NETWORK ANALYSIS WITH NEGATIVE LINKS

By

Tyler Scott Derr

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science – Doctor of Philosophy

2020

ABSTRACT

NETWORK ANALYSIS WITH NEGATIVE LINKS

By

Tyler Scott Derr

As we rapidly continue into the information age, the rate at which data is produced has created an unprecedented demand for novel methods to effectively extract insightful patterns. We can then seek to understand the past, make predictions about the future, and ultimately take actionable steps towards improving our society. Thus, due to the fact that much of today's big data can be represented as graphs, emphasis is being taken to harness the natural structure of data through network analysis. Traditionally, network analysis has focused on networks having only positive links, or unsigned networks. However, in many real-world systems, relations between nodes in a graph can be both positive and negative, or signed networks. For example, in online social media, users not only have positive links such as friends, followers, and those they trust, but also can establish negative links to those they distrust, towards their foes, or block and unfriend users.

Thus, although signed networks are ubiquitous due to their ability to represent negative links in addition to positive links, they have been significantly under explored. In addition, due to the rise in popularity of today's social media and increased polarization online, this has led to both an increased attention and demand for advanced methods to perform the typical network analysis tasks when also taking into consideration negative links. More specifically, there is a need for methods that can measure, model, mine, and apply signed networks that harness both these positive and negative relations. However, this raises novel challenges, as the properties and principles of negative links are not necessarily the same as positive links, and furthermore the social theories that have been used in unsigned networks might not apply with the inclusion of negative links.

The chief objective of this dissertation is to first analyze the distinct properties negative links have as compared to positive links and towards improving network analysis with negative links by researching the utility and how to harness social theories that have been established in a holistic

view of networks containing both positive and negative links. We discover that simply extending unsigned network analysis is typically not sufficient and that although the existence of negative links introduces numerous challenges, they also provide unprecedented opportunities for advancing the frontier of the network analysis domain. In particular, we develop advanced methods in signed networks for measuring node relevance and centrality (i.e., *signed network measuring*), present the first generative signed network model and extend/analyze balance theory to signed bipartite networks (i.e., *signed network modeling*), construct the first signed graph convolutional network which learns node representations that can achieve state-of-the-art prediction performance and then furthermore introduce the novel idea of transformation-based network embedding (i.e., *signed network mining*), and *apply signed networks* by creating a framework that can infer both link and interaction polarity levels in online social media and constructing an advanced comprehensive congressional vote prediction framework built around harnessing signed networks.

To my wife, parents, siblings, and entire family for their love and support.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisor, Dr. Jiliang Tang, for his guidance, encouragement, inspiration, and support through my Ph.D. I have learned so much from him during these years ranging from how to find good research problems, writing papers, and designing presentations, to ways of efficiently managing a research lab and mentoring students. He has worked tirelessly to provide me with countless opportunities and learning experiences, which are what has led to my growth as an independent researcher. I could not imagine having a better mentor, feel honored to have been his student, and will always appreciate his suggestions and advice. He is also a dear friend and has guided me with his knowledge and experience in both my work and life. Dr. Tang, I cannot thank you enough.

I would like to thank my committee members Dr. Charu Aggarwal, Dr. Kenneth Frank, Dr. Anil Jain, Dr. Pang-Ning Tan, and Dr. Kaitlin Torphy, for their insightful comments and helpful suggestions. I had met Dr. Charu Aggarwal for the first time at the SIAM International Conference on Data Mining in 2017 and since then he has provided invaluable insights and guidance for much of my research in signed networks. I had met Dr. Kenneth Frank for the first time when Dr. Tang and I gave a guest lecture on signed network analysis in Dr. Frank's social network course. He has opened my eyes to social network analysis from a computational social science perspective and this has prepared me for being better equipped as an interdisciplinary researcher, especially for applying data science techniques to social science problems. Dr. Anil Jain had greatly helped me during my preparation for my faculty job market interviews by providing numerous tips based on his many experiences while also helping to broaden my perspectives on research. Although I have not had the pleasure of taking Dr. Pang-Ning Tan's data mining course, I have found myself continuously referencing his Introduction to Data Mining textbook throughout my Ph.D. studies. Through discussions with Dr. Tan I was also able to strengthen the motivation for some of the presented methods and now have numerous new ideas in mind to continue my research beyond this dissertation. I met Dr. Kaitlin Torphy through the Teachers in Social Media Project at MSU. I have

learned a lot from Dr. Torphy in the domain of education research and she has mentored me on some topics in educational data mining. I look forward to a continued collaboration with them all after the completion of my dissertation.

I was fortunate to have been an intern at HRL Laboratories. It was here that I had the privilege of having the amazing mentors: Dr. Kang-Yu Ni and Dr. Jiejun Xu. It was thanks to you both that I gained valuable insights into new problems, techniques, and valuable domain knowledge that has helped shape my research trajectory. Thank you both for everything. I would also like to thank Dr. Tsai-Ching Lu for his insightful comments during my time at HRL.

