similarly, the learned embeddings can also be utilized to advance various downstream graph mining tasks.

- *Generalize feature learning in a dynamic environment*: We generalize the above-mentioned feature learning (including feature selection and network embedding) algorithms in a dynamic environment when the attributed networks are continuously evolving over time. The purpose of this study is to develop effective yet efficient online learning algorithms that can quickly adapt to the changes of attributed networks by updating the learning results on the fly, which in turn enables us to gain real-time insights from the underlying system.

- *Advance applications on attributed networks*: We also attempt to advance real-world applications on attributed networks across different disciplines. Different from the aforementioned learning algorithms that are agnostic to specific tasks, here we pay more attention to incorporating the objectives of specific applications and domain knowledge into the learning process, in a sense that the developed algorithms will be customized for the targeted applications.

An overview of my dissertation research is summarized in Figure 1.2.

1.4 Organization

The remainder of this dissertation is organized as follows. Chapter 2 reviews related work on feature selection, network embedding, dynamic network analytics, and attributed network analytics. Chapter 3 introduces the proposed novel feature selection algorithms for learning on static attributed networks. Chapter 4 investigates attributed networks from a different perspective with attributed network embedding. Chapter 5 discusses how to develop an online feature selection algorithm for dynamic attributed networks. Chapter 6 generalizes the attributed network embedding in a dynamic setting by presenting an efficient online algorithm. Chapter 7 discusses the application of node classification on attributed networks and shows how to capture the personalized patterns of each node for relational learning. Chapter 8 studies the anomaly detection problem on attributed networks and develops a general detection framework with residual analysis. Chapter 9 investigates the streaming link prediction problem on fast-evolving attributed networks and the proposed algorithm supports the real-time prediction of missing links on the fly. Finally, Chapter 10 concludes the dissertation and visions the paths for future work.

Chapter 2

LITERATURE REVIEW

In this chapter, we review the related work from the following perspectives: (1) feature selection; (2) network embedding; (3) dynamic network analytics; and (4) attributed network analytics.

2.1 Feature Selection

Feature selection is imperative in alleviating the curse of dimensionality (Cover and Thomas, 2012; Guyon and Elisseeff, 2003; Li *et al.*, 2017a; Li and Liu, 2017) by finding a subset of features of high quality and is essentially useful when the original features are indispensable for model interpretation and knowledge distillation. Concerning different selection strategies, they can be broadly grouped into three categories: filter methods, wrapper methods, and embedded methods. Filter methods are independent of any learning algorithms and are thereby very efficient, they rely on some data characteristics such as distance, consistency, dependency, and correlation to measure the strength of each feature individually (Gu *et al.*, 2012; He *et al.*, 2005; Peng *et al.*, 2005; Robnik-Šikonja and Kononenko, 2003). Wrapper methods use the prediction power of a predefined learning algorithm to evaluate the quality of the selected features. They are inevitably computational expensive since the search space grows exponentially with the number of features (Dy and Brodley, 2000; Guyon and Elisseeff, 2003; Liu and Yu, 2005). Embedded methods is a tradeoff between these two models which combines feature selection and model construction (Cai *et al.*, 2010b; Guyon *et al.*, 2002; Nie *et al.*, 2010). Therefore, they are usually comparably efficient to filters and are comparably accurate to wrappers. Depending on whether label in-

formation is involved, existing methods can be broadly classified as supervised (Tang *et al.*, 2014) and unsupervised algorithms (Alelyani *et al.*, 2013). Supervised feature selection directly makes use of the discriminative information embedded in the class labels to find features to differentiate instances from different classes. Since label information is time-consuming and expensive to obtain, there is a surge of interests on unsupervised feature selection. Without label information in guiding the selection phase, existing efforts seek for alternative criteria to assess the relevance of features, including data similarity (He *et al.*, 2006; Zhao and Liu, 2007; Zhao *et al.*, 2013), local and global discriminative information (Cai *et al.*, 2010b; Du and Shen, 2015; Li *et al.*, 2012, 2018b; Qian and Zhai, 2013; Yang *et al.*, 2011), and data reconstruction error (Farahat *et al.*, 2013; Li *et al.*, 2017d; Zhao *et al.*, 2016b). Traditional feature selection algorithms cannot be directly applied on networked data as the *i.i.d.* assumption does not hold. (Gu and Han, 2011) first studied supervised feature selection on networked data, in particular, a graph regularized sparse learning framework is developed to capture the correlation between network structure and node attributes. (Tang and Liu, 2012a) further investigated how to find relevant features from social media data by incorporating various types of social relations, and it was later extended to jointly find relevant instances and features simultaneously (Tang and Liu, 2013) since both instances and features could be noisy. The above-mentioned attempts, however, are limited with the use of label information, which is often tedious to obtain in practice. LUFS (Tang and Liu, 2012b) was among one of the first unsupervised feature selection frameworks on networks. In particular, it leverages the community structure of nodes to facilitate the selection of relevant features, which is performed in two separate steps. (Li *et al.*, 2016b) proposed a robust framework NetFS to embed the community detection into feature selection, and the proposed framework is robust to the noise among the observed links. However, it is argued that both LUFS

and NetFS fail to take advantage of the fine-grained link information for feature selection. Therefore, (Wei *et al.*, 2015, 2016) proposed novel frameworks based on the partial order relations among node pairs to exploit link information directly. These efforts, however, fail to capture the finer-grained tie strength information embedded on the network. Thus, (Li *et al.*, 2019) developed an adaptive framework by characterizing the optimal neighborhood structure around each node for unsupervised feature selection. Additionally, feature selection on signed networks (Cheng *et al.*, 2017), and heterogeneous networks (Wei *et al.*, 2017a) also have been investigated.

2.2 Network Embedding

Learning meaningful and discriminative representations of nodes in a network is essential for various network analytical tasks as it avoids the laborious manual feature engineering process. Additionally, as the node embedding representations are often learned in a task-agnostic fashion, they are generalizable to a number of downstream learning tasks such as node classification (Perozzi *et al.*, 2014b), community detection (Wang *et al.*, 2017b), link prediction (Grover and Leskovec, 2016), and visualization (Tang *et al.*, 2016). On top of that, it also has broad impacts in advancing many real-world applications, ranging from recommendation (Wang *et al.*, 2018), polypharmacy side effects prediction (Zitnik *et al.*, 2018), to name disambiguation (Zhang and Al Hasan, 2017). The basic idea is to represent each node by a low-dimensional vector in which the relativity information among nodes on the original network is maximally transcribed. The story of network embedding can be dated back to the early 2000s, when myriad of graph embedding algorithms (Belkin and Niyogi, 2002; Roweis and Saul, 2000; Tenenbaum *et al.*, 2000) were developed, as a part of the general dimensionality reduction techniques. Graph embedding first builds an affinity graph based on the feature representations of data instances and then embeds the affinity

graph into a low-dimensional feature space. Along this line, we witnessed a surge of factorization based network embedding methods in recent years, with the target to decompose a carefully designed affinity matrix in capturing the first-order (Belkin and Niyogi, 2002; Ahmed *et al.*, 2013), higher-order (Cao *et al.*, 2015; Ou *et al.*, 2016) node proximity, or community structure (Tang and Liu, 2009; Wang *et al.*, 2017b) of the underlying network. Despite their empirical success, the factorization based network embedding methods have at least a quadratic time complexity w.r.t. the number of nodes, prohibiting their practical usage on large-scale networks. The recent advances of network representation learning are largely influenced by the WORD2VEC (Mikolov *et al.*, 2013) model in the NLP community. The seminal work of DEEPWALK (Perozzi *et al.*, 2014b) first makes an analogy between truncated random walks on a network and sentences in a corpus, and then learns the embedding representations of nodes with the same principle as WORD2VEC. Typical embedding methods along this line include LINE (Tang *et al.*, 2015b), NODE2VEC (Grover and Leskovec, 2016), and PTE (Tang *et al.*, 2015a). Recent work found that the embedding methods with negative sampling (e.g., DEEPWALK, LINE, PTE, and NODE2VEC) can be unified into a matrix factorization framework with closed-form solutions (Qiu *et al.*, 2018), which bridges the gap between these two families of network embedding methods. Aforementioned methods mainly adopt a shallow model and the expressibility of the learned embedding representations are rather limited. As a remedy, researchers also resort to deep learning techniques (Cao *et al.*, 2016; Chang *et al.*, 2015; Kipf and Welling, 2016; Veličković *et al.*, 2017; Wang *et al.*, 2016) to learn more complex and nonlinear mapping functions. Many popular deep graph embedding algorithms can be found in the Deep Graph Library[1]. In addition to the raw network structure, real-world networks are often presented with different properties, thus there is

[1]https://www.dgl.ai/

a growing interest to learn the embedding representations of networks from different perspectives, such as attributed network (Huang *et al.*, 2017a,b; Yang *et al.*, 2015), heterogeneous networks (Chen and Sun, 2017; Dong *et al.*, 2017), multi-dimensional networks (Ma *et al.*, 2018; Zhang *et al.*, 2018), signed networks (Wang *et al.*, 2017a; Yuan *et al.*, 2017), and dynamic networks (Li *et al.*, 2017c; Zhou *et al.*, 2018).

2.3 Dynamic Network Analytics

Many real-world networks are not static but are continuously evolving with a different rate (Aggarwal and Subbian, 2014; Spiliopoulou, 2011; Zhang, 2010). Hence, the results of many network mining tasks will become stale and need to be updated to keep freshness. Along this line, various dynamic learning algorithms have been proposed to incrementally adjust the end results from the previous time stamp, with applications in low-rank approximation (Chen and Tong, 2015; Sarwar *et al.*, 2002; Tong *et al.*, 2008), community detection (Chakrabarti *et al.*, 2006; Chi *et al.*, 2009; Kim and Han, 2009; Ning *et al.*, 2007; Tang *et al.*, 2008), classification (Aggarwal and Li, 2011; Guo *et al.*, 2014; Jian *et al.*, 2018), link prediction (Li *et al.*, 2014b, 2018a; Sarkar *et al.*, 2012; Zhao *et al.*, 2016a), and anomaly detection (Aggarwal *et al.*, 2011; Manzoor *et al.*, 2016; Ranshous *et al.*, 2015; Yu *et al.*, 2018). For example, (Tong *et al.*, 2008) proposed an efficient way to sample columns and/or rows from the network adjacency matrix for low-rank approximation. (Ning *et al.*, 2007) proposed an incremental approach to perform spectral clustering on networks dynamically. (Aggarwal and Li, 2011) proposed a random-walk based method to perform dynamic classification in content-based networks. In (Zhu *et al.*, 2016), a temporal latent space model is proposed for temporal link prediction on dynamic networks. (Gupta *et al.*, 2012) proposed to detect evolutionary outliers by leveraging both time and community information. In certain scenarios, the edge stream representing the in-

teractions among nodes are continuously arriving at an unprecedented rate, posing additional challenges to dynamic network analytics as the size of edge stream could be massive and cannot be easily stored, which further exacerbates the consequent learning tasks. Hence, most of the existing efforts are dedicated to designing effective data structures to summarize the observed network structure in real-time. For example, (Aggarwal *et al.*, 2010) first proposed to cluster a small graph (or a collection or edges) in a streaming fashion by using the hash-based compression techniques. The similar sketching mechanism is extended to the scenario when node attributes are attached to the continuously generated nodes (Zhao and Yu, 2013). In (Aggarwal *et al.*, 2011), a reservoir sampling method is presented to maintain the structural summary of the underlying network stream for clustering and outlier detection. In terms of classification, (Aggarwal, 2011) employed a min-hash based approach to model the dependencies between the sketched subgraphs and the class labels. Another prevalent way to model dynamic networks is through tensors, and a more detailed review can be found in (Papalexakis *et al.*, 2017). Contrast to the maintenance models which attempt to replenish the staleness of the model, the other line of work tries to quantify and understand the evolution mechanisms of the dynamic networks. For example, (Leskovec *et al.*, 2005) found that dynamic networks gradually get densified over time, and the diameter of the network also shrinks. In (Leskovec *et al.*, 2008), the authors scrutinized the dynamic networks from a microscopic perspective and developed a network model to simulate the generation process of dynamic networks. In (Nigam *et al.*, 2018), the authors studied the co-evolution patterns of opinions and network connections. A more detailed review of dynamic network analytics is in (Aggarwal and Subbian, 2014).

2.4 Attributed Network Analytics

As attributed networks are becoming widely used to model and characterize a variety of real-world information systems, we have witnessed an increasing amount of research efforts in mining attributed networks in recent years. Existing research progress on attributed networks can be summarized into two parts: (1) analytical studies and modeling of attributed networks; and (2) predictive modeling of attributed networks. The first line of work focused on understanding the interplay between the node attributes and the network structure, as well as the underlying generation mechanisms of attributed networks. One of the most widely observed patterns among linked individuals on an attributed network is the assortativity, which implies that similar nodes are connected to one another more often than dissimilar nodes. In (Newman, 2003, 2018b), the authors found that the social factors including influence and homophily induce the assortativity patterns, and tried to understand the assortativity effect by quantifying the correlation of node attributes and the network structure. (Rabbany *et al.*, 2017) generalized to quantify the structural correlations of a single attribute or a pair of attributes. Other studies tried to understand the dynamic patterns of attributed networks (Crandall *et al.*, 2008) and differentiate social influence and homophily effects with randomization tests (La Fond and Neville, 2010). Later on, a network sampling schema (Robles *et al.*, 2016) and a generative network model (Pfeiffer III *et al.*, 2014) are developed for a better characterization of attributed networks. Additionally, (Eswaran *et al.*, 2018) discovered some interesting patterns about the structural properties of attribute-induced subgraph and proposed to generate synthetic graphs by matching the observed patterns. The second line of work made use of the interplay between node attributes and network structure to perform various predictive tasks on attributed networks, including subgraph matching (Du *et al.*, 2017;

Fang *et al.*, 2016; Sakr *et al.*, 2012; Tong *et al.*, 2007), node classification (Hamilton *et al.*, 2017; Kipf and Welling, 2016; Li *et al.*, 2017e; Singh and Gordon, 2008; Velickovic *et al.*, 2017; Zhu *et al.*, 2007), network clustering (Akoglu *et al.*, 2012; Bojchevski and Günnemann, 2018; Gunnemann *et al.*, 2013; Ruan *et al.*, 2013; Yang *et al.*, 2009, 2013), link prediction (Barbieri *et al.*, 2014; Gao *et al.*, 2011b; Gong *et al.*, 2014; Li *et al.*, 2018a; Menon and Elkan, 2011; Wei *et al.*, 2017b; Yin *et al.*, 2010), anomaly detection (Gao *et al.*, 2010; Li *et al.*, 2017b; Peng *et al.*, 2018; Perozzi *et al.*, 2014a; Perozzi and Akoglu, 2016; Sánchez *et al.*, 2013), pattern mining (Gunnemann *et al.*, 2010; Lee *et al.*, 2016; Silva *et al.*, 2012), ranking (Gao *et al.*, 2011a; Hsu *et al.*, 2017), and network alignment (Heimann *et al.*, 2018; Zhang and Tong, 2016; Zhou and De la Torre, 2012). In this dissertation, we propose novel learning algorithms for attributed networks from a new perspective with feature learning. In addition, we systematically investigate online algorithms for attributed networks in a dynamic environment to replenish the end results, which is rather underexplored in the existing literature. It should also be noted that in addition to the attributed networks, another widely used way to fuse different information modalities is through tensors and it is beyond the focus of this dissertation (Araujo *et al.*, 2017; Papalexakis *et al.*, 2017).

Part I

Learning Algorithms in A Static Environment

Chapter 3

FEATURE SELECTION ON ATTRIBUTED NETWORKS

To facilitate the computational understanding of attributed networks, we first propose to investigate attributed networks at the attribute level by developing novel feature selection algorithms to find a high-quality feature subset that is tightly correlated with the network structure. After the feature selection phase, the relevant feature subset can be taken as input for off-the-shelf machine learning and data mining algorithms.

3.1 Overview

On attributed networks, as the node features are often in a high-dimensional feature space, the illusion that all features are dovetailed with the network topological structure is not always true. As in the case of academic collaboration networks, features like gender are rather more independent of network structure than discerning features like research interests. On top of that, deviating or noisy features that are not consistent with the network topology may jeopardize the discovery of actionable and explainable patterns upon it. These observations necessitate the usage of feature selection algorithms (Guyon and Elisseeff, 2003; Li *et al.*, 2017a) to find a set of relevant features closely correlated with the network structure. As it is easy to amass substantial amounts of unlabeled data while label information is costly to obtain, we focus on unsupervised feature selection for attributed networks. Existing unsupervised feature selection algorithms cannot be directly applied or are not suitable for attributed networks because of their distinct characteristics: (1) on attributed networks, data instances are not independent and identically distributed (*i.i.d.*) but inherently interconnected with each other; (2) in addition to noisy features in the

content space, the observed networks are typically very noisy on grounds of imperfect measurements (Newman, 2018a). To this end, we first propose a general unsupervised framework NetFS (Li *et al.*, 2016b) that is robust to the noisy network structure. In fact, as NetFS models the network structure at a macro-level by community analysis, thus it fails to exploit the finer-grained tie strength information and may lead to suboptimal results. Hence, we also develop an adaptive unsupervised feature selection framework ADAPT (Li *et al.*, 2019) by characterizing and leveraging the finer-grained tie strength information embedded on the network.

Before illustrating the details of the proposed frameworks, we first define the problem of unsupervised feature selection for attributed networks as follows.

Problem 1. Unsupervised Feature Selection for Attributed Networks

Given: *An attributed network G represented by the content matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ (where the feature space is $\mathcal{F}$) and the adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}_{\geq 0}$, where n and d denote the number of nodes and features, respectively*[1]*. The number of selected features is specified as m ($m << d$).*

Select: *A subset of most relevant features $\mathcal{S} \subset \mathcal{F}$ ($|\mathcal{S}| = m$) that is tightly correlated with the network structure*[2]*.*

3.2 Proposed Robust Framework – NetFS

In this section, we introduce the proposed robust unsupervised feature selection framework NetFS in detail. An illustration of NetFS is illustrated in Figure 3.1. Firstly, to capture the inherent interactions among networked instances, we introduce

[1]For undirected network, the adjacency matrix is symmetric such that $\mathbf{A} = \mathbf{A}'$. To model the network information on directed networks, we specify $\mathbf{A} = \max(\mathbf{A}, \mathbf{A}')$, where $\max(*, *)$ is an element-wise maximum operation.

[2]The exact optimization function is given in the following sections.

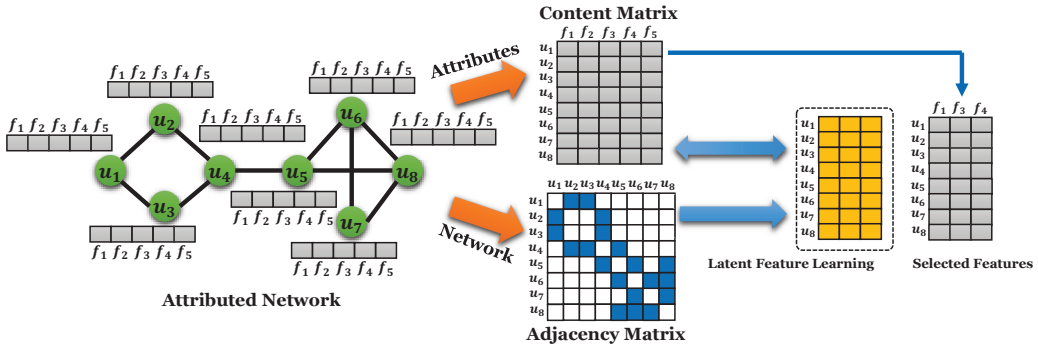


Figure 3.1: An Illustration of the Robust Feature Selection Framework NetFS.

the concept of latent representations to uncover some hidden attributes encoded in the network structure. Secondly, to alleviate the adverse effects of noisy links, we propose to embed the latent representation learning into the feature selection phase. In this way, these two phases influence and help each other iteratively.

3.2.1 Problem Formulation

We first discuss how to model the latent representations from the network structure, and then introduce how to embed the latent representation learning into the content information for feature selection.

Modeling Link Information with Latent Representation. On attributed networks, data instances connect to each other due to a variety of factors. For example, in social networks, these factors can be movie fans, sports enthusiasts, colleagues, family members, etc; in coauthor networks, factors include similar research interests, same affiliations, etc. These hidden factors are often referred as latent representations since they can describe a set of diverse affiliation factors hidden in a network. Latent representations of different instances interact with each other to form link information, and the instances with similar latent representations are more likely to be connected with each other than the instances with dissimilar latent representations.

Uncovering latent representations has received increasing attention recently in data mining and machine learning communities (Airoldi *et al.*, 2008; Newman and Girvan, 2004; Tang and Liu, 2009). Here, we model the latent representations from link information by symmetric nonnegative matrix factorization (SymNMF) (He *et al.*, 2011; Kuang *et al.*, 2012). The principle of SymNMF is consistent with network clustering such that each networked instance consists of a mixture of latent attributes. Mathematically, it decomposes the adjacency matrix $\mathbf{A}$ into a product of a nonnegative matrix $\mathbf{U}$ and its transpose $\mathbf{U}'$ in a low-dimensional latent space:

$$\min_{\mathbf{U}\geq 0} \|\mathbf{A} - \mathbf{U}\mathbf{U}'\|_F^2, \tag{3.1}$$

where $\mathbf{U} \in \mathbb{R}_{\geq 0}^{n\times c}$ is the latent representations of all n instances, and c is the number of latent factors that we need to specify.

Embedding Latent Representation Learning into Feature Selection. Considering the fact that the link information on attributed networks could be noisy and incomplete, latent representations that are directly derived from link information may jeopardize feature selection on the content space. In addition, according to the homophily effect (McPherson *et al.*, 2001) and social influence (Marsden and Friedkin, 1993) in social science, content information will affect and is dependent on the latent representations from the network structure. Therefore, it is desirable to embed latent representation learning into the feature selection phase on the content space. As a result, latent representation learning and feature selection could help and boost each other. Content information can help learn better latent representations which are robust to noisy links, and better latent representations can fill the gap of scarce label information and rich link information to guide feature selection.

As latent factors encode some hidden attributes of instances, they should be related to some attributes of networked instances. Therefore, we take $\mathbf{U}$ as a constraint

to model the content information through a multivariate linear regression model:

$$\min_{\mathbf{W}} \|\mathbf{XW} - \mathbf{U}\|_F^2, \tag{3.2}$$

where $\mathbf{W} \in \mathbb{R}^{d \times c}$ is a transformation matrix. Each row vector $\mathbf{W}(i,:)$ measures the importance of the i-th feature. To achieve feature selection, we add an $\ell_{2,1}$-norm regularization term on $\mathbf{W}$ for a joint sparsity among all c latent factors:

$$\min_{\mathbf{W}} \|\mathbf{XW} - \mathbf{U}\|_F^2 + \alpha\|\mathbf{W}\|_{2,1}, \tag{3.3}$$

where parameter α controls the sparsity of the model.

By combining the objective functions in Eq. (3.1) and Eq. (3.3), the final objective function that embeds latent representation learning into feature selection phase can be formulated as follows:

$$\min_{\mathbf{U}\geq 0,\mathbf{W}} \mathcal{J}(\mathbf{W},\mathbf{U}) = \|\mathbf{XW} - \mathbf{U}\|_F^2 + \alpha\|\mathbf{W}\|_{2,1} + \frac{\beta}{2}\|\mathbf{A} - \mathbf{UU}'\|_F^2, \tag{3.4}$$

where β is a parameter to balance the latent representation modeling and the feature selection in the content space. It can be observed from Eq. (3.4) that when $\mathbf{W}$ is fixed, the latent representation learning phase is not only associated with the adjacency matrix $\mathbf{A}$, but also the content matrix $\mathbf{X}$. In this way, the learned latent representations can capture their inherent correlations and are more robust to noisy links. When the latent representations $\mathbf{U}$ is fixed, they will take the role of label information to steer feature selection in a supervised way.

3.2.2 Optimization Solution

Now, we talk about how to solve the optimization problem of NetFS. The objective function in Eq. (3.4) is not convex w.r.t. $\mathbf{U}$ and $\mathbf{W}$ simultaneously. Besides, due to the $\ell_{2,1}$-norm regularization term, it is also not smooth. We adopt an alternating optimization scheme to solve this problem.

When $\mathbf{U}$ is fixed, the objective function is convex w.r.t. $\mathbf{W}$. Therefore, we take the derivative of $\mathcal{J}(\mathbf{W}, \mathbf{U})$ with respect to $\mathbf{W}$ and set it to be zero, then we have:

$$\mathbf{X}'(\mathbf{X}\mathbf{W} - \mathbf{U}) + \alpha\mathbf{D}\mathbf{W} = 0, \tag{3.5}$$

where $\mathbf{D} \in \mathbb{R}^{d\times d}$ is a diagonal matrix such that $\mathbf{D}(i,i) = \frac{1}{2\|\mathbf{W}(i,:)\|_2}$. It should be noted that in practice, $\|\mathbf{W}(i,:)\|_2$ could be very close to zero. Therefore, we define $\mathbf{D}(i,i) = \frac{1}{2\|\mathbf{W}(i,:)\|_2+\epsilon}$, where ϵ is a very small constant. Since $\mathbf{X}'\mathbf{X}$ is a positive semidefinite matrix, $\alpha\mathbf{D}$ is a diagonal matrix with positive entries, thus their summation $\mathbf{X}'\mathbf{X}+\alpha\mathbf{D}$ is positive definite. Therefore, $\mathbf{W}$ has a closed-form solution, which is:

$$\mathbf{W} = (\mathbf{X}'\mathbf{X} + \alpha\mathbf{D})^{-1}\mathbf{X}'\mathbf{U}. \tag{3.6}$$

By substituting the above solution of $\mathbf{W}$ into Eq. (3.4), we get:

$$\begin{aligned}\min_{\mathbf{U}\geq 0} \mathcal{J}(\mathbf{U}) &= tr(\mathbf{U}'\mathbf{U}) - tr(\mathbf{W}'\mathbf{M}\mathbf{W}) + \frac{\beta}{2}\|\mathbf{A} - \mathbf{U}\mathbf{U}'\|_F^2 \\ &= tr(\mathbf{U}'(\mathbf{I}_n - \mathbf{X}\mathbf{M}^{-1}\mathbf{X}')\mathbf{U}) + \frac{\beta}{2}\|\mathbf{A} - \mathbf{U}\mathbf{U}'\|_F^2,\end{aligned} \tag{3.7}$$

where $\mathbf{M} = \mathbf{X}'\mathbf{X} + \alpha\mathbf{D}$. The problem in Eq. (3.7) is a standard bound-constrained optimization problem, we propose to use projected gradient descent (Lin, 2007) to solve it. Now the objective function can be reformulated as:

$$\min_{\mathbf{U}\geq 0} \mathcal{J}(\mathbf{U}) = tr(\mathbf{U}'(\mathbf{I}_n - \mathbf{X}\mathbf{M}^{-1}\mathbf{X}')\mathbf{U}) + \frac{\beta}{2}\|\mathbf{A} - \mathbf{U}\mathbf{U}'\|_F^2. \tag{3.8}$$

Let $\mathbf{U}_t$ be the update of $\mathbf{U}$ at the t-th iteration. It is updated by the following rule:

$$\mathbf{U}_{t+1} = P[\mathbf{U}_t - s_t\nabla\mathcal{J}(\mathbf{U}_t)], \tag{3.9}$$

where $P[\mathbf{U}_t - s_t\nabla\mathcal{J}(\mathbf{U}_t)]$ is a box projection operator which maps a point to a bounded nonnegative region. s_t is the step size at the t-th iteration and can be determined by the Armijo rule (Bertsekas, 1999). To be more specific, $s_t = \theta^{a_t}$, where a_t is the first nonnegative integer such that the condition $\mathcal{J}(\mathbf{U}_{t+1}) - \mathcal{J}(\mathbf{U}_t) \leq \sigma\langle\nabla\mathcal{J}(\mathbf{U}_t), (\mathbf{U}_{t+1} - \mathbf{U}_t)\rangle$ is satisfied, where θ and σ are two predefined parameters between 0 and 1, $\langle\mathbf{A}, \mathbf{B}\rangle$ represents the inner product operation between two matrix $\mathbf{A}$ and $\mathbf{B}$.

Algorithm 1 Proposed Feature Selection Framework NetFS.

Input: Attributed network G, the number desired features m, parameters α and β.

Output: The top-m ranked features

Initialize $\mathbf{D}_k$ as an identity matrix, and set $k = 0$;
while objective function value in Eq. (3.4) not converge **do**
    Compute $\mathbf{M}_k = \mathbf{X}'\mathbf{X} + \alpha\mathbf{D}_k$;
    Obtain $\mathbf{U}_{k+1}$ by projected gradient descent in Eq. (3.9);
    Obtain $\mathbf{W}_{k+1}$ by Eq. (3.6);
    Update $\mathbf{D}_{k+1}$ with $\mathbf{W}_{k+1}$;
    $k = k + 1$;
end while
Rank features according to $\|\mathbf{W}(i,:)\|_2$.

3.2.3 Time Complexity Analysis

The pseudocode of the proposed NetFS framework is shown in Algorithm 1. In each iteration, when $\mathbf{W}$ is fixed, we use projected gradient descent method to update $\mathbf{U}$, the computational cost to obtain the gradient in Eq. (3.9) is $O(n^2d) + O(nd^2) + O(n^2c)$. Then we fix $\mathbf{U}$ to update $\mathbf{W}$, we can first precompute $\mathbf{X}'\mathbf{X}$ once, which requires the $O(nd^2)$ in the worst case; later on, the computation in Eq. (3.6) requires $O(d^3) + O(d^2c) + O(ndc)$. In practice, the computational of the matrix inversion operation can be further accelerated by solving a linear equation.

3.2.4 Convergence Analysis

In summary, the objective function in Eq. (3.4) is solved through an alternating way. When $\mathbf{W}$ is fixed, we use projected gradient descent through Eq. (3.9) to update $\mathbf{U}$; then we fix $\mathbf{U}$ and employ Eq. (3.6) to update $\mathbf{W}$, and the diagonal matrix $\mathbf{D}$ is

updated as well. Later on, we sort the features in a descending order according to the matrix $\mathbf{W}$ and return top-m ones. Specifically, the larger the value $\|\mathbf{W}(i,:)\|_2$ is, the more important the i-th feature is. Next, we show the objective function of NetFS in Eq. (3.4) is guaranteed to converge with the proposed optimization algorithm.

Lemma 1. *The following inequality holds if* $\mathbf{W}_k(i,:)$ *and* $\mathbf{W}_{k+1}(i,:)$ *are non-zero vectors (i=1,2,...,d) (Nie* et al.*, 2010):*

$$\|\mathbf{W}_{k+1}\|_{2,1} - \sum_i \frac{\|\mathbf{W}_{k+1}(i,:)\|_2^2}{2\|\mathbf{W}_k(i,:)\|_2} \leq \|\mathbf{W}_k\|_{2,1} - \sum_i \frac{\|\mathbf{W}_k(i,:)\|_2^2}{2\|\mathbf{W}_k(i,:)\|_2}. \tag{3.10}$$

Theorem 1. *The alternating optimization procedure will decrease the objective function value of Eq. (3.4).*

Proof. During the $(k+1)$-th iteration, when $\mathbf{W}_k$ is fixed, we update $\mathbf{U}$ with projected gradient descent, which decreases the objective function $\mathcal{J}(\mathbf{U})$ for appropriate choices of step size. Therefore, we have:

$$\mathcal{J}(\mathbf{U}_{k+1}, \mathbf{W}_k) \leq \mathcal{J}(\mathbf{U}_k, \mathbf{W}_k). \tag{3.11}$$

Then when $\mathbf{U}_{k+1}$ is fixed, we obtain the optimal solution $\mathbf{W}_{k+1}$ through Eq. (3.6) and $\mathbf{W}_{k+1}$ is the solution of the following objective function:

$$\min_{\mathbf{W}} \|\mathbf{X}\mathbf{W} - \mathbf{U}_{k+1}\|_F^2 + \alpha tr(\mathbf{W}'\mathbf{D}_k\mathbf{W}). \tag{3.12}$$

Therefore, we have the following inequality:

$$\begin{aligned}
&\|\mathbf{X}\mathbf{W}_{k+1} - \mathbf{U}_{k+1}\|_F^2 + \alpha tr(\mathbf{W}'_{k+1}\mathbf{D}_k\mathbf{W}_{k+1}) \leq \|\mathbf{X}\mathbf{W}_k - \mathbf{U}_{k+1}\|_F^2 + \alpha tr(\mathbf{W}'_k\mathbf{D}_k\mathbf{W}_k) \\
\Rightarrow\quad &\|\mathbf{X}\mathbf{W}_{k+1} - \mathbf{U}_{k+1}\|_F^2 + \alpha\|\mathbf{W}_{k+1}\|_{2,1} - \alpha(\|\mathbf{W}_{k+1}\|_{2,1} - \sum_i \frac{\|\mathbf{W}_{k+1}(i,:)\|_2^2}{2\|\mathbf{W}_k(i,:)\|_2}) \\
\leq\quad &\|\mathbf{X}\mathbf{W}_k - \mathbf{U}_{k+1}\|_F^2 + \alpha\|\mathbf{W}_k\|_{2,1} - \alpha(\|\mathbf{W}_k\|_{2,1} - \sum_i \frac{\|\mathbf{W}_k(i,:)\|_2^2}{2\|\mathbf{W}_k(i,:)\|_2}).
\end{aligned} \tag{3.13}$$

Integrating the results in Lemma 1, we have the following:

$$\begin{aligned} &\|\mathbf{X}\mathbf{W}_{k+1} - \mathbf{U}_{k+1}\|_F^2 + \alpha\|\mathbf{W}_{k+1}\|_{2,1} \leq \|\mathbf{X}\mathbf{W}_k - \mathbf{U}_{k+1}\|_F^2 + \alpha\|\mathbf{W}_k\|_{2,1} \\ \Rightarrow \quad &\mathcal{J}(\mathbf{U}_{k+1}, \mathbf{W}_{k+1}) \leq \mathcal{J}(\mathbf{U}_{k+1}, \mathbf{W}_k) \leq \mathcal{J}(\mathbf{U}_k, \mathbf{W}_k), \end{aligned} \tag{3.14}$$

which completes the proof. ■

3.3 Experimental Evaluation of NetFS

We conduct experiments to assess the performance of NetFS in performing unsupervised feature selection on attributed networks. We first introduce the datasets and experimental settings before presenting details of the experiments.

Datasets. Three real-world attributed networks, BlogCatalog, Flickr, and Epinions are used for evaluation. BlogCatalog is a social blog directory in which users can post their blogs under different predefined categories. The tags of blogs from users form the feature information, while the major categories of blogs by users are considered as ground truth. Flickr is an image sharing website, users can provide tags for the photos they upload which provide feature information. Besides, users interact with others forming link information. Photos are organized under some predefined categories, which are used as ground truth. To facilitate the comparison with the online version of NetFS that is designed for the dynamic attributed networks (will be introduced later in Chapter 5), we randomly disturb 0.1% edges and 0.1% feature values over 20 time stamps for static attributed networks BlogCatalog and Flickr. Epinions is a product review website in which users can share their reviews about products. Users themselves can also build trust relations to seek advice from others. Features are formed by the bag-of-words model, while the major categories of reviews by users are taken as ground truth of class labels. Since the time information when users build trust relationships is available, we crawled and collected the site at 17 different time stamps to form a dynamic attributed network. Then for all these three datasets,

Table 3.1: Detailed Information of the Datasets Used for NetFS and TeFS.

	BlogCatalog	Flickr	Epinions
# of nodes	5,196	7,575	5,665
# of features	8,189	12,047	10,382
# of links	171,743	239,738	97,123
# of classes	6	9	24
# of time stamps	20	20	17

we rerun NetFS and other baselines at each time stamp and report the average evaluation performance across different time stamps. Some statistics of these datasets are listed in Table 3.1.