I joined the Data Science and Engineering (DSE) Lab at the end of the Fall 2016 semester as Dr. Jiliang Tang's first Ph.D. student when he was establishing the lab. During my Ph.D. study, I have had the pleasure and fortune of having supportive and encouraging friends and colleagues. I am thankful to all my collaborators from outside the DSE Lab: Dr. Charu Aggarwal, Dr. Kevin Chen-Chuan Chang, Dr. Yi Chang, Pouya Esmalian, Dr. Kenneth Frank, Amin Javari, Dr. Qing Li, Dr. Hui Liu, Dr. Zitao Liu, Dr. Kaitlin Torphy, Dr. Jianping Wang, Dr. Lingfei Wu, Dr. Jiejun Xu, and Dr. Dawei Yin. I am thankful to all of my colleagues from the DSE Lab: Ibrahim Ahmed, Dr. Meznah Almutairy, Aaron Brookhouse, Jamell Dacon, Wenqi Fan, Bryan Hendryx, Dr. Jiangtao Huang, Wei Jin, Cassidy Johnson, Hamid Karimi, Juanhui Li, Yaxin Li, Haochen Liu, Hua Liu, Xiaorui Liu, Yao Ma, Mitansh Madan, Daniel K. Ofori-Dankwa, Namratha Shah, Hannah Striebel, Pegah Varghaei, Chenxing Wang, Wentao Wang, Xiaoyang Wang, Xin Wang, Yiqi Wang, Zhiwei Wang, Han Xu, Hansheng Zhao, and Xiangyu Zhao. In particular, thanks to Zhiwei Wang for collaboration on my first co-author paper; thanks to Chenxing Wang and Dr. Suhang Wang for collaboration on my first first-author paper; thanks to my DSE collaborators Aaron Brookhouse, Jamell Dacon, Wenqi Fan, Dr. Jiangtao Huang, Wei Jin, Cassidy Johnson, Hamid Karimi, Haochen Liu, Xiaorui Liu, Yao Ma, Chenxing Wang, Wentao Wang, Yiqi Wang, Zhiwei Wang, and Xiangyu Zhao, I will remember staying awake to meticulously revise our papers until the last moment of the deadlines. During my time in the DSE Lab I have learned so much from you all. I look forward to continued collaboration and seeing all your future great achievements; I know first-hand you are

all in good hands having Dr. Jiliang Tang as an advisor, mentor, and friend.

Finally, I would again like to thank my wife, parents, siblings, and entire family for their love and support. This dissertation is dedicated to them.

TABLE OF CONTENTS

LIST OF TABLES

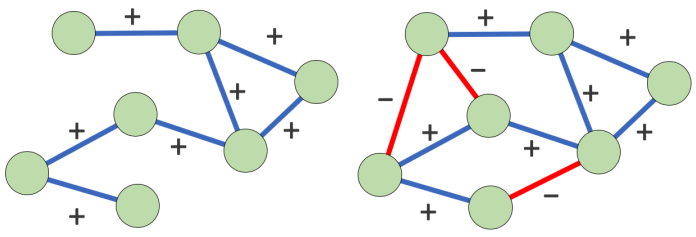
LIST OF FIGURES

LIST OF ALGORITHMS

CHAPTER 1

INTRODUCTION

Most existing network analysis has solely focused on unsigned networks (or networks with only positive links) as shown in Figure 1.1(a). However, in many real-world systems, relations can be both positive and negative. For instance, social media users not only have positive links such as friends (e.g., Facebook and Slashdot), followers (e.g., Twitter), and trust (e.g., Epinions), but also can establish negative links such as foes (e.g., Slashdot), distrust (e.g., Epinions), blocked and unfriended users (e.g., Facebook and Twitter). These relations can be represented as networks with both positive and negative links (or signed networks) as shown in Figure 1.1 (b). The introduction of negative links in signed networks not only increases the complexity of the representation, but also poses tremendous challenges for traditional unsigned network analysis. It is evident from recent researches that negative links in signed networks have distinct properties from positive links [1]. Meanwhile, the fundamental principles and theories of signed networks are substantially different from those of unsigned networks [2, 1]. Hence, signed network analysis cannot be carried out by simply extending unsigned network analysis. On the other hand, the existence of negative links also brings about unprecedented opportunities for network analysis. First, negative links have been proven to have significant added value over positive links in various analytical tasks [3, 4, 5, 6]. Second, analogous to unsigned networks, we can have similar analysis tasks for signed networks; however, negative links in signed networks make them applicable to a broader range of applications and tasks [7].



(a) Unsigned network (b) Signed network

Figure 1.1: A visualization of an unsigned and signed network.

Therefore, in this dissertation, I investigate network analysis with negative links (i.e., signed network analysis) focusing on each of the four primary directions. In particular, I propose novel frameworks to tackle the challenges associated with measuring, modeling, mining, and applying signed networks.

1.1 Research Challenges

While analyzing and making predictions in signed networks, we are faced with several challenges that must be overcome:

- The first major challenge is that the properties and theories of signed networks deviate from traditional unsigned networks. In Chapter 2 we perform an initial investigation to analyze multiple network properties for both positive and negative links, which empirically show their many differences on a representative set of signed networks. Furthermore, many of the unsigned network methods are built with classical network theories, but recent work has shown they might not apply to signed networks. Thus, this leads to needing dedicated efforts for new ways of constructing signed network analysis methods built in accordance to signed network specific theories.

- For measuring signed networks, we have the challenge of polarity being introduced. More specifically, when seeking to define how related two nodes are in a signed network (i.e., signed node relevance) we must not only consider the strength of their relationship, but also whether it is in a positive or negative way. This also requires balancing between situations such as two users having many common friends while also disagreeing on many other users (e.g., one user trusts the third parties while the other distrusts them), and the situation of two users having only a single friend in common, but having no differing opinions on others. Similarly, for defining a signed node centrality, the polarities of relations pose the specific challenge of needing to now differ from the "infamous" users who are important for negative reasons from the "famous" users who are popular for positive reasons.

- For signed generative network modeling the major challenge is that this domain has never been explored before to the best of our knowledge. Hence, just as unsigned generative modeling requires elegant strategies to allow the generation of networks maintaining network properties such as a correct degree distribution and the level of local clustering, signed networks require similar level of finesse. More specifically, while generating a synthetic signed network if we seek to maintain the correct ratio of positive to negative links this will conflict with maintaining the correct distribution of balanced to unbalanced triangles due to these signed triangles requiring to be of a certain form to adhere to the signed network social theory. Then, when seeking to harness the same social balance theory for bipartite networks new challenges are introduced for signed bipartite networks. This is due to prior works primarily using balance theory in the form of triangles, but there are inherently no triangles in bipartite networks, which thus require investigating other useful network subgraph structures beyond signed triangles.

- Recently network embedding methods and graph neural networks have shown to provide significant improvements in a wide range of network analysis tasks, such predicting missing links in the network or classifying the node type for those missing labels. The first challenge here is that many methods are based on random walks to build up a context of the local neighborhood and utilizes contrastive loss in relation to randomly sampled nodes in the network. In addition, many also make use of aggregating features from surrounding nodes under the assumption of homophily. However, this will not suffice for signed networks, since a traditional random walker is not sufficient to differentiate the context from both a positive and negative perspective, while homophily may also not apply on negative links (as it does on positive links). Furthermore, the situation becomes even more complex the further the aggregation distance is from the focal node.

The chief objective of this dissertation is to first analyze the distinct properties of negative links as compared to positive links and towards improving network analysis with negative links by

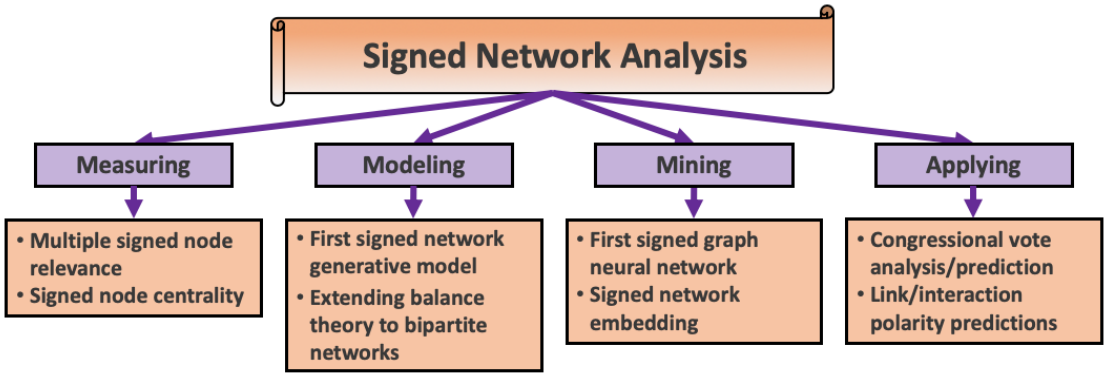


Figure 1.2: An overview of the research contributions presented in this dissertation.

researching the utility and how to harness social theories that have been established in a holistic view of networks containing both positive and negative links. We discover that simply extending unsigned network analysis is typically not sufficient and that although the existence of negative links introduces numerous challenges, they also provide unprecedented opportunities for advancing the frontier of the network analysis domain. An overview of my dissertation research is summarized in Figure 1.2.

1.2 Contributions

The contributions of this dissertation can be summarized as follows:

- We develop advanced methods in signed networks for measuring node relevance and centrality (i.e., *signed network measuring*);
- Presenting the first generative signed network model and extend/analyze balance theory in signed bipartite networks (i.e., *signed network modeling*);
- Construct the first signed graph convolutional network and introduce the novel idea of network transformation based signed network embedding, which both are able to learn node representations that can achieve state-of-the-art prediction performance on two representative link-oriented tasks (i.e., *signed network mining*);
- *Applying signed networks* by creating a framework that can infer both link and interaction

polarity levels in online social media, and constructing an advanced comprehensive congressional vote prediction framework built around harnessing signed networks.

1.3 Organization

The remainder of this dissertation is organized as follows. In Chapter 2, we introduce the preliminaries, including basic definitions, and social theories used in network analysis. In Chapter 3, we introduce multiple signed network node relevance measures and a signed centrality measure of which all are developed in accordance to signed social theories. Then, in Chapter 4, we introduce the first proposed generative network model for signed networks and furthermore introduce models for making predictions in signed bipartite graphs by harnessing structural balance theory through our novel defined signed butterfly network substructures. Chapter 5 presents our work on mining signed networks through the development of node level representations that can be used to solve traditional signed network mining tasks, such as predicting missing signed links and the unknown polarity of existing links. This is performed by developing the first signed graph convolutional network and introducing the novel idea of network transformation based embeddings. Chapter 6 presents our work on applying signed network analysis techniques to important interdisciplinary research in predicting future congressional votes, while also developing a method to help alleviate the cold-start problem of predicting the polarity of direct and indirect links between users in online platforms. Finally, Chapter 7 concludes the dissertation and presents promising future research directions.

CHAPTER 2

FOUNDATIONS AND PRELIMINARIES

In this section, we briefly introduce basic network definitions and social theories for networks consisting of only positive links (i.e., unsigned networks), only negative links, and networks containing both positive and negative links. This provides the groundwork for later introducing our novel methodologies for measuring, modeling, mining, and applying signed networks in the later chapters.