Experimental Settings. Following the standard way to assess unsupervised feature selection (Cai *et al.*, 2010b; Li *et al.*, 2012; Yang *et al.*, 2011), we compare different methods in terms of the clustering performance[3]. Two commonly used clustering performance metrics, i.e., *normalized mutual information* (NMI) and *clustering accuracy* (ACC) are used.

Let C and C' denote the clustering results from ground truth class labels and the predicted cluster labels, respectively. The mutual information between two clusters C and C' is:

$$MI(C, C') = \sum_{c_i \in C, c'_j \in C'} p(c_i, c'_j) log \frac{p(c_i, c'_j)}{p(c_i)p(c'_j)}, \tag{3.15}$$

where $p(c_i)$ and $p(c'_j)$ are the probabilities of instances in cluster c_i and c'_j, respectively. $p(c_i, c'_j)$ indicates the probability of instances in cluster c_i and in c'_j at the same time. Then, NMI is defined as:

$$NMI(C, C') = \frac{MI(C, C')}{\max(H(C), H(C')}, \tag{3.16}$$

[3]As mentioned above, the average clustering performance across different time stamps is reported.

where $H(C)$ and $H(C')$ represent the entropies of clusterings C and C', respectively.

Let p_i and q_i be the clustering result and the ground truth label for instance u_i, respectively. Then, clustering accuracy (ACC) is defined as:

$$ACC = \frac{1}{n}\sum_{i=1}^{n}\delta(q_i, map(p_i)), \tag{3.17}$$

where n is the total number of instances, $\delta(.)$ is an indicator function such that $\delta(x, y) = 1$ if $x = y$, otherwise $\delta(x, y) = 0$. $map(x)$ permutes the predicted cluster labels to match the ground truth as much as possible.

The proposed NetFS is measured against the following state-of-the-art unsupervised feature selection algorithms:

- LC: Laplacian score (He *et al.*, 2005) evaluates feature importance via its ability of locality preservation.
- SPEC: It is an extension of Laplacian Score where features are selected using spectral analysis (Zhao and Liu, 2007).
- NDFS: Features are selected via joint nonnegative spectral analysis and $\ell_{2,1}$-norm regularization (Li *et al.*, 2012).
- LUFS: Social dimensions are first extracted from link information, then they are utilized to guide feature selection in the content space (Tang and Liu, 2012b).

In LS and NDFS, as suggested by the original papers (He *et al.*, 2005; Li *et al.*, 2012), we set the number of neighborhood size as 5 to construct the affinity matrix. For NDFS, LUFS, and NetFS, the number of clusters or pseudo labels is specified to be the number of classes. NDFS and LUFS have different regularization parameters, we set these parameters by the suggestions from the original papers (Li *et al.*, 2012; Tang and Liu, 2012b). In the proposed NetFS, we also have two regularization parameters α and β. In the experiments, we empirically set α as 10 and β as 0.1.

Table 3.2: Clustering Results Evaluation of NetFS and Baselines on BlogCatalog.

metric	ACC (%)									
# of features	200	400	600	800	1000	1200	1400	1600	1800	2000
LS	26.83	28.41	27.34	24.35	31.87	27.62	28.05	29.47	30.81	31.98
SPEC	18.34	18.01	18.65	19.32	21.01	22.87	21.65	22.32	24.50	24.37
NDFS	24.12	30.82	32.56	31.78	34.35	32.95	33.85	**44.67**	41.67	43.22
LUFS	21.30	21.89	31.65	32.01	32.36	33.45	34.12	42.40	41.34	**43.83**
NetFS	**49.99**	**43.04**	**43.25**	**42.89**	**42.09**	**43.56**	**43.44**	43.31	**43.08**	43.59
metric	NMI (%)									
# of features	200	400	600	800	1000	1200	1400	1600	1800	2000
LS	8.21	7.54	5.86	4.54	7.93	5.03	5.55	5.64	6.06	6.83
SPEC	0.21	0.19	0.54	0.68	0.81	1.90	3.11	2.15	5.43	3.36
NDFS	10.19	16.50	18.67	13.68	17.05	14.36	14.98	22.77	23.20	**25.51**
LUFS	4.21	4.05	14.19	14.98	16.11	16.98	18.93	**27.61**	**26.79**	24.48
NetFS	**33.21**	**23.45**	**24.04**	**23.18**	**23.83**	**23.63**	**25.49**	25.98	23.85	24.72

Each feature selection algorithm is first applied to select features, then K-means clustering is performed based on the selected features. Since K-means may converge to local minima, we repeat the process 20 times and report the average. Normally, the higher the ACC and NMI values are, the better the selected features are.

Quality of Selected Features by NetFS. We compare the quality of selected features by NetFS and other baseline methods on the three aforementioned datasets. The number of selected features are varied among $\{200, 400, ..., 2000\}$. The comparison results are shown in Table 3.2, Table 3.3, and Table 3.4.

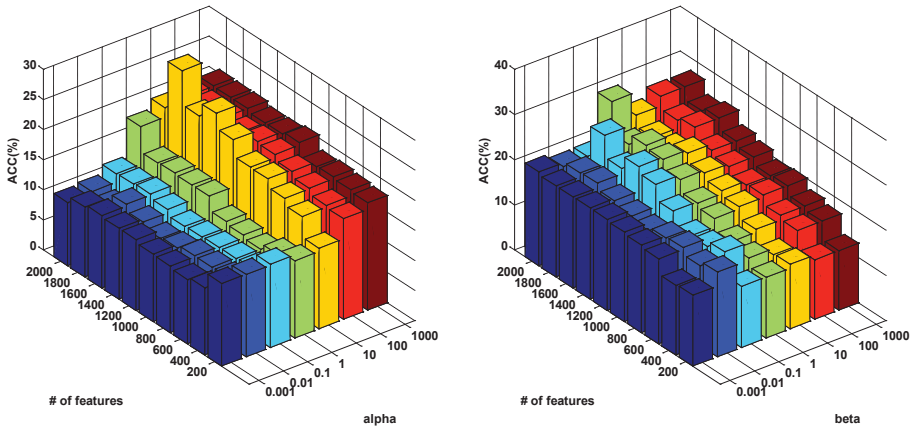
We make the following observations: (1) NetFS outperforms traditional unsupervised feature selection algorithms in almost all cases by obtaining better clustering performance. We also perform pairwise Wilcoxon signed-rank test (Demšar, 2006) between NetFS and these baseline methods, the test results show NetFS is significantly better, with a 0.05 significance level. A major reason is that traditional

Table 3.3: Clustering Results Evaluation of NetFS and Baselines on Flickr.

metric	ACC (%)									
# of features	200	400	600	800	1000	1200	1400	1600	1800	2000
LS	12.26	12.71	12.38	13.07	13.26	13.09	14.25	15.88	17.23	16.60
SPEC	12.14	12.42	13.18	13.53	14.61	14.36	14.51	14.69	14.92	14.36
NDFS	15.41	17.25	26.27	28.56	**35.54**	33.24	37.90	38.65	41.57	**44.28**
LUFS	11.89	19.24	20.08	22.56	23.24	28.58	28.52	31.44	34.79	39.72
NetFS	**23.04**	**31.52**	**33.60**	**36.21**	35.52	**42.56**	**46.46**	**41.35**	**47.42**	35.78
metric	NMI (%)									
# of features	200	400	600	800	1000	1200	1400	1600	1800	2000
LS	0.67	0.72	0.64	1.55	1.81	2.49	2.50	4.31	4.67	3.38
SPEC	0.18	0.54	0.74	1.55	1.41	1.38	1.68	1.64	1.72	1.81
NDFS	3.49	5.82	9.00	10.72	17.44	16.21	18.92	22.77	24.06	**30.68**
LUFS	1.52	7.98	9.56	13.21	12.88	14.62	14.41	15.88	20.04	25.20
NetFS	**11.61**	**16.55**	**20.28**	**20.56**	**21.67**	**23.38**	**26.54**	**25.40**	**28.91**	25.42

Table 3.4: Clustering Results Evaluation of NetFS and Baselines on Epinions.

metric	ACC (%)									
# of features	200	400	600	800	1000	1200	1400	1600	1800	2000
LS	**15.29**	13.45	13.08	12.81	11.47	11.62	12.26	12.95	12.86	11.06
SPEC	14.20	12.87	12.76	11.08	10.26	10.08	10.55	11.26	11.50	10.64
NDFS	12.81	11.82	12.41	12.40	12.63	14.52	14.02	15.46	14.75	14.89
LUFS	13.21	11.28	11.42	11.59	12.56	14.37	13.54	14.32	13.61	16.88
NetFS	14.04	**16.66**	**18.27**	**20.48**	**20.46**	**20.98**	**24.82**	**23.79**	**23.91**	**27.32**
metric	NMI (%)									
# of features	200	400	600	800	1000	1200	1400	1600	1800	2000
LS	1.42	2.05	2.28	2.72	2.08	2.13	2.22	2.31	2.28	2.13
SPEC	1.80	2.07	2.38	2.53	2.58	2.87	2.82	2.39	2.61	2.64
NDFS	2.29	2.50	2.92	3.23	3.83	4.00	4.24	4.49	4.98	5.67
LUFS	2.32	2.52	2.40	2.71	3.33	3.87	3.92	4.42	4.39	5.13
NetFS	**3.87**	**5.80**	**6.91**	**6.93**	**8.92**	**10.04**	**10.83**	**11.45**	**11.65**	**10.26**



(a) Effect of α. (b) Effect of β.

Figure 3.2: Parameter Study of NetFS on Epinions Dataset.

algorithms can only handle *i.i.d.* data while NetFS exploits both content information and network structure to obtain good features. (2) NetFS and LUFS deal with network and content information differently. LUFS performs network structure modeling and feature selection separately and the feature selection performance is highly dependent on the quality of extracted latent representations, thus it is very sensitive to the noise among the links. In contrast, NetFS embeds the latent representation learning phase into the feature selection. Therefore, content information is used adaptively to obtain better latent factor representations because better latent factors can contribute to selecting more relevant features. (3) On BlogCatalog, NetFS works well with only a few hundred of features. Flickr and Epinions datasets have more features than BlogCatalog, but NetFS still achieves good clustering performance with only around 1/8 and 1/7 of total features, respectively.

Parameter Sensitivity Study. NetFS has two important parameters - α controls the sparsity of the model while β balances the latent representation learning and feature selection phases. We fix one parameter each time and vary the other one to see how it affects the feature selection performance. As the settings mentioned

above, we assess the feature selection performance in terms of clustering with different number of selected features. In Figure 3.2(a), we present the clustering performance of NetFS on Epinions dataset in terms of ACC. We first fix the parameter β to be 0.1 and vary the other parameter α as $\{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$. As shown in Figure 3.2(a), with an increase of α, the clustering performance first increases then becomes stable between 1 and 1000. The reason is that when α is small, the sparsity of the model is low, which is not suitable for feature selection. Then we fix α to be 10 and vary β in $\{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$, the results are presented in Figure 3.2(b), we can observe that the clustering performance is less sensitive to β compared with α, the performance is relatively better when β is around 1. The clustering performance is relatively more sensitive to the number of selected features, which is still an open problem in unsupervised feature selection.

3.4 Proposed Adaptive Framework - ADAPT

In this section, we present the proposed adaptive unsupervised feature selection framework ADAPT. Even though the aforementioned robust framework NetFS (Li *et al.*, 2016b) and several other research efforts (Tang and Liu, 2012b; Wei *et al.*, 2015, 2016) enable the selection of relevant features in an unsupervised manner, these attempts, however, overwhelmingly focus on binary relations (e.g., coauthors or not) among nodes which only yield a coarse indication of the heterogeneity of relations. As indicated by tie strength theory (Gilbert and Karahalios, 2009; Granovetter, 1973; Xiang *et al.*, 2010), the strength of links could vary remarkably over the full spectrum (e.g., from close friends to acquaintances), thus treating all links equally may result in the selection of a suboptimal feature set. For example, in academic collaboration networks, informative features such as research interests should not only be able to make a distinction between coauthors and non-coauthors but also should distinguish

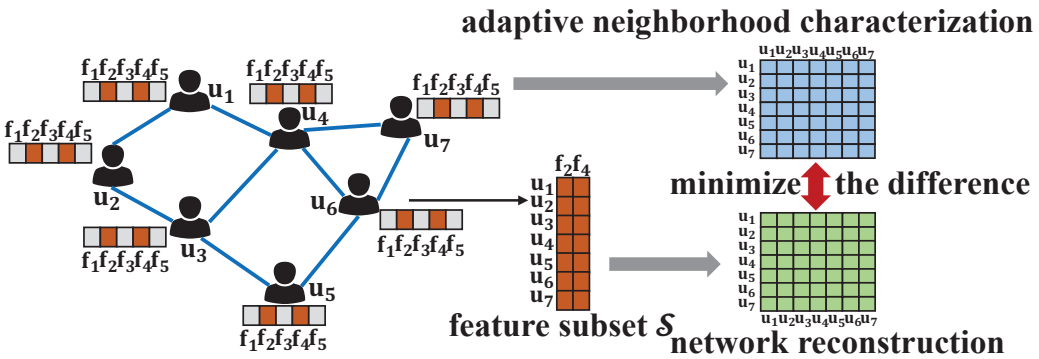


Figure 3.3: An Illustration of the Adaptive Feature Selection Framework ADAPT.

collaborators with strong ties from the ones with weak ties.

ADAPT assumes that the observed features can be employed to obtain the adaptive neighborhood structure around each node. Then to find a subset of the most informative features, ADAPT further assumes that the network structure can be regenerated (i.e., network reconstruction) through these informative features via a probabilistic framework. At last, to capture the inherent correlation between network structure and node attributes, ADAPT imposes a constraint on the network reconstruction process to ensure that it preserves the adaptive neighborhood structure measured by the tie strength. An illustration of these three aforementioned steps of ADAPT is shown in Figure 3.3.

3.4.1 Problem Formulation

Here, we will elaborate on these three aforementioned phases in detail.

Characterizing the Adaptive Neighborhood. First, we embark on the definition of the adaptive neighborhood structure around each node.

Definition 3. Adaptive Neighborhood Structure: *The adaptive neighborhood structure around the node v_i is encoded in the tie strength vector $\mathbf{a}_i = [a_{i1}, ...a_{in}]' \in \mathbb{R}^n$ with the following constraints: (i) $\mathbf{1}'\mathbf{a}_i = 1$; (ii) $a_{ij} \geq 0$, $\forall v_j \in \mathcal{N}(v_i)$; (iii) $a_{ij} = 0$,*

$\forall v_j \notin \mathcal{N}(v_i)$. *We denote the constraint that is imposed on* $\mathbf{a}_i$ *as* Ω_i.

Motivated by the *dyadic hypothesis* (Granovetter, 1973) in sociology, the characterization insinuates that the strength of a tie is largely determined by how similar the characteristics of two end nodes are w.r.t. node attributes (Xiang *et al.*, 2010). However, it faces several unique challenges. Firstly, as label information of nodes is both time and labor intensive to acquire, we are in short of reliable ground truths to measure whether a node pair is indeed similar or not. Second, for each individual on the network, the number of its strongly connected nodes and the number of weakly connected nodes could differ remarkably.

To tackle these challenges, we introduce the concept of pseudo class labels to portray the characteristics of nodes. For each node v_i, we use $\mathbf{y}_i \in \{0,1\}^c$ to denote its pseudo class label vector (c is the number of pseudo classes) and we assume it can be obtained by applying a mapping function $f(.) : \mathbb{R}^d \to \mathbb{R}^c$ on its feature vector $\mathbf{x}_i$:

$$\mathbf{y}_i = f(\mathbf{x}_i) + \epsilon_i \ \ (\forall i = 1, ..., n), \tag{3.18}$$

where $\epsilon_i \in \mathbb{R}$ $(i = 1, ..., n)$ are independent noisy terms.

With the concept of pseudo labels, we can characterize the adaptive neighborhood structure around each node. Specifically, we assume that the pseudo label of each node v_i $(i = 1, ..., n)$ can be estimated via a weighted average of the noisy pseudo labels of its neighbors on the attributed network. Let the estimated pseudo class label vector of node v_i be $\hat{f}(\mathbf{x}_i)$, then it holds that $\hat{f}(\mathbf{x}_i) \approx \sum_{j=1}^{n} a_{ij}\mathbf{y}_j$. More concretely, to obtain the optimal weight vector $\mathbf{a}_i$, we can minimize the Manhattan distance[4] between the estimator and the ground truth pseudo class label (without noise) as:

$$\min \| \sum_{j=1}^{n} a_{ij}\mathbf{y}_j - f(\mathbf{x}_i)\|_1 \ \text{ s.t. } \mathbf{a}_i \in \Omega_i, \ \ (\forall i = 1, ..., n). \tag{3.19}$$

[4]We choose to use the Manhattan distance for the sake of simplicity, but it can be extended to other distance measures.

In particular, the node v_j has a stronger tie with node v_i (w.r.t. the pseudo label estimation) if the corresponding value a_{ij} is higher, and vice versa. It is also in line with the *dyadic hypothesis* (Granovetter, 1973) as higher node attribute similarity implies similar pseudo labels, which leads to stronger tie strength.

Reconstruct the Network. Our target is to find a subset of features $\mathcal{S}$ that are the most tightly correlated to the network structure, thus we assume that the observed links $\mathcal{E}$ on the network can be reconstructed from the feature subset $\mathcal{S}$. We first define how to measure the node similarity w.r.t. a subset of informative features $\mathcal{S}$.

Definition 4. ***(Node Similarity w.r.t.*** $\mathcal{S}$*):** *Given a feature subset $\mathcal{S}$ and the corresponding feature indicator vector $\mathbf{w} \in \{0,1\}^d$, the node similarity between two nodes v_i and v_j on the attributed network G is defined as $s_{ij} = \mathbf{x}_i' diag(\mathbf{w})\mathbf{x}_j$.*

From a generative point of view, given the node v_i, we assume that the probability of an edge (e.g., $(v_i, v_j) \in \mathcal{E}$) can be decided by quantifying how similar the two end nodes v_i and v_j are in the feature space $\mathcal{S}$. Then for each observed link $(v_i, v_j) \in \mathcal{E}$, the probability that v_j is in the context of v_i is determined by the softmax function as follows:

$$p(v_j|v_i) = \frac{\exp(\mathbf{x}_i' \text{diag}(\mathbf{w})\mathbf{x}_j)}{\sum_{m=1}^{n} \exp(\mathbf{x}_i' \text{diag}(\mathbf{w})\mathbf{x}_m)}. \tag{3.20}$$

The above formulation implies that the more similar node v_j and node v_i is in the feature space $\mathcal{S}$, the more likely we can reconstruct the observed link $(v_i, v_j) \in \mathcal{E}$.

Capturing the Correlation between Network and Node Attributes. To capture the correlation between node attributes and network structure, for each node v_i, we enforce its conditional distribution vector $\mathbf{p}_i = [p(v_1|v_i), ..., p(v_n|v_i)]'$ to preserve the optimal adaptive neighbor structure specified by the tie strength vector $\mathbf{a}_i$. This target can be achieved by minimizing the KL divergence between the distributions $\mathbf{p}_i$

and $\mathbf{a}_i$. By summing up the KL divergence for all nodes, we obtain:

$$\min \sum_{i=1}^{n} \gamma_i D_{KL}(\mathbf{a}_i || \mathbf{p}_i) \text{ s.t. } \mathbf{1}'\mathbf{w} = m;\ w_m \in \{0,1\},\ (\forall m = 1, ..., d), \tag{3.21}$$

where γ_i is introduced to show the prestige of node v_i, and its value can be determined by various node centrality measures such as degree centrality and PageRank (Zafarani *et al.*, 2014). We can further expand the above objective function value as follows:

$$\sum_{(i,j)\in\mathcal{E}} \gamma_i a_{ij} \log(a_{ij}) - \sum_{(i,j)\in\mathcal{E}} \gamma_i a_{ij} \log p(v_i|v_j). \tag{3.22}$$

In the above equation, the computation of the conditional probability $p(v_j|v_i)$ is very expensive due to the summation of all possible terms $\sum_{m=1}^{n} \exp(\mathbf{x}_i' \text{diag}(\mathbf{w}) \mathbf{x}_m)$ in the denominator of the softmax function, especially when the number of nodes n is large. To address this issue, we make use of the negative sampling approach proposed in (Mikolov *et al.*, 2013) to reformulate the optimization problem in Eq. (3.21):

$$\begin{aligned} \min \sum_{(i,j)\in\mathcal{E}} \gamma_i a_{ij} \log(a_{ij}) - \sum_{(i,j)\in\mathcal{E}} \gamma_i a_{ij} \underbrace{\big(\log\sigma(s_{ij}) + \sum_{m=1}^{K} \mathbb{E}_{v_m \sim P_v} [\log\sigma(-s_{im})] \big)}_{\Theta_{ij}} \\ \text{s.t. } \mathbf{1}'\mathbf{w} = m;\ w_m \in \{0,1\},\ (\forall m = 1, ..., d). \end{aligned} \tag{3.23}$$

where $\sigma(x)$ is the sigmoid function, and K is the number of negative samples. Given the node v_i, the task now is to distinguish its neighborhood nodes v_j from other K nodes randomly drawn from the noisy distribution P_v – proportional to the node degree distribution raised to the power of 3/4 (Mikolov *et al.*, 2013).

The optimization problem in Eq. (3.23) is NP-hard due to the discrete nature of $\mathbf{w}$. Thus we relax the discrete constraint on $\mathbf{w}$ by reformulating it as a real-valued vector in the range of $[0, 1]$ (Wei *et al.*, 2015). Furthermore, we rewrite the constraint $\mathbf{1}'\mathbf{w} = m$ in the Lagrangian, resulting in the following optimization problem:

$$\begin{aligned} \min \sum_{(i,j)\in\mathcal{E}} \gamma_i a_{ij} \log(a_{ij}) - \sum_{(i,j)\in\mathcal{E}} \gamma_i a_{ij} \Theta_{ij} + \alpha \|\mathbf{w}\|_1 \\ \text{s.t. } 0 \le w_m \le 1\ (\forall m = 1, ..., d), \end{aligned} \tag{3.24}$$

where α controls the sparsity of feature indicator vector $\mathbf{w}$.

3.4.2 Optimization Solution

We discuss how to obtain the adaptive neighborhood structure $\mathbf{a}_i$ for each node v_i and the feature indicator vector $\mathbf{w}$.

First, we learn the optimal adaptive neighborhood structure $\mathbf{a}_i$ for each node through Eq. (3.19). However, it is difficult as $\mathbf{a}_i$ is related to the mapping function $f(.)$ and the noisy pseudo class label $\mathbf{y}_i$, both are unknown in an unsupervised scenario. Following (Anava and Levy, 2016), we reformulate the problem in Eq. (3.19) into the following one which yields an upper bound guarantee of high confidence:

$$\min_{\mathbf{a}_i} \|\mathbf{a}_i\|_2 + M \sum_{j=1}^{n} a_{ij} d(\mathbf{x}_j, \mathbf{x}_i), \;\; \text{s.t. } \mathbf{a}_i \in \Omega_i, \tag{3.25}$$

where M is a positive constant and $d(.,.)$ is a distance function. The Lagrangian of the above problem is as follows:

$$L = \|\mathbf{a}_i\|_2 + \mathbf{a}_i' \mathbf{u}_i + \lambda(1 - \sum_{j=1}^{n} a_{ij}) - \sum_{v_j \in \mathcal{N}(v_i)} \theta_j a_{ij} + \sum_{v_j \notin \mathcal{N}(v_i)} \eta_j a_{ij}, \tag{3.26}$$

where $\mathbf{u}_i = [M.d(\mathbf{x}_1, \mathbf{x}_i), ..., M.d(\mathbf{x}_n, \mathbf{x}_i)]'$. The parameters λ, η_j $(\forall v_j \notin \mathcal{N}(v_i))$ and $\theta_j \geq 0$ $(\forall v_j \in \mathcal{N}(v_i))$ are the Lagrange multipliers. As Eq. (3.25) is convex, thus any solution that satisfies the KKT condition guarantees a global optimum. By setting the derivative of the Lagrangian w.r.t. $\mathbf{a}_i$ to zero, we obtain:

$$\frac{a_{ij}}{\|\mathbf{a}_i\|_2} = \begin{cases} \lambda + \theta_j - u_{ij}, & \forall v_j \in \mathcal{N}(v_i) \\ \lambda - \eta_j - u_{ij}, & \forall v_j \notin \mathcal{N}(v_i). \end{cases} \tag{3.27}$$

Let $\mathbf{a}_i^*$ be the optimal solution, according to the complementary slackness condition, for any $a_{ij}^* > 0$, we obtain $a_{ij}^* / \|\mathbf{a}_i^*\|_2 = \lambda - u_{ij}$. And the optimal solution of $\mathbf{a}_i^*$ is:

$$a_{ij}^* = \frac{\{\lambda - u_{ij}\} \cdot \mathbf{1}\{\lambda > u_{ij}\}}{\sum_{j=1}^{n} \{\lambda - u_{ij}\} \cdot \mathbf{1}\{\lambda > u_{ij}\}}, \tag{3.28}$$

for $\forall v_j \in \mathcal{N}(v_i)$. According to the definition of $\mathbf{a}_i$, it has a cutoff effect as $a_{ij} = 0$ if $v_j \notin \mathcal{N}(v_i)$. In addition to that, it can be observed that $\mathbf{a}_i$ also has a cutoff effect for $v_j \in \mathcal{N}(v_i)$ when the condition $\lambda > u_{ij}$ is satisfied. In other words, for each node v_i, there exists $0 \leq k_i^* \leq |\mathcal{N}(v_i)|$ such that only the tie strengths to these k_i^* neighbors are nonzero. Then by squaring and summing Eq. (3.28) over all nonzero entries in $\mathbf{a}_i^*$ and solving the equation[5], we have:

$$\lambda = \frac{1}{k_i^*}\Big(\sum_{i=1}^{k_i^*} u_{ij} + \sqrt{(\sum_{j=1}^{k_i^*} u_{ij})^2 - k_i^* \sum_{j=1}^{k_i^*} u_{ij}^2 + k_i^*}\Big). \tag{3.29}$$

Following (Anava and Levy, 2016), a greedy algorithm is leveraged to add neighbors v_j of node v_i according to the value of u_{ij} until the optimal number of neighbors k_i^* is fulfilled (until the condition $\lambda > u_{ij}$ does not hold any more).

Second, to update the feature indicator vector $\mathbf{w}$, we plug the optimal solution of $\mathbf{a}_i^*$ into Eq. (3.24), resulting in the following optimization problem:

$$\begin{aligned} &\min_{\mathbf{w}} \sum_{(i,j)\in\mathcal{E}} \gamma_i a_{ij}^* \log(a_{ij}^*) - \sum_{(i,j)\in\mathcal{E}} \gamma_i a_{ij}^* \Theta_{ij} + \alpha\|\mathbf{w}\|_1 \\ &\text{s.t. } 0 \leq w_m \leq 1 \ \ (\forall m = 1, ..., d). \end{aligned} \tag{3.30}$$

We apply projected stochastic gradient descent method to optimize the above objective function, by sampling a mini-batch of edges each step. Suppose the edge (v_i, v_j) is sampled, then the objective function in Eq. (3.30) is reduced to:

$$\mathcal{L}(i,j) = \underbrace{\gamma_i a_{ij}^* \log(a_{ij}^*)}_{\mathcal{L}_1(i,j)} + \underbrace{(-\gamma_i a_{ij}^* \log\sigma(s_{ij}))}_{\mathcal{L}_2(i,j)} + \underbrace{(-\gamma_i a_{ij}^* \sum_{m=1}^{K} \mathbb{E}_{v_m\sim P_v}[\log\sigma(-s_{im})])}_{\mathcal{L}_3(i,j)} + \alpha\|\mathbf{w}\|_1. \tag{3.31}$$

The first term $\mathcal{L}_1(i,j)$ is independent of $\mathbf{w}$, while the partial derivative of $\mathcal{L}_2(i,j)$,

[5]Without the loss of generality, we assume that for each node v_i, the index of the other nodes are ordered in an ascending order w.r.t. u_{ij},

and $\mathcal{L}_3(i,j)$ w.r.t. w_k can be calculated as follows:

$$\begin{aligned}\frac{\partial \mathcal{L}_2(i,j)}{\partial w_k} &= -\gamma_i a^*_{ij}(1-\sigma(s_{ij}))x_{ik}x_{jk} \\ \frac{\partial \mathcal{L}_3(i,j)}{\partial w_k} &= \gamma_i a^*_{ij}\sum_{m=1}^{K}\mathbb{E}_{v_m\sim P_v}\sigma(s_{im})x_{ik}x_{mk}.\end{aligned} \tag{3.32}$$

Normally, the ℓ_1-norm regularization term on $\mathbf{w}$ is not smooth and its derivative is not achievable everywhere over its feasible region. However, as we relax the constraint on $\mathbf{w}$ to make it in the range of [0,1], the partial derivative of $\alpha\|\mathbf{w}\|_1$ w.r.t. w_k is α. With the above partial derivatives, the update step of w_k is given as follows:

$$w_k \leftarrow P\big[w_k - \rho\frac{\partial \mathcal{L}(ij)}{\partial w_k}\big], \tag{3.33}$$

where $P[x]$ maps $x \in \mathbb{R}$ in the bounded region of [0,1]. Meanwhile, ρ is the learning rate, and can be adaptively adjusted to facilitate the convergence.

3.4.3 Time Complexity Analysis

The detailed pseudocode of ADAPT is illustrated in Algorithm 2. We first make use of the observed features to update the tie strength vector $\mathbf{a}_i$ for each node. The complexity of updating $\mathbf{a}_i$ is $O(k_i + d|\mathcal{N}(v_i)| + |\mathcal{N}(v_i)|\log|\mathcal{N}(v_i)|)$, where k_i denotes the optimal number of neighbors for node v_i on the attributed network. Then in each epoch, we update the feature indicator vector $\mathbf{w}$ by sampling a mini-batch of edges and the negative samples. For each epoch, the complexity of updating $\mathbf{w}$ is $O(edK) + \sum_{i=1}^{n} O(dk_i|\mathcal{N}(v_i)|)$. The negative sampling in each epoch takes $O(e)$ with the alias table (Li *et al.*, 2014a).

3.5 Experimental Evaluation of ADAPT

We perform experiments on real-world attributed networks of various types to validate the effectiveness of the proposed ADAPT framework.

Algorithm 2 Proposed Feature Selection Framework ADAPT.

Input: Attributed network G, the number desired features m, parameters α and M.

Output: The top-m ranked features.

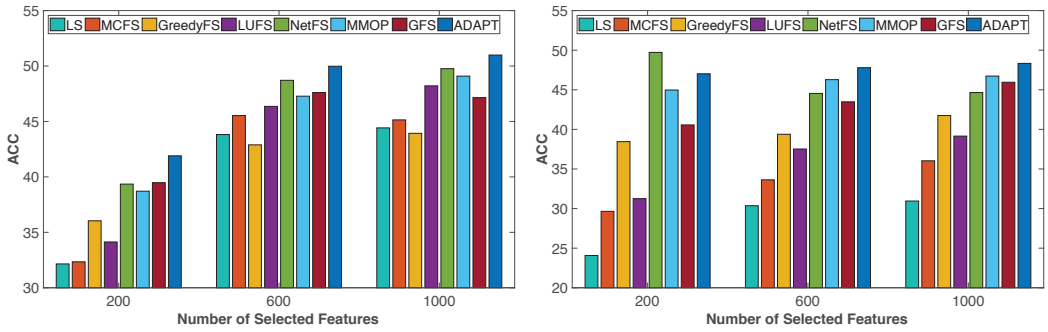
Initialize the feature indicator vector $\mathbf{w}$;
Update $\mathbf{a}_i$ via the greedy algorithm in section 3.4.2 ($\forall i = 1, ..., n$);
for $epoch = 1 : maxepoch$ **do**
    for $batch = 1 : numbatch$ **do**
        Sample a mini-batch of edges and negative samples;
        Update w_k ($\forall k = 1, ..., d$) according to Eq. (3.33) ;
    end for
end for
Rank features according to the entries in $\mathbf{w}$.

Datasets. As we focus on unsupervised feature selection in a static setting, in addition to the aforementioned BlogCatalog and Flickr, we also use another two static attributed networks Wiki (Yang *et al.*, 2015) and ACM (Huang *et al.*, 2018) for evaluation. Wiki is a collection of Wikipedia documents that are inherently connected with each other via hyperlinks. Each document is categorized into a number of predefined classes. The dataset contains 2,405 Wikipedia documents from 19 different classes. Each document is described by 4,973-dimensional TF-IDF features. In total, there are 12,178 hyperlinks among these Wikipedia documents. ACM is a subgraph of citation network of papers published before 2016 in ACM organized venues. Each publication is described by the bag-of-words features based on the abstract and is categorized into one of the 9 predefined classes such as machine learning and data mining. In the dataset, we have 16,484 publications, 8,337 features, and 71,980 links.

Experimental Settings. The experimental settings are the same as the evaluation of NetFS. The following baseline methods of different categories are compared:

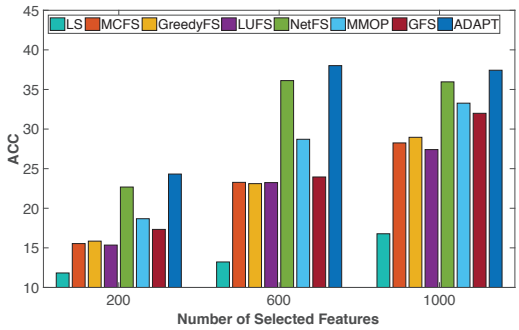
- LS (He *et al.*, 2006): It is a baseline method mentioned previously in section 3.3.
- MCFS (Cai *et al.*, 2010b): It performs feature selection based on spectral analysis and sparse regression.
- GreedyFS (Farahat *et al.*, 2013): It selects features in a greedy manner by measuring the reconstruction error of the data.
- LUFS (Tang and Liu, 2012b): It is also a baseline method mentioned previously in section 3.3.
- NetFS (Li *et al.*, 2016b): It is our developed robust feature selection framework for attributed networks.
- MMOP (Wei *et al.*, 2015): It selects features on attributed networks that can maximally preserve the partial order relations among nodes.
- GFS (Wei *et al.*, 2016): It finds relevant features on attributed networks by modeling network and attributes with a generative process.

Among them, LS, MCFS, GreedyFS are conventional methods with node attributes only. They respectively belong to the similarity based, sparse learning based, and reconstruction based methods, which are the three most widely used categories of unsupervised feature selection methods. LUFS and NetFS exploit the link information at a coarse granularity level via community analysis, while MMOP and GFS directly make use of the link information for unsupervised feature selection but treat all links equally. As these methods are from different categories, their comparisons against ADAPT could further reveal the superiority of the developed framework. For LS and MCFS, we set the number of neighborhood size to be 5. In GreedyFS, the number of feature partitions is specified as 5. For MCFS, LUFS, and NetFS,

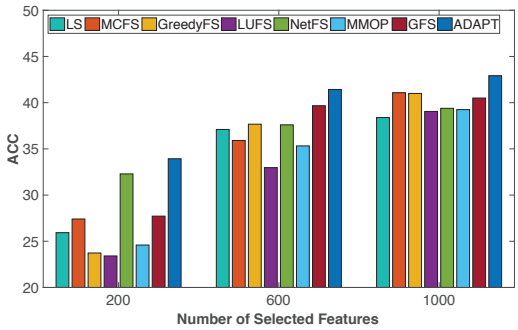


(a) Wiki

(b) BlogCatalog



(c) Flickr

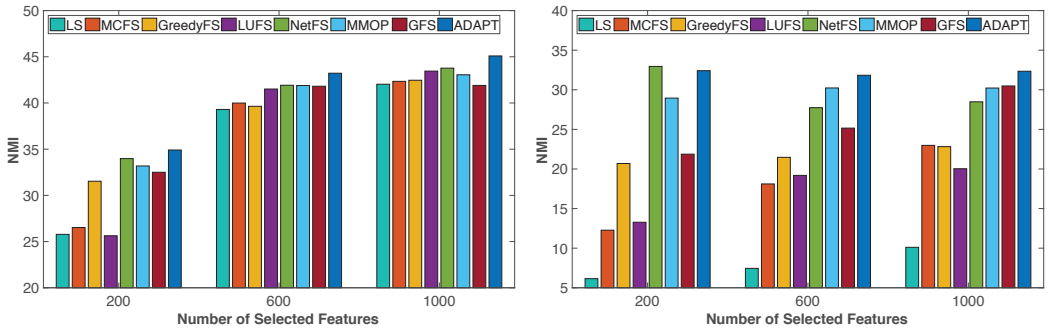


(d) ACM

Figure 3.4: Clustering Results (ACC) Evaluation of ADAPT and Baselines.