2.1 Basic Notations and Definitions

A signed network $\mathcal{G}$ is composed of a set of N nodes (i.e., users) $\mathcal{U} = \{u_1, u_2, \ldots, u_N\}$, a set of positive links $\mathcal{E}^+$ and a set of negative links $\mathcal{E}^-$. We represent directed signed links between users in an adjacency matrix, $\mathbf{A} \in \mathbb{R}^{N \times N}$, where $\mathbf{A}_{ij} = 1$ if u_i has a positive link to u_j, -1 if u_i creates a negative link to u_j, and 0 when u_i has no link to u_j. Furthermore, we can separate a signed network into two networks, one containing only positive links and the other with only negative links, which we can represent in the adjacency matrices $\mathbf{A}^+ \in \mathbb{R}^{N \times N}$ and $\mathbf{A}^- \in \mathbb{R}^{N \times N}$, respectively. We represent a positive link from u_i to u_j with $\mathbf{A}^+_{ij} = 1$ and $\mathbf{A}^+_{ij} = 0$ otherwise. Similarly, we represent a negative link from u_i to u_j with $\mathbf{A}^-_{ij} = 1$ and $\mathbf{A}^-_{ij} = 0$ otherwise. Note that we can similarly define an undirected signed network such that there are no directions associated with their signed links. Therefore, unlike the directed signed links, when u_i and u_j have a positive (or negative link) undirected link then $\mathbf{A}_{ij} = \mathbf{A}_{ij} = 1$ (or $\mathbf{A}_{ij} = \mathbf{A}_{ij} = -1$). In other words, an undirected signed network will have a symmetric adjacency matrix. We furthermore note that $\mathbf{A} = \mathbf{A}^+ - \mathbf{A}^-$.

2.2 Unsigned Network Properties and Theories

In networks consisting of only positive links (i.e., unsigned networks, $\mathbf{A}^+$) there are a few well known properties that are quite universally studied, such as the degree distribution, reciprocity, transitivity, and clustering coefficient, which we define here. Additionally, we introduce the social

network theory of homophily [8].

2.2.1 Degree Distribution and Network Density

In a network, observing the distribution of node degrees can provide great insight. One of the first analyses of the degree distribution of a network was done in [9]. They studied a friendship network among children in a school and noticed that while many children were selected as a friend by only a few of the other students, there were a few children that had been selected as a friend by very many other students. This led to later works, such as [10], observing that many real-world (unsigned) networks have degree distributions that follow a power-law (more specifically in [10] they studied citation networks). Essentially what many have observed is that for both in- and out-degrees of positive links in unsigned networks follow power-law distributions – most nodes have a small degrees while a few nodes have large degrees, and the networks with power-law degree distributions are commonly called "scale-free" networks [11].

In comparison, the network density is the property that compares the specific connectivity of a given unsigned network (which is defined by their links $\mathcal{E}+$) to a the maximum number of possible edges in that network. We note that for an unsigned network having the maximum number of edges is denoted as a fully-connected network and assuming no self loops (i.e., edges connecting a node to itself) it has $\binom{N}{2} = \frac{1}{2}N(N-1)$ edges in the undirected setting and $N(N-1)$ edges in the directed setting. Thus, the density of ρ_u and ρ_d in the undirected and directed unsigned network setting can be defined as follows:

$$\rho_u = \frac{|\mathcal{E}^+|}{\binom{N}{2}} = \frac{2|\mathcal{E}^+|}{N(N-1)} = \frac{||\mathbf{A}^+||_+}{N(N-1)} \qquad \rho_d = \frac{|\mathcal{E}^+|}{2\binom{N}{2}} = \frac{|\mathcal{E}^+|}{N(N-1)} = \frac{||\mathbf{A}^+||_+}{N(N-1)} \tag{2.1}$$

where we assume $\mathcal{E}^+$ and $\mathbf{A}^+$ are appropriately constructed in the undirected and directed settings (i.e., $\mathbf{A}^+$ is symmetrical in the undirected setting) for the respective definitions of unsigned network density.

2.2.2 Network Reciprocity

Links in directed social networks can be generally categorized into reciprocal (two-way) and parasocial (one-way) links [12]. Reciprocal links among nodes in unsigned networks are usually treated as the basis to create stable social ties and play an important role in the formation and evolution of networks [13]. Reciprocity is uniquely defined on directed networks. More specifically, given a pair of users (u_i, u_j), we define an edge u_i to u_j being reciprocated if there also exists a link from u_j back to u_i. Note that this is the shortest loop in a simple directed network (assuming no self-loops). An example of a reciprocal link would be if you follow someone on the social media site Twitter and they also follow you back.

While works such as [14] have examined relationships between in-degrees and out-degrees of social networks, it was in [15] where it is believed the first measurement of reciprocity appeared [16]. We can more formally define reciprocity r [17] for an directed unsigned network as the percentage of positive links that are reciprocated as follows:

$$r = \frac{1}{|\mathcal{E}^+|} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \mathbf{A}^+_{ij} \mathbf{A}^+_{ji} = \frac{1}{|\mathcal{E}^+|} \operatorname{Tr}\left((\mathbf{A}^+)^2\right) \tag{2.2}$$

where $\operatorname{Tr}(\cdot)$ is the trace of a matrix. Note that $\mathbf{A}^+_{ij}\mathbf{A}^+_{ji} = 1$ if and only if there is a reciprocal link between u_i and u_j, and $\mathbf{A}^+_{ij}\mathbf{A}^+_{ji} = 0$ otherwise.

2.2.3 Transitivity and Clustering Coefficient

In an unsigned network transitivity can be explained as "a friend's friend is a friend". Thus, the network only has perfect transitivity if each of the networks components are fully connected. However, in most real-world unsigned networks this is not going to be true for the entire network. Thus, we can measure the level of partial transitivity as follows. We first observe the wedges of the users u_i, u_k, and u_j, which are paths of length two, such as the wedge consisting of the edges u_i friends with u_k and u_k friends with u_j. We then consider this specific wedge closed if we also have u_i friends with u_j (i.e., forms a triangle). The clustering coefficient (which was first measured in [18]) is then the fraction of wedges that are closed into triangles. More formally, we define the

clustering coefficient c in an unsigned network as:

$$c = \frac{\text{(total number of closed wedges)}}{\text{(total number of wedges)}} = \frac{||\mathbf{A}^+ \circ (\mathbf{A}^+)^2||_+}{||(\mathbf{A}^+)^2||_+} \tag{2.3}$$

where $\circ$ denotes the Hadamard product (i.e., element-wise product) of two matrices and we use $||\cdot||_+$ to denote the summation of all elements of a matrix. Note that many real-world unsigned networks observe relatively higher clustering coefficients as compared to random graphs. Furthermore, a lot of prior work (first starting in [19]) have shown evidence that transitivity and having a higher clustering coefficient is both common and important in social networks.