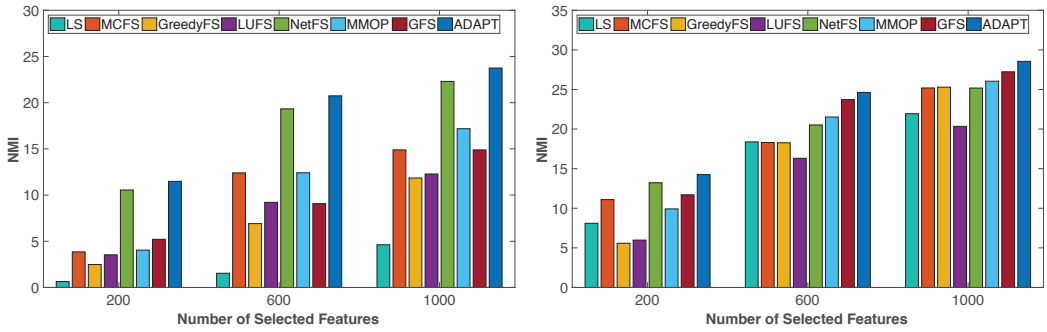
the number of clusters or pseudo labels is the true number of classes. In ADAPT, the number of negative samples K is 5. In addition, different methods have different sets of regularization parameters, to have a fair comparison, we tune these parameters via grid search and the best average clustering results are reported.

Quality of Selected Features by ADAPT. We investigate the effectiveness of the proposed ADAPT by comparing the clustering performance against other baseline methods after feature selection. The number of selected features is varied in the range of $\{200, 600, 1000\}$. The comparison results are shown in Figure 3.4 and Figure 3.5. We make the following observations from these figures: (1) ADAPT obtains the best clustering performance in almost all cases w.r.t. different numbers of selected features and the improvement is statistically significant at the level of 0.05



(a) Wiki

(b) BlogCatalog



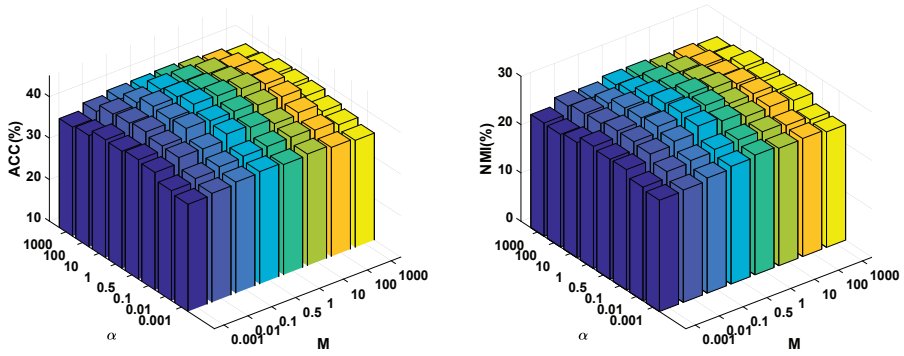
(c) Flickr

(d) ACM

Figure 3.5: Clustering Results (NMI) Evaluation of ADAPT and Baselines.

(with pairwise Wilcoxon signed rank test). (2) ADAPT is superior to LUFS and NetFS which model the network information at the community level. Meanwhile, it also achieves better performance than MMOP and GFS which treat all observed links equally. The improvement of ADAPT over these methods corroborate the importance of characterizing adaptive neighborhood structure around each node for feature selection. As LUFS is very sensitive to the noisy and incomplete network structure, its performances are the worst among these five methods. (3) LS, MCFS, and GreedyFS are conventional unsupervised feature selection methods which only make use of the node attribute information. Their performance is inferior to NetFS, MMOP, and GFS in many cases. The observation supports the assumption that network structure complements node attribute information for feature selection. (4) The

improvement of feature selection algorithms with network information (i.e., ADAPT, NetFS) over conventional feature selection methods (i.e., LS, MCFS, GreedyFS) is relatively higher on the BlogCatalog and Flickr datasets. The reason is that the density of networks (the ratio between the number of edges and the number of nodes) is much higher on these two datasets, thus network information could provide more constraints in finding relevant features.



(a) ACC (b) NMI

Figure 3.6: Parameter Study of ADAPT on ACM Dataset.

Parameter Sensitivity Study. ADAPT has two important model parameters: (1) M controls the number of optimal neighbors around each node for tie strength characterization; and (2) α controls the sparsity of the feature indicator vector $\mathbf{w}$. To investigate how the variation of these two model parameters affects the feature selection performance, we vary them among $\{0.001, 0.01, 0.1, 0.5, 1, 10, 100, 1000\}$. We only show the parameter study results on the ACM dataset (with 1000 selected features) as we have similar observations on the other datasets. Firstly, as can be observed from Figure 3.6, when we vary the value of M, the clustering performance first increases, then reaches its peak, and then gradually decreases. It implies that finding a suitable number of optimal neighbors around each node could advance feature selection. Secondly, we can observe that when the parameter α is between 0.1 and 10,

the clustering performance is relatively stable. Hence, it is safe to tune α in a wide range without jeopardizing the clustering performance too much.

3.6 Summary

Two distinct but highly correlated data representations are naturally observed on real-world attributed networks. In many cases, high-dimensional node attributes increase the possibility of including noisy, redundant, and network topologically irrelevant features, which hinder us to gain insights from such networks. Without the label supervision, we make several efforts to find a subset of features that are can be fused with network topology seamlessly for synergistic knowledge discovery. We first develop a robust unsupervised feature selection framework NetFS. NetFS first uses latent representations to capture the inherent correlations of nodes on the network, then we propose to embed the latent representation learning process into feature selection. Therefore, these two phases could help and boost each other to obtain good features, and the proposed model is more robust to noisy links. Later on, to capture the finer-grained tie strength information embedded on the network, we make the initial investigation to develop a principled adaptive unsupervised feature selection framework ADAPT on attributed networks. Specifically, the proposed framework characterizes the optimal adaptive neighborhood structure around each node for unsupervised feature selection on attributed networks. We also perform empirical evaluations on various real-world attributed networks, and the experimental results demonstrate the effectiveness of the proposed frameworks.

Chapter 4

NETWORK EMBEDDING ON ATTRIBUTED NETWORKS

Aside from feature selection which investigates the attributed networks at the attribute level, network embedding yields a different angle to scrutinize task-agnostic learning algorithms on attributed networks at the node level. Specifically, we aim to map each node to a low-dimensional vector space such that node proximity in terms of both network topology and node attributes are both well preserved. Since the node embeddings are obtained independent of any specific learning tasks, they can be generalized to a wide range of downstream applications.

4.1 Overview

Network embedding (Grover and Leskovec, 2016; Perozzi *et al.*, 2014b; Tang *et al.*, 2015b) has attracted a surge of research attention in recent years. The basic idea is to preserve the node proximity in the embedded Euclidean space, based on which the performance of various network mining tasks such as node classification (Aggarwal and Li, 2011; Bhagat *et al.*, 2011), community detection (Tang *et al.*, 2008; Yang *et al.*, 2009), and link prediction (Barbieri *et al.*, 2014; Liben-Nowell and Kleinberg, 2007; Wang *et al.*, 2011) can be enhanced. However, a vast majority of existing work are predominately designed for plain networks. They inevitably ignore the node attributes that could be potentially complementary in learning better embedding representations, especially when the network suffers from high sparsity. However, assessing a vector representation for each node in the joint space of geometrical structure and node attributes is difficult due to the bewildering combination of heterogeneous sources. Thus, it is challenging for existing network embedding algorithms to be

directly applied. To handle the challenge, we present a novel attributed network embedding framework DANE-O (Li *et al.*, 2017c) to learn discriminative node embedding representations. Even though network topology and node attributes are two distinct data representations, they are inherently correlated. In addition, the raw data representations could be noisy and even incomplete, individually. Hence, it is of paramount importance to seek a noise-resilient consensus embedding to capture their individual properties and inherent correlations.

We first illustrate the studied problem of attributed network embedding before presenting the details of the proposed solution DANE-O.

Problem 2. Network Embedding for Static Attributed Networks

Given: *An attributed network G represented by the content matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ and the adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}_{\geq 0}$, where n and d denote the number of nodes and features, respectively*[1]*. The dimensionality of final consensus node embeddings is specified as l.*

Learn: *Embedding representation $\mathbf{Y} \in \mathbb{R}^{n \times l}$ for all nodes by preserving the node proximity w.r.t. two different sources of information*[2]*.*

4.2 Proposed Consensus Framework - DANE-O

In this section, we present an offline model DANE-O that works in a static setting in finding a consensus embedding representation for nodes on attributed networks. An illustration of the proposed DANE-O framework is shown in Figure 4.1. As we can see from the figure, the whole framework consists of two steps. In the first step, we attempt to learn the intermediate embedding representations for each data

[1]We adopt the same strategy as Chapter 3 to convert directed networks into undirected ones.

[2]The exact optimization function is given in the following subsections.

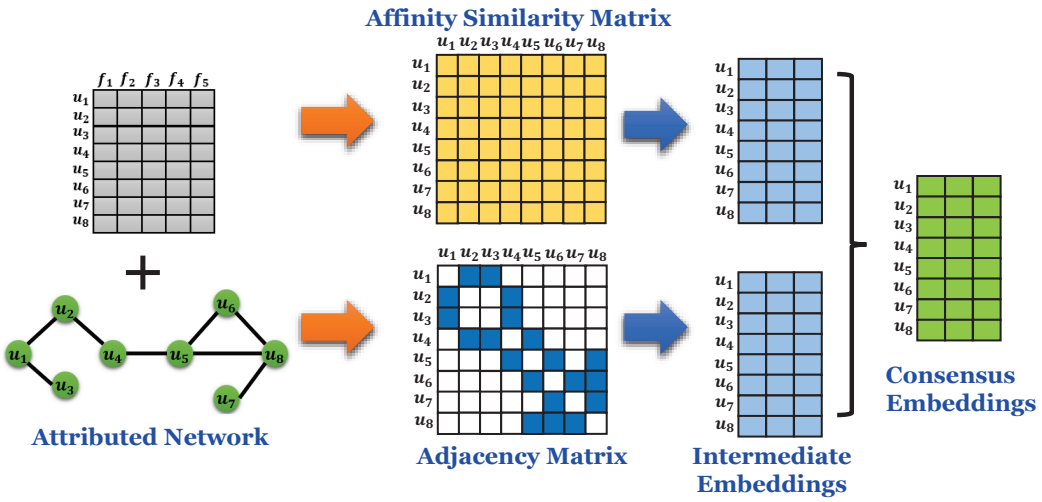


Figure 4.1: An Illustration of the Consensus Embedding Framework DANE-O.

modality by reducing the noise. Then in the second step, we aim to leverage the inherent correlations among these two intermediate embeddings for a final consensus embedding of all nodes on the attributed network. In the consensus embedding space, the node proximity information w.r.t. both the network structure and node attribute similarity are well preserved.

4.2.1 Problem Formulation

We present the details of learning intermediate node embeddings and learning consensus node embeddings.

Learning Intermediate Node Embeddings. Network topology and node attributes on attributed networks are presented in different representations. Typically, either of these two representations could be incomplete and noisy, presenting great challenges to embedding representation learning. For example, social networks are very sparse as a large amount of users only have a limited number of links (Adamic and Huberman, 2000). Thus, network embedding could be jeopardized as links are inadequate to provide enough node proximity information. Fortunately, rich node at-

tributes are readily available and could be potentially helpful to mitigate the network sparsity in finding better embeddings. Hence, it is more desired to make these two representations compensate each other for consensus embeddings. However, as mentioned earlier, both representations could be noisy and the existence of noise could degenerate the learning of consensus embedding. Hence, it motivates us to reduce the noise of these two raw data representations before learning consensus embeddings.

Let $\mathbf{A} \in \mathbb{R}^{n\times n}_{\geq 0}$ be the adjacency matrix and $\mathbf{D_A}$ be the diagonal matrix with $\mathbf{D_A}(i,i) = \sum_{j=1}^{n} \mathbf{A}(i,j)$, then $\mathbf{L_A} = \mathbf{D_A} - \mathbf{A}$ is a Laplacian matrix. According to the spectral theory (Belkin and Niyogi, 2002; Von Luxburg, 2007), by mapping each node on the network to a k-dimensional embedded space, i.e., $\mathbf{y}_i \in \mathbb{R}^k$ ($k \ll n$), the noise on the network can be substantially reduced. A rational choice of the embedding $\mathbf{Y_A} = [\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_n]' \in \mathbb{R}^{n\times k}$ is to minimize the following loss function:

$$\frac{1}{2}\sum_{i,j} \mathbf{A}(i,j)\|\mathbf{y}_i - \mathbf{y}_j\|_2^2. \tag{4.1}$$

It ensures that connected nodes are close to each other in the embedded space. Meanwhile, the orthogonal constraint $\mathbf{Y}'_\mathbf{A}\mathbf{D_A}\mathbf{Y_A} = \mathbf{I}$ is often imposed to avoid arbitrary scaling factor of the embedding. Thus, the problem boils down to solving the following generalized eigen-problem:

$$\mathbf{L_A}\mathbf{a} = \lambda\mathbf{D_A}\mathbf{a}. \tag{4.2}$$

Let $\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_n$ be the eigenvectors of the above generalized eigen-problem and the corresponding eigenvalues are sorted in an ascending order $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$. It is easy to verify that $\mathbf{1}$ is the only eigenvector for the eigenvalue $\lambda_1 = 0$. Then the k-dimensional embedding $\mathbf{Y_A} \in \mathbb{R}^{n\times k}$ of the network structure is given by the top-k eigenvectors starting from $\mathbf{a}_2$, i.e., $\mathbf{Y_A} = [\mathbf{a}_2, ..., \mathbf{a}_k, \mathbf{a}_{k+1}]$. For the ease of presentation, in the following part, we refer these k eigenvectors and their eigenvalues as the top-k eigenvectors and eigenvalues, respectively.

Akin to the network structure, noise in the node attribute space can be reduced in a similar fashion with spectral embedding. Specifically, we first normalize the attributes of each node and obtain the cosine similarity matrix $\mathbf{S}$. Afterwards, we obtain the top-k eigenvectors $\mathbf{Y_X} = [\mathbf{b}_2, ..., \mathbf{b}_{k+1}]$ of the generalized eigen-problem corresponding to the cosine similarity matrix $\mathbf{S}$.

Learning Consensus Node Embeddings. The noisy data problem is resolved by finding two intermediate embeddings $\mathbf{Y_A}$ and $\mathbf{Y_X}$. We now take advantage of them to seek a consensus embedding. However, since they are obtained individually, these two embeddings may not be compatible and in the worst case, they may be independent of each other. To capture their interdependence and to make them compensate each other, we propose to maximize their correlations (or equivalently minimize their disagreements) by Canonical Correlation Analysis (Hardoon *et al.*, 2004). In particular, we seek two projection vectors $\mathbf{p_A}$ and $\mathbf{p_X}$ such that the correlation of $\mathbf{Y_A}$ and $\mathbf{Y_X}$ is maximized after projection. It is equivalent to solving the following optimization problem:

$$\begin{aligned} &\max_{\mathbf{p_A}, \mathbf{p_X}} \mathbf{p}_\mathbf{A}'\mathbf{Y}_\mathbf{A}'\mathbf{Y}_\mathbf{A}\mathbf{p}_\mathbf{A} + \mathbf{p}_\mathbf{A}'\mathbf{Y}_\mathbf{A}'\mathbf{Y}_\mathbf{X}\mathbf{p}_\mathbf{X} + \mathbf{p}_\mathbf{X}'\mathbf{Y}_\mathbf{X}'\mathbf{Y}_\mathbf{A}\mathbf{p}_\mathbf{A} + \mathbf{p}_\mathbf{X}'\mathbf{Y}_\mathbf{X}'\mathbf{Y}_\mathbf{X}\mathbf{p}_\mathbf{X} \\ &\text{s.t.} \quad \mathbf{p}_\mathbf{A}'\mathbf{Y}_\mathbf{A}'\mathbf{Y}_\mathbf{A}\mathbf{p}_\mathbf{A} + \mathbf{p}_\mathbf{X}'\mathbf{Y}_\mathbf{X}'\mathbf{Y}_\mathbf{X}\mathbf{p}_\mathbf{X} = 1. \end{aligned} \tag{4.3}$$

Let γ be the Lagrange multiplier for the constraint, by setting the derivative of the Lagrange function w.r.t. $\mathbf{p_A}$ and $\mathbf{p_X}$ to zero, we obtain the solution for $[\mathbf{p_A}; \mathbf{p_X}]$, which are the eigenvectors of the following generalized eigen-problem:

$$\begin{bmatrix} \mathbf{Y}_\mathbf{A}'\mathbf{Y}_\mathbf{A} & \mathbf{Y}_\mathbf{A}'\mathbf{Y}_\mathbf{X} \\ \mathbf{Y}_\mathbf{X}'\mathbf{Y}_\mathbf{A} & \mathbf{Y}_\mathbf{X}'\mathbf{Y}_\mathbf{X} \end{bmatrix} \begin{bmatrix} \mathbf{p_A} \\ \mathbf{p_X} \end{bmatrix} = \gamma \begin{bmatrix} \mathbf{Y}_\mathbf{A}'\mathbf{Y}_\mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{Y}_\mathbf{X}'\mathbf{Y}_\mathbf{X} \end{bmatrix} \begin{bmatrix} \mathbf{p_A} \\ \mathbf{p_X} \end{bmatrix}. \tag{4.4}$$

Later on, to obtain a consensus embedding representation, we could take the top-l eigenvectors of the above generalized eigen-problem and stack them together. Suppose the projection matrix $\mathbf{P} \in \mathbb{R}^{2k \times l}$ is the concatenated top-l eigenvectors, the final consensus embedding representation can be computed as $\mathbf{Y} = [\mathbf{Y_A}, \mathbf{Y_X}] \times \mathbf{P}$.

4.2.2 Time Complexity Analysis

The complexity of computing the attribute similarity matrix is $O(n^2d)$. Later on, the computation of obtaining the two intermediate embeddings requires $O(n^2k)$, and the computation of the final consensus embedding needs $O(k^2l)$.

4.3 Experimental Evaluation

We evaluate the effectiveness of the proposed consensus embedding framework DANE-O in a static setting. The details of the used datasets and experimental settings are introduced before presenting details of the experimental results.

Datasets. We use four datasets BlogCatalog, Flickr, Epinions, and DBLP for experimental evaluation. The BlogCatalog and Flickr datasets have been introduced before in Chapter 3. Similar as NetFS, to facilitate the comparison between DANE-O and its online version DANE (will be introduced later in Chapter 6), we first apply the perturbation by adding 0.1% new edges and changing 0.1% attribute values on BlogCatalog and Flickr datasets across 20 time stamps to generate two semi-synthetic datasets. In the Epinions dataset, the nodes denote users and the links represent the trust relations among users, the attributes of users are obtained using the bag-of-words model from users' reviews on products. The major category of reviews on products is taken as ground truth. The dataset spans over 16 time stamps and was collected a different period of time against the Epinions dataset in Table 3.1. In the last dataset DBLP, we extracted a DBLP co-author network for the authors that publish at least two papers between the years of 2001 and 2016 from seven different areas. Bag-of-words model is applied to the paper title to obtain the attribute information, and the major area the authors publish is considered as ground truth. Then we rerun DANE-O and baseline methods on these datasets at each time stamp

Table 4.1: Detailed Information of the Datasets Used for DANE-O and DANE.

	BlogCatalog	Flickr	Epinions	DBLP
# of nodes	5,196	7,575	14,180	23,393
# of features	8,189	12,047	9,936	8,945
# of links	173,468	242,146	227,642	289,478
# of classes	6	9	20	7
# of time stamps	10	10	16	16

and the average evaluation performance is presented. The detailed statistics of the datasets are shown in Table 4.1.

Experimental Settings. One commonly adopted way to evaluate the quality of the embedding representations (Perozzi *et al.*, 2014b; Tang *et al.*, 2015b) is by the following two unsupervised and supervised tasks: network clustering and node classification. First, we validate the effectiveness of the embedding representations of the proposed framework on the network clustering task. Two standard clustering performance metrics, i.e., *clustering accuracy* (ACC) and *normalized mutual information* (NMI) are used. In particular, after obtaining the embedding representation of each node on the attributed network, we perform K-means clustering based on the embedding representations. The K-means algorithm is repeated 10 times and the average results are reported since K-means may converge to the local minima due to different initializations. Normally, better clustering performance implies better node embedding representations. Another way to assess the embedding is by the node classification task. Specifically, we split the embedding representations of all nodes via a 10-fold cross-validation, using 90% of nodes to train a classification model by logistic regression and the rest 10% nodes for the testing. The whole process is repeated 10 times and the average performance are reported. Three evaluation metrics,

classification accuracy (AC), *F1-Macro* and *F1-Micro* are used. *F1-Macro* can be considered as a weighted average of F1-measure over all c class labels:

$$\text{F1-Micro} = \frac{\sum_{i=1}^{c} 2\text{TP}^i}{\sum_{i=1}^{c}(2\text{TP}^i + \text{FP}^i + \text{FN}^i)}. \tag{4.5}$$

F1-Macro is an arithmetic average of F1-measure of all output labels:

$$\text{F1-Macro} = \frac{1}{c}\sum_{i=1}^{c} \frac{2\text{TP}^i}{(2\text{TP}^i + \text{FP}^i + \text{FN}^i)}. \tag{4.6}$$

TP^i, FP^i and FN^i denote the number of true positives, false positives and false negatives in the i-th class label, respectively. The higher the *classification accuracy*, *F1-Macro*, and *F1-Micro* values are, the better the learned embeddings are.

How to determine the optimal number of embedding dimensions is still an open research problem, thus we vary the embedding dimension as $\{10, 20, ..., 100\}$ and the best evaluation results are reported.

The proposed consensus embedding framework DANE-O is measured against the following baseline methods on the two aforementioned tasks:

- DEEPWALK (Perozzi *et al.*, 2014b): It learns network embeddings by the usage of word2vec (Mikolov *et al.*, 2013) and truncated random walk techniques.
- LINE (Tang *et al.*, 2015b): It learns embeddings by preserving the first and second-order node proximity information.
- DANE-N: It is a variation of DANE-O with only network information.
- DANE-A: It is a variation of DANE-O with only attribute information.
- CCA (Hardoon *et al.*, 2004): It directly uses the network structure and attributes for a joint low-dimensional representation.

Table 4.2: Clustering Results Evaluation of DANE-O and Baselines.

Datasets		BlogCatalog		Flickr		Epinions		DBLP	
Methods		ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI
Network	DEEPWALK	49.85	30.51	40.70	24.29	13.31	12.72	53.61	32.54
	LINE	50.20	29.53	42.93	26.01	14.34	12.65	51.61	30.74
	DANE-N	37.05	21.84	31.89	18.91	12.01	11.95	56.61	31.54
Attributes	DANE-A	62.32	45.95	63.80	48.29	16.12	11.62	47.37	20.64
Network+Attributes	CCA	33.42	11.86	24.39	10.89	10.85	8.61	26.42	18.60
	LCMF	55.72	40.38	27.03	13.06	12.86	10.73	42.27	26.48
	LANE	65.06	48.89	65.45	52.58	32.18	22.09	55.80	31.84
	DANE-O	**80.31**	**59.46**	**67.33**	**53.04**	**34.11**	**23.07**	**59.14**	**35.31**

Table 4.3: Classification Results Evaluation of DANE-O and Baselines.

Datasets		BlogCatalog			Flickr			Epinions			DBLP		
Methods		AC	Micro	Macro	AC	Micro	Macro	AC	Micro	Macro	AC	Micro	Macro
Network	DEEPWALK	68.05	67.15	68.18	60.08	58.93	59.08	22.12	17.43	20.10	74.38	69.65	72.37
	LINE	70.20	69.88	70.91	61.03	60.90	60.01	23.54	17.17	21.05	72.97	67.56	70.97
	DANE-N	66.97	66.06	67.78	49.37	47.82	49.34	21.25	20.57	21.88	71.99	65.33	71.94
Attributes	DANE-A	80.23	79.86	80.23	76.66	75.59	76.60	23.76	21.57	22.00	63.92	54.80	62.97
Network+Attributes	CCA	48.63	49.96	49.63	27.09	26.54	26.09	11.53	9.43	10.56	45.67	42.08	43.83
	LCMF	84.41	89.01	89.26	66.27	66.75	65.71	19.14	9.22	10.14	69.71	68.01	68.42
	LANE	87.52	87.52	87.93	77.54	77.81	77.26	27.74	28.45	28.87	72.15	71.09	73.48
	DANE-O	**89.34**	**89.15**	**89.23**	**79.68**	**79.52**	**79.95**	**31.23**	**31.28**	**31.35**	**77.21**	**74.96**	**75.48**

- LCMF (Zhu *et al.*, 2007): It maps network and attributes to a shared latent space by collective matrix factorization.

- LANE (Huang *et al.*, 2017b): It is a label informed attributed network embedding method, and we use the variant LANE w/o Label.

We follow the suggestions of the original papers to set the parameters of all these baseline methods for network embedding.

Quality of Learned Embeddings. To evaluate the effectiveness of embedding representations, we first compare DANE-O with baseline methods on network clustering

which is naturally an unsupervised learning task. The average clustering performance comparison w.r.t. ACC and NMI are presented in Table 4.2. We make the following observations: (1) DANE-O consistently outperforms all baseline methods on four datasets by achieving better clustering performance. The Wilcoxon signed-rank test is performed between DANE-O and other baselines and shows that DANE-O is significantly better with both 0.01 and 0.05 significance levels. (2) DANE-O and LANE achieve better clustering performance than network embedding methods such as DEEPWALK, LINE, and DANE-N; and attribute embedding method DANE-A. The improvements indicate that attribute information is complementary to pure network topology and can help learn more informative embedding representations. (3) Meanwhile, DANE-O also outperforms the CCA and LCMF which also leverage node attributes. The reason is that although these methods learn a low-dimensional representation by using both sources, they are not explicitly designed to preserve the node proximity. Also, their performance degenerates when the data is very noisy.

Next, we assess the effectiveness of embedding representations on a supervised learning task - node classification. The classification results in terms of three different measures are shown in Table 4.3. The following findings can be inferred from the table: (1) Generally, we have similar observations as the clustering task. The methods which only use link information or node attributes (e.g., DEEPWALK, LINE, DANE-N, DANE-A) and methods which do not explicitly model node proximity (e.g., CCA, LCMF) give poor classification results. (2) The embeddings learned by DANE-O help train a more discriminative classification model by obtaining higher classification performance. In addition, pairwise Wilcoxon signed-rank test shows that DANE-O is significantly better. (3) For the node classification task, the attribute embedding method DANE-A works better than the network embedding method in the BlogCatalog, Flickr, and Epinions datasets. The reason is that in these datasets,

the class labels are more closely related to the attribute information than the network structure. However, it is a different case for the DBLP dataset in which the labels of authors are more closely related to the coauthor relationships.

4.4 Summary

The prevalence of attributed networks in many real-world applications presents new challenges for many graph mining problems because of its natural heterogeneity. In this chapter, we investigate the attributed networks from a new perspective at the node level by developing novel network embedding algorithms. The main target is trying to map each node on the attributed network into a new low-dimensional feature space while we want to ensure that in the new feature space the node proximity w.r.t. the original network structure and attribute similarity can be well preserved. Later on, we can leverage the learned embeddings for various applications by applying off-the-shelf machine learning and data mining algorithms. Methodologically, we propose a consensus embedding framework DANE-O to fuse topological structure and instance attributes into a unified embedding space by maximizing their interactions, and the problem boils down to solving a series of generalized eigen-problems. To validate the effectiveness of the learned consensus embeddings, we conduct experiments on two learning tasks - network clustering and node classification. Empirical experimental evaluations show the promising performance of the proposed framework DANE-O.

Part II

Learning Algorithms in A Dynamic Environment

Chapter 5

FEATURE SELECTION ON DYNAMIC ATTRIBUTED NETWORKS

Feature selection has shown to be useful in deriving actionable patterns from attributed networks. Nonetheless, most of existing methods predominantly focus on a static setting, while fail to characterize the evolution facts of network structure and node features. Given the dynamic nature of attributed networks, it is necessary and of vital importance to perform feature selection incrementally to adapt the changes in a timely manner, which is of fundamental importance in many time-critical applications such as disaster relief and viral marketing.

5.1 Overview

As per the dynamic network theory (Westaby, 2012) in psychology and social science, network and features often co-evolve over time, and a small disturbance of network structure may result in a ripple effect on the drifts of feature patterns, and vice versa. With these unique characteristics, existing approaches probing dynamic networks are either correcting and adjusting the staleness of network mining algorithms or understanding the underlying evolution mechanisms (Aggarwal and Subbian, 2014). Despite the fundamental importance of analyzing attributed networks in a dynamic environment, the development of sophisticated learning models to find relevant features in an online fashion is still in its infancy. In this chapter, we study a novel problem about how to perform online feature selection on dynamic attributed networks. We also focus on an unsupervised scenario as label information of nodes is time and labor intensive to obtain. In particular, we present an online framework TeFS to employ the temporal smoothness property of attributed networks between

consecutive time stamps in updating the feature selection results incrementally.

We use G^t to represent the attributed network observed at time stamp t, and use the adjacency matrix $\mathbf{A}^t \in \mathbb{R}^{n\times n}_{\geq 0}$ and $\mathbf{X}^t \in \mathbb{R}^{n\times d}$ to represent the corresponding network structure and node feature information, respectively. For the ease of discussion, we assume that the number of nodes and the number of features are constant between two consecutive time stamps. The studied problem is formally defined as follows.

Problem 3. Feature Selection on Dynamic Attributed Networks

Given: *A dynamic attributed network that is observed at a series of time stamps t, $t+1$, ..., $t+i$, where the node feature information is represented by a set of content matrices $\{\mathbf{X}^t, \mathbf{X}^{t+1}, ..., \mathbf{X}^{t+i}\}$ (where the content feature space is $\mathcal{F}$) and network structure is encoded in a set of adjacency matrices $\{\mathbf{A}^t, \mathbf{A}^{t+1}, ..., \mathbf{A}^{t+i}\}$. The number of selected features is specified as m.*

Select: *A subset of most relevant features $\mathcal{S}^t \subset \mathcal{F}, \mathcal{S}^{t+1} \subset \mathcal{F}, ..., \mathcal{S}^{t+i} \subset \mathcal{F}$ that is tightly correlated with the network structure at each time stamp in a timely manner.*

5.2 Proposed Online Framework - TeFS

One widely adopted framework to analyze evolutionary network is to leverage the temporal smoothness property (Aggarwal and Subbian, 2014) which assumes that the structure of the network does not change significantly within a short period of time. In particular, given two consecutive time stamps t_1 and t_2, it attempts to incrementally adjust the data mining results at time stamp t_1 by taking advantage of the results from t_1 and the small perturbations between t_1 and t_2. Specifically, we build our model on the basis of the previously proposed NetFS framework and employ the temporal smoothness in updating the feature selection results incrementally. Also,

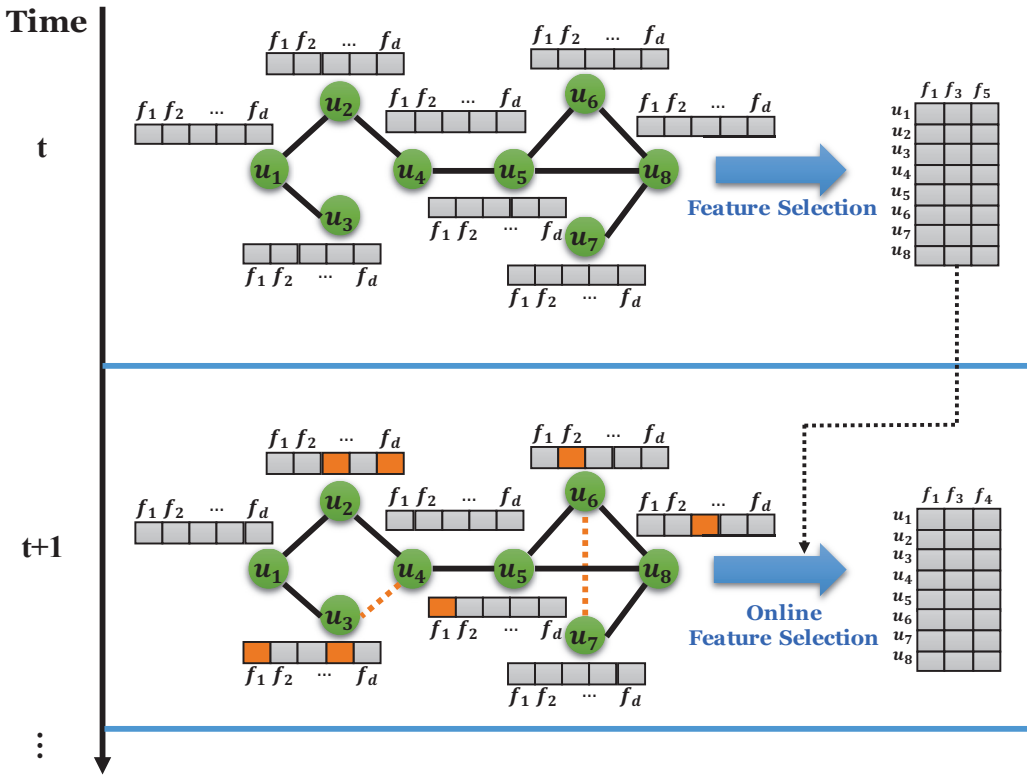


Figure 5.1: An Illustration of the Online Feature Selection Framework TeFS.

we present an efficient optimization schema to solve the optimization problem of the proposed TeFS. An illustration of the proposed TeFS is shown in Figure 5.1.

5.2.1 *Problem Formulation*

Here, we present the detailed formulation of the proposed TeFS framework of online feature selection on dynamic attributed networks.