2.2.4 Network Homophily

In an unsigned social network homophily can be summarized with the proverb "birds of a feather flock together" [8]. Coming from roots in sociology [20], it has been applied to social networks and used to explain the phenomenon that users who are similar are more likely to become friends with each other. This theory has been used in numerous social network areas, such as explaining the evolution/growth of networks [21] and in the recently developed graph neural networks [22], which are a generalization of deep neural networks to graph structured data. More specifically, the latter utilizes this fact they can aggregate node/user features from their local structural neighborhood (e.g., a user's set of friends) to better represent themselves. However, it is not necessarily the case that the same will apply for negative links.

2.3 Signed Network Datasets, Properties, and Theories

Now, having defined the basic network properties and social theory governing unsigned networks, here will investigate and discuss them further in the signed network setting. Furthermore, we will introduce the two major social theories that have been developed for networks containing both positive and negative links.

First we introduce a few representative signed network datasets. Then, we present an analysis for the same set of unsigned network properties on negative links and introduce signed network social theories that help explain the discovered differences, help motivate the need for dedicated

Table 2.1: Statistics of four signed social networks.

Network	# Users (N)	# Positive Edges ($\|\mathcal{E}^+\|$)	# Negative Edges ($\|\mathcal{E}^-\|$)
Bitcoin-Alpha	3,784	22,651	1,556
Bitcoin-OTC	5,901	32,271	3,438
Slashdot	79,116	392,179	123,218
Epinions	131,828	717,667	123,705

signed network analysis efforts, and guidance towards building our novel methodologies presented in the later chapters.

2.3.1 Signed Network Datasets

For the majority of this dissertation we study four signed network datasets we have collected, namely Bitcoin-Alpha[1], Bitcoin-OTC[2], Slashdot[3] and Epinions[4]. Some basic statistics of these four signed network datasets are demonstrated in Table 4.2. Below we describe more details about these datasets.

The Bitcoin-Alpha and Bitcoin-OTC networks are signed networks that we collected from publicly available data from their respective websites[5]. We note that smaller and less comprehensive versions of this data had previously been collected in [23]. The two Bitcoin sites are open market websites that allow users to buy and sell things. Due to the anonymity behind users' Bitcoin account, users of these websites form trust networks to prevent against scammers (e.g., fake users who are just attempting to have another user send them bitcoins, but never deliver their end of the deal, which is usually the delivery of some other monetary good). In addition to the signed networks, users in both websites can specify scores in the range [1,10] (or [-10,-1]) to indicate the positive (or negative) tie strength. Note that negative links in both websites are visible to the public.

The Slashdot dataset was obtained from [24]. Slashdot focuses on providing technology news

[1] http://www.btcalpha.com

[2] https://www.bitcoin-otc.com

[3] http://www.slashdot.org

[4] http://www.epinions.com

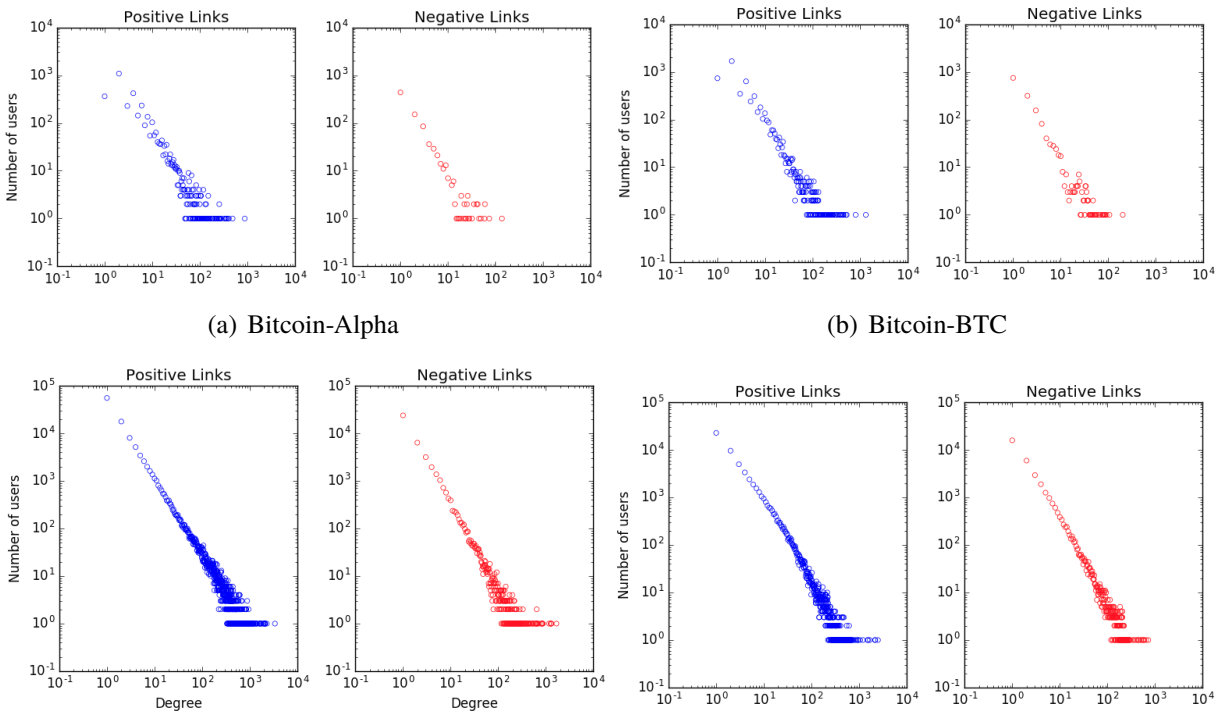
[5] The Bitcoin-Alpha and Bitcoin-OTC data was exhaustively crawled on December 18$^{\text{th}}$ of 2016.