Modeling the Temporal Smoothness for Online Feature Selection. Regarding the temporal smoothness assumption, at a new time stamp $t+1$, both network structure and feature information evolve smoothly, i.e., $\|\mathbf{X}^{t+1}-\mathbf{X}^{t}\|_0$ and $\|\mathbf{A}^{t+1}-\mathbf{A}^{t}\|_0$ are very small. Therefore, we have:

$$\mathbf{X}^{t+1} = \mathbf{X}^{t} + \Delta\mathbf{X}, \quad \mathbf{A}^{t+1} = \mathbf{A}^{t} + \Delta\mathbf{A}, \tag{5.1}$$

where $\Delta\mathbf{X}$ and $\Delta\mathbf{A}$ indicate the changes between t and $t+1$, respectively. Then the feature selection problem at the new time stamp $t+1$ can be formulated by solving the following optimization problem:

$$\begin{aligned}\min_{\mathbf{U}^{t+1}\geq 0,\mathbf{W}^{t+1}} \mathcal{J}(\mathbf{W}^{t+1},\mathbf{U}^{t+1}) =&\|(\mathbf{X}^t+\Delta\mathbf{X})\mathbf{W}^{t+1}-\mathbf{U}^{t+1}\|_F^2+\alpha\|\mathbf{W}^{t+1}\|_{2,1}\\ &+\frac{\beta}{2}\|(\mathbf{A}^t+\Delta\mathbf{A})-\mathbf{U}^{t+1}\mathbf{U}^{t+1\prime}\|_F^2.\end{aligned} \tag{5.2}$$

Let $\mathbf{W}^{t+1}$ and $\mathbf{U}^{t+1}$ be represented as:

$$\mathbf{W}^{t+1}=\mathbf{W}^t+\Delta\mathbf{W},\quad \mathbf{U}^{t+1}=\mathbf{U}^t+\Delta\mathbf{U}, \tag{5.3}$$

then Eq. (5.2) can be reformulated as:

$$\begin{aligned}\min_{\mathbf{U}^t+\Delta\mathbf{U}\geq 0,\mathbf{W}^t+\Delta\mathbf{W}}&\|(\mathbf{X}^t+\Delta\mathbf{X})(\mathbf{W}^t+\Delta\mathbf{W})-(\mathbf{U}^t+\Delta\mathbf{U})\|_F^2\\ &+\frac{\beta}{2}\|(\mathbf{A}^t+\Delta\mathbf{A})-(\mathbf{U}^t+\Delta\mathbf{U})(\mathbf{U}^t+\Delta\mathbf{U})'\|_F^2+\alpha\|\mathbf{W}^t+\Delta\mathbf{W}\|_{2,1}.\end{aligned} \tag{5.4}$$

Lemma 2. *$\ell_{2,1}$-norm on matrix is a valid norm and it satisfies the triangle inequality* $\|\mathbf{A}+\mathbf{B}\|_{2,1}\leq\|\mathbf{A}\|_{2,1}+\|\mathbf{B}\|_{2,1}$.

Proof. The ℓ_p-norm of a vector $\mathbf{x}\in\mathbb{R}^n$ is defined as $\|\mathbf{x}\|_p=(\sum_{i=1}^n\|x_i\|^p)^{1/p}$ and it satisfies the triangle inequality such that $(\sum_{i=1}^n\|x_i+y_i\|^p)^{1/p}\leq(\sum_{i=1}^n\|x_i\|^p)^{1/p}+(\sum_{i=1}^n\|y_i\|^p)^{1/p}$ for any vectors $\mathbf{x}\in\mathbb{R}^n$ and $\mathbf{y}\in\mathbb{R}^n$. By setting $x_i=\|\mathbf{a}_i\|_2$ and $y_i=\|\mathbf{b}_i\|_2$, we obtain the following inequality:

$$\big(\sum_{i=1}^n|\|\mathbf{a}_i+\mathbf{b}_i\|_2|^p\big)^{1/p}\leq\big(\sum_{i=1}^n|\|\mathbf{a}_i\|_2+\|\mathbf{b}_i\|_2|^p\big)^{1/p}\leq\big(\sum_{i=1}^n\|\mathbf{a}_i\|^p\big)^{1/p}+\big(\sum_{i=1}^n\|\mathbf{b}_i\|^p\big)^{1/p}, \tag{5.5}$$

which is equivalent to $\|\mathbf{A}+\mathbf{B}\|_{2,1}\leq\|\mathbf{A}\|_{2,1}+\|\mathbf{B}\|_{2,1}$ when $p=1$. ■

According to triangle inequality of Frobenius norm and the above lemma, we have the following from Eq. (5.4):

$$\begin{aligned}&\|\mathbf{X}^t\mathbf{W}^t+\Delta\mathbf{X}(\mathbf{W}^t+\Delta\mathbf{W})+\mathbf{X}^t\Delta\mathbf{W}-\mathbf{U}^t-\Delta\mathbf{U}\|_F^2\\ \leq&\|\Delta\mathbf{X}\mathbf{W}^t+\mathbf{X}^t\Delta\mathbf{W}+\Delta\mathbf{X}\Delta\mathbf{W}-\Delta\mathbf{U}\|_F^2+\|\mathbf{X}^t\mathbf{W}^t-\mathbf{U}^t\|_F^2,\end{aligned} \tag{5.6}$$

$$\begin{aligned}&\|\mathbf{A}^t + \Delta\mathbf{A} - \mathbf{U}^t(\mathbf{U}^{t'} + \Delta\mathbf{U}') - \Delta\mathbf{U}(\mathbf{U}^{t'} + \Delta\mathbf{U}')\|_F^2 \\ \leq &\|\Delta\mathbf{A} - \Delta\mathbf{U}\mathbf{U}^{t'} - \mathbf{U}^t\Delta\mathbf{U}' - \Delta\mathbf{U}\Delta\mathbf{U}'\|_F^2 + \|\mathbf{A}^t - \mathbf{U}^t\mathbf{U}^{t'}\|_F^2,\end{aligned} \tag{5.7}$$

and

$$\|\mathbf{W}^t + \Delta\mathbf{W}\|_{2,1} \leq \|\mathbf{W}^t\|_{2,1} + \|\Delta\mathbf{W}\|_{2,1}. \tag{5.8}$$

Integrating Eq. (5.6), Eq. (5.7) and Eq. (5.8), we find that the solutions of $\mathbf{W}^t$ and $\mathbf{U}^t$ in the following part

$$\min_{\mathbf{W}^t, \mathbf{U}^t \geq 0} \mathcal{J}(\mathbf{W}^t, \mathbf{U}^t) = \|\mathbf{X}^t\mathbf{W}^t - \mathbf{U}^t\|_F^2 + \frac{\beta}{2}\|\mathbf{A}^t - \mathbf{U}^t\mathbf{U}^{t'}\|_F^2 + \alpha\|\mathbf{W}^t\|_{2,1} \tag{5.9}$$

can be obtained from the previous time stamp t. Therefore, we can only optimize the following part to approximate the solution of Eq. (5.4):

$$\begin{aligned}&\min_{\mathbf{U}^t + \Delta\mathbf{U} \geq 0, \Delta\mathbf{W}} \|\Delta\mathbf{X}\mathbf{W}^t + \mathbf{X}^t\Delta\mathbf{W} + \Delta\mathbf{X}\Delta\mathbf{W} - \Delta\mathbf{U}\|_F^2 \\ &+\frac{\beta}{2}\|\Delta\mathbf{A} - \Delta\mathbf{U}\mathbf{U}^{t'} - \mathbf{U}^t\Delta\mathbf{U}' - \Delta\mathbf{U}\Delta\mathbf{U}'\|_F^2 + \alpha\|\Delta\mathbf{W}\|_{2,1}.\end{aligned} \tag{5.10}$$

In the above objective function, the model parameters are the perturbation variables $\Delta\mathbf{W}$ and $\Delta\mathbf{U}$.

5.2.2 Optimization Solution

The objective function in Eq. (5.10) is also not a convex function w.r.t. $\Delta\mathbf{W}$ and $\Delta\mathbf{U}$ simultaneously. Thus we adopt the alternating optimization algorithm as NetFS (Li *et al.*, 2016b) to solve this problem. After the objective function converges to a local optimal solution, we sort the features in descending order according to the new transformation matrix $\mathbf{W}^t + \Delta\mathbf{W}$ and return top-m ranked features. Typically, the larger the value $\|(\mathbf{W}^t + \Delta\mathbf{W})(i,:)\|_2$ is, the more important the i-th feature is.

5.2.3 Time Complexity Analysis

The detailed online feature selection framework TeFS that performs feature selection in an incremental manner at each time stamp is illustrated as follows. We first obtain $\mathbf{W}^t$ and $\mathbf{U}^t$ at the first time stamp t by NetFS for a good initialization. Then in each iteration, we first fix $\Delta\mathbf{U}$ to update $\Delta\mathbf{W}$ which needs $O(d^3)$ operations. The term of $\Delta\mathbf{XW}$ can be pre-computed before iteration to improve the computational efficiency. Assume that the total number of nonzero elements in $\Delta\mathbf{U} - \Delta\mathbf{XW}$ and $\mathbf{U}^t$ are $\tilde{a}$ and a, respectively. Then the number of operations to obtain $\Delta\mathbf{W}$ is $O(nd^2 + \tilde{a}d)$. Otherwise, the cost is $O(nd^2 + ad)$ if we rerun NetFS. Next, we fix $\Delta\mathbf{W}$ to update $\Delta\mathbf{U}$, the major computational cost involves in the computation of the gradient for the projected gradient descent method. Suppose the total number of nonzero elements in $\Delta\mathbf{A}$ and $\mathbf{A}^t$ are $\tilde{b}$ and b, respectively. The computation cost of gradient of $\mathcal{J}(\Delta\mathbf{W}, \Delta\mathbf{U})$ w.r.t. $\Delta\mathbf{U}$ is $O(nc^2 + \tilde{b}c + n^2d + n\tilde{a})$. On the other hand, if we do not choose the incremental method, the cost of obtaining gradient is $O(nc^2 + bc + n^2d + na)$. Since $\tilde{a} < a$ and $\tilde{b} \ll b$, TeFS is more efficient than its offline counterpart NetFS.

5.3 Experimental Evaluation

We conduct experiments to assess the performance of the proposed TeFS. In particular, we attempt to answer the following two research questions: (1) How is the quality of selected features by TeFS? (2) How efficiency is the proposed online feature selection framework TeFS compared with its offline version NetFS?

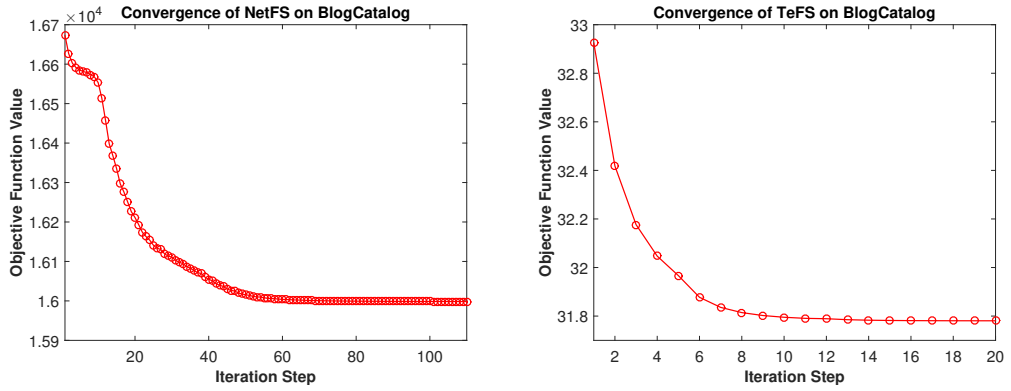
Datasets. The experiments are conducted on the three datasets BlogCatalog, Flickr, and Epinions introduced in section 3.3. It should be noted that the evolution of network structure and node attributes of these datasets are all very smooth.

Table 5.1: Clustering Results Comparison Between TeFS and NetFS.

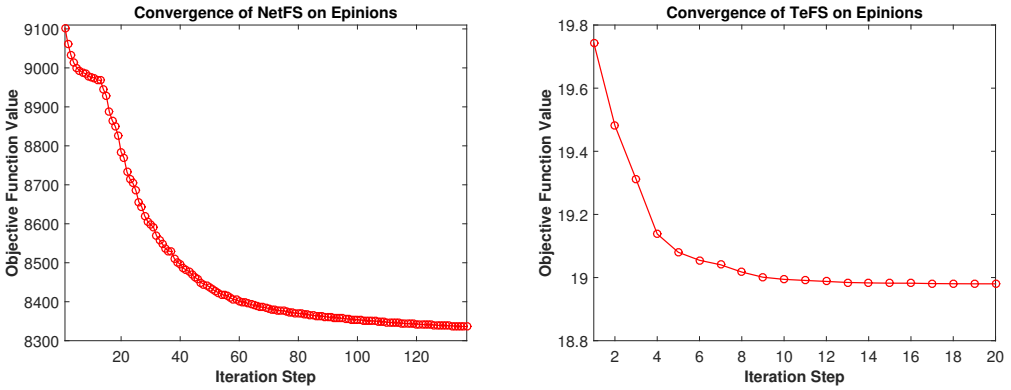
# of features	200	400	600	800	1000	1200	1400	1600	1800	2000
BlogCatalog – ACC (%)										
NetFS	49.99	43.04	43.25	42.89	42.09	43.56	43.44	43.31	43.08	43.59
TeFS	50.13	42.70	42.46	42.96	43.35	42.45	43.20	43.42	43.27	44.18
Flickr – ACC (%)										
NetFS	23.04	31.52	33.60	36.21	35.52	42.56	46.46	41.35	47.42	35.78
TeFS	23.14	31.40	34.28	35.49	35.28	41.89	45.88	42.21	46.90	36.17
Epinions – ACC (%)										
NetFS	14.04	16.66	18.27	20.48	20.46	20.98	24.82	23.79	23.91	27.32
TeFS	14.03	16.87	18.30	20.29	20.70	21.06	24.79	23.62	23.94	27.78
BlogCatalog – NMI (%)										
NetFS	33.21	23.45	24.04	23.18	23.83	23.63	25.49	25.98	23.85	24.72
TeFS	34.72	24.03	22.89	23.48	25.25	22.71	25.40	25.75	24.22	26.00
Flickr – NMI (%)										
NetFS	11.61	16.55	20.28	20.56	21.67	23.38	26.54	25.40	28.91	25.42
TeFS	11.92	15.89	21.25	20.89	21.34	22.48	27.51	25.46	29.28	25.29
Epinions – NMI (%)										
NetFS	3.87	5.80	6.91	6.93	8.92	10.04	10.83	11.45	11.65	10.26
TeFS	4.56	5.76	6.55	6.82	8.99	11.30	10.47	12.42	11.48	10.75

Experimental Settings. Firstly, we follow the same evaluation mechanism as NetFS to assess the quality of selected features. The average clustering performance on the selected features at different time stamps are presented. Here, we mainly compare the performance of TeFS and NetFS as NetFS have already shown to outperform many other baseline methods. Secondly, to assess the efficiency of the proposed online framework TeFS, we record its cumulative running time over all time stamps and compare it with its offline counterpart NetFS that reruns at each time stamp. In the experiments, we set $\alpha = 10$ and $\beta = 0.1$.

Effectiveness Evaluation. We first compare the clustering performance between



(a) NetFS on BlogCatalog. (b) TeFS on BlogCatalog.



(c) NetFS on Epinions. (d) TeFS on Epininons.

Figure 5.2: Convergence Rate Comparison Between NetFS and TeFS.

the online method TeFS and the offline method NetFS. The comparison results are shown in Table 5.1. We can also find that TeFS achieves comparable clustering performance as NetFS. It indicates that even though we adopt an approximation method to reduce the computational cost, it does not bring any negative effects by jeopardizing the clustering performance.

Efficiency Evaluation. In this part, we first investigate the convergence rate of TeFS and its offline version NetFS. Figure 5.2 shows the convergence comparison results on BlogCatalog and Epinions datasets at a specific time stamp. It can be clearly seen from the figure that the objective function value of TeFS decreases and

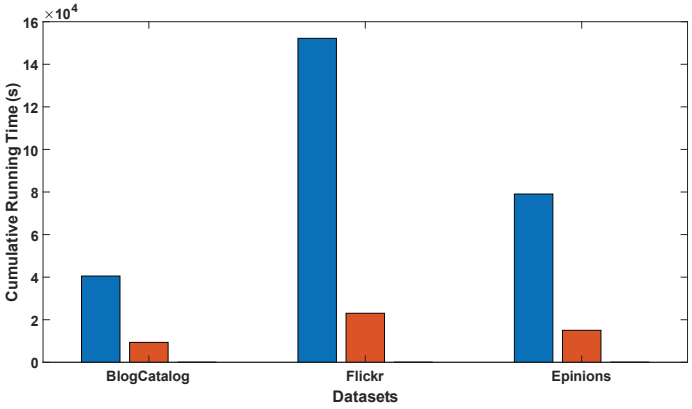


Figure 5.3: Cumulative Running Time Comparison Between NetFS and TeFS.

reaches a stable state much more quickly than its offline version NetFS. TeFS converges within 20 iterations while NetFS needs more than 100 iterations to converge. This observation suggests that TeFS has a faster convergence rate. Also, we compare their cumulative running time on all these three dynamic attributed datasets. The cumulative running time comparison results are shown in Figure 5.3, the results show that the proposed TeFS is significantly more efficient than NetFS, the speedup is 4.3×, 6.6× and 5.3× on BlogCatalog, Flickr, and Epinions, respectively.

5.4 Summary

In this chapter, we investigate the attributed networks in a dynamic environment and propose an online unsupervised feature selection framework TeFS for real-time insights. The proposed framework takes both the content drift and the network structure changes into account to find relevant features in an online manner. In particular, the proposed framework is based on the temporal smoothness property of dynamic networks. Instead of rerunning the offline model from scratch each time stamp when variations happen, we propose an efficient way to update the feature selection results from previous time stamps incrementally. Experimental results on

real-world dynamic attributed networks validate the effectiveness and efficiency of the proposed online feature selection framework TeFS.

Chapter 6

NETWORK EMBEDDING ON DYNAMIC ATTRIBUTED NETWORKS

Attributed network embedding seeks low-dimensional node vector representations in which the network topological structure and node attribute proximity can be maximally preserved. Despite its empirical success, a fundamental assumption behind existing efforts is that the data is static and given a prior. Nonetheless, most real-world attributed networks are intrinsically dynamic with both structure and content changes. These changing characteristics motivate us to seek an effective yet efficient embedding presentation to capture the evolving patterns timely, which is of fundamental importance for learning in a dynamic environment.

6.1 Overview

More often than not, attributed networks often exhibit high dynamics. On one hand, we often observe the addition/deletion of edges and nodes, examples include co-author relations between scholars in an academic network and friendships among users in a social network. Meanwhile, node attributes also change naturally such that new content patterns may emerge and outdated content patterns will fade. For example, humanitarian and disaster relief related topics become popular on social media sites after the earthquakes as users continuously post related content. One natural question to ask is when attributed networks evolve, how to correct and adjust the staleness of the end embedding results for network analysis, which will shed light on the understanding of their evolving nature. Hence, it necessitates the design of an efficient online algorithm that can give embedding representations promptly. The studied problem is formulated as follows.

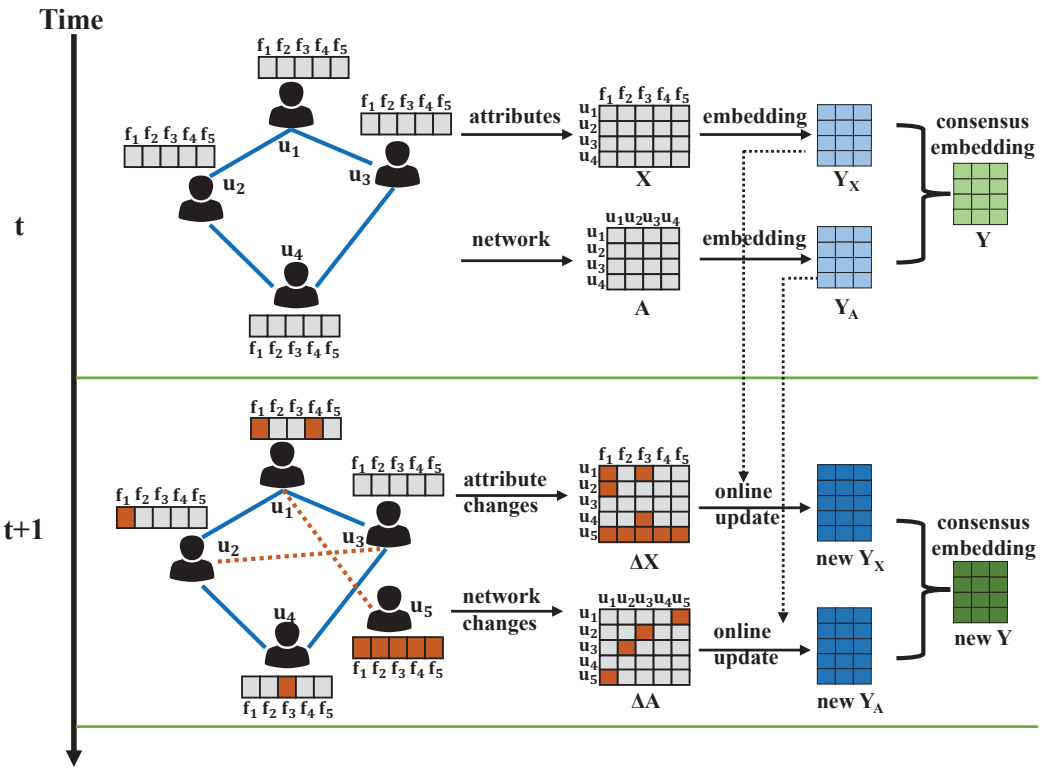


Figure 6.1: An Illustration of the Online Embedding Framework DANE.

Problem 4. Network Embedding for Dynamic Attributed Networks

Given: *A dynamic attributed network that is observed at a series of time stamps* t, $t+1$, ..., $t+i$, *where the node feature information is represented by a set of content matrices* $\{\mathbf{X}^t, \mathbf{X}^{t+1}, ..., \mathbf{X}^{t+i}\}$ *and network structure is encoded in a set of adjacency matrices* $\{\mathbf{A}^t, \mathbf{A}^{t+1}, ..., \mathbf{A}^{t+i}\}$. *The dimensionality of final consensus node embeddings is specified as* l.

Learn: *Embedding representation* $\mathbf{Y}^t \in \mathbb{R}^{n \times l}$ *for all nodes at each time stamp.*

6.2 Proposed Online Framework - DANE

The proposed online embedding model DANE is motivated by the observation that most of the real-world networks, with no exception for attributed networks, often

evolve smoothly in the temporal dimension between two consecutive time stamps (Aggarwal and Subbian, 2014; Chi *et al.*, 2007; Li *et al.*, 2016a). Previously in Chapter 4, we have presented an effective consensus embedding framework for static attributed networks, which boils down to solving a series of generalized eigen-problems. Therefore, the core idea to enable online update of the embeddings is to develop an efficient way to update the top eigenvectors and eigenvalues of the generalized eigensystem. Otherwise, we have to perform generalized eigen-decomposition each time stamp, which is not practical due to its high time complexity. The workflow of the proposed online embedding model is shown in Figure 6.1.

6.2.1 Problem Formulation

Without loss of generality, we use the network topology as an example to illustrate the proposed algorithm for online embedding.

We use $\Delta\mathbf{A}$ and $\Delta\mathbf{X}$ to denote the perturbation of network structure and node attributes between two consecutive time stamps t and $t+1$, respectively. With these, the diagonal matrix and Laplacian matrix of $\mathbf{A}$ and $\mathbf{X}$ also evolve smoothly such that:

$$\begin{aligned} \mathbf{D}_{\mathbf{A}}^{t+1} &= \mathbf{D}_{\mathbf{A}}^{t} + \Delta\mathbf{D}_{\mathbf{A}}, \quad \mathbf{L}_{\mathbf{A}}^{t+1} = \mathbf{L}_{\mathbf{A}}^{t} + \Delta\mathbf{L}_{\mathbf{A}}, \\ \mathbf{D}_{\mathbf{X}}^{t+1} &= \mathbf{D}_{\mathbf{X}}^{t} + \Delta\mathbf{D}_{\mathbf{X}}, \quad \mathbf{L}_{\mathbf{X}}^{t+1} = \mathbf{L}_{\mathbf{X}}^{t} + \Delta\mathbf{L}_{\mathbf{X}}. \end{aligned} \tag{6.1}$$

By the matrix perturbation theory (Stewart and Sun, 1990), we have the following equation in embedding the network structure at the new time stamp:

$$(\mathbf{L}_{\mathbf{A}}^{t} + \Delta\mathbf{L}_{\mathbf{A}})(\mathbf{a} + \Delta\mathbf{a}) = (\lambda + \Delta\lambda)(\mathbf{D}_{\mathbf{A}}^{t} + \Delta\mathbf{D}_{\mathbf{A}})(\mathbf{a} + \Delta\mathbf{a}). \tag{6.2}$$

For a specific eigen-pair $(\lambda_i, \mathbf{a}_i)$, we have the following equation:

$$(\mathbf{L}_{\mathbf{A}}^{t} + \Delta\mathbf{L}_{\mathbf{A}})(\mathbf{a}_i + \Delta\mathbf{a}_i) = (\lambda_i + \Delta\lambda_i)(\mathbf{D}_{\mathbf{A}}^{t} + \Delta\mathbf{D}_{\mathbf{A}})(\mathbf{a}_i + \Delta\mathbf{a}_i). \tag{6.3}$$

The problem now is how to compute the change of the i-th eigen-pair $(\Delta\mathbf{a}_i, \Delta\lambda_i)$ by taking advantage of the sparse perturbation matrices $\Delta\mathbf{D}$ and $\Delta\mathbf{L}$.

Computing eigenvalue change $\Delta\lambda_i$. By expanding the above equation, we have:

$$\begin{aligned}\mathbf{L}_\mathbf{A}^t\mathbf{a}_i + \Delta\mathbf{L}_\mathbf{A}\mathbf{a}_i + \mathbf{L}_\mathbf{A}^t\Delta\mathbf{a}_i + \Delta\mathbf{L}_\mathbf{A}\Delta\mathbf{a}_i \;=\; \lambda_i\mathbf{D}_\mathbf{A}^t\mathbf{a}_i + \lambda_i\Delta\mathbf{D}_\mathbf{A}\mathbf{a}_i + \Delta\lambda_i\mathbf{D}_\mathbf{A}^t\mathbf{a}_i \\ + \Delta\lambda_i\Delta\mathbf{D}_\mathbf{A}\mathbf{a}_i + (\lambda_i\mathbf{D}_\mathbf{A}^t + \lambda_i\Delta\mathbf{D}_\mathbf{A} + \Delta\lambda_i\mathbf{D}_\mathbf{A}^t + \Delta\lambda_i\Delta\mathbf{D}_\mathbf{A})\Delta\mathbf{a}_i.\end{aligned} \tag{6.4}$$

The higher order terms, i.e., $\Delta\lambda_i\Delta\mathbf{D}_\mathbf{A}\mathbf{a}_i$, $\lambda_i\Delta\mathbf{D}_\mathbf{A}\Delta\mathbf{a}_i$, $\Delta\lambda_i\mathbf{D}_\mathbf{A}^t\Delta\mathbf{a}_i$, and $\Delta\lambda_i\Delta\mathbf{D}_\mathbf{A}\Delta\mathbf{a}_i$ can be removed as they have limited effects on the accuracy of the generalized eigensystem (Golub and Van Loan, 2012). With the fact that $\mathbf{L}_\mathbf{A}^t\mathbf{a}_i = \lambda_i\mathbf{D}_\mathbf{A}^t\mathbf{a}_i$, we have:

$$\Delta\mathbf{L}_\mathbf{A}\mathbf{a}_i + \mathbf{L}_\mathbf{A}^t\Delta\mathbf{a}_i = \lambda_i\Delta\mathbf{D}_\mathbf{A}\mathbf{a}_i + \Delta\lambda_i\mathbf{D}_\mathbf{A}^t\mathbf{a}_i + \lambda_i\mathbf{D}_\mathbf{A}^t\Delta\mathbf{a}_i. \tag{6.5}$$

Multiplying both sides with $\mathbf{a}_i'$, we now have:

$$\mathbf{a}_i'\Delta\mathbf{L}_\mathbf{A}\mathbf{a}_i + \mathbf{a}_i'\mathbf{L}_\mathbf{A}^t\Delta\mathbf{a}_i = \lambda_i\mathbf{a}_i'\Delta\mathbf{D}_\mathbf{A}\mathbf{a}_i + \Delta\lambda_i\mathbf{a}_i'\mathbf{D}_\mathbf{A}^t\mathbf{a}_i + \lambda_i\mathbf{a}_i'\mathbf{D}_\mathbf{A}^t\Delta\mathbf{a}_i. \tag{6.6}$$

Since both $\mathbf{L}_\mathbf{A}^t$ and $\mathbf{D}_\mathbf{A}^t$ are symmetric, we have $\mathbf{a}_i'\mathbf{L}_\mathbf{A}^t\Delta\mathbf{a}_i = \lambda_i\mathbf{a}_i'\mathbf{D}_\mathbf{A}^t\Delta\mathbf{a}_i$. Therefore, Eq. (6.6) can be reformulated as follows:

$$\mathbf{a}_i'\Delta\mathbf{L}_\mathbf{A}\mathbf{a}_i = \lambda_i\mathbf{a}_i'\Delta\mathbf{D}_\mathbf{A}\mathbf{a}_i + \Delta\lambda_i\mathbf{a}_i'\mathbf{D}_\mathbf{A}^t\mathbf{a}_i. \tag{6.7}$$

Through this, the variation of eigenvalue, i.e., $\Delta\lambda_i$, is:

$$\Delta\lambda_i = \frac{\mathbf{a}_i'\Delta\mathbf{L}_\mathbf{A}\mathbf{a}_i - \lambda_i\mathbf{a}_i'\Delta\mathbf{D}_\mathbf{A}\mathbf{a}_i}{\mathbf{a}_i'\mathbf{D}_\mathbf{A}^t\mathbf{a}_i}. \tag{6.8}$$

Lemma 3. *In the generalized eigen-problem $\mathbf{A}\mathbf{v} = \lambda\mathbf{B}\mathbf{v}$, if $\mathbf{A}$ and $\mathbf{B}$ are both Hermitian matrices and $\mathbf{B}$ is positive semidefinite, the eigenvalue λ are real; and eigenvectors are $\mathbf{B}$-orthogonal, i.e., $\mathbf{v}_i'\mathbf{B}\mathbf{v}_j = 0$ and $\mathbf{v}_i'\mathbf{B}\mathbf{v}_i = 1$ $(i \neq j)$ (Parlett, 1980).*

Corollary 1. $\mathbf{a}_i'\mathbf{D}_\mathbf{A}^t\mathbf{a}_i = 1$ *and* $\mathbf{a}_i'\mathbf{D}_\mathbf{A}^t\mathbf{a}_j = 0$ $(i \neq j)$.

Proof. Both $\mathbf{D}_\mathbf{A}^t$ and $\mathbf{L}_\mathbf{A}^t$ are symmetric and are also Hermitian matrices. Meanwhile, the Laplacian matrix $\mathbf{L}_\mathbf{A}^t$ is a positive semidefinite matrix, which completes the proof. ■

Therefore, the variation of the eigenvalue λ_i is as follows:

$$\Delta\lambda_i = \mathbf{a}_i'\Delta\mathbf{L}_\mathbf{A}\mathbf{a}_i - \lambda_i\mathbf{a}_i'\Delta\mathbf{D}_\mathbf{A}\mathbf{a}_i. \tag{6.9}$$

Computing eigenvector change $\Delta\mathbf{a}_i$. As the network structure often evolves smoothly between two consecutive time stamps, we assume that the perturbation of the eigenvectors $\Delta\mathbf{a}_i$ lies in the column space that is composed by the top-k eigenvectors at time stamp t such that $\Delta\mathbf{a}_i = \sum_{j=2}^{k+1}\alpha_{ij}\mathbf{a}_j$, where α_{ij} is a weight indicating the contribution of the j-th eigenvector $\mathbf{a}_j$ in approximating the new i-th eigenvector. Next, we show how to determine these weights such that the perturbation $\Delta\mathbf{a}_i$ can be estimated. By plugging $\Delta\mathbf{a}_i = \sum_{j=2}^{k+1}\alpha_{ij}\mathbf{a}_j$ into Eq. (6.5) and using the fact that $\mathbf{L}_\mathbf{A}^t\sum_{j=2}^{k+1}\alpha_{ij}\mathbf{a}_j = \mathbf{D}_\mathbf{A}^t\sum_{j=2}^{k+1}\alpha_{ij}\lambda_j\mathbf{a}_j$, we obtain the following:

$$\Delta\mathbf{L}_\mathbf{A}\mathbf{a}_i + \mathbf{D}_\mathbf{A}^t\sum_{j=2}^{k+1}\alpha_{ij}\lambda_j\mathbf{a}_j = \lambda_i\Delta\mathbf{D}_\mathbf{A}\mathbf{a}_i + \Delta\lambda_i\mathbf{D}_\mathbf{A}^t\mathbf{a}_i + \lambda_i\mathbf{D}_\mathbf{A}^t\sum_{j=2}^{k+1}\alpha_{ij}\mathbf{a}_j. \tag{6.10}$$

By multiplying eigenvector $\mathbf{a}_p'\,(2 \le p \le k+1, p \ne i)$ on both sides of Eq. (6.10), and with the orthonormal property from Corollary 6.2.1, we obtain the following:

$$\begin{aligned}&\mathbf{a}_p'\Delta\mathbf{L}_\mathbf{A}\mathbf{a}_i + \mathbf{a}_p'\mathbf{D}_\mathbf{A}^t\sum_{j=2}^{k+1}\alpha_{ij}\lambda_j\mathbf{a}_j = \lambda_i\mathbf{a}_p'\Delta\mathbf{D}_\mathbf{A}\mathbf{a}_i + \Delta\lambda_i\mathbf{a}_p'\mathbf{D}_\mathbf{A}^t\mathbf{a}_i + \lambda_i\mathbf{a}_p'\mathbf{D}_\mathbf{A}^t\sum_{j=2}^{k+1}\alpha_{ij}\mathbf{a}_j\\ \Rightarrow\quad &\mathbf{a}_p'\Delta\mathbf{L}_\mathbf{A}\mathbf{a}_i + \alpha_{ip}\lambda_p = \lambda_i\mathbf{a}_p'\Delta\mathbf{D}_\mathbf{A}\mathbf{a}_i + \alpha_{ip}\lambda_i.\end{aligned} \tag{6.11}$$

Hence, the weight α_{ip} can be determined by:

$$\alpha_{ip} = \frac{\mathbf{a}_p'\Delta\mathbf{L}_\mathbf{A}\mathbf{a}_i - \lambda_i\mathbf{a}_p'\Delta\mathbf{D}_\mathbf{A}\mathbf{a}_i}{\lambda_i - \lambda_p}. \tag{6.12}$$

After eigenvector perturbation, we still need to make the orthonormal condition holds for new eigenvectors, thus we have $(\mathbf{a}_i + \Delta\mathbf{a}_i)'(\mathbf{D}_\mathbf{A} + \Delta\mathbf{D}_\mathbf{A})(\mathbf{a}_i + \Delta\mathbf{a}_i) = 1$. By expanding it and removing the second-order and third-order terms, we obtain:

$$2\mathbf{a}_i'\mathbf{D}_\mathbf{A}^t\Delta\mathbf{a}_i + \mathbf{a}_i'\Delta\mathbf{D}_\mathbf{A}^t\mathbf{a}_i = 0. \tag{6.13}$$

Then the solution of α_{ii} is $\alpha_{ii} = -\frac{1}{2}\mathbf{a}_i'\Delta\mathbf{D_A}\mathbf{a}_i$. With the above solutions, the perturbation of eigenvector $\mathbf{a}_i$ is given as follows:

$$\Delta\mathbf{a}_i = -\frac{1}{2}\mathbf{a}_i'\Delta\mathbf{D_A}\mathbf{a}_i\mathbf{a}_i + \sum_{j=2,j\neq i}^{k+1}(\frac{\mathbf{a}_j'\Delta\mathbf{L_A}\mathbf{a}_i - \lambda_i\mathbf{a}_j'\Delta\mathbf{D_A}\mathbf{a}_i}{\lambda_i - \lambda_j})\mathbf{a}_j. \tag{6.14}$$

Overall, the i-th eigen-pair $(\Delta\lambda_i, \Delta\mathbf{a}_i)$ can be updated on the fly by Eq. (6.9) and Eq. (6.14), the pseudocode of the updating process is illustrated in Algorithm 3. The first input is the top-k eigen-pairs of the generalized eigen-problem, and they can be computed by standard methods like power iteration and Lanczos method (Golub and Van Loan, 2012). Another input is the variation of the diagonal matrix and the Laplacian matrix. For the top-k eigen-pairs, we update eigenvalues in Line 2 and update eigenvectors in Line 3.

Algorithm 3 Updating of Embedding Results of the Network by DANE.

Input: Top-k eigen-pairs of the generalized eigen-problem $\{(\lambda_2, \mathbf{a}_2),...,(\lambda_{k+1}, \mathbf{a}_{k+1})\}$ at time t, variation of the diagonal matrix $\Delta\mathbf{L_A}$ and Laplacian matrix $\Delta\mathbf{D_A}$.

Output: Top-k eigen-pairs $\{(\lambda_2^{(t+1)}, \mathbf{a}_2^{(t+1)}),...,(\lambda_{k+1}^{(t+1)}, \mathbf{a}_{k+1}^{(t+1)})\}$ at time step $t+1$.

1: **for** $i = 2$ to $k+1$ **do**
2: Calculate the variation of $\Delta\lambda_i$ by Eq. (6.9);
3: Calculate the variation of $\Delta\mathbf{a}_i$ by Eq. (6.14);
4: $\lambda_i^{(t+1)} = \lambda_i + \Delta\lambda_i$; $\mathbf{a}_i^{(t+1)} = \mathbf{a}_i + \Delta\mathbf{a}_i$;
5: **end for**

6.2.2 Time Complexity Analysis

Firstly, we need to compute the perturbation terms $\mathbf{\Delta A}$ and $\mathbf{\Delta X}$. $\mathbf{\Delta A}$ can be directly computed while $\mathbf{\Delta X}$ can also be computed efficiently in an incremental way by focusing on the nodes whose attributes or connections change. As both $\mathbf{\Delta A}$ and $\mathbf{\Delta X}$ are very sparse, it also enables us to compute $\mathbf{\Delta L_A}$, $\mathbf{\Delta L_X}$, $\mathbf{\Delta D_A}$, and

$\mathbf{\Delta D_X}$ efficiently. Later on, to update the top-k eigenvalues of the network and node attributes in an online fashion, it requires $O(k(d_a+l_a))$ and $O(k(d_x+l_x))$, respectively. Also, the online updating of the top-k eigenvectors for the network and attributes are $O(k^2(d_a + l_a + n))$ and $O(k^2(d_x + l_x + n))$, respectively. After that, the complexity for the consensus embedding is $O(k^2 l)$. Therefore, the computational complexity of the proposed online model over T time stamps are $O(Tk^2(n + l + l_a + l_x + d_a + d_x))$. As can be shown, since $\mathbf{\Delta L_A}$, $\mathbf{\Delta L_X}$, $\mathbf{\Delta D_A}$, and $\mathbf{\Delta D_X}$ are often very sparse, thus l_a, l_x, d_a, d_x are usually very small, meanwhile we have $k \ll n$ and $l \ll n$. Based on the above analysis, the proposed online embedding algorithm DANE is much more efficient than rerunning the offline method DANE-O repeatedly.

6.3 Experimental Evaluation

We evaluate the effectiveness and efficiency of the proposed online embedding framework DANE on dynamic attributed networks. In particular, we attempt to answer the following two research questions: (1) How effective are the embeddings obtained by DANE on different learning tasks? (2) How fast is the proposed framework DANE compared with other offline embedding methods?

Datasets. We use the same datasets that appear in section 4.3 for the evaluation. Among them, BlogCatalog and Flickr are semi-synthetic dynamic attributed networks while Epinions and DBLP are real-world dynamic attributed networks. On these datasets, the evolution is very small between two consecutive time stamps.

Experimental Settings. The experimental protocol is also the same as that in section 4.3. We first compare the embeddings learned by DANE and DANE-O on two different learning tasks - node classification and network clustering. Secondly, we also compare the cumulative running time between DANE and other baseline methods mentioned in section 4.3.

Table 6.1: Clustering Results Comparison Between DANE and DANE-O.

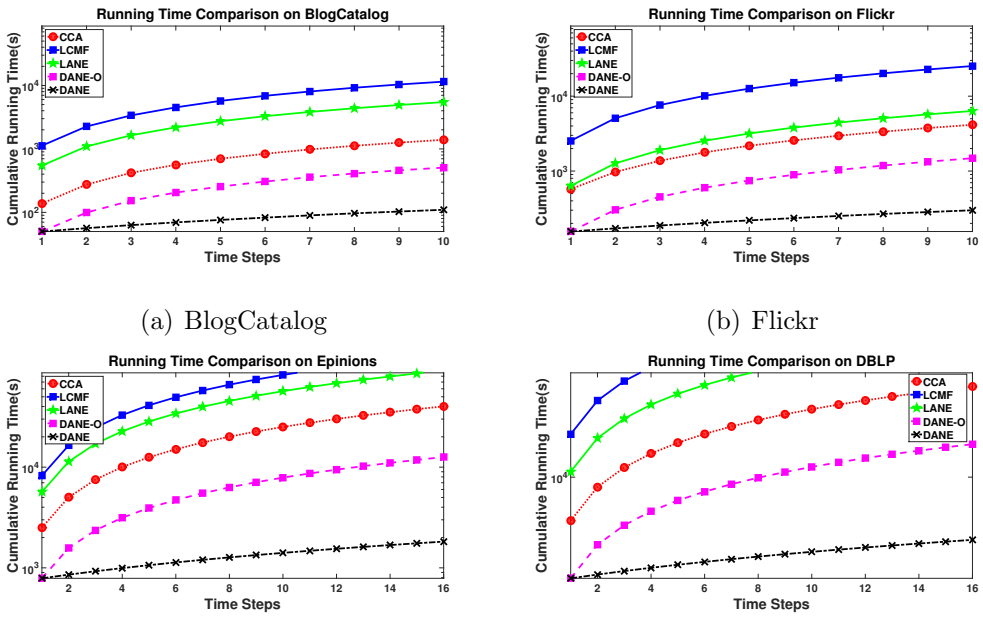
Datasets	BlogCatalog		Flickr		Epinions		DBLP	
Methods	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI
DANE-O	80.31	59.46	67.33	53.04	34.11	23.07	59.14	35.31
DANE	79.69	59.32	67.24	52.19	34.52	22.36	57.68	34.87

Table 6.2: Classification Results Comparison Between DANE and DANE-O.

Datasets	BlogCatalog			Flickr			Epinions			DBLP		
Methods	AC	Micro	Macro	AC	Micro	Macro	AC	Micro	Macro	AC	Micro	Macro
DANE-O	89.34	89.15	89.23	79.68	79.52	79.95	31.23	31.28	31.35	77.21	74.96	75.48
DANE	89.09	88.78	88.94	79.56	78.94	79.56	30.87	30.93	30.81	76.64	74.53	75.69

Effectiveness Evaluation. We compare the proposed DANE with the aforementioned offline framework DANE-O on the network clustering and node classification tasks. In particular, we need to rerun the offline method DANE-O at each time stamp and the average performance across different time stamps is compared. The comparison results are shown in Table 6.1 and Table 6.2. From these two tables, we can find that even though DANE leverages matrix perturbation theory to update the embedding representations, its performance is very close to DANE-O which reruns at each time stamp. It implies that the online embedding model does not sacrifice too much informative information in terms of embedding.

Efficiency Evaluation. We report the cumulative running time (in log scale) of different embedding methods in Figure 6.2. As can be observed, DANE is much faster than all these comparison methods. In all these datasets, it terminates within one hour while some offline methods need several hours or even days to run. It can also be shown that both DANE and DANE-O are much faster than all other of-

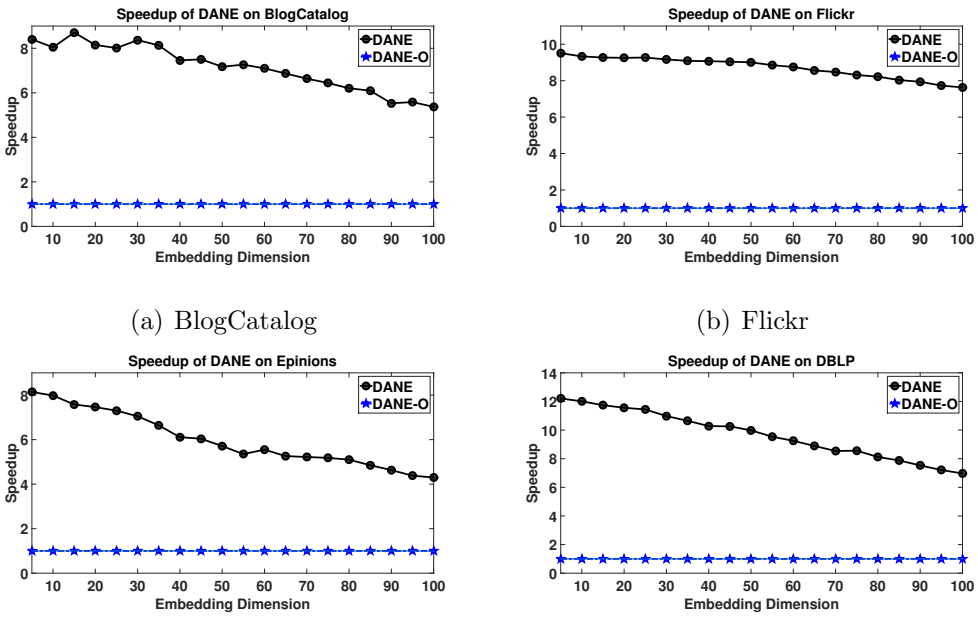


(a) BlogCatalog (b) Flickr

(c) Epinions (d) DBLP

Figure 6.2: Cumulative Running Time of Different Network Embedding Methods.

fline methods. To be more specific, for example, DANE is 84×, 21× and 14× faster than LCMF, CCA, and LANE respectively on Flickr dataset. To further investigate the superiority of DANE against its offline version DANE-O, we compare the speedup rate of DANE against DANE-O w.r.t. different embedding dimensions in Figure 6.3. As can be observed, when the embedding dimension is small (around 10), DANE achieves around 8×, 10×, 8×, 12× speedup on BlogCatalog, Flickr, Epinions, and DBLP, respectively. When the embedding dimensionality gradually increases, the speedup of DANE decreases, but it is still significantly faster than DANE-O. With all the above observations, we can draw a conclusion that the proposed DANE framework is able to learn informative embeddings for attributed networks efficiently without jeopardizing the classification and the clustering performance.



(a) BlogCatalog

(b) Flickr

(c) Epinions

(d) DBLP

Figure 6.3: Running Time Speedup of DANE Against DANE-O.

6.4 Summary

Real-world attributed networks are often not static such that interactions among networked instances tend to evolve gradually, and the attributes also change accordingly. In this chapter, we study a novel research problem: how to learn embedding representations for nodes on dynamic attributed networks in an online manner to enable downstream learning tasks. As the offline embedding at each time stamp boils down to solving a series of generalized eigen-problems, we argue that the core idea to enable online embedding learning is to incrementally update the solutions of generalized eigen-problems. In particular, we leverage the temporal smoothness properties of the evolution patterns and update the node embeddings on the fly with matrix perturbation theory. We also analyze the time complexity of the proposed framework and show its superiority over offline methods with experimental studies.

Part III

Applications

Chapter 7

PERSONALIZED RELATIONAL LEARNING ON ATTRIBUTED NETWORKS

We have discussed how to build general and one-size-fits-all feature learning solutions to enable learning on attributed networks. In this chapter, we focus on one important application on attributed networks - the relational learning problem and show how to characterize the inherent properties of the studied problem for a more customized solution. Relational learning exploits relationships among instances manifested in a network to improve the predictive performance of many network mining tasks, and it encompasses node classification as one of the central problems in the network domain[1]. In many cases, individuals in a network are highly idiosyncratic. They not only connect to each other with a composite of factors but also are often described by some content information of high dimensionality specific to each individual. Therefore, it would be more appealing to tailor the prediction for each individual while alleviating the issue related to the curse of dimensionality.

7.1 Overview

Inferring missing labels of nodes in a network could advance many real-world applications such as recommendation, personalized search, and crowdsourcing. However, the label information is rather limited on networks as the labeling process requires human attention and maybe very expensive; or itself is naturally unavailable due to some privacy issues. The limited access to label information necessities the usage of relational learning (Koller *et al.*, 2007; Singh and Gordon, 2008; Tang and Liu, 2009), which leverages the network structure that is readily available and a small subset of

[1]In this chapter, we use *relational learning* and *node classification* interchangeably.

labeled nodes to assign unlabeled nodes to some predefined groups.

Most, if not all, individuals in a network are highly idiosyncratic. To give a palpable understanding, we can observe that in social media, content information (e.g., blogs, posts, images) by different users could be quite diverse and personal, with a variety of foci. Also, user-generated content is often high-dimensional and may jeopardize the prediction performance on unseen nodes due to the curse of dimensionality (Li *et al.*, 2017a; Li and Liu, 2017). Therefore, it is desired to tailor the prediction for each node on the network with only a small subset of relevant features. In other words, for each instance, we would like to use a subset of discriminative personalized features in conjunction with some shared features for prediction, while these personalized features could vary for different nodes. Consequently, the model is interpretable as we can explain why we make such a prediction.

In this chapter, we study a novel problem of personalized relational learning on attributed networks. This problem has not been previously studied, mainly because of the following challenges: (1) As per the fact that labeled nodes are scarce while network structure is readily observed, it is indispensable to design a relational model such that nodes could borrow strength from its neighbors in building a more accurate predictive model. (2) Social identity theory (Tajfel, 2010) suggests that individuals in a network often exhibit different personalized patterns, but also, they more or less share some common behaviors to some extent. Relational learning should be able to seize these natures. (3) Traditional relational learning approaches often use a global pattern for the prediction purpose. Thus it is still not clear how to customize the learning and prediction for each individual node. The studied problem of personalized relational learning is formulated as follows.

Problem 5. Personalized Relational Learning

Given: *An attributed network G represented by the content matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ and the adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}_{\geq 0}$, where n and d denote the number of nodes and features, respectively. Among these n nodes, assume m nodes are labeled while the rest $n - m$ nodes are unlabeled. We use $\mathcal{Y} = \{c_1, c_2, ..., c_k\}$ to denote the label set of these nodes and $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_m]' \in \{0, 1\}^{m \times k}$ is the corresponding one-hot label indicator matrix for the labeled nodes, where the j-the element in $\mathbf{y}_i$ is 1 if the i-th node is associated with class label c_j, otherwise 0.*

Train: *A classifier to predict the missing labels for the unlabeled nodes. During the learning phase, we would like to tailor the learning process for each node by employing a subset of features locally associated with the node itself and a small subset of features relevant to all nodes.*

7.2 Proposed Framework - PRL

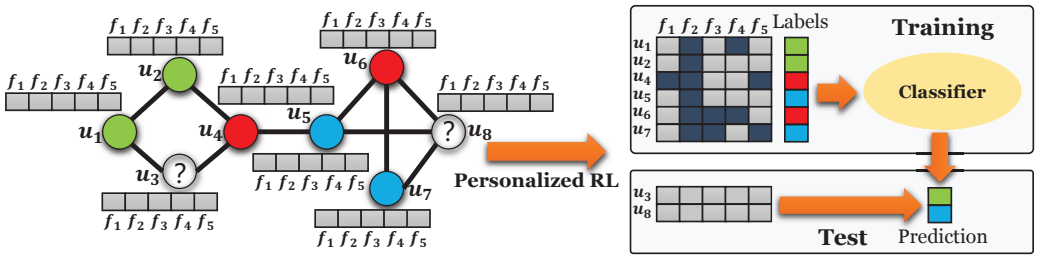


Figure 7.1: An Illustration of the Personalized Relational Learning Framework PRL.

In this section, we show how to build a personalized learning model to address relational learning problem in detail. The workflow of the proposed PRL framework is illustrated in Figure 7.1. From the figure, we can see that in the training phase, we have three sources of information, i.e., the network structure $\mathbf{A}$ for all nodes, feature matrix $\mathbf{X}$, and labels $\mathbf{Y}$ for labeled nodes. We first show how the proposed PRL framework finds some relevant features shared by all nodes (e.g., feature f_2) and also,

a small subset of discriminative features that are locally associated with each specific node (e.g., feature f_4 for u_1) to build a personalized predictive model, i.e., a classifier. Second, as label information is rather limited on real-world networks, we show how PRL makes use of rich network structure to make nodes borrow strength from each other to improve the prediction performance.

Modeling Node Features for Personalized Relational Learning. In order to infer the missing labels of unlabeled nodes, one simple and straightforward way is to build a global model for all nodes on the node features. However, one drawback is that it assumes that all nodes share the exact same patterns. In other words, it conjectures that all nodes share the same feature weight, and the feature weight derived from labeled nodes could be directly shifted to unlabeled nodes. Despite the fact that nodes in a network share some common patterns to some extent, they are often regarded as being highly idiosyncratic, showing distinct behaviors. The idiosyncrasy of nodes has been heavily observed in reality and also is supported by social identity theory (Tajfel, 2010) in sociology. It motivates us to build a predictive model to capture both global and personalized behaviors of nodes on the network. Next, we first introduce the framework to model the common node patterns and then extend it to model the personalized nature of each individual.

To uncover common behaviors shared by all nodes and to alleviate the curse of dimensionality, we embed feature selection into a linear multi-class classification model, resulting in the following objective function:

$$\min_{\tilde{\mathbf{W}}} \sum_{i=1}^{m} \|\mathbf{x}_i \tilde{\mathbf{W}} - \mathbf{y}_i\|_2^2 + \gamma \|\tilde{\mathbf{W}}\|_{2,1}, \tag{7.1}$$

where $\tilde{\mathbf{W}} \in \mathbb{R}^{d \times k}$ is the global feature weight shared by all nodes, and the term $\gamma \|\tilde{\mathbf{W}}\|_{2,1}$ is imposed to achieve joint feature sparsity across k different classes.

To apprehend personalized behaviors of each single node, we also assume that

each node is also associated with a local variable $\mathbf{W}^i$. In this way, the class labels of labeled nodes can be approximated by a conjunction of global model parameter $\tilde{\mathbf{W}}$ and a localized variable $\mathbf{W}^i$. In this way, Eq. (7.1) can be reformulated as:

$$\min_{\tilde{\mathbf{W}},\mathbf{W}^i} \sum_{i=1}^{m} \|\mathbf{x}_i(\tilde{\mathbf{W}} + \mathbf{W}^i) - \mathbf{y}_i\|_2^2 + \gamma\|\tilde{\mathbf{W}}\|_{2,1}. \tag{7.2}$$

Similar to the modeling of global behaviors, personalized behavior is also encoded in a small subset of features that is locally associated with each individual. To put it in another way, we would like to achieve feature sparsity within each localized model parameter $\mathbf{W}^i$. It can be mathematically formulated by solving an exclusive group lasso problem (Kong *et al.*, 2014, 2016; Zhou *et al.*, 2010). In particular, each $\mathbf{W}^i$ is regarded as a group, exclusive group lasso encourages intra-group level competition but discourages inter-group level competition. As a result, a small subset of discriminative personalized features can be obtained within each $\mathbf{W}^i$. Therefore, we first impose an $\ell_{2,1}$-norm sparse regularization on $\mathbf{W}^i$ for intra-group level feature sparsity across k different class labels. Afterwards, we put ℓ_2-norm at the inter-group level for non-sparsity. With the intra-level sparsity and inter-level non-sparsity regularization term $\sum_{i=1}^{m} \|\mathbf{W}^i\|_{2,1}^2$, the node features $\mathbf{X}$ for personalized relational learning can be formally formulated as:

$$\min_{\tilde{\mathbf{W}},\mathbf{W}^i} \sum_{i=1}^{m} \|\mathbf{x}_i(\tilde{\mathbf{W}} + \mathbf{W}^i) - \mathbf{y}_i\|_2^2 + \beta \sum_{i=1}^{m} \|\mathbf{W}^i\|_{2,1}^2 + \gamma\|\tilde{\mathbf{W}}\|_{2,1}, \tag{7.3}$$

where parameters β and γ are used to balance the sparsity of personalized and shared features, respectively.

Modeling Network Information for Personalized Relational Learning. The objective function in Eq. (7.3) builds a predictive learning model with the supervision of node labels $\mathbf{Y}$. However, as mentioned above that in many cases, the portion of labeled nodes is very limited, either because of the labor and time consuming labeling process or labels themselves are just unavailable due to some privacy issues.

Fortunately, a rich source of network structure is readily observable and could be potentially helpful to build a more informative predictive model.

Even though individuals in a highly connected network exhibit some unique behaviors, as indicated by social categorization theory (Hornsey, 2008), these personalized individual behaviors are well organized and can be categorized into various groups. For example, groups can indicate different foci of user interests, such as sports, literature, and arts. Here the challenges center around inferring of personalized patterns and obtaining their group structures simultaneously. In this work, we take advantage of the network structure to cluster the personalized patterns based on node connectivity. In particular, we force linked nodes to borrow strength from each other in learning personalized patterns to fortify the prediction model by the network lasso regularization term (Hallac *et al.*, 2015):

$$\min_{\mathbf{W}} \sum_{i,j=1}^{m} \mathbf{A}_{ij} \|\mathbf{W}^i - \mathbf{W}^j\|_F. \tag{7.4}$$

The advantages of the above regularization term are two folds. First, the Frobenius norm of the difference between $\mathbf{W}^i$ and $\mathbf{W}^j$ not only makes them close to each other if they are connected, i.e., $\mathbf{A}_{ij} = 1$, but also incentivizes them to be the same. In this way, since many localized feature weights $\mathbf{W}^i$ are made to be the same, they are automatically grouped into several clusters. Second, when the label information cannot provide us enough guide to learn the localized parameter, Eq. (7.4) provides us a way to borrow strength from neighbors for the model parameter learning.

Learning and Inference. By combing Eq. (7.3) and Eq. (7.4), the final objective function of the proposed PRL framework can be formulated as:

$$\begin{aligned} \min_{\tilde{\mathbf{W}}, \mathbf{W}^i} J(\tilde{\mathbf{W}}, \mathbf{W}^i) &= \sum_{i=1}^{m} \|\mathbf{x}_i(\tilde{\mathbf{W}} + \mathbf{W}^i) - \mathbf{y}_i\|_2^2 \\ +\alpha \sum_{i,j=1}^{m} \mathbf{A}(i,j) \|\mathbf{W}^i - \mathbf{W}^j\|_F &+ \beta \sum_{i=1}^{m} \|\mathbf{W}^i\|_{2,1}^2 + \gamma \|\tilde{\mathbf{W}}\|_{2,1}, \end{aligned} \tag{7.5}$$

where α is a model parameter to control the contribution of network structure in helping personalized relational learning. Also, it controls how the nodes are clustered according to their localized feature parameters $\mathbf{W}^i$. By solving the above optimization problem, we can obtain $\tilde{\mathbf{W}}$ that captures the global feature pattern and a set of $\mathbf{W}^i$ $(i = 1, ..., m)$ that capture the personalized feature pattern for each labeled node.

Now we discuss how to make a prediction for unlabeled nodes by the built classifier which is a conjunction of $\tilde{\mathbf{W}}$ and $\mathbf{W}^i$. During the prediction phase, we first find the linked neighbors for a new unlabeled node u_l on the network G; then if we successfully find some labeled neighbors, we take the averaged feature parameters (conjunction of global and personalized) of its neighbors as the new feature weight $\overline{\mathbf{W}}^l$; otherwise, we use the averaged feature parameters (conjunction of global and personalized) of all labeled nodes as the new feature weight $\overline{\mathbf{W}}^l$. After we obtain the feature weight for the new unlabeled node u_l, its class labels can be predicted by $c^* = \arg\max_{c_j \in \mathcal{Y}}(|\mathbf{x}_l \overline{\mathbf{W}}^l|_j)$.

Optimization Solution. The objective function of PRL in Eq. (7.5) involves two sets of variables: (1) the global variable $\tilde{\mathbf{W}}$ that captures the global patterns of nodes; and (2) the localized variable $\mathbf{W}^i$ that encodes personalized behaviors of each individual node. The objective function is not convex w.r.t. $\tilde{\mathbf{W}}$ and $\mathbf{W}^i$ $(i = 1, ..., m)$ simultaneously. In addition to that, the objective function is also not smooth. Motivated by (Yamada *et al.*, 2017), we present an effective alternating algorithm to solve it, thus in each iteration, the model parameters could be updated with a closed-form solution. More details about the optimization can be found in (Li *et al.*, 2017e).

7.3 Experimental Evaluation

In this section, we conduct experiments to evaluate the effectiveness of the proposed framework PRL. We first introduce the used datasets, and experimental settings before presenting detailed results of the experiments. At last, we investigate the

parameter sensitivity study of PRL.

Datasets. We use three real-world networks for evaluation, and all of them are publicly available. Cora and Citeseer are real-world academic networks (Rossi and Ahmed, 2015) while BlogCatalog is a social media network (Li *et al.*, 2015). The Cora dataset is a citation network with 2,708 publications and 5,429 citations. Each publication is described by a set of 1,433 words which are considered as features. All these features are 0/1-valued. All publications are categorized into 7 classes according to their subjects. The Citeseer dataset is another citation network with 3,312 publications and 4,732 links. They are grouped into 6 classes. Similar to Cora, each publication is associated with a total of 3,703 0/1-valued features. The BlogCatalog dataset is a social blogging dataset with 5,196 users. The tag information of blogs by users are regarded as features; the feature number is 1,638. A total number of 171,743 links are observed. The ground truth is the major category (among 6 categories) of blogs posted by users. We adopt the same mechanism mentioned before to transform directed networks into undirected ones.

Experimental Settings. We select several representative basline methods for a fair comparison.

- NMF: Non-negative Matrix Factorization (NMF) (Lee and Seung, 2001) has proven to be effective in many real-world applications by reducing the feature dimensionality. We consider it as a baseline method to first obtain the low-rank node feature representation and then apply discriminative learning methods.

- wvRN: Weighted-Vote Relational Neighbor Classifier (wvRN) (Macskassy and Provost, 2003) is a local neighborhood based classifier. It makes the prediction for unlabeled nodes by a weighted vote score of its labeled neighbors.

- SocDim: Social Dimensions (Tang and Liu, 2009) is one of the state-of-the-art

relational learning approaches with only network information. It first adopts modularity maximization (Newman, 2006) to extract latent representations and then utilize them as features for discriminative learning.

- GNMF: Graph Regularized NMF (Cai *et al.*, 2010a) is based on the assumption that latent representations of connected nodes are also similar to each other. After getting the low-rank feature representation, we take them as input to a typical learning method.

- FsNet: It (Gu and Han, 2011) aims to select a subset of relevant features on the node feature space. In particular, it exploits a linear regression model to capture the node features and adopt graph regularization to make use of the network structure. We employ discriminative learning methods to build a predictive model based on the selected features.

The vast majority of relational learning methods heavily depend on the extracted feature representations. Among these comparison methods, NMF, SocDim, GNMF, and FsNet are typical feature-based relational learning methods. They first extract latent features and then employ typical discriminative methods to build a classifier to enable the prediction on unlabeled nodes. In the experiments, we follow a commonly adopted setting (Tang and Liu, 2009) to use linear SVM for discriminative learning. For each method, we randomly choose $p\%$ of nodes for training and the rest $1-p\%$ for testing. As we often have limited access to labeled nodes in practice, we choose a relatively small value for p by varying it in the range of $\{1, 2, ..., 9, 10\}$. For each p, we run the experiments 10 times and report the average classification performance. Two widely used evaluation criteria based on F1-measure, i.e., Micro-F1 and Macro-F1 are used to measure the multi-class and multi-label classification problems.

Effectiveness Evaluation. We evaluate the performance of PRL by comparing its

Table 7.1: Classification Results Evaluation of PRL and Baselines on Cora.

Training Ratio		1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Micro-F1	NMF	0.3914	0.4531	0.4748	0.4980	0.5229	0.5342	0.5402	0.5460	0.5639	0.5494
	wvRN	0.3230	0.3402	0.3626	0.3751	0.3929	0.4109	0.4221	0.4399	0.4502	0.4643
	SocDim	0.3322	0.3942	0.4414	0.4797	0.4996	0.5315	0.5467	0.5636	0.5872	0.5945
	GNMF	0.3936	0.4510	0.4798	0.5137	0.5216	0.5415	0.5477	0.5586	0.5726	0.5740
	FsNet	0.3880	0.4516	0.4829	0.5079	0.5231	0.5274	0.5384	0.5413	0.5444	0.5399
	PRL	**0.4254**	**0.4908**	**0.5324**	**0.5506**	**0.5688**	**0.5811**	**0.5989**	**0.6170**	**0.6266**	**0.6315**
Macro-F1	NMF	0.3133	0.3874	0.4178	0.4409	0.4829	0.4960	0.5041	0.5053	0.5262	0.5038
	wvRN	0.1198	0.1617	0.2064	0.2374	0.2721	0.3045	0.3273	0.3556	0.3755	0.3979
	SocDim	0.3077	0.3808	0.4256	0.4628	0.4814	0.5123	0.5311	0.5469	0.5688	0.5769
	GNMF	0.3173	0.3906	0.4300	0.4674	0.4793	0.4999	0.5061	0.5212	0.5340	0.5404
	FsNet	0.3074	0.3905	0.4269	0.4626	0.4836	0.4892	0.5040	0.5074	0.5133	0.5109
	PRL	**0.3833**	**0.4098**	**0.4881**	**0.4968**	**0.5324**	**0.5539**	**0.5637**	**0.5791**	**0.5906**	**0.6039**

Table 7.2: Classification Results Evaluation of PRL and Baselines on Citeseer.

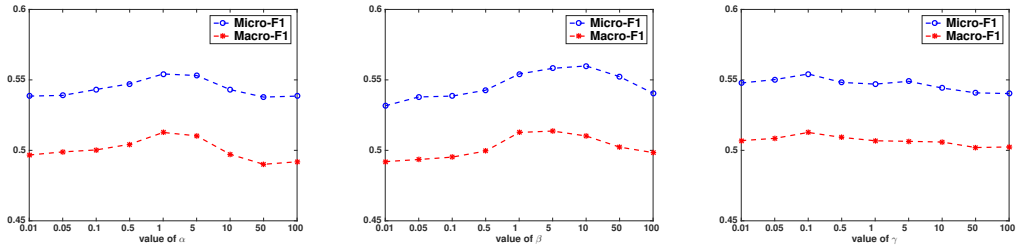
Training Ratio		1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Micro-F1	NMF	0.4236	0.4704	0.4749	0.4926	0.4978	0.5062	0.5264	0.5329	0.5363	0.5412
	wvRN	0.2264	0.2412	0.2548	0.2655	0.2779	0.2884	0.3003	0.3118	0.3206	0.3313
	SocDim	0.2701	0.2996	0.3254	0.3447	0.3523	0.3682	0.3750	0.3855	0.3934	0.4044
	GNMF	0.4296	0.4768	0.4981	0.5124	0.5235	0.5243	0.5253	0.5357	0.5435	0.5535
	FsNet	0.4301	0.4657	0.5125	0.5142	0.5202	0.5301	0.5344	0.5417	0.5524	0.5576
	PRL	**0.4356**	**0.4851**	**0.5296**	**0.5307**	**0.5505**	**0.5535**	**0.5568**	**0.5691**	**0.5725**	**0.5762**
Macro-F1	NMF	0.3732	0.4271	0.4347	0.4548	0.4589	0.4671	0.4881	0.4961	0.4977	0.5021
	wvRN	0.0887	0.1172	0.1421	0.1626	0.1843	0.2023	0.2221	0.2393	0.2532	0.2700
	SocDim	0.2453	0.2815	0.3056	0.3264	0.3333	0.3476	0.3544	0.3644	0.3712	0.3821
	GNMF	0.3820	0.4346	0.4565	0.4723	0.4837	0.4862	0.4865	0.4967	0.5061	0.5141
	FsNet	0.3677	0.4183	0.4683	0.4714	0.4797	0.4835	0.4949	0.5030	0.5089	0.5167
	PRL	**0.3993**	**0.4356**	**0.4751**	**0.4862**	**0.5103**	**0.5142**	**0.5220**	**0.5231**	**0.5287**	**0.5295**

classification performance with other methods on the three above-mentioned datasets. The comparison results are shown in Table 7.1, Table 7.2 and Table 7.3. The model parameters could be determined by cross-validation, and a detailed sensitivity study will be investigated later. We make the following observations from these three tables.

Table 7.3: Classification Results Evaluation of PRL and Baselines on BlogCatalog.

Training Ratio		1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Micro-F1	NMF	0.5342	0.5938	0.6235	0.6531	0.6570	0.6617	0.6739	0.6787	0.6854	0.6938
	wvRN	0.2516	0.3181	0.3511	0.3928	0.4204	0.4372	0.4598	0.4727	0.4849	0.5026
	SocDim	0.3697	0.4419	0.4741	0.5078	0.5340	0.5502	0.5680	0.5831	0.5947	0.5952
	GNMF	0.5632	0.6063	**0.6504**	0.6587	0.6634	0.6743	0.6771	0.6873	0.6880	0.6927
	FsNet	0.5363	0.6096	0.6240	0.6308	0.6467	0.6359	0.6422	0.6444	0.6408	0.6433
	PRL	**0.6009**	**0.6127**	0.6341	**0.6622**	**0.6767**	**0.6939**	**0.7117**	**0.7184**	**0.7235**	**0.7365**
Macro-F1	NMF	0.5279	0.5856	0.6184	0.6479	0.6529	0.6579	0.6693	0.6748	0.6804	0.6885
	wvRN	0.2276	0.3043	0.3416	0.3836	0.4123	0.4299	0.4495	0.4607	0.4722	0.4902
	SocDim	0.3651	0.4372	0.4690	0.5023	0.5293	0.5429	0.5599	0.5754	0.5863	0.5869
	GNMF	0.5533	0.6006	0.6236	0.6544	0.6571	0.6689	0.6733	0.6819	0.6925	0.6963
	FsNet	0.5189	0.6010	0.6175	0.6306	0.6452	0.6338	0.6417	0.6436	0.6398	0.6426
	PRL	**0.5720**	**0.6153**	**0.6447**	**0.6697**	**0.6661**	**0.6923**	**0.7143**	**0.7153**	**0.7228**	**0.7335**

(1) In most cases, when we gradually increase the number of labeled nodes from 1% to 10%, the classification performance increases for all methods in the table. (2) Our proposed personalized relational learning framework PRL outperforms all baseline methods in almost all cases. Meanwhile, PRL is significantly better with a significance level in both 0.01 and 0.05, with Wilcoxon signed-rank test. (3) Both wvRN and SocDim are relational learning methods with only network information; their classification performance is inferior to relational learning approaches incorporating node features such as GNMF, FsNet, and PRL. The results support the importance of leveraging both sources of information for relational learning. (4) GNMF is an extension of NMF that uses graph regularization to make the latent representation consistent with the network topological structure. It obtains higher Micro-F1 and Macro-F1 than NMF in most cases, suggesting that the exploration of rich network information is helpful and could improve relational learning. (5) FsNet selects a common set of relevant features, while our proposed method could be regarded as a personalized feature selection framework. The improvement of PRL over FsNet val-



(a) Effect of parameter α. (b) Effect of parameter β. (c) Effect of parameter γ.

Figure 7.2: Parameter Study of PRL on Citeseer Dataset.

idates the necessity of employing personalized features for relational learning, which has an added value over a set of shared features.

Parameter Sensitivity Study. In PRL, α balances the contribution of node features and network structure for relational learning, β and γ controls the sparsity of personalized features of each individual and the common feature in the model learning phase. To investigate the effects of these three parameters, we fix one parameter each time and vary the other two to see how it affects the classification performance. The portion of training data in the study is set to be 5%. First, we fix the parameters β as 1 and γ as 0.1 and vary the value of α among $\{0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100\}$. As can be observed from Figure 7.2(a), when we gradually increase α, the classification performance first increases and reaches its peak and then gradually decreases. The best performance is achieved when α is between 1 and 5. Next, we fix $\alpha = 1$ and $\gamma = 0.1$ and vary β as $\{0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100\}$. The study results are shown in Figure 7.2(b). We observe that when β is small, the classification performance is relatively lower. The reason is that when β is small, the contribution of the personalized feature selection is limited; on the other hand, a reasonable β enables us to find better localized features customized for each node, which in turn benefit the prediction performance. At last, we fix $\alpha = 1$ and $\beta = 1$, and vary the third variable

γ. In particular, when γ is small, personalized features will dominate the objective function, the performance is high. As we continuously increase γ, the performance gradually decreases, but the change is small.