since 1997. More specifically, the technology stories are written by either editors or submitted by the users and then other users are allowed to comment on these published stories. In addition to being an website allowing social connections, it added a novel component allowing users to flag others with negative sentiment in addition to the more traditional positive only sentiment (e.g., friends feature). In more detail, this unique feature was established in 2002 where the website has since allowed users to explicitly mark other users as their friends (i.e., positive links) or foes (i.e., negative links) (i.e., their "Zoo" feature). Note that negative links in Slashdot are only visible to users who are currently logged into the system (i.e., negative relations are not publicly available, but require signing up as a member to the site).

In addition, we have collected a dataset from the product review site Epinions where users can establish trust (i.e., positive) and distrust (i.e., negative) links. In addition, users can write reviews for items from certain pre-defined categories. Furthermore, users can then rate the "helpfulness" of those reviews on a scale of 1 to 6 (with a higher score denoting the user found the review more helpful). While there have been numerous multiple versions of this dataset [25, 2, 26, 27], we primarily only use the explicitly created trust and distrust links made between users. Later in Section 6.1 we will utilize these helpfulness ratings (obtained from [27]), but otherwise we only use the more commonly used basic signed network version from [2]. We note that negative links in Epinions are totally invisible to the public, but were obtained from Epinions staff for research purposes in this dataset. However, the helpfulness ratings are publicly available.

2.3.2 Data Analysis on Signed Networks Properties

Now, having defined the basic network properties and social theory governing unsigned networks, here we investigate and discuss them further in the signed network setting. More specifically, we perform an initial study of these properties on both the networks consisting of only negative links (i.e., $\mathbf{A}^-$) and also positive links (i.e., $\mathbf{A}^+$) from four real-world signed networks (as shown in Table 4.2). In addition, as previous studies suggested that balance theory is helpful to explain social phenomena in signed networks [4] and status theory is another influential social theory in signed



(a) Bitcoin-Alpha (b) Bitcoin-BTC

(c) Epinions (d) Slashdot

Figure 2.1: Degree distributions in signed social networks.

networks [2], we also define and discuss some insights from both of these signed social network theories.

Degree Distributions: Here, we analyze the undirected degree distributions for the positive (i.e., $\mathcal{E}^+$ and negative (i.e., $\mathcal{E}^-$) links. For each user, we calculate and combine the numbers of in- and out-degrees for both positive and negative links. The distributions of the undirected degrees for positive and negative links in our four signed networks are demonstrated in Figure 2.1. From the figure, it is clearly observed that the degree distributions of positive and negative links in all four signed networks also follow power-law distributions. For instance, a few nodes give a large number of negative links; while many nodes only give few negative links. Note that this conclusion can be approximated by observing a linear relationship in the distribution plots when having a log-log scale. Hence, we can see that positive and negative links indeed empirically have at least one similar property so far in regards to having a power-law degree distribution.

Table 2.2: Probability of links being reciprocal in signed social networks.

Datasets	Positive Links	Negative Links
Bitcoin-Alpha	85.4%	18.0%
Bitcoin-OTC	83.8%	17.8%
Slashdot	30.7%	7.4%
Epinions	34.8%	3.8%

Network Density: From Table 4.2 we can easily observe that the negative links are sparser than positive links. In other words, the density of the positive links (i.e., defined in Eq. (2.1)) is significantly larger than that of the negative links (i.e., redefining Eq. (2.1) with $\mathcal{E}^-$ and $\mathbf{A}^-$ instead of $\mathcal{E}^+$ and $\mathbf{A}^+$), which is obvious from the fact that both the positive and negative linked networks have the same number of users (i.e., the same denominator in Eq. (2.1 and its negative link variant). Thus, we can already start to notice that negative links are having different global behaviors from positive links.

Reciprocal Links: For a pair of users (u_i, u_j), there are four types of reciprocal links – $(u_i + u_j, u_j + u_i)$, $(u_i + u_j, u_j - u_i)$, $(u_i - u_j, u_j - u_i)$ and $(u_i - u_j, u_j + u_i)$, where $u_i + u_j$ (or $u_i - u_j$) denotes that there is a positive link (or a negative link) from u_i to u_j. We checked our four signed networks and found that among four types of reciprocal links, there are few $(u_i + u_j, u_j - u_i)$ and $(u_i - u_j, u_j + u_i)$. Therefore, our analysis on reciprocal links focuses on $(u_i + u_j, u_j + u_i)$ and $(u_i - u_j, u_j - u_i)$. We calculate if u_i has a positive link (or a negative link) to u_j, how likely u_j also has a positive link (or a negative link) to u_i. The results on four signed networks are shown in Table 2.2.

From the table, we first make the observation that the percent of reciprocal positive links is much higher than that of reciprocal negative links in all four signed social networks. Next, we notice that in all four websites, positive links are always visible to the public, the percent of reciprocal positive links in Bitcoin-Alpha and Bitcoin-OTC is much higher than that in Slashdot and Epinions. Users in Bitcoin Alpha and OTC exchange bitcoins with others; while users share free content (news or

Table 2.3: Tie strengths of positive and negative links in signed social networks.

		Bitcoin-Alpha		Bitcoin-OTC	
		Avg. Strength	# Links	Avg. Strength	# Links
Overall Link Tie Strengths	Positive	1.998	22,651	1.968	32,271
	Negative	-6.319	1,556	-7.538	3,438
Non-reciprocal Tie Strengths	Positive	2.225	3,054	2.215	4,868
	Negative	-6.120	1,029	-7.540	2,467
Positive Reciprocal Tie Strengths	$(\boldsymbol{u_i}\ \boldsymbol{u_j}, u_j + u_i)$	1.955	19,350	1.915	27,044
	$(\boldsymbol{u_i}\ \boldsymbol{u_j}, u_j - u_i)$	2.611	247	2.571	359
Negative Reciprocal Tie Strengths	$(\boldsymbol{u_i} - \boldsymbol{u_j}, u_j - u_i)$	-7.079	280	-7.920	612
	$(\boldsymbol{u_i} - \boldsymbol{u_j}, u_j + u_i)$	-6.291	247	-6.875	359

reviews) with others in Slashdot and Epinions. Thus, Bitcoin Alpha and OTC users need much stronger social ties for bitcoin trading in the online worlds than users in Slashdot and Epinions to consume online free content. Finally, we note that the percent of reciprocal negative links in Bitcoin-Alpha and Bitcoin-OTC is much higher than that in Slashdot, where the percent of reciprocal negative links in Slashdot is much higher than that in Epinions. Four websites have different access controls to negative links. In Bitcoin Alpha and OTC, negative links are totally visible to the public; only users who login to the Slashdot can see negative links; while negative links are totally private in Epinions. Exposing negative links may cause revenges that consequently could lead to more reciprocal negative relations [28].

Furthermore, since the Bitcoin-Alpha and Bitcoin-OTC are weighted signed networks, we also investigate if there are any differences/similarities between positive and negative when it comes to the strength of the ties. More specifically, we first calculated the average positive and negative link strengths, which are reported in Table 2.3. The first observation is that the magnitude of negative links is significantly higher than that of positive links in both datasets. One interpretation of this could be that users are conservative in their judgement of others they have done successful transactions with, but are very aggressive in their judgement of other users when having a negative experience. We can also notice the ranking of tie strength magnitude between: (i) non-reciprocal links, (ii) reciprocal links with the same sign, and (iii) reciprocal links with opposing signs. For

positive links in both datasets we have the largest magnitude is (iii), followed by (i) and (ii). In comparison, for negative links in both datasets (ii) is the largest, but then for Bitcoin-Alpha we have (iii) followed by (i) and Bitcoin-OTC we have (i) followed by (iii). It is interested that the strongest negative links are those where the two users both have expressed a negative sentiment towards each other by creating a reciprocal negative link However, for positive links, the largest magnitude is coming from positive links that are reciprocated by a negative link, which is somehow unintuitive. In summary, there are more differences than commonalities for positive and negative links when it comes to the tie strengths in signed networks.

2.3.3 Signed Network Theories

In this section, we will define and introduce the two most influential signed network theories, namely that of balance and status. This will then help motivate the need for dedicated signed network analysis efforts beyond the differences in network properties previously discussed.

2.3.3.1 Balance Theory in Signed Networks

It was in [29] that balance theory was first developed at the individual level with the general notion that “the friend of my friend is my friend" and “the enemy of my enemy is my friend". Then, later in [30] structural balance theory was introduced at the group level with a network perspective. A signed network is defined as being balanced if and only if all the cycles in the network contain an even number of negative links. It had been proved in [31] that a signed network is balanced if and only if the nodes can be partitioned into two mutually exclusive subsets such that all links within the subsets are positive, while the links having an endpoint in each of the two subsets should be negative. However, it is rare to have real-world signed networks that are completely balanced.

There have been multiple methods to measure the level of balance in a signed network. It was in [32] and [33] that the ratio of balanced to unbalanced cycles were used to calculate the level of balance of a signed network. Later methods were then developed, such as in [34] where they performed a clustering of the signed network and then analyzed the number of postive links

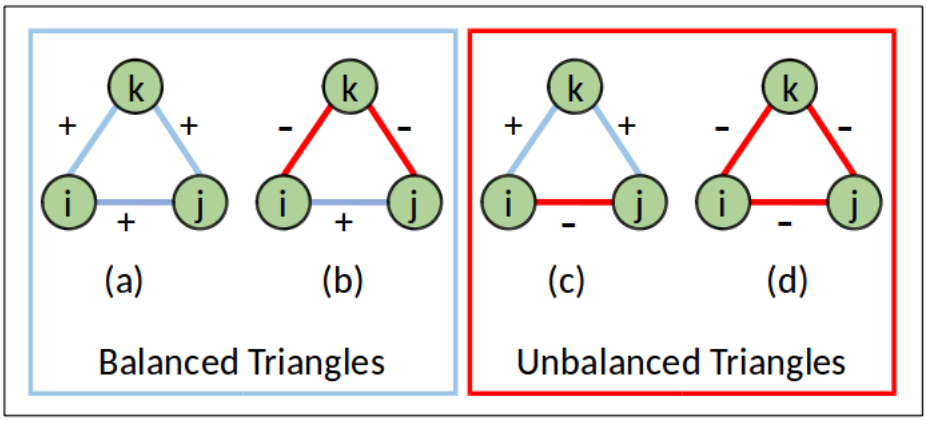


Figure 2.2: Visualizing balance theory in the form of signed triangles.

between clusters and negative links within clusters. This led to other methods similarly defined using clusterings [35, 36]. In addition, in [37] a signed spectral theory approach was taken. It should also be noted that it was in [34] where the notion of weak structural balance came, which based on the clustering ignores the assumption that "the enemy of my enemy is my friend" is a balanced relation. In other words, weak structral balance does not concern whether the enemy of my enemy is either my friend or my enemy, and thus only takes into consideration that a friend of a friend should be a friend as compared to an enemy.