7.4 Summary

Node classification targets to use network structure and node features of a small number of labeled nodes (if available) to build a predictive learning model; then employs the built model to infer missing labels for unlabeled nodes. Existing methods on this line assume that all nodes have a common pattern by sharing the same feature weight. However, as nodes on networks are highly idiosyncratic, their associated node features are quite diverse and personalized. Hence, it would be appealing to tailor the learning and prediction by using a set of personalized features specific to the node, and a set of common features shared by all nodes. Toward this goal, we propose a novel personalized relational learning framework PRL. As we can customize the learning and prediction for each individual, the proposed model is also human interpretable. Experiments on real-world networks show the effectiveness of the proposed model.

Chapter 8

ANOMALY DETECTION ON ATTRIBUTED NETWORKS

In this chapter, we investigate another important problem on attributed networks - the anomaly detection problem. Attributed networks are pervasive in different domains, ranging from social networks, gene regulatory networks, to financial transaction networks. This kind of rich network representation presents challenges for the anomaly detection problem due to the heterogeneity of two data representations. A vast majority of existing algorithms assume certain properties of anomalies are given a prior and attempt to detect anomalies within a specific context. Since various types of anomalies in real-world attributed networks co-exist, the assumption that prior knowledge regarding anomalies is available does not hold. To solve the challenge, we study the anomaly detection problem generally from a residual analysis perspective, which has been shown to be effective in traditional anomaly detection problems – with a mild assumption that residuals of anomalies have a significant deviation from the normal samples.

8.1 Overview

Anomaly detection (a.k.a. outlier detection) (Aggarwal, 2015; Chandola *et al.*, 2009) aims to discover rare instances that do not conform to the patterns of majority. Recently, there is a growing interest to perform anomaly detection on attributed networks (Gao *et al.*, 2010; Perozzi *et al.*, 2014a; Perozzi and Akoglu, 2016; Sánchez *et al.*, 2014). To facilitate the detection of anomalous nodes, a straightforward way is to assume that some properties of anomalies are known in advance, examples include structural anomaly, contextual anomaly, and community anomaly, among others. In

fact, different types of anomalies are often mixed together on attributed networks, and it is hard to identify all of them when we have no prior knowledge of data. Besides, new types of anomalies may continuously arise over time especially in an adversarial environment. Therefore, it is beneficial and desirable to explore and spot anomalies in a general sense. Residual analysis (Cook and Weisberg, 1982; She and Owen, 2011), which is initiated to study the residuals between true data and estimated data for regression problems, provides a mild assumption for anomaly detection – instances with large residual errors are more likely to be anomalies. Although it provides a general way to find anomalies, it is a non-trivial to be applied on attributed networks: (1) we have heterogeneous data sources on attributed networks, it is insufficient to consider residuals from a single data source; (2) instances on attributed networks are not independent and identically distributed (*i.i.d.*), the interactions among them further complicate the residual modeling process.

In this chapter, we provide a principled way to identify and detect anomalies via the principle of residual analysis. In particular, we develop a novel framework RADAR by investigating: (1) How to characterize the residuals of attribute information to spot anomalies when there is no prior knowledge of anomalies? (2) How to exploit coherence between attribute residuals and network information to identify anomalies? The problem statement is formulated as follows.

Problem 6. Anomaly Detection on Attributed Networks

Given: *An attributed network G represented by the content matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ and the adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}_{\geq 0}$, where n and d denote the number of nodes and features, respectively.*

Find: *A set of nodes that are rare and differ singularly from the majority reference instances. In particular, we would like to rank the nodes by their degree of*

abnormality such that the abnormal ones are ranked in higher positions.

8.2 Proposed Framework - RADAR

In this section, we give details about how to model attribute information and network information to detect anomalies generally from a residual analysis perspective.

Modeling Attributed Information. We start from the situation when only attribute information is available. Let $\tilde{\mathbf{X}}$ denotes the estimated attribute information, then the approximation error $\mathbf{X} - \tilde{\mathbf{X}}$, i.e, residuals, can be exploited to determine contextual anomaly as content patterns of anomalies deviate significantly from majority normal instances (Tong and Lin, 2011). One natural way to build $\tilde{\mathbf{X}}$ is by using some representative instances (Yu *et al.*, 2006). For a certain instance, if its attribute information can be approximated by some representative instances, it is of low probability to be anomalous. On the opposite side, if the instance cannot be well represented by some representative instances, its attribute information does not conform to the patterns of majority reference instances. In other words, we would like to use the attribute information of some representative instances to reconstruct $\mathbf{X}$. Mathematically, it is formulated as:

$$\min_{\mathbf{W}} \|\mathbf{X} - \mathbf{W}'\mathbf{X}\|_F^2 + \alpha\|\mathbf{W}\|_{2,0}, \tag{8.1}$$

where $\mathbf{W} \in \mathbb{R}^{n\times n}$ is a coefficient matrix such that the attribute information of each instance (a row of $\mathbf{X}$) can be reconstructed by a linear combination of other instances; the row sparsity constraint $\|\mathbf{W}\|_{2,0}$ ensures that only the attribute information of a few representative instances are employed to reconstruct $\mathbf{X}$, α is a parameter to control the row sparsity. However, the problem in Eq. (8.1) is NP-hard due to the $\ell_{2,0}$-norm term. $\|\mathbf{W}\|_{2,1}$ is the minimum convex hull of $\|\mathbf{W}\|_{2,0}$ and we can minimize $\|\mathbf{W}\|_{2,1}$ to obtain the same results as $\|\mathbf{W}\|_{2,0}$, and it is also widely used in other learning tasks

such as feature selection. In this way, we reformulate Eq. (8.1) as:

$$\min_{\mathbf{W}} \|\mathbf{X} - \mathbf{W}'\mathbf{X}\|_F^2 + \alpha\|\mathbf{W}\|_{2,1}. \tag{8.2}$$

Let $\Theta = \mathbf{X} - \mathbf{W}'\mathbf{X} - \mathbf{R}$ be a random error matrix. Θ is usually assumed to follow a multi-dimensional normal distribution. $\mathbf{R}$ is the residual matrix from the reconstruction process in Eq. (8.2). The residual matrix $\mathbf{R}$ can be used to determine anomalies since the attribute patterns of anomalous instances and normal instances are quite different, a large norm of $\mathbf{R}(i,:)$ indicates the instance has a higher probability to be an anomaly (Tang and Liu, 2013). In addition, in many applications like rumor detection (Wu *et al.*, 2017), malicious URL detection (Sahoo *et al.*, 2017), and rare category detection (Zhou *et al.*, 2015), the number of anomalies is much smaller than the number of normal instances, therefore we add $\|\mathbf{R}\|_{2,1}$ on the basis of Eq. (8.2) to achieve row sparsity to constrain the number of abnormal instances. The objective function can be reformulated as:

$$\min_{\mathbf{W},\mathbf{R}} \|\mathbf{X} - \mathbf{W}'\mathbf{X} - \mathbf{R}\|_F^2 + \alpha\|\mathbf{W}\|_{2,1} + \beta\|\mathbf{R}\|_{2,1}. \tag{8.3}$$

where β controls the row sparsity of residual matrix $\mathbf{R}$.

Modeling Network Information. We model the residuals of attribute information to spot anomalies in Eq. (8.3). However, on attributed networks, some types of anomalies are not solely described at a contextual level. Therefore, we need to exploit the correlation between attribute and network information to detect anomalies in a more general way. According to well-received social science theory such as homophily (McPherson *et al.*, 2001), instances with similar patterns are more likely to be linked together on attributed networks. Similarly, when we reconstruct $\mathbf{X}$ by the attribute information of some representative instances, the homophily effect should also hold. It indicates that if two instances are linked together on the network, after

attribute reconstruction by representative (normal) instances, their attribute patterns in the residual matrix $\mathbf{R}$ should also be similar. If the attributed network is an undirected network, it can be mathematically formulated by minimizing the following term:

$$\begin{aligned} &\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{A}(i,j)\|\mathbf{R}(i,:)-\mathbf{R}(j,:)\|_2^2 \\ =&tr(\mathbf{R}'(\mathbf{D}-\mathbf{A})\mathbf{R}) = tr(\mathbf{R}'\mathbf{L}\mathbf{R}), \end{aligned} \tag{8.4}$$

where $\mathbf{D}$ is a diagonal matrix with $\mathbf{D}(i,i) = \sum_{j=1}^{n}\mathbf{A}(i,j)$, $\mathbf{L}$ is a Laplacian matrix. If the attributed network is a directed network, the graph regularization term in Eq. (8.4) cannot be used directly since the adjacency matrix $\mathbf{A}$ is not symmetric. To model the network information on directed networks, we follow (Li *et al.*, 2016b) to use $\mathbf{A} = max(\mathbf{A}, \mathbf{A}')$. Then the Laplacian matrix is in the same form as the undirected networks.

The Joint Framework for Anomaly Detection. The objective function in Eq. (8.3) is based on a strong assumption that instances are independent and identically distributed (*i.i.d.*). However, it is not the case on networks such that instances are interconnected with each other, the interactions among instances also complicate the residual modeling process. Therefore, we propose to integrate the network modeling term in Eq. (8.4) on the basis of Eq. (8.3) to capture the coherence between attribute residual information and network information, the objective function of the RADAR framework can be formulated as follows:

$$\min_{\mathbf{W},\mathbf{R}} \|\mathbf{X}-\mathbf{W}'\mathbf{X}-\mathbf{R}\|_F^2 + \alpha\|\mathbf{W}\|_{2,1} + \beta\|\mathbf{R}\|_{2,1} + \gamma tr(\mathbf{R}'\mathbf{L}\mathbf{R}), \tag{8.5}$$

where γ is a parameter to balance the contribution of attribute reconstruction and network modeling.

It can be observed that without any prior knowledge about anomalies, we build a general learning framework (Eq. (8.5)) to detect anomalous instances generally by

exploiting both attribute and network information as well as their correlations. By learning and analyzing the residual matrix $\mathbf{R}$, it enables the ranking of anomalies according to their residual values. Different from making a binary decision of anomalies, anomaly ranking is easier to be interpreted. It makes further exploration possible as decision markers can check the degrees of deviation manually.

The objective function in Eq. (8.5) is not convex in terms of $\mathbf{W}$ and $\mathbf{R}$ simultaneously. Besides, it is also not smooth due to the existence of $\ell_{2,1}$-norm regularization term. We use an alternating way to optimize this problem and more details of the optimization process and its convergence analysis can be found in (Li *et al.*, 2017b).

8.3 Experimental Evaluation

In this section, we conduct experiments to evaluate the effectiveness of the proposed anomaly detection framework RADAR. In particular, we investigate the following two research questions: (1) How is the anomaly detection performance of the proposed RADAR when measured against other representative anomaly detection methods? (2) Does the utilization of coherence between attribute residuals and network information help find anomalous instances otherwise remain undiscovered? Before discussing details of the experiments, we first introduce the datasets and the experimental settings.

Datasets. We use three real-world attributed network datasets for the evaluation, and all these datasets have been used in previous research (Müller *et al.*, 2013; Sánchez *et al.*, 2013). Among them, Disney dataset and Books dataset come from the Amazon co-purchase networks. Disney is a co-purchase network of movies, the attributes include prices, ratings, number of reviews, etc. The ground truth (anomalies) are manually labeled by high school students. The dataset contains 124 nodes, 334 edges, 28 attributes, and the ratio of anomalies is 4.8%. The second dataset, Books, is a

co-purchase network of books, it has similar attributes as Disney dataset. The ground truth (anomalies) are obtained by amazonfail tag information. There are 1,418 nodes, 3,695 edges, 21 attributes in total. And 28 nodes are considered to be anomalous. Enron is an email network dataset, spam messages are taken as ground truth. There are 13,533 nodes (including 6 anomalies), 176,987 edges and 18 attributes.

Experimental Settings. The criteria of AUC (Area Under ROC Curve) is applied to evaluate the performance of anomaly detection algorithms. According to the ground truth and the results by anomaly detection algorithms, there are four possible outcomes: anomaly is recognized as anomaly (TP), anomaly is recognized as normal (FN), normal is recognized as anomaly (FP), and normal is recognized as normal (TN). Therefore, the detection rate (dr) and false alarm rate (flr) are defined as:

$$dr = \frac{TP}{TP + FN}, \ flr = \frac{FP}{FP + TN}. \tag{8.6}$$

Then the ROC curve is a plot of detection rate (dr) vs. false alarm rate (flr). From the statistical perspective, AUC value represents the probability that a randomly chosen abnormal instance is ranked higher than a normal instance. If the AUC value approaches 1, the method is of high quality.

We compare the proposed RADAR with four baseline methods:

- LOF (Breunig *et al.*, 2000): LOF detects anomalies in a contextual level and only uses attribute information.
- SCAN (Xu *et al.*, 2007): SCAN detects anomalies in a structural level and only considers network information.
- CODA (Gao *et al.*, 2010): CODA detects anomalies within the context of communities where these instances deviate significantly from other members in the same community.

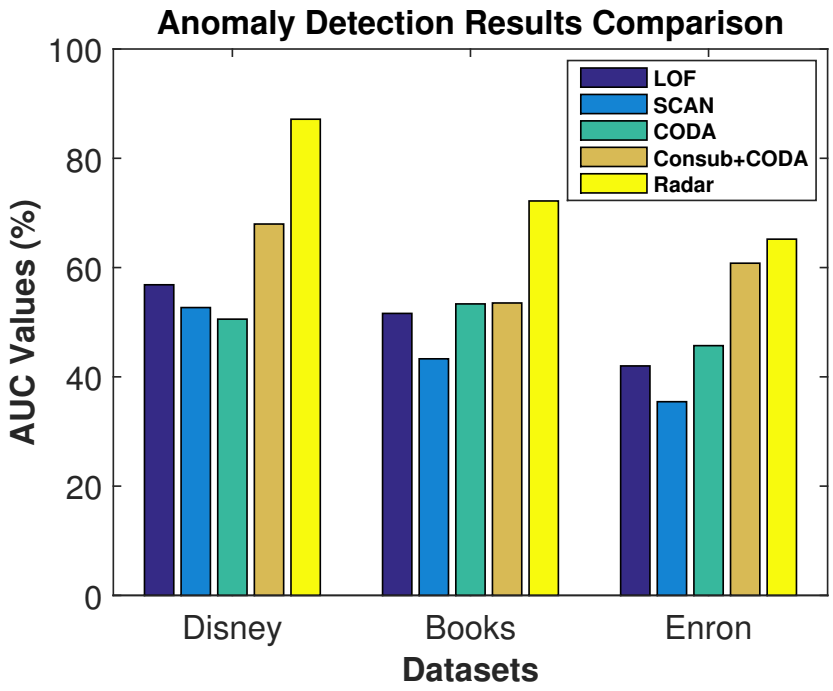


Figure 8.1: Anomaly Detection Results Evaluation by RADAR and Baselines.

- CONSUB+CODA (Sánchez *et al.*, 2013): It performs subspace selection as a pre-processing step and then applies CODA to detect subspace community anomalies on attributed networks.

Among them, LOF, SCAN, CODA covers three types of widely defined anomalies on attributed networks (contextual anomaly, structural anomaly, and community anomaly). CONSUB+CODA is able to find subspace community anomalies by taking subspace selection as a pre-processing step. The proposed RADAR framework has three different regularization parameters, for a fair comparison, we tune these parameters by a "grid-search" strategy from $\{10^{-3}, 10^{-2}, ..., 10^{2}, 10^{3}\}$. Details about the effects of these parameters will be investigated later.

Effectiveness of Radar. The experimental results in terms of AUC values are presented in Figure 8.1. By comparing the performance of different methods, we can observe that the proposed RADAR framework always obtains the best anomaly

detection performance. The reason is that on real-world attributed networks, nodes are annotated as anomalies due to a variety of reasons. Our RADAR algorithm provides a general way to detect anomalies globally and does not depend on specific properties of anomalies. We also perform one-tailed t-test between RADAR and other baseline methods and the test results show that RADAR is significantly better (with a 0.05 significance level). Therefore, we can get an answer for the first question that the proposed RADAR framework outperforms other representative anomaly detection algorithms for attributed networks.

Effectiveness of Coherence Modeling. We study the second question to investigate how the coherence between attribute residuals and network information affects anomaly detection. We compare RADAR with the following variants by varying γ:

- *Residual-based method*: We set the parameter γ to be zero, therefore, only residuals of attribute information is taken into consideration. The detected anomalies can be considered as contextual anomalies.

- *Network-based method*: We set the parameter γ to be a large number, therefore, the contribution from attribute residuals can be ignored. The detected anomalies can be considered as structural anomalies.

First, we compare the anomaly detection results by the proposed RADAR, the residual-based method, and the network-based method on Disney dataset. The AUC values of these three methods are 87.1%, 77.68%, 74.29%, respectively. It indicates that by exploiting the correlation between attribute residuals and network information, the anomaly detection performance indeed improves. We only present the comparison results on Disney dataset as we have similar observations on the other two datasets. Second, we compare the overlap of detected anomalies by each pair of method (RADAR and residual-based method, RADAR and network-based method,

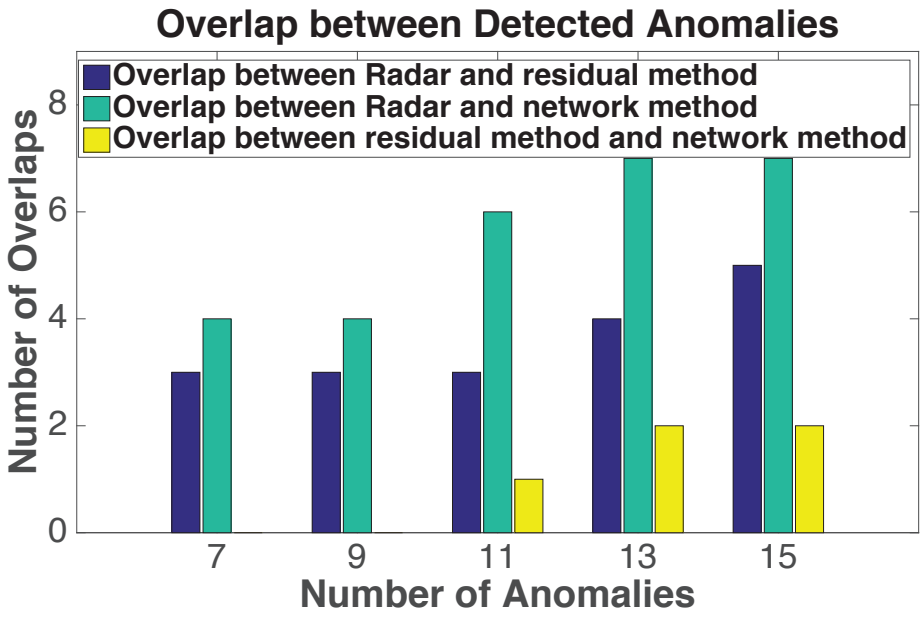


Figure 8.2: Anomalies Overlap Comparison between RADAR and Its Variants.

residual-based method and network-based method) in Figure 8.2. As can be observed, when we vary the number of detected anomalies, the overlap of anomalies between RADAR and residual-based method, RADAR and network-based method are always larger than the overlap between residual-based method and network-based method. This phenomenon shows that by exploiting the correlation between attribute residuals and network structure, we can find anomalies otherwise undiscovered by a single source of information. It also shows the potential to detect anomalies generally via residual analysis.

Parameter Study. There are three parameters in the proposed framework. Among them, β and γ are relatively more important. The parameter β controls the number of anomalies, while γ balances the contribution of attribute information and network information for anomaly detection. We investigate how these two parameters affect the anomaly detection results on Disney dataset. The performance variance result is shown in Figure 8.3 (α is fixed to be 0.5). We observe that when β is small, the

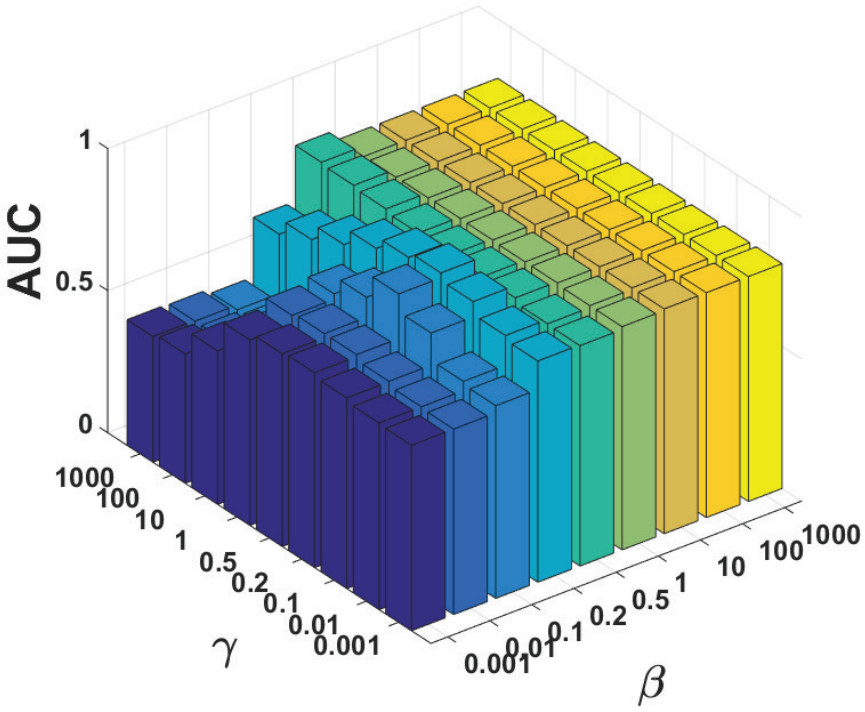


Figure 8.3: Parameter Study of RADAR on Disney Dataset.

AUC values are relatively low, the anomaly detection performance is not sensitive to the parameters when β and γ are in the range of 0.1 to 1000, and 0.001 to 10, respectively. The performance is the best when both β and γ are around 0.2.

8.4 Summary

In this chapter, we study the application of anomaly detection on attributed networks, and our goal is to detect the nodes whose behaviors or patterns deviate significantly from other majority nodes on the network. Methodologically, we propose a learning framework RADAR to characterize attribute reconstruction residual and its correlation with network information to detect anomalies. Through learning and probing the residuals of the reconstruction process, we are able to spot anomalies in a global view when properties of anomalies are unknown. Experiments on real-world

datasets show that our framework yields better AUC values compared to baseline methods which define anomalies in a specific context. Besides, the coherence between attribute residuals and network structure can help uncover anomalies otherwise undiscovered by a single source of information.

Chapter 9

STREAMING LINK PREDICTION ON DYNAMIC ATTRIBUTED NETWORKS

Link prediction targets to predict the future node interactions mainly based on the current network snapshot. It is a key step in understanding the formation and evolution of the underlying networks; and has practical implications in many real-world applications, ranging from friendship recommendation, click through prediction, to targeted advertising (Getoor and Diehl, 2005). Most existing efforts are devoted to plain networks and assume the availability of network structure in memory before link prediction takes place. However, this assumption is untenable as many real-world networks are affiliated with rich node attributes, and often, the network structure and node attributes are both dynamically evolving at an unprecedented rate. Even though recent studies show that node attributes have an added value to network structure for accurate link prediction, it still remains a daunting task to support link prediction in an online fashion on such dynamic attributed networks. As changes in the dynamic attributed networks are often transient and can be endless, link prediction algorithms need to be efficient by making only one pass of the data with limited memory overhead.

9.1 Overview

As per the fact that node attributes are complementary for link prediction while both network structure and node attributes exhibit high dynamics, we investigate a novel problem of streaming link prediction on dynamic attributed networks in this chapter. The following challenges have to be addressed simultaneously: (1) **Near Real-Time Prediction**: Dynamic attributed networks are characterized by stream-

ing nodes/edges of high velocity. Also, the evolution of networks is often mixed with the changes of node attributes at an unsynchronized rate. As changes are essential components of the system and could occur at any time, link prediction algorithms require to be efficient and are performed in a streaming fashion to predict missing links in close to near real-time. (2) **One-Pass of the Data**: The entire size of the network and affiliated node attributes are often unknown at a particular moment and could even be infinite in the worst case. Hence, the streaming link prediction algorithms need to be pass-efficient to make only one pass of the data as the further passes are either expensive or naturally impossible. (3) **Space Efficiency**: Data is continuously being generated, the huge volume of data makes the dynamic attributed network hard to be materialized in memory, which necessitates the design of a cost-effective data synopsis with limited memory overhead to summarize the ever-increasing network structure and node attributes. (4) **Concept Drift**: With the accumulation of new nodes/edges and the changes of node attributes, the underlying network topology and the content patterns continuously evolve over time, resulting in the emerging of unseen patterns and the fading of existing patterns, which may significantly impact the link prediction performance. This phenomenon is often referred to as *concept drift* in data stream mining. In this regard, link prediction algorithms should be able to tackle the issue of concept drift. (5) **Data Heterogeneity**: Even though network structure and node attributes are presented in different modalities, they are often not mutually independent and could influence each other. Link prediction algorithms are supposed to seize the inherent interconnections for accurate prediction.

Here, we present the problem formulation for the studied problem.

Problem 7. Streaming Link Prediction on Dynamic Attributed Networks

Given: *A dynamic attributed network* $G = (G^{t_0}, G^{t_1}, ...)$ *with fast-evolving node/edge*

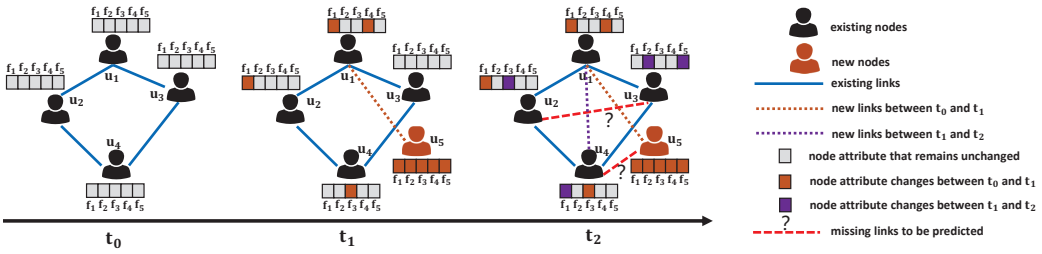


Figure 9.1: An Illustration of the Studied Problem of Streaming Link Prediction on Dynamic Attributed Networks.

streams and changes of node attributes across multiple time stamps $(t_0, t_1, ...)$.

Predict: *If there exists an edge* $e = (u, v)$ *for a pair of nodes* (u, v) *at a particular time stamp* t*, which are not connected previously before time stamp* t.

An illustration of the studied problem is shown in Figure 9.1. The streaming link prediction problem supports the prediction of missing links in an online fashion. For example, as shown in the figure, given a dynamic attributed network at three different time stamps t_0, t_1, and t_2 with both network structure and node attribute changes, the streaming link prediction problem predicts the missing links that may appear at time stamp t, where $t > t_2$.

9.2 Proposed Framework - SLIDE

Notations We first introduce some basic concepts and notations that will be used in this chapter. The SVD of $\mathbf{A} \in \mathbb{R}^{n \times m}$ is denoted as $\text{SVD}(\mathbf{A}) = \mathbf{U}\mathbf{\Sigma}\mathbf{V}'$, $\mathbf{U}$ is an $n \times n$ orthogonal matrix with the columns being left singular vectors $[\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_n]$, $\mathbf{V}$ is a $m \times m$ orthogonal matrix with the columns being the right singular vectors $[\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_m]$, $\mathbf{\Sigma} = \text{diag}(\sigma_1, ..., \sigma_n)$ is a $n \times m$ diagonal matrix, where $\sigma_1 \geq \sigma_2 \geq , ..., \geq \sigma_r$ are the singular values of $\mathbf{A}$ and r is the rank of matrix $\mathbf{A}$. The best rank-k $(k \leq r)$ approximation of matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ is $\mathbf{A}_k = \text{argmin}_{\text{rank}(\mathbf{X}) \leq k} \|\mathbf{A} - \mathbf{X}\|_F$ and

it can be computed as $\mathbf{A}_k = \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k' = \sum_{i=1}^{k}\sigma_i\mathbf{u}_i\mathbf{v}_i'$, where $\mathbf{U}_k$, $\mathbf{\Sigma}_k$, $\mathbf{V}_k$ are the truncated matrices consisting of the top-k left singular vectors, singular values, and right singular vectors, respectively.

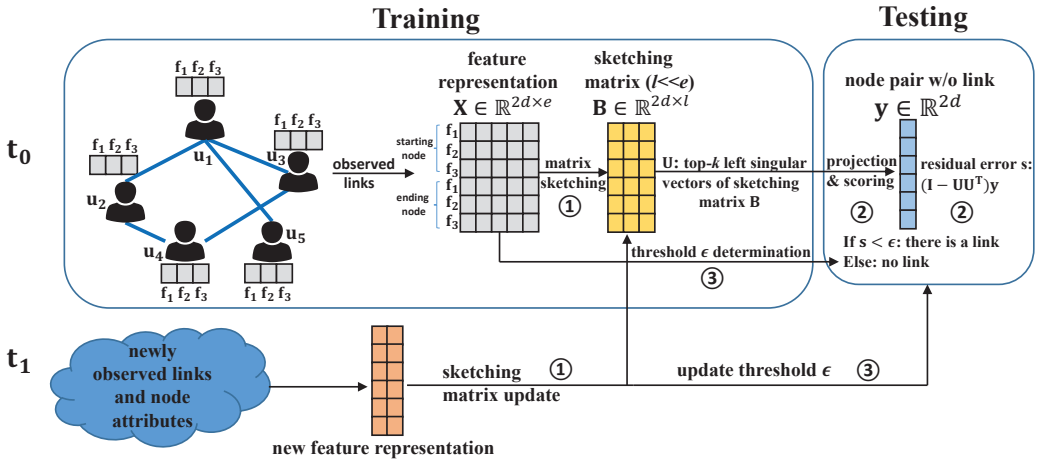


Figure 9.2: An Illustration of the Streaming Link Prediction Framework SLIDE.

In this section, we present the proposed framework SLIDE for streaming link prediction on dynamic attributed networks. The basic idea is to maintain and update a low-rank sketching matrix with limited memory overhead to summarize the currently observed links and node attributes. Then given the attributes of a pair of unconnected nodes, we leverage the low-rank sketching matrix to determine if there exists a link between these two end nodes in the future. The low-rank sketching matrix is continuously being updated when new links and new node attributes are observed. The workflow of the proposed SLIDE is shown in Figure 9.2. As can be observed from the figure, the proposed framework consists of three essential components: (1) maintain and update a sketching matrix to summarize the currently observed data, including all the observed links and node attributes; (2) predict missing links on the fly with the up-to-date sketching matrix; (3) calculate and update the threshold which is used to determine the existence of links.

Summarization with Matrix Sketching. On a typical dynamic attributed network, a massive amount of edges are continuously arriving at a fast pace. Meanwhile, node attributes also change naturally such that new content patterns may emerge and outdated content patterns will fade. To explicitly store such a dynamic attributed network at a particular time stamp t, we need $O(n_t^2 + n_t d)$ space in the worst case (d is often much smaller than n_t), where n_t is the number of nodes on the network until t and d is the dimensionality of node attributes. The materialization of the attributed networks becomes infeasible when n_t is very large. Hence, it is of vital importance to use cost-effective data synopsis to summarize all observed data including the links and node attributes. Nonetheless, designing a full streaming model with limited and constant memory space is a challenging problem and most of the existing efforts on graph streams are devoted to the so-called *semi-streaming* models (Feigenbaum *et al.*, 2005; McGregor, 2014), which requires $O(n_t \text{polylog}(n_t))$ space. These semi-streaming models are intractable if the available memory is not proportional to the number of nodes n_t on the network. In addition, due to the heterogeneity of two information sources on attributed networks, the resulted data synopsis is expected to summarize both information sources simultaneously.

Motivated by the recent advances in full streaming models for conventional data streams, we propose to use the frequent directions algorithm (Liberty, 2013) to maintain a low-rank sketching matrix (with limited memory overhead) to make a structural summary of the currently observed data. One major merit of the frequent directions algorithm is that it operates in a streaming fashion and makes only one pass of the data. However, the frequent directions algorithm cannot be directly applied to dynamic attributed networks for a low-rank approximation in real-time. The reason is that frequent directions algorithm is proposed to summarize conventional data streams where columns of the input matrix are added incrementally and the row of

the input matrix is fixed. On dynamic attributed networks, even though the node attributes are presented as a data stream (if the number of node attributes is fixed), the changes of the underlying network structure (often encoded in an adjacency matrix), per se, cannot be simply generalized as a conventional data stream. To this end, we propose to represent the dynamic attributed networks as the feature representation based on the observed links. The similar feature representation mechanism is also widely used in many other learning tasks, such as factorization machines (Rendle, 2010) and contextual-bandit collaborative filtering (Li *et al.*, 2010a).

Definition 5. *(***Feature Representation of Dynamic Attributed Networks***): Given a dynamic attributed network across multiple time stamps* $G = (G^{t_0}, G^{t_1}, ...)$*, its feature representation at a particular time stamp* t *is represented* $\mathbf{F}^t \in \mathbb{R}^{2d \times |\mathcal{E}^t|}$*. Each column* $\mathbf{f} \in \mathbb{R}^{2d}$ *in the feature representation* $\mathbf{F}^t$ *corresponds to an edge in* G^t*. Now assume that two end nodes of the edge is* u_i *and* u_j *(*$i < j$*), then the corresponding feature representation, i.e.,* $\mathbf{f}$*, can be represented as* $\mathbf{f} = [\mathbf{X}^t_{i*}, \mathbf{X}^t_{j*}]'$*, where* $\mathbf{X}^t$ *is the node attributes of the dynamic attributed network at time stamp* t*.*

By transforming the dynamic attributed networks into feature representations, the changes can be presented as new columns in a conventional data stream. In particular, new columns are introduced in two scenarios: (1) the arrival of new edges; and (2) node attribute information changes. The first scenario is straightforward and easy to understand. Regarding the second scenario, if the node attributes change, then the feature representations of edges that these nodes involved in should also change, and we represent these changes as new columns in the data stream. Now we assume that we can store all the historically generated data, and the feature representation of the dynamic attributed networks until time stamp t is represented as $\mathbf{G}^t \in \mathbb{R}^{2d \times c_t}$, where c_t is the number of columns in $\mathbf{G}^t$, and $c_t \geq |\mathcal{E}^t|$.

Even though we have reformulated the changes in the dynamic attributed networks as new columns in a data stream, the number of columns in $\mathbf{G}^t$ are still too large to be stored in memory, especially when the number of edges is in the scale of billion or trillion. Instead of storing the feature representations of the underlying dynamic attributed network, the frequent directions algorithm is employed to maintain a low-rank sketching matrix, and the sketching matrix can well summarize the observed data with a theoretical guarantee. Specifically, the low-rank sketching matrix $\mathbf{B}^t \in \mathbb{R}^{2d \times l}$ (l is often small) approximates the matrix $\mathbf{G}^t$ well such that $\mathbf{B}^t(\mathbf{B}^t)' \approx \mathbf{G}^t(\mathbf{G}^t)'$. More accurately, the approximation error is bounded by the conditions that: (1) $\mathbf{G}^t(\mathbf{G}^t)' \succeq \mathbf{B}^t(\mathbf{B}^t)'$; and (2) $\|\mathbf{G}^t(\mathbf{G}^t)' - \mathbf{B}^t(\mathbf{B}^t)'\| \leq 2\|\mathbf{G}^t\|_F^2/l$ (Liberty, 2013).