However, it is still common to use the definition based on the ratio of triangles (even though it is less computationally efficient taking $O(N^3)$ when utilizing matrix operations on the adjacency matrix). In regards to analyzing the level of balance in our signed networks we first adopt s_{ij} to denote the link sign between two users u_i and u_j where $s_{ij} = 1$ (or $s_{ij} = -1$) if the positive (or negative) link between u_i and u_j. As previously methods, and given that typically we focus on triads (or 3-cycles) [4], a triad of three users (u_i, u_j, u_k) is balanced if $s_{ik} = 1$ and $s_{jk} = 1$, then $s_{ik} = 1$; or $s_{ij} = 1$ and $s_{jk} = -1$, then $s_{ik} = -1$. Therefore, for a triad, there are four possible sign combinations: (a) $(+,+,+)$; (b) $(+,-,-)$; (c) $(+,+,-)$; and (d) $(-,-,-)$, while only (a) and (b) are balanced. We visualize these four signed triangle combinations in Figure 2.2.

Note that balance theory is only applicable to undirected signed network, and thus we perform some basic preprocessing to ignore the link directions when applying it to directed signed networks following the discussions in [4]. More specifically, in our specific mapping from a directed to undi-

rected signed network we count each of the four sign combinations and find that 92.0%, 91.5%, 94.5% and 92.4% of triads in Bitcoin-Alpha, Bitcoin-OTC, Slashdot and Epinions are balanced, respectively. This is in line with prior works measuring the level of balance in signed networks and it has also been shown that the ratio of balance to unbalance triangles increases over time [28].

2.3.3.2 Status Theory in Signed Networks

Based on some of the observations found in [3] status theory was later developed in [2]. Unlike balance theory, which is defined in an undirected signed network based on users liking or disliking (i.e., positive or negative links), status theory takes a separate perspective in directed signed networks. Rather than assuming a positive link from u_i to u_j implies that u_j is a friend of u_i, it considers that perhaps u_i instead thinks that u_j has a higher status. Similarly, a negative link from u_i to u_j might not imply that u_i dislikes u_j, but perhaps u_i just believes that u_j is of a lower status in the network. Thus, from a triangle perspective, status theory is based on the concept of consistency in the logical deductions of the directed relations. We note that directed signed triangles can take 12 different forms, where 4 are cyclic and 8 are acyclic.

Determining if a triangle adhere to status theory can be done by the following three steps: 1) first flip negative links to positive and reverse their direction; and 2) if the triangle is acyclic, then it adheres to status theory. For example, if we have the cyclic signed triangle consisting of the three positive links $u_i \xrightarrow{+} u_k$, $u_k \xrightarrow{+} u_j$, and $u_j \xrightarrow{+} u_i$, then based on status theory these three links respectively imply that u_i believes that u_k has a higher status then them (i.e., $u_i < u_k$), u_k believes that u_j has a higher status than them (i.e., $u_j > u_k$), and u_j believes that u_i has a higher status than them (i.e., $u_i > u_j$). However, we then from the first two links we have that $u_i < u_k < u_j$, but then the third link with u_j thinking that $u_i > u_j$ creates a contradiction. We note that prior works have shown that many real-world signed networks have the majority (and almost all) triangles adhering to status theory quite similar to balance theory [2]. It can be calculated that when converting directed signed network triangles to undirected signed network triangles that only 6 agree between

the theories of balance and status while the other 6 are in disagreement. For example, our prior example of the cyclic signed triangle of three positive links does not adhere to status theory, but aligns with triangle (a) from Figure 2.2 and thus adheres to balance theory.

CHAPTER 3

MEASURING NETWORKS WITH NEGATIVE LINKS

In this chapter[1], we investigate network measurements for networks having negative links. A network metric is a mathematical expression that allows the expression of information contained in a network to be output in numerical form. It is then via these metrics that we can define network measurements through the use of algorithms and/or mathematical formula that can be used to compare and/or rank users, pairs of users, subgroups (i.e., communities), or entire networks.

Node relevance, which measures how relevant two nodes are in a social network, is one of the keystones of social network analysis. This has been shown by their usage in diverse social network analysis tasks and applications such as link prediction [38, 39], node classification [40], community detection [41], search and recommendations [42]. The vast majority of existing node relevance measurements have been designed for unsigned networks (or social networks with only positive links) [11, 43]. We note that these measurements can be divided into local and global measurements according to the information used – local measurements only use local neighborhood information such as common neighbors; while global measurements utilize the whole structural information of the network such as Random Walk with Restart [44]. Thus, in Section 3.1, we present our proposed node relevance measures for signed networks and discuss their relationship to balance theory.

Node centrality is a fundamental network measurement that has a diverse set of applications [45, 46, 47, 48, 49] across many domains, such as economics [45], biology [46], and urban infrastructure [47, 48] to sociology, which the later has a plethora of applications [49]. In general, the task is to construct a ranking of nodes based on how "central" or "important" they are in the network. Most of the previous work has been for unsigned networks. Due to the inclusion of negative links, existing unsigned centrality measures are not directly applicable to signed networks.

[1]Tyler Derr, Chenxing Wang, Suhang Wang, and Jiliang