Let $\mathbf{D}^t \in \mathbb{R}^{2d \times m_t}$ denote the new columns generated between time stamp $t-1$ and the following time stamp t such that $\mathbf{G}^t = [\mathbf{G}^{t-1}, \mathbf{D}^t]$, where m_t is the number of new columns generated between time stamps $t-1$ and t. As mentioned above, the generation of new columns pertains to the arrival of new edges or the changes of node attributes. Here the challenges center around how to maintain and update the sketching matrix $\mathbf{B}^t$ based on the newly generated data in $\mathbf{D}^t$. At the very beginning, the sketching matrix $\mathbf{B}^t$ ($t = 0$) is set to be empty, then new columns presented in a data stream are continuously being inserted into the sketching matrix $\mathbf{B}^t$ until there are no empty columns anymore. Then frequent directions algorithm "shrinks" l orthogonal vectors by the same amount to make space for new data in the future. Concretely, the computation of SVD is necessary each time when the sketching matrix $\mathbf{B}^t$ is full. The original frequent directions algorithm assumes that one column arrives at each time stamp, (Huang and Kasiviswanathan, 2015; Huang *et al.*, 2015) further extended it to tackle the case when more than one columns arrive at each time stamp. As we have more than one column in $\mathbf{D}^t$ in most cases, we leverage this general solution to maintain and update the low-rank sketching matrix $\mathbf{B}^t$, upon which the

Algorithm 4 Maintain and Update the Sketching Matrix $\mathbf{B}^t$.

Input: Sketching matrix $\mathbf{B}^{t-1} \in \mathbb{R}^{2d \times l}$, new data $\mathbf{D}^t \in \mathbb{R}^{2d \times m_t}$.

Output: New sketching matrix $\mathbf{B}^t \in \mathbb{R}^{2d \times l}$, and $\tilde{\mathbf{U}}_l^t \in \mathbb{R}^{2d \times l}$.

1: $\mathbf{C}^t = [\mathbf{B}^{t-1}, \mathbf{D}^t] \in \mathbb{R}^{2d \times (l+m_t)}$;

2: $[\tilde{\mathbf{U}}_l^t, \tilde{\mathbf{\Sigma}}_l^t, \tilde{\mathbf{V}}_l^t] = \mathrm{SVD}_l(\mathbf{C}^t)$;

3: $\hat{\mathbf{\Sigma}}_l^t = \mathrm{diag}(\sqrt{\tilde{\sigma}_1^2 - \tilde{\sigma}_l^2}, \sqrt{\tilde{\sigma}_2^2 - \tilde{\sigma}_l^2}, ..., \sqrt{\tilde{\sigma}_{l-1}^2 - \tilde{\sigma}_l^2}, 0)$;

4: $\mathbf{B}^t = \tilde{\mathbf{U}}_l^t \hat{\mathbf{\Sigma}}_l^t$;

patterns in the observed links and node attributes can be summarized accurately. The detailed pseudocode of the summarization phase using matrix sketching is presented in Algorithm 4. As the number of new columns m_t is much smaller than the number of columns in $\mathbf{G}^t$, we only need to perform SVD on a low-rank matrix (Line 2); its computation is efficient with a complexity of $O(2d(m_t + l)l)$. Also, it is space efficient with a maximum overhead of $O(2d(max_t\{m_t\} + l))$ across all time stamps. All in all, the summarization phase makes only one pass of the data and is both computational and space efficient.

Infer Missing Links with Sketching Matrix. The low-rank sketching matrix $\mathbf{B}^t$ makes a structural summarization of the up-to-date observed data on dynamic attributed networks. Hence, we can leverage it to predict missing links on the fly. To show the underlying mechanism of the link prediction phase, we first assume that the feature representation $\mathbf{G}^t$ of the dynamic attributed network until time stamp t is available (which actually not). The original feature representation $\mathbf{G}^t$ could be very noisy, containing a certain amount of noisy and irrelevant attributes which may degrade the link prediction performance. On top of that, the link information of networks may also be noisy and even erroneous from a network analysis perspective. To alleviate the negative impacts from these noisy attributes and noisy links, we propose to use principal component analysis (PCA) (Jolliffe, 2011) to reduce the

noise hidden in the data stream. Formally, PCA projects the data in $\mathbf{G}^t$ onto several principal components such that the total data variance is minimized, and these principal components correspond to the top-k eigenvectors of the estimated covariance matrix $\frac{1}{c_t}\mathbf{G}^t(\mathbf{G}^t)'$, which is also equivalent to find the top-k left singular vectors of the matrix $\mathbf{G}^t$. Here, we denote the concatenation of the top-k eigenvectors as $\mathbf{U}_k^t \in \mathbb{R}^{2d\times k}$, and the principal components $\mathbf{U}_k^t$ is often regarded as a good rank-k basis to reconstruct all the data in the original representation $\mathbf{G}^t$. In addition, by using the linear combination of columns in $\mathbf{U}_k^t$ to represent columns in $\mathbf{G}^t$, the noisy information contained can be greatly reduced. As $\mathbf{U}_k^t$ provides a noise-resilient abstraction of patterns of connected node pairs, we can leverage the orthogonal basis to predict missing links for unconnected node pairs. In particular, let $\mathbf{y} \in \mathbb{R}^{2d}$ denotes the feature representation of an unconnected node pair for testing, if $\mathbf{y}$ is close to the space composed of columns in $\mathbf{U}_k^t$, most likely there will be a link between the starting node and the ending node of $\mathbf{y}$; otherwise, if $\mathbf{y}$ cannot be well reconstructed by the orthogonal basis $\mathbf{U}_k^t$, it implies that $\mathbf{y}$ deviates from the patterns of connected node pairs and the possibility that the two end nodes of $\mathbf{y}$ connected in the near future is low (Huang and Kasiviswanathan, 2015). The residual error of the reconstruction phase is $\|\mathbf{I} - \mathbf{U}_k^t(\mathbf{U}_k^t)'\mathbf{y}\|_2^2$. It can be observed that to obtain the residual error for each pair of unconnected nodes, only low-rank matrix multiplication operations are involved, and the computation cost is $O(2dk)$. Afterwards, we can use the residual error to predict whether there exists a link between a pair of unconnected nodes. Specifically, the smaller the residual error of a pair of unconnected nodes is, the higher the chance that they will be linked together in the near future.

The above phase assumes that the feature representation $\mathbf{G}^t$ is readily available before the link prediction takes place and the orthogonal basis $\mathbf{U}_k^t$ is the top-k left singular vectors of $\mathbf{G}^t$. However, as mentioned previously, the storage of the whole

feature representation $\mathbf{G}^t$ is intractable when the underlying attributed network is large; while on the other hand, the sketching matrix $\mathbf{B}^t$ is not only a low-rank matrix with light memory overhead, but also can approximate the original matrix $\mathbf{G}^t$ well. In this regard, we attempt to perform SVD on the low-rank matrix $\mathbf{B}^t$ instead of $\mathbf{G}^t$ to obtain the orthogonal basis, i.e., the top-k left singular vectors. Here, we denote these k left singular vectors of $\mathbf{B}^t$ as $\tilde{\mathbf{U}}_k^t \in \mathbb{R}^{2d \times k}$. And according to Algorithm 4, the sketching matrix $\mathbf{B}^t$ can be efficiently maintained and updated, and its top-l left singular vectors are $\tilde{\mathbf{U}}_l^t$. In this way, the approximation of the top-k left singular vectors of $\mathbf{G}^t$ can be directly obtained from $\tilde{\mathbf{U}}_l$ as long as the condition of $k \leq l$ is satisfied. And the residual error of the feature vector $\mathbf{y}$ is $\|\mathbf{I} - \tilde{\mathbf{U}}_k^t(\tilde{\mathbf{U}}_k^t)'\mathbf{y}\|_2^2$.

Threshold Determination. For the above phase, the potential links between unconnected nodes can be determined by verifying if the residual error is below a threshold. One simple solution to specify the threshold is to set it as a fixed value. Nonetheless, with the accumulation of new edges and new node attributes in a data stream over time, the intrinsic patterns of data change over time, and this phenomenon is often referred to as *concept drift* in data stream mining (Tsymbal, 2004). To this end, it is more appealing to continuously update the threshold value for link prediction such that the up-to-date patterns of data can be well captured. Concretely, we propose to obtain the threshold automatically from the presented data stream instead of manually setting it up. For each observed link, i.e., a column in $\mathbf{G}^t$, we calculate its residual error, where $\tilde{\mathbf{U}}_k^t$ can be obtained by the top-k left singular vectors of the current sketching matrix $\mathbf{B}^t$ (Algorithm 4). Let us denote the collection of residual errors of links in $\mathbf{G}^t$ as $R = \{error_1, error_2, ..., error_{c_t}\}$, then the residual error threshold that is used to check the existence of new links can be determined as the largest error among R. In this way, we do not need to manually specify the threshold value and it can be automatically determined from the observed links. Hence, the

Algorithm 5 SLIDE to Predict Missing Links at Time Stamp t.

Input: Sketching matrix $\mathbf{B}^{t-1} \in \mathbb{R}^{2d \times l}$ and its top-l left singular vectors $\tilde{\mathbf{U}}_l^{t-1}$, new data $\mathbf{D}^t \in \mathbb{R}^{2d \times m_t}$, residual error threshold ϵ^{t-1}, feature representation $\mathbf{y}$ of an unconnected node pair.

Output: If there exists a link between the two end nodes of $\mathbf{y}$.

1: Obtain the new sketching matrix $\mathbf{B}^t$ and its top-l left singular vectors $\tilde{\mathbf{U}}_l^t$ by Algorithm 4;
2: Obtain the top-k singular vectors $\tilde{\mathbf{U}}_k^t$ from $\tilde{\mathbf{U}}_l^t$ ($k \leq l$);
3: Calculate the residual error of links in $\mathbf{D}^t$;
4: Update the residual error threshold ϵ^t;
5: Calculate the residual error of $\mathbf{y}$ by $\|\mathbf{I} - \tilde{\mathbf{U}}_k^t(\tilde{\mathbf{U}}_k^t)^T)\mathbf{y}\|_2^2$;
6: **if** error of $\mathbf{y} \leq \epsilon^t$ **then**
7: There exists a future link between the two end nodes of $\mathbf{y}$;
8: **else**
9: The two end nodes of $\mathbf{y}$ will not be connected;
10: **end if**

whole procedure of the proposed SLIDE is summarized in Algorithm 5.

9.3 Experimental Evaluation

We perform experiments on real-world dynamic attributed networks to validate the effectiveness and efficiency of the proposed SLIDE framework. We attempt to answer two research questions: (1) How accurate is SLIDE in predicting missing links? (2) How efficient is SLIDE when measured against other offline methods?

Datasets. We collect three real-world dynamic attributed networks for experimental validation, these datasets range from social media networks to coauthor networks. The detailed descriptions of these three dynamic attributed networks are listed below. (1) Epinions is a product review site in which users build trust relationships to seek

advice from others and share their reviews about products. We take each user as a node and regard his/her reviews as node attributes. In particular, we first use the bag-of-words model to extract features from user reviews and then employ the above-mentioned unsupervised feature selection methods for networked data (Li *et al.*, 2016b) to find the top 100 important features closely correlated with the network topology. Both the network structure and the node attributes are evolving over time. In the collected dataset, there are 25 time stamps (with an interval of one month), the total number of nodes is 14,180 and the total number of edges is 308,136. (2) DBLP is an extracted coauthor network for the authors who publish at least three papers from the year of 1995 to 2011. On the network, each author corresponds to a node. And similar to Epinions, we apply the bag-of-words model and feature selection on the title of their publications to find the most relevant 100 node attributes, i.e., words. As authors gradually form new coauthor relations and their research interests evolve over time, the underlying network is naturally a dynamic attributed network. The resulted dataset has 100,924 nodes and 764,392 edges over 17 time stamps. (3) ACM is a similar coauthor network as DBLP. We extract a subgraph consisting of the authors who publish at least three papers in the year of 1995 and 2015, and apply the same mechanism as before to extract 100 important node attributes. Therefore, we obtain a dynamic attributed network with a total amount of 122,567 nodes and 1,551,554 edges over 16 different time stamps.

Experimental Settings. To verify the effectiveness and efficiency of the proposed SLIDE framework, we compare SLIDE with the following baseline link prediction methods from three different categories: (1) with only network structure; (2) with only node attributes; and (3) with both sources of information.

- Common Neighbors (CN) (Liben-Nowell and Kleinberg, 2007): CN quantifies the number of common users between node pairs for link prediction.

- Jaccard Coefficient (JC) (Liben-Nowell and Kleinberg, 2007): JC calculates the similarity of pairs of nodes for link prediction with Jaccard coefficient.

- Adamic-Adar (AA) (Liben-Nowell and Kleinberg, 2007): AA is an extension of CN which penalizes the common neighbors with high node degrees.

- Rooted PageRank (Tong *et al.*, 2006): It performs random walk with start from a root node and then determines the scores of links to other nodes from the root node.

- NMF (Lin, 2007): It conducts non-negative matrix factorization on the adjacency matrix of the network to calculate the scores of unconnected node pairs.

- SimAttr (Yin *et al.*, 2010): It calculates cosine similarity on node attributes and uses the similarity score to rank links.

- FactLog (Menon and Elkan, 2011): It adopts matrix factorization and incorporates both network structure and node attributes in a joint framework for link prediction. The loss function is set to be the log loss.

- AttriRank (Hsu *et al.*, 2017): It performs PageRank on the attributed networks and then the score of each node pair is determined as the product of the PageRank scores of two end nodes.

Among them, CN, JC, AA, Rooted PageRank, and NMF use only network information; SimAttr on the other hand only takes advantage of node attribute information; FactLog, AttriRank, and SLIDE are in the third category by combining both sources of information together for link prediction.

We first investigate the effectiveness of the proposed framework SLIDE. Given a dynamic attributed network with T different time stamps, for each time stamp t

$(1 \leq t \leq T)$, in the training phase, we first perform link prediction with the attributed network G^t, and then test the link prediction performance on G^{t+1}. It should be noted that as most of these baseline methods cannot handle cold-start nodes, we choose to predict the missing links for the nodes that appear in both G^t and G^{t+1}. More investigation on the link prediction for cold-start nodes will be presented later. As a final result, we output the average link prediction performance over $T-1$ test periods. In the experiments, we set the number of columns in the sketching matrix l according to the suggestions of (Huang and Kasiviswanathan, 2015). Meanwhile, we specify the parameter k the same as l.

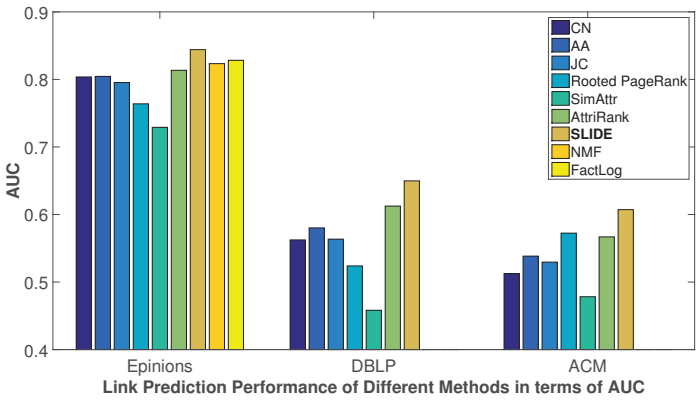
Suppose the number of new links for testing between time stamp t and $t+1$ is e_t, all these baseline methods can be regarded as a ranking model which returns the top e_t possible links from G_t and then compares with the ground truth links in G^{t+1}. To make a fair comparison between SLIDE and baseline methods, in the evaluation, we do not use the residual error threshold ϵ, instead, we rank the candidate links according to the residual errors. Three commonly used evaluation metrics are used to compare the link prediction performance of different methods. They are *area under the curve* (AUC) (Chang *et al.*, 2016), *mean average precision* (MAP) (Li *et al.*, 2010b), *half-life utility* (HLU) (Pan *et al.*, 2008). The higher the values of AUC, MAP, and HLU are, the better the prediction performance is. Specifically, at each time stamp during the testing phase, we treat all the e_t links that will happen at the next time stamp $t+1$ as positive samples, and the other links as negative links.

Different from SLIDE that only makes one pass of the data to predict missing link on the fly, all baseline methods are offline methods that require the access of the whole historical data each time when changes occur. In other words, they need to explicitly materialize the whole attributed networks in memory before link prediction takes place. To have a fair comparison between SLIDE and the baseline methods in

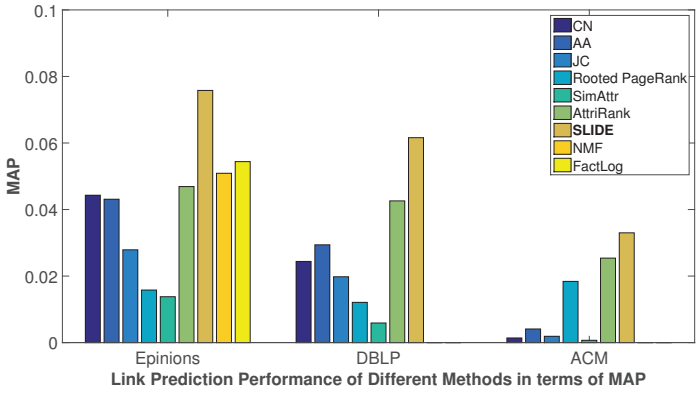
terms of efficiency, we allow the storage of the historical data for baseline methods in memory and compare their cumulative running time over all time stamps.

Effectiveness of SLIDE. First, we investigate the effectiveness of SLIDE by comparing its link prediction performance with the aforementioned baseline methods. The average link prediction results over multiple time stamps are presented in Figure 9.3. We make the following observations from the figure: (1) The proposed streaming link prediction framework SLIDE outperforms all baseline methods in almost all cases. We also perform a pairwise Wilcoxon signed-rank test between SLIDE and these baseline methods. The comparison results indicate that the proposed SLIDE framework is significantly better than others, with a significance level of 0.05. (2) CN, AA, JC, Rooted PageRank and NMF only leverage network structure for link prediction, and their performance is superior to SimAttr which relies on node attributes. It implies that the link prediction performance is influenced more by the network structure rather than the node attributes. (3) The methods FactLog, AttriRank and SLIDE that leverage two sources of information achieve better link prediction performance than methods with only one source of information. The observation supports the assumption that node attribute information compliments to network structure for link prediction. (4) We do not report the link prediction results of NMF and FactLog on DBLP and ACM datasets as we run out of memory for these two methods. The reason is that these two methods are both matrix factorization based methods and cannot be easily scaled to large-scale networks.

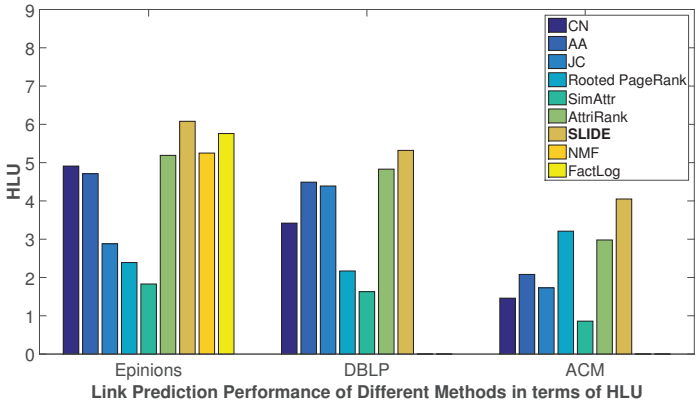
Efficiency of SLIDE. We investigate the second research question about the efficiency of SLIDE. Specifically, we report the cumulative running time of different methods across all time stamps in Table 9.1. As all the baseline methods mentioned are designed for static networks by assuming the materialization of the network structure in memory, we have to rerun these baseline methods repeatedly each time when



(a) AUC



(b) MAP



(c) HLU

Figure 9.3: Link Prediction Results Evaluation Between SLIDE and Baselines.

Table 9.1: Cumulative Running Time Comparison Between SLIDE and Baselines.

	Epinions	DBLP	ACM
CN	1245.41s	>3 hours	>3 hours
AA	6243.32s	>3 hours	>3 hours
JC	1489.81s	>3 hours	>3 hours
Rooted PageRank	305.41s	>3 hours	>3 hours
NMF	774.7s	>3 hours	>3 hours
SimAttr	259.09s	>3 hours	>3 hours
FactLog	7140.18s	>3 hours	>3 hours
AttriRank	2081.53s	>3 hours	>3 hours
SLIDE	**25.67s**	**291.31s**	**689.93s**

there are changes on the attributed networks. As can be observed from the table, our proposed SLIDE framework is significantly faster than all baseline methods. The overall running time of SLIDE on Epinions, DBLP, and ACM are 25.67 seconds, 291.31 seconds and 689.93 seconds, respectively. Specifically, SLIDE is 49×, 243×, 58×, 12×, 30×, 10×, 278×, and 81× faster than CN, AA, JC, Rooted PageRank, NMF, SimAttr, FactLog, AttriRank, respectively in Epinions. On DBLP and ACM datasets, the cumulative running time of all baseline methods cost more than 3 hours while our method finishes within minutes. In addition to that, as our proposed SLIDE framework maintains and updates a low-rank sketching matrix with light memory overhead, it is also much more space efficient than most baseline methods. All in all, SLIDE achieves promising link prediction performance within a favorable amount of running time with limited memory costs.

Link Prediction for Cold-Start Users. It has been widely known that in conventional link prediction problems, new users often suffer from the cold-start problems

Table 9.2: Link Prediction Results Evaluation for Cold-Start Users on Epinions.

Metrics	AUC	MAP	HLU
SimAttr	0.6828	0.0286	3.60
AttriRank	0.7067	0.0409	4.29
SLIDE	0.7532	0.0523	4.94

since we often do not have any data about newly joined users. Fortunately, the rich node attributes can help mitigate this critical issue when link information is not available. To investigate how well the proposed SLIDE framework handles new users for cold-start link prediction problem, we compare SLIDE with SimAttr and AttriRank, as these two methods can also handle the cold-start problem by leveraging node attributes. We focus on the Epinions dataset to investigate the cold-start problem as in DBLP and ACM datasets, authors create coauthor relations with other scholars the same time when they publish a paper and is therefore not suitable for cold-start problem study. In particular, in Epinions, users can first write reviews about products and then build trust relations with others, and we predict missing links for these new users by using their attribute information before they build any trust relations. The link prediction performance comparison in terms of these new users is illustrated in Table 9.2. It can be shown that SLIDE obtains better link prediction performance than SimAttr and AttriRank for the cold-start problem. The reason is that SLIDE summarizes the connectivity patterns of linked nodes in the sketching matrix; the orthogonal basis from the sketching matrix is noise resilient and can help us predict missing links more accurately.

9.4 Summary

A vast majority of existing link prediction algorithms are designed for static networks and assume that the whole network structure is materialized in memory before link prediction happens. However, many real-world networks are naturally dynamic and are characterized by frequent updates. The updates are often transient and could even be infinite, which puts the applicability of conventional link prediction algorithms in jeopardy. In addition to that, rich node attributes are prevalent and often have a strong connection with the network topology, and they may also change adaptively over time. It remains a daunting task to support the link prediction on such dynamic attributed networks in an online fashion due to some unique challenges. In this chapter, we study the novel problem of streaming link prediction on dynamic attributed networks and propose a sophisticated link prediction framework - SLIDE. In particular, we leverage a cost-effective matrix sketching technique to make a summarization of the current observed data by making only one pass of the data, and the sketching matrix, in turn, is used to infer the missing links. We also perform empirical experimental evaluations on real-world datasets, the results imply that SLIDE not only can predict the missing links more accurately but also is much more computationally efficient than competitors.

Part IV

Conclusion and Future Work

Chapter 10

CONCLUSION AND FUTURE WORK

In this chapter, we summarize the key contributions made in this dissertation and discuss promising future research directions.

10.1 Conclusion

Attributed networks naturally appear in a variety of high-impact domains and pose a number of fascinating research questions. In this dissertation, we are dedicated to developing principled learning algorithms and investigate novel applications on attributed networks, in order to have a better computational understanding and gain deeper insights into such unique data representation. The main thrusts of our research work in exploring attributed networks are summarized as follows.

- **Learning Algorithms in A Static Environment**: The first part of the dissertation focuses on developing principled learning algorithms for attributed networks in a static environment from two aspects with feature selection and network embedding. These research efforts are essential in building generalizable learning algorithms on attributed networks, especially when the supervision information is not available. Meanwhile, the end results of feature selection and network embedding can facilitate various downstream applications. For feature selection, we first propose a robust unsupervised framework for attributed networks - named NetFS (Chapter 3). Without label information to provide the guidelines, NetFS regards the latent representation of nodes as pseudo class labels and then embeds them as constraints to steer the selection of informa-

tive features. To fully exploit the finer-grained tie strength of links embedded on the network, we also propose an adaptive unsupervised feature selection framework ADAPT (Chapter 3). ADAPT assumes that the observed links in the network can be generated through informative features with a probabilistic framework. Then, to capture the inherent correlation between network structure and node attributes, ADAPT imposes a constraint on the link generation process to ensure that it preserves the adaptive neighborhood structure measured by the tie strength over the full spectrum. Different from feature selection which keeps a subset of original features, we also propose a noise-resilient consensus attributed network embedding framework DANE-O (Li *et al.*, 2017c) (Chapter 4) to project two different data modalities of attributed networks into a consensus space while maximizing their correlations. The utility of developed learning algorithms is demonstrated by their superior performance in the network clustering and node classification tasks.

- **Learning Algorithms in A Dynamic Environment**: Given the rapidly evolving nature of real-world attributed networks, conventional offline learning algorithms would suffer from serious computational bottlenecks, especially when fast-response is desired. Hence, it is of vital importance to develop online algorithms to quickly adapt the changes for real-time insights. My contributions in building online algorithms for attributed networks in a dynamic environment are in the following two aspects. Firstly, we investigate how to model the temporal dynamics of node attributes and network structure for online feature selection. The key idea of the developed online algorithm TeFS (Chapter 5) is to leverage the temporal smoothness property by assuming small changes of attributed networks within a short period of time. We also study attributed

network embedding in a dynamic setting by presenting an online embedding learning framework - DANE (Chapter 5). The essential idea is to replenish the freshness of the end embedding results with matrix perturbation theory. The effectiveness and efficiency of the developed online feature selection and network embedding algorithms are validated on various real-world datasets.

- **Applications of Attributed Networks**: In addition to building general learning algorithms through feature learning (including feature selection and attributed network embedding), tailoring the learning process for specific applications is another research direction we have pursued. Developing application-aware learning algorithms is more desired when we have a clear understanding of the application domain and its unique needs. We have worked on a variety of different applications on the attributed networks and in this dissertation, we will use three representative applications - personalized node classification, anomaly detection, and streaming link prediction to showcase how application understanding and learning algorithms mutually enhance each other. Firstly, we argue that the attribute information of different nodes could be quite diverse and even the same content could convey entirely different meanings. Hence, it would be more appealing to customize the learning and prediction for each node on the network by capturing its unique patterns. To solve this problem, we propose a personalized node classification framework PRL (Chapter 7) by finding a small subset of features customized for each individual and some common features shared by all for the node categorization. The proposed model not only makes accurate predictions but also is interpretable as we can explain why we make such a prediction. Secondly, we investigate the anomaly detection problem on attributed networks, where different types of anomalies are

often mixed together. Hence, we develop a general framework RADAR (Chapter 8) that explores and spots anomalies through the means of residual analysis. Empirical studies not only show the promising detection performance of the developed framework but also demonstrate its capability in finding anomalies that are otherwise undiscovered by other methods. At last, we study the streaming link prediction problem to support the prediction of missing links in an online fashion on dynamic attributed networks. The proposed SLIDE framework (Chapter 9) maintains and updates a low-rank sketching matrix to summarize all observed data, and we further leverage the sketching matrix for inference on the fly. The whole procedure is cost-effective, effective, and efficient.

10.2 Future Work

Attributed networks provide a powerful tool to model various modern information infrastructures, while effective and efficient learning algorithms play a central role in distilling insights or values from such networks in making impacts. My current research on attributed networks and feature learning poses a wealth of fascinating but challenging research questions that I plan to address in the future.

- **Scalable Feature Learning on Dynamic Attributed Networks.** Even though we have proposed a number of online algorithms to handle the dynamics of attributed networks for knowledge discovery on the fly. Most of these algorithms, however, are required to store the data in memory and access the data multiple times. In fact, real-world applications such as social media consistently and continuously generate massive semi-structured and unstructured data at an unprecedented rate, and the size of data is often measured in terabytes or even petabytes. These large-scale user-generated data is difficult to be materialized in memory; thus dynamic feature learning algorithms (including feature selec-

tion and embedding learning) need to be pass-efficient to make only one pass of the data as the further passes are either expensive or naturally impossible. To tackle the challenges, on one hand, we attempt to design cost-effective data synopsis with limited memory overhead to summarize the ever-increasing network structure and node attributes for learning. On the other hand, we also plan to study how dynamic feature learning algorithms can be designed in a distributed manner to deal with the massive amount of evolving and connected data. Hence, I plan to establish fundamental dynamic models on distributed platforms (e.g., Spark and Hadoop) to support the online processing of attributed networks for real-time insights.

- **Adversarial Attack and Learning on Attributed Networks.** Many researchers have shown that machine learning models, especially deep learning architectures, can be easily fooled or attacked. It results in serious societal concerns as machine learning models are often deployed in security-related applications, such as spam detection, financial fraud, and system diagnosis. Meanwhile, as most machine learning models are black-box in nature, it further exacerbates the vulnerabilities of the underlying system without intuitive model interpretation. Due to the strong modeling power of attributed networks in the physical world, we plan to investigate the adversarial learning on attributed networks, which is more challenging than attack and defense research on image and text data due to the discrete nature of node connections, as well as the bewildering combination of heterogeneous information sources. Our investigations will be in two folds: (1) *Attack* - Whether the attributed networks are vulnerable to data poisoning attacks? How to attack the attributed network when it is fast evolving? What kind of attacks (e.g., addition/deletion of edges,

perturbation of node features) has the highest catastrophic effects? (2) *Defense* - How to learn task-agnostic feature representations on attributed networks in the presence of adversarial attacks? How to build more robust task-oriented learning algorithms on attributed networks? How to measure the robustness of the dynamic system?

- **Interplay between Feature Selection and Deep Learning.** The success of deep learning can partly be attributed to the carefully designed deep neural network architectures. However, when there is little prior knowledge about the underlying characteristics of data structure, the quality of input features can be instrumental for deep learning's success. Meanwhile, deep learning models require more training samples to train a generalizable model to prevent overfitting, while in many applications such as health informatics and bioinformatics, we often only have a limited amount of data, far from sufficient. In this regard, we will investigate how to leverage the strength of feature selection to shatter the barriers of deep learning for *small* data. Another challenge deep learning users often face is model interpretability, or a black box, in the sense that the learned feature representation is generally not understandable by human experts. The success of feature selection in building interpretable model motivates us to research if feature selection can help improve the comprehensibility of deep learning to make its models more transparent. Besides, we plan to lay down the theoretical foundations of feature selection for deep learning. Many intriguing questions await: (1) Do we really need so much data to perform deep learning? (2) How many instances do we need to guarantee a certain level of generation accuracy? (3) How to measure the feasibility of feature selection for deep learning?

- **Facilitate Human-in-the-Loop Learning for Graph Mining.** Graph or network data accounts for a large portion of real-world datasets. The ultimate goal of graph mining is to develop principled learning algorithms to help users to better understand and make sense of network data. However, most of the current methodologies are mostly data-driven and conducted in a passive fashion, and thus are inadequate to apprehend idiosyncratic human intents and demands. In other words, network mining algorithms will work much better when accounting the valuable human cognition and physiological characteristics, by continuously adapting the learning strategies driven by human feedback. Hence, we attempt to develop human-centric learning frameworks and prototype systems to facilitate human-in-the-loop learning on networks. We hope the research can fundamentally change the sense-making process of humans in various domains (e.g., social media, e-commerce, education, and security) and improve the utilities of existing data-driven network mining algorithms and systems. We have done some preliminary work on human-in-the-loop learning in anomaly detection (Ding *et al.*, 2019), and questions recommendation of online education (Teng *et al.*, 2018). In the future, we will continue our explorations to investigate: (1) How to transform the human cognition into concrete data or knowledge to advance human-in-the-loop learning on networks? (2) How to incorporate the diversified human needs into the interactive learning process? (3) How to adapt the human-in-the-loop learning to large-scale, heterogeneous, and dynamic networks?

Abufouda, M. and K. A. Zweig, "Link classification and tie strength ranking in online social networks with exogenous interaction networks", arXiv preprint arXiv:1708.04030 (2017).

Adamic, L. A. and B. A. Huberman, "Power-law distribution of the world wide web", Science **287**, 5461, 2115–2115 (2000).

Aggarwal, C. and K. Subbian, "Evolutionary network analysis: A survey", ACM Computing Surveys **47**, 1, 10 (2014).

Aggarwal, C. C., "On classification of graph streams", in "SIAM International Conference on Data Mining", pp. 652–663 (2011).

Aggarwal, C. C., "Outlier analysis", in "Data Mining", (2015).

Aggarwal, C. C. and N. Li, "On node classification in dynamic content-based networks", in "SIAM International Conference on Data Mining", pp. 355–366 (2011).

Aggarwal, C. C., Y. Zhao and P. S. Yu, "On clustering graph streams", in "SIAM International Conference on Data Mining", pp. 478–489 (2010).

Aggarwal, C. C., Y. Zhao and P. S. Yu, "Outlier detection in graph streams", in "IEEE International Conference on Data Engineering", pp. 399–409 (2011).

Ahmed, A., N. Shervashidze, S. Narayanamurthy, V. Josifovski and A. J. Smola, "Distributed large-scale natural graph factorization", in "International Conference on World Wide Web", pp. 37–48 (2013).

Airoldi, E. M., D. M. Blei, S. E. Fienberg and E. P. Xing, "Mixed membership stochastic blockmodels", Journal of Machine Learning Research **9**, Sep, 1981–2014 (2008).

Akoglu, L., H. Tong, B. Meeder and C. Faloutsos, "Pics: Parameter-free identification of cohesive subgroups in large attributed graphs", in "SIAM International Conference on Data Mining", pp. 439–450 (2012).

Alelyani, S., J. Tang and H. Liu, "Feature selection for clustering: A review", in "Data Clustering: Algorithms and Applications", pp. 29–60 (CRC, 2013).

Anava, O. and K. Levy, "k*-nearest neighbors: From global to local", in "Advances in Neural Information Processing Systems", pp. 4916–4924 (2016).

Araujo, M. R., P. M. P. Ribeiro and C. Faloutsos, "Tensorcast: Forecasting with context using coupled tensors", in "IEEE International Conference on Data Mining", pp. 71–80 (2017).

Barbieri, N., F. Bonchi and G. Manco, "Who to follow and why: Link prediction with explanations", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 1266–1275 (2014).

Belkin, M. and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering", in "Advances in Neural Information Processing Systems", pp. 585–591 (2002).

Bertsekas, D. P., *Nonlinear Programming* (Athena Scientific, 1999).

Bhagat, S., G. Cormode and S. Muthukrishnan, "Node classification in social networks", in "Social Network Data Analytics", pp. 115–148 (Springer, 2011).

Bishop, C., "Pattern recognition and machine learning", Pattern Recognition and Machine Learning (2006).

Bojchevski, A. and S. Günnemann, "Bayesian robust attributed graph clustering: Joint learning of partial anomalies and group structure", in "AAAI Conference on Artificial Intelligence", pp. 2738–2745 (2018).

Breunig, M. M., H.-P. Kriegel, R. T. Ng and J. Sander, "Lof: Identifying density-based local outliers", in "ACM sigmod record", vol. 29, pp. 93–104 (ACM, 2000).

Cai, D., X. He, J. Han and T. S. Huang, "Graph regularized nonnegative matrix factorization for data representation", IEEE Transactions on Pattern Analysis and Machine Intelligence **33**, 8, 1548–1560 (2010a).

Cai, D., C. Zhang and X. He, "Unsupervised feature selection for multi-cluster data", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 333–342 (2010b).

Cao, S., W. Lu and Q. Xu, "Grarep: Learning graph representations with global structural information", in "ACM International on Conference on Information and Knowledge Management", pp. 891–900 (2015).

Cao, S., W. Lu and Q. Xu, "Deep neural networks for learning graph representations", in "AAAI Conference on Artificial Intelligence", pp. 1145–1152 (2016).

Chakrabarti, D., R. Kumar and A. Tomkins, "Evolutionary clustering", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 554–560 (2006).

Chandola, V., A. Banerjee and V. Kumar, "Anomaly detection: A survey", ACM Computing Surveys **41**, 3, 15 (2009).

Chang, S., W. Han, J. Tang, G.-J. Qi, C. C. Aggarwal and T. S. Huang, "Heterogeneous network embedding via deep architectures", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 119–128 (2015).

Chang, S., Y. Zhang, J. Tang, D. Yin, Y. Chang, M. A. Hasegawa-Johnson and T. S. Huang, "Positive-unlabeled learning in streaming networks", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 755–764 (2016).

Chen, C. and H. Tong, "Fast eigen-functions tracking on dynamic graphs", in "SIAM International Conference on Data Mining", pp. 559–567 (2015).

Chen, T. and Y. Sun, "Task-guided and path-augmented heterogeneous network embedding for author identification", in "ACM International Conference on Web Search and Data Mining", pp. 295–304 (2017).

Cheng, K., J. Li and H. Liu, "Unsupervised feature selection in signed social networks", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 777–786 (2017).

Chi, Y., X. Song, D. Zhou, K. Hino and B. L. Tseng, "Evolutionary spectral clustering by incorporating temporal smoothness", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 153–162 (2007).

Chi, Y., X. Song, D. Zhou, K. Hino and B. L. Tseng, "On evolutionary spectral clustering", ACM Transactions on Knowledge Discovery from Data **3**, 4, 17 (2009).

Cook, R. D. and S. Weisberg, *Residuals and Influence in Regression* (New York: Chapman and Hall, 1982).

Cover, T. M. and J. A. Thomas, *Elements of Information Theory* (John Wiley & Sons, 2012).

Crandall, D., D. Cosley, D. Huttenlocher, J. Kleinberg and S. Suri, "Feedback effects between similarity and social influence in online communities", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 160–168 (2008).

Demšar, J., "Statistical comparisons of classifiers over multiple data sets", Journal of Machine Learning Research **7**, Jan, 1–30 (2006).

Ding, K., J. Li and H. Liu, "Interactive anomaly detection on attributed networks", in "ACM International Conference on Web Search and Data Mining", pp. 357–365 (2019).

Dong, Y., N. V. Chawla and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 135–144 (2017).

Doreian, P., "Network autocorrelation models: Problems and prospects", Spatial Statistics: Past, Present, Future pp. 369–89 (1989).

Du, B., S. Zhang, N. Cao and H. Tong, "First: Fast interactive attributed subgraph matching", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 1447–1456 (2017).

Du, L. and Y.-D. Shen, "Unsupervised feature selection with adaptive structure learning", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 209–218 (2015).

Dy, J. G. and C. E. Brodley, "Feature subset selection and order identification for unsupervised learning", in "International Conference on Machine Learning", pp. 247–254 (2000).

Eswaran, D., R. Rabbany, A. W. Dubrawski and C. Faloutsos, "Social-affiliation networks: Patterns and the soar model", in "Joint European Conference on Machine Learning and Knowledge Discovery in Databases", pp. 105–121 (2018).

Fang, Y., R. Cheng, S. Luo and J. Hu, "Effective community search for large attributed graphs", VLDB Endowment **9**, 12, 1233–1244 (2016).

Farahat, A. K., A. Ghodsi and M. S. Kamel, "Efficient greedy feature selection for unsupervised learning", Knowledge and Information Systems **35**, 2, 285–310 (2013).

Feigenbaum, J., S. Kannan, A. McGregor, S. Suri and J. Zhang, "On graph problems in a semi-streaming model", Theoretical Computer Science **348**, 2-3, 207–216 (2005).

Friedman, J., T. Hastie and R. Tibshirani, *The Elements of Statistical Learning* (Springer Series in Statistics, 2001).

Gao, B., T.-Y. Liu, W. Wei, T. Wang and H. Li, "Semi-supervised ranking on very large graphs with rich metadata", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 96–104 (2011a).

Gao, J., F. Liang, W. Fan, C. Wang, Y. Sun and J. Han, "On community outliers and their efficient detection in information networks", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 813–822 (2010).

Gao, S., L. Denoyer and P. Gallinari, "Temporal link prediction by integrating content and structure information", in "ACM International Conference on Information and Knowledge Management", pp. 1169–1174 (2011b).

Getoor, L. and C. P. Diehl, "Link mining: A survey", ACM SIGKDD Explorations Newsletter **7**, 2, 3–12 (2005).

Getoor, L. and B. Taskar, *Introduction to Statistical Relational Learning* (MIT Press Cambridge, 2007).

Gilbert, E. and K. Karahalios, "Predicting tie strength with social media", in "SIGCHI Conference on Human Factors in Computing Systems", pp. 211–220 (2009).

Golub, G. H. and C. F. Van Loan, *Matrix Computations*, vol. 3 (2012).

Gong, N. Z., A. Talwalkar, L. Mackey, L. Huang, E. C. R. Shin, E. Stefanov, E. R. Shi and D. Song, "Joint link prediction and attribute inference using a social-attribute network", ACM Transactions on Intelligent Systems and Technology **5**, 2, 27 (2014).

Granovetter, M. S., "The strength of weak ties", American Journal of Sociology **78**, 6, 1360–1380 (1973).

Grover, A. and J. Leskovec, "node2vec: Scalable feature learning for networks", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 855–864 (2016).

Gu, Q., C. Aggarwal and J. Han, "Unsupervised link selection in networks", in "International Conference on Artificial Intelligence and Statistics", pp. 298–306 (2013).

Gu, Q. and J. Han, "Towards feature selection in network", in "ACM International Conference on Information and Knowledge Management", pp. 1175–1184 (2011).

Gu, Q., Z. Li and J. Han, "Generalized fisher score for feature selection", in "International Conference on Uncertainty in Artificial Intelligence", pp. 266–273 (2012).

Gunnemann, S., I. Farber, B. Boden and T. Seidl, "Subspace clustering meets dense subgraph mining: A synthesis of two paradigms", in "IEEE International Conference on Data Mining", pp. 845–850 (2010).

Gunnemann, S., I. Farber, S. Raubach and T. Seidl, "Spectral subspace clustering for graphs with feature vectors", in "IEEE International Conference on Data Mining", pp. 231–240 (2013).

Guo, T., X. Zhu, J. Pei and C. Zhang, "Snoc: Streaming network node classification", in "IEEE International Conference on Data Mining", pp. 150–159 (2014).

Gupta, M., J. Gao, Y. Sun and J. Han, "Integrating community matching and outlier detection for mining evolutionary community outliers", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 859–867 (2012).

Guyon, I. and A. Elisseeff, "An introduction to variable and feature selection", Journal of Machine Learning Research **3**, Mar, 1157–1182 (2003).

Guyon, I., J. Weston, S. Barnhill and V. Vapnik, "Gene selection for cancer classification using support vector machines", Machine Learning **46**, 1-3, 389–422 (2002).

Hallac, D., J. Leskovec and S. Boyd, "Network lasso: Clustering and optimization in large graphs", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 387–396 (2015).

Hamilton, W., Z. Ying and J. Leskovec, "Inductive representation learning on large graphs", in "Advances in Neural Information Processing Systems", pp. 1024–1034 (2017).

Hardoon, D. R., S. Szedmak and J. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods", Neural Computation **16**, 12, 2639–2664 (2004).

He, X., D. Cai and P. Niyogi, "Laplacian score for feature selection", in "Advances in Neural Information Processing Systems", pp. 507–514 (2005).

He, X., D. Cai and P. Niyogi, "Laplacian score for feature selection", in "Advances in Neural Information Processing Systems", pp. 507–514 (2006).

He, Z., S. Xie, R. Zdunek, G. Zhou and A. Cichocki, "Symmetric nonnegative matrix factorization: Algorithms and applications to probabilistic clustering", IEEE Transactions on Neural Networks **22**, 12, 2117–2131 (2011).

Heimann, M., H. Shen, T. Safavi and D. Koutra, "Regal: Representation learning-based graph alignment", in "ACM International Conference on Information and Knowledge Management", pp. 117–126 (2018).

Hornsey, M. J., "Social identity theory and self-categorization theory: A historical review", Social and Personality Psychology Compass **2**, 1, 204–222 (2008).

Hsu, C.-C., Y.-A. Lai, W.-H. Chen, M.-H. Feng and S.-D. Lin, "Unsupervised ranking using graph structures and node attributes", in "ACM International Conference on Web Search and Data Mining", pp. 771–779 (2017).

Huang, H. and S. P. Kasiviswanathan, "Streaming anomaly detection using randomized matrix sketching", VLDB Endowment **9**, 3, 192–203 (2015).

Huang, H., S. Yoo and S. P. Kasiviswanathan, "Unsupervised feature selection on data streams", in "ACM International on Conference on Information and Knowledge Management", pp. 1031–1040 (2015).

Huang, X., J. Li and X. Hu, "Accelerated attributed network embedding", in "SIAM International Conference on Data Mining", pp. 633–641 (2017a).

Huang, X., J. Li and X. Hu, "Label informed attributed network embedding", in "ACM International Conference on Web Search and Data Mining", pp. 731–739 (2017b).

Huang, X., Q. Song, J. Li and X. Hu, "Exploring expert cognition for attributed network embedding", in "ACM International Conference on Web Search and Data Mining", pp. 270–278 (2018).

Jensen, D., J. Neville and B. Gallagher, "Why collective inference improves relational classification", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 593–598 (2004).

Jian, L., J. Li and H. Liu, "Toward online node classification on streaming networks", Data Mining and Knowledge Discovery **32**, 1, 231–257 (2018).

Jolliffe, I., *Principal Component Analysis* (Springer, 2011).

Keogh, E. and A. Mueen, "Curse of dimensionality", in "Encyclopedia of Machine Learning", pp. 257–258 (Springer, 2011).

Kim, M.-S. and J. Han, "A particle-and-density based evolutionary clustering method for dynamic networks", VLDB Endowment **2**, 1, 622–633 (2009).

Kipf, T. N. and M. Welling, "Semi-supervised classification with graph convolutional networks", arXiv preprint arXiv:1609.02907 (2016).

Koller, D., N. Friedman, S. Džeroski, C. Sutton, A. McCallum, A. Pfeffer, P. Abbeel, M.-F. Wong, D. Heckerman, C. Meek *et al.*, *Introduction to Statistical Relational Learning* (MIT press, 2007).

Kong, D., R. Fujimaki, J. Liu, F. Nie and C. Ding, "Exclusive feature learning on arbitrary structures via $\ell_{1,2}$ -norm", in "Advances in Neural Information Processing Systems", pp. 1655–1663 (2014).

Kong, D., J. Liu, B. Liu and X. Bao, "Uncorrelated group lasso", in "AAAI Conference on Artificial Intelligence", pp. 1765–1771 (2016).

Kuang, D., C. Ding and H. Park, "Symmetric nonnegative matrix factorization for graph clustering", in "SIAM International Conference on Data Mining", pp. 106–117 (2012).

La Fond, T. and J. Neville, "Randomization tests for distinguishing social influence and homophily effects", in "International Conference on World Wide Web", pp. 601–610 (2010).

Lee, D. D. and H. S. Seung, "Algorithms for non-negative matrix factorization", in "Advances in Neural Information Processing Systems", pp. 556–562 (2001).

Lee, J., K. Park and S. Prabhakar, "Mining statistically significant attribute associations in attributed graphs", in "IEEE International Conference on Data Mining", pp. 991–996 (2016).

Leskovec, J., L. Backstrom, R. Kumar and A. Tomkins, "Microscopic evolution of social networks", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 462–470 (2008).

Leskovec, J., J. Kleinberg and C. Faloutsos, "Graphs over time: Densification laws, shrinking diameters and possible explanations", in "ACM SIGKDD International Conference on Knowledge Discovery in Data Mining", pp. 177–187 (2005).

Li, A. Q., A. Ahmed, S. Ravi and A. J. Smola, "Reducing the sampling complexity of topic models", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 891–900 (2014a).

Li, J., K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang and H. Liu, "Feature selection: A data perspective", ACM Computing Surveys **50**, 6, 94 (2017a).

Li, J., K. Cheng, L. Wu and H. Liu, "Streaming link prediction on dynamic attributed networks", in "ACM International Conference on Web Search and Data Mining", pp. 369–377 (2018a).

Li, J., H. Dani, X. Hu and H. Liu, "Radar: Residual analysis for anomaly detection in attributed networks", in "International Joint Conference on Artificial Intelligence", pp. 2152–2158 (2017b).

Li, J., H. Dani, X. Hu, J. Tang, Y. Chang and H. Liu, "Attributed network embedding for learning in a dynamic environment", in "ACM International Conference on Information and Knowledge Management", pp. 387–396 (2017c).

Li, J., R. Guo, C. Liu and H. Liu, "Adaptive unsupervised feature selection on attributed networks", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", (2019).

Li, J., X. Hu, L. Jian and H. Liu, "Toward time-evolving feature selection on dynamic networks", in "IEEE International Conference on Data Mining", pp. 1003–1008 (2016a).

Li, J., X. Hu, J. Tang and H. Liu, "Unsupervised streaming feature selection in social media", in "ACM International on Conference on Information and Knowledge Management", pp. 1041–1050 (2015).

Li, J., X. Hu, L. Wu and H. Liu, "Robust unsupervised feature selection on networked data", in "SIAM International Conference on Data Mining", pp. 387–395 (2016b).

Li, J. and H. Liu, "Challenges of feature selection for big data analytics", IEEE Intelligent Systems **32**, 2, 9–15 (2017).

Li, J., J. Tang and H. Liu, "Reconstruction-based unsupervised feature selection: An embedded approach", in "International Joint Conference on Artificial Intelligence", pp. 2159–2165 (2017d).

Li, J., L. Wu, H. Dani and H. Liu, "Unsupervised personalized feature selection", in "AAAI Conference on Artificial Intelligence", pp. 3514–3521 (2018b).

Li, J., L. Wu, O. R. Zaïane and H. Liu, "Toward personalized relational learning", in "SIAM International Conference on Data Mining", pp. 444–452 (2017e).

Li, L., W. Chu, J. Langford and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation", in "International Conference on World Wide Web", pp. 661–670 (2010a).

Li, X., N. Du, H. Li, K. Li, J. Gao and A. Zhang, "A deep learning approach to link prediction in dynamic networks", in "SIAM International Conference on Data Mining", pp. 289–297 (2014b).

Li, Y., J. Hu, C. Zhai and Y. Chen, "Improving one-class collaborative filtering by incorporating rich user information", in "ACM International Conference on Information and Knowledge Management", pp. 959–968 (2010b).

Li, Z., Y. Yang, J. Liu, X. Zhou, H. Lu *et al.*, "Unsupervised feature selection using nonnegative spectral analysis", in "AAAI Conference on Artificial Intelligence", pp. 1026–1032 (2012).

Liben-Nowell, D. and J. Kleinberg, "The link-prediction problem for social networks", Journal of the American Society for Information Science and Technology **58**, 7, 1019–1031 (2007).

Liberty, E., "Simple and deterministic matrix sketching", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 581–588 (2013).

Lin, C.-J., "Projected gradient methods for nonnegative matrix factorization", Neural Computation **19**, 10, 2756–2779 (2007).

Liu, H. and L. Yu, "Toward integrating feature selection algorithms for classification and clustering", IEEE Transactions on Knowledge and Data Engineering **17**, 4, 491–502 (2005).

Ma, Y., Z. Ren, Z. Jiang, J. Tang and D. Yin, "Multi-dimensional network embedding with hierarchical structure", in "ACM International Conference on Web Search and Data Mining", pp. 387–395 (2018).

Macskassy, S. A. and F. Provost, "A simple relational classifier", Tech. rep., DTIC Document (2003).

Manzoor, E., S. M. Milajerdi and L. Akoglu, "Fast memory-efficient anomaly detection in streaming heterogeneous graphs", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 1035–1044 (2016).

Marsden, P. V. and N. E. Friedkin, "Network studies of social influence", Sociological Methods & Research **22**, 1, 127–151 (1993).

McGregor, A., "Graph stream algorithms: A survey", ACM SIGMOD Record **43**, 1, 9–20 (2014).

McPherson, M., L. Smith-Lovin and J. M. Cook, "Birds of a feather: Homophily in social networks", Annual Review of Sociology **27**, 1, 415–444 (2001).

Menon, A. K. and C. Elkan, "Link prediction via matrix factorization", in "Joint European Conference on Machine Learning and Knowledge Discovery in Databases", pp. 437–452 (2011).

Mikolov, T., I. Sutskever, K. Chen, G. S. Corrado and J. Dean, "Distributed representations of words and phrases and their compositionality", in "Advances in Neural Information Processing Systems", pp. 3111–3119 (2013).

Mislove, A., B. Viswanath, K. P. Gummadi and P. Druschel, "You are who you know: Inferring user profiles in online social networks", in "ACM International Conference on Web Search and Data Mining", pp. 251–260 (2010).

Müller, E., P. I. Sánchez, Y. Mülle and K. Böhm, "Ranking outlier nodes in subspaces of attributed graphs", in "IEEE International Conference on Data Engineering Workshops", pp. 216–222 (2013).

Namata, G. M., H. Sharara and L. Getoor, "A survey of link mining tasks for analyzing noisy and incomplete networks", in "Link Mining: Models, Algorithms, and Applications", pp. 107–133 (Springer, 2010).

Newman, M., "Network structure from rich but noisy data", Nature Physics **14**, 6, 542 (2018a).

Newman, M., *Networks* (Oxford university press, 2018b).

Newman, M. E., "Mixing patterns in networks", Physical Review E **67**, 2, 026126 (2003).

Newman, M. E., "Modularity and community structure in networks", Proceedings of the National Academy of Sciences **103**, 23, 8577–8582 (2006).

Newman, M. E. and M. Girvan, "Finding and evaluating community structure in networks", Physical Review E **69**, 2, 026113 (2004).

Nie, F., H. Huang, X. Cai and C. H. Ding, "Efficient and robust feature selection via joint l2, 1-norms minimization", in "Advances in Neural Information Processing Systems", pp. 1813–1821 (2010).

Nigam, A., K. Shin, A. Bahulkar, B. Hooi, D. Hachen, B. K. Szymanski, C. Faloutsos and N. V. Chawla, "One-m: Modeling the co-evolution of opinions and network connections", in "Joint European Conference on Machine Learning and Knowledge Discovery in Databases", pp. 122–140 (2018).

Ning, H., W. Xu, Y. Chi, Y. Gong and T. Huang, "Incremental spectral clustering with application to monitoring of evolving blog communities", in "SIAM International Conference on Data Mining", pp. 261–272 (2007).

Ou, M., P. Cui, J. Pei, Z. Zhang and W. Zhu, "Asymmetric transitivity preserving graph embedding", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 1105–1114 (2016).

Pan, R., Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz and Q. Yang, "One-class collaborative filtering", in "IEEE International Conference on Data Mining", pp. 502–511 (2008).

Papalexakis, E. E., C. Faloutsos and N. D. Sidiropoulos, "Tensors for data mining and data fusion: Models, applications, and scalable algorithms", ACM Transactions on Intelligent Systems and Technology (TIST) **8**, 2, 16 (2017).

Parlett, B. N., *The Symmetric Eigenvalue Problem*, vol. 7 (1980).

Peng, H., F. Long and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy", IEEE Transactions on Pattern Analysis and Machine Intelligence **27**, 8, 1226–1238 (2005).

Peng, Z., M. Luo, J. Li, H. Liu and Q. Zheng, "Anomalous: A joint modeling approach for anomaly detection on attributed networks", in "International Joint Conference on Artificial Intelligence", pp. 3513–3519 (2018).

Perozzi, B. and L. Akoglu, "Scalable anomaly ranking of attributed neighborhoods", in "SIAM International Conference on Data Mining", pp. 207–215 (2016).

Perozzi, B. and L. Akoglu, "Discovering communities and anomalies in attributed graphs: Interactive visual exploration and summarization", ACM Transactions on Knowledge Discovery from Data **12**, 2, 24 (2018).

Perozzi, B., L. Akoglu, P. Iglesias Sánchez and E. Müller, "Focused clustering and outlier detection in large attributed graphs", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 1346–1355 (2014a).

Perozzi, B., R. Al-Rfou and S. Skiena, "Deepwalk: Online learning of social representations", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 701–710 (2014b).

Pfeiffer III, J. J., S. Moreno, T. La Fond, J. Neville and B. Gallagher, "Attributed graph models: Modeling network structure with correlated attributes", in "International Conference on World Wide Web", pp. 831–842 (2014).

Qian, M. and C. Zhai, "Robust unsupervised feature selection", in "International Joint Conference on Artificial Intelligence", pp. 1621–1627 (2013).

Qiu, J., Y. Dong, H. Ma, J. Li, K. Wang and J. Tang, "Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec", in "ACM International Conference on Web Search and Data Mining", pp. 459–467 (2018).

Rabbany, R., D. Eswaran, A. W. Dubrawski and C. Faloutsos, "Beyond assortativity: Proclivity index for attributed networks (prone)", in "Pacific-Asia Conference on Knowledge Discovery and Data Mining", pp. 225–237 (2017).

Ranshous, S., S. Shen, D. Koutra, S. Harenberg, C. Faloutsos and N. F. Samatova, "Anomaly detection in dynamic networks: A survey", Wiley Interdisciplinary Reviews: Computational Statistics **7**, 3, 223–247 (2015).

Rendle, S., "Factorization machines", in "IEEE International Conference on Data Mining", pp. 995–1000 (2010).

Rezaei, A., B. Perozzi and L. Akoglu, "Ties that bind: Characterizing classes by attributes and social ties", in "International Conference on World Wide Web Companion", pp. 973–981 (2017).

Robles, P., S. Moreno and J. Neville, "Sampling of attributed networks from hierarchical generative models", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 1155–1164 (2016).

Robnik-Šikonja, M. and I. Kononenko, "Theoretical and empirical analysis of relieff and rrelieff", Machine Learning **53**, 1-2, 23–69 (2003).

Rossi, R. and N. Ahmed, "The network data repository with interactive graph analytics and visualization", in "AAAI Conference on Artificial Intelligence", (2015).

Roweis, S. T. and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding", Science **290**, 5500, 2323–2326 (2000).

Ruan, Y., D. Fuhry and S. Parthasarathy, "Efficient community detection in large networks using content and links", in "International Conference on World Wide Web", pp. 1089–1098 (2013).

Sahoo, D., C. Liu and S. C. Hoi, "Malicious url detection using machine learning: A survey", arXiv preprint arXiv:1701.07179 (2017).

Sakr, S., S. Elnikety and Y. He, "G-sparql: A hybrid engine for querying large attributed graphs", in "ACM International Conference on Information and Knowledge Management", pp. 335–344 (2012).

Sánchez, P. I., E. Müller, O. Irmler and K. Böhm, "Local context selection for outlier ranking in graphs with multiple numeric node attributes", in "International Conference on Scientific and Statistical Database Management", p. 16 (2014).

Sánchez, P. I., E. Muller, F. Laforet, F. Keller and K. Bohm, "Statistical selection of congruent subspaces for mining attributed graphs", in "IEEE International Conference on Data Mining", pp. 647–656 (2013).

Sarkar, P., D. Chakrabarti and M. Jordan, "Nonparametric link prediction in dynamic networks", arXiv preprint arXiv:1206.6394 (2012).

Sarwar, B., G. Karypis, J. Konstan and J. Riedl, "Incremental singular value decomposition algorithms for highly scalable recommender systems", in "International Conference on Computer and Information Science", pp. 27–28 (2002).

Sen, P., G. Namata, M. Bilgic, L. Getoor, B. Galligher and T. Eliassi-Rad, "Collective classification in network data", AI magazine **29**, 3, 93 (2008).

She, Y. and A. B. Owen, "Outlier detection using nonconvex penalized regression", Journal of the American Statistical Association **106**, 494, 626–639 (2011).

Silva, A., W. Meira Jr and M. J. Zaki, "Mining attribute-structure correlated patterns in large attributed graphs", VLDB Endowment **5**, 5, 466–477 (2012).

Singh, A. P. and G. J. Gordon, "Relational learning via collective matrix factorization", in "SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 650–658 (2008).

Spiliopoulou, M., "Evolution in social networks: A survey", in "Social Network Data Analytics", pp. 149–175 (Springer, 2011).

Stewart, G. and J. Sun, "Matrix perturbation theory", (1990).

Tajfel, H., *Social Identity and Intergroup Relations* (Cambridge University Press, 2010).

Tang, J., S. Alelyani and H. Liu, "Feature selection for classification: A review", in "Data Classification: Algorithms and Applications", pp. 37–64 (CRC Press, 2014).

Tang, J. and H. Liu, "Feature selection with linked data in social media", in "SIAM International Conference on Data Mining", pp. 118–128 (2012a).

Tang, J. and H. Liu, "Unsupervised feature selection for linked social media data", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 904–912 (2012b).

Tang, J. and H. Liu, "Coselect: Feature selection with instance selection for social media data", in "SIAM International Conference on Data Mining", pp. 695–703 (2013).

Tang, J., J. Liu, M. Zhang and Q. Mei, "Visualizing large-scale and high-dimensional data", in "International Conference on World Wide Web", pp. 287–297 (2016).

Tang, J., M. Qu and Q. Mei, "Pte: Predictive text embedding through large-scale heterogeneous text networks", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 1165–1174 (2015a).

Tang, J., M. Qu, M. Wang, M. Zhang, J. Yan and Q. Mei, "Line: Large-scale information network embedding", in "International Conference on World Wide Web", pp. 1067–1077 (2015b).

Tang, L. and H. Liu, "Relational learning via latent social dimensions", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 817–826 (2009).

Tang, L., H. Liu, J. Zhang and Z. Nazeri, "Community evolution in dynamic multi-mode networks", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 677–685 (2008).

Taskar, B., E. Segal and D. Koller, "Probabilistic classification and clustering in relational data", in "International Joint Conference on Artificial Intelligence", pp. 870–878 (2001).

Tenenbaum, J. B., V. De Silva and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction", Science **290**, 5500, 2319–2323 (2000).

Teng, S.-Y., J. Li, L. P.-Y. Ting, K.-T. Chuang and H. Liu, "Interactive unknowns recommendation in e-learning systems", in "IEEE International Conference on Data Mining", pp. 497–506 (2018).

Tong, H., C. Faloutsos, B. Gallagher and T. Eliassi-Rad, "Fast best-effort pattern matching in large attributed graphs", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 737–746 (2007).

Tong, H., C. Faloutsos and J.-Y. Pan, "Fast random walk with restart and its applications", in "International Conference on Data Mining", pp. 613–622 (2006).

Tong, H. and C.-Y. Lin, "Non-negative residual matrix factorization with application to graph anomaly detection", in "SIAM International Conference on Data Mining", pp. 143–153 (2011).

Tong, H., S. Papadimitriou, J. Sun, P. S. Yu and C. Faloutsos, "Colibri: Fast mining of large static and dynamic graphs", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 686–694 (2008).

Tsymbal, A., "The problem of concept drift: Definitions and related work", Computer Science Department, Trinity College Dublin **106**, 2, 58 (2004).

Veličković, P., G. Cucurull, A. Casanova, A. Romero, P. Lio and Y. Bengio, "Graph attention networks", arXiv preprint arXiv:1710.10903 (2017).

Velickovic, P., G. Cucurull, A. Casanova, A. Romero, P. Lio and Y. Bengio, "Graph attention networks", arXiv preprint arXiv:1710.10903 **1**, 2 (2017).

Von Luxburg, U., "A tutorial on spectral clustering", Statistics and Computing **17**, 4, 395–416 (2007).

Wang, D., P. Cui and W. Zhu, "Structural deep network embedding", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 1225–1234 (2016).

Wang, D., D. Pedreschi, C. Song, F. Giannotti and A.-L. Barabasi, "Human mobility, social ties, and link prediction", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 1100–1108 (2011).

Wang, S., J. Tang, C. Aggarwal, Y. Chang and H. Liu, "Signed network embedding in social media", in "SIAM International Conference on Data Mining", pp. 327–335 (2017a).

Wang, X., P. Cui, J. Wang, J. Pei, W. Zhu and S. Yang, "Community preserving network embedding", in "AAAI Conference on Artificial Intelligence", pp. 203–209 (2017b).

Wang, Z., Z. Jiang, Z. Ren, J. Tang and D. Yin, "A path-constrained framework for discriminating substitutable and complementary products in e-commerce", in "ACM International Conference on Web Search and Data Mining", pp. 619–627 (2018).

Wei, X., B. Cao and S. Y. Philip, "Unsupervised feature selection on networks: A generative view", in "AAAI Conference on Artificial Intelligence", pp. 2215–2221 (2016).

Wei, X., B. Cao and P. S. Yu, "Unsupervised feature selection with heterogeneous side information", in "ACM International Conference on Information and Knowledge Management", pp. 2359–2362 (2017a).

Wei, X., S. Xie and P. S. Yu, "Efficient partial order preserving unsupervised feature selection on networks", in "SIAM International Conference on Data Mining", pp. 82–90 (2015).

Wei, X., L. Xu, B. Cao and P. S. Yu, "Cross view link prediction by learning noise-resilient representation consensus", in "International Conference on World Wide Web", pp. 1611–1619 (2017b).

Westaby, J. D., *Dynamic Network Theory: How Social Networks Influence Goal Pursuit* (American Psychological Association, 2012).

Wu, L., J. Li, X. Hu and H. Liu, "Gleaning wisdom from the past: Early detection of emerging rumors in social media", in "SIAM International Conference on Data Mining", pp. 99–107 (2017).

Xiang, R., J. Neville and M. Rogati, "Modeling relationship strength in online social networks", in "International Conference on World Wide Web", pp. 981–990 (2010).

Xu, X., N. Yuruk, Z. Feng and T. A. Schweiger, "Scan: A structural clustering algorithm for networks", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 824–833 (2007).

Yamada, M., T. Koh, T. Iwata, J. Shawe-Taylor and S. Kaski, "Localized lasso for high-dimensional regression", in "Artificial Intelligence and Statistics", pp. 325–333 (2017).

Yang, C., Z. Liu, D. Zhao, M. Sun and E. Y. Chang, "Network representation learning with rich text information", in "International Joint Conference on Artificial Intelligence", pp. 2111–2117 (2015).

Yang, J., J. McAuley and J. Leskovec, "Community detection in networks with node attributes", in "IEEE International Conference on Data Mining", pp. 1151–1156 (2013).

Yang, T., R. Jin, Y. Chi and S. Zhu, "Combining link and content for community detection: A discriminative approach", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 927–936 (2009).

Yang, Y., H. T. Shen, Z. Ma, Z. Huang and X. Zhou, "$\ell_{2,1}$-norm regularized discriminative feature selection for unsupervised learning", in "International Joint Conference on Artificial Intelligence", pp. 1589–1594 (2011).

Yin, Z., M. Gupta, T. Weninger and J. Han, "A unified framework for link recommendation using random walks", in "International Conference on Advances in Social Networks Analysis and Mining", pp. 152–159 (2010).

Yu, K., J. Bi and V. Tresp, "Active learning via transductive experimental design", in "International Conference on Machine Learning", pp. 1081–1088 (2006).

Yu, W., W. Cheng, C. C. Aggarwal, K. Zhang, H. Chen and W. Wang, "Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 2672–2681 (2018).

Yuan, S., X. Wu and Y. Xiang, "Sne: Signed network embedding", in "Pacific-Asia Conference on Knowledge Discovery and Data Mining", pp. 183–195 (2017).

Zafarani, R., M. A. Abbasi and H. Liu, *Social Media Mining: An Introduction* (Cambridge University Press, 2014).

Zhang, B. and M. Al Hasan, "Name disambiguation in anonymized graphs using network embedding", in "ACM International Conference on Information and Knowledge Management", pp. 1239–1248 (2017).

Zhang, H., L. Qiu, L. Yi and Y. Song, "Scalable multiplex network embedding", in "International Joint Conference on Artificial Intelligence", pp. 3082–3088 (2018).

Zhang, J., "A survey on streaming algorithms for massive graphs", in "Managing and Mining Graph Data", pp. 393–420 (Springer, 2010).

Zhang, S. and H. Tong, "Final: Fast attributed network alignment", in "ACM SIGKDD International Conference on Knowledge Discovery and Data Mining", pp. 1345–1354 (2016).

Zhang, Y., H. Lin, Z. Yang, J. Wang, Y. Li and B. Xu, "Protein complex prediction in large ontology attributed protein-protein interaction networks", IEEE/ACM Transactions on Computational Biology and Bioinformatics **10**, 3, 729–741 (2013).

Zhao, P., C. Aggarwal and G. He, "Link prediction in graph streams", in "IEEE International Conference on Data Engineering", pp. 553–564 (2016a).

Zhao, Y. and P. S. Yu, "On graph stream clustering with side information", in "SIAM International Conference on Data Mining", pp. 139–150 (2013).

Zhao, Z., X. He, D. Cai, L. Zhang, W. Ng and Y. Zhuang, "Graph regularized feature selection with data reconstruction", IEEE Transactions on Knowledge and Data Engineering **28**, 3, 689–700 (2016b).

Zhao, Z. and H. Liu, "Spectral feature selection for supervised and unsupervised learning", in "International Conference on Machine Learning", pp. 1151–1157 (2007).

Zhao, Z., L. Wang, H. Liu and J. Ye, "On similarity preserving feature selection", IEEE Transactions on Knowledge and Data Engineering **25**, 3, 619–632 (2013).

Zhou, D., J. He, K. S. Candan and H. Davulcu, "Muvir: Multi-view rare category detection", in "International Conference on Artificial Intelligence", pp. 4098–4104 (2015).

Zhou, F. and F. De la Torre, "Factorized graph matching", in "IEEE Conference on Computer Vision and Pattern Recognition", pp. 127–134 (2012).

Zhou, L., Y. Yang, X. Ren, F. Wu and Y. Zhuang, "Dynamic network embedding by modeling triadic closure process", in "AAAI Conference on Artificial Intelligence", pp. 571–578 (2018).

Zhou, Y., R. Jin and S. C.-H. Hoi, "Exclusive lasso for multi-task feature selection", in "International Conference on Artificial Intelligence and Statistics", pp. 988–995 (2010).

Zhu, L., D. Guo, J. Yin, G. Ver Steeg and A. Galstyan, "Scalable temporal latent space inference for link prediction in dynamic social networks", IEEE Transactions on Knowledge and Data Engineering **28**, 10, 2765–2777 (2016).

Zhu, S., K. Yu, Y. Chi and Y. Gong, "Combining content and link for classification using matrix factorization", in "ACM SIGIR conference on Research and Development in Information Retrieval", pp. 487–494 (2007).

Zitnik, M., M. Agrawal and J. Leskovec, "Modeling polypharmacy side effects with graph convolutional networks", arXiv preprint arXiv:1802.00543 (2018).

Biographical Sketch

Jundong Li is a Ph.D. candidate of Computer Science and Engineering at Arizona State University. He obtained his BEng degree from College of Computer Science and Technology, Zhejiang University in 2012, and MSc degree from Department of Computing Science, University of Alberta in 2014. His research interests are in data mining, machine learning, and their applications in social media. As a result of his research work, he has published more than 50 papers in high-impact venues (including KDD, WWW, IJCAI, AAAI, WSDM, CIKM, ICDM, SDM, ECMLPKDD, CSUR, TKDD, TIST, DMKD, etc), with over 1,000 citation count. He is a tutorial speaker at KDD 2017 and KDD 2019. He also leads the development of an open-source feature selection repository (scikit-feature) which has been reported by several news articles and blogs. He regularly serves on program committees for major international conferences and reviews for multiple journals. He also worked as a research intern at Yahoo! Research in 2016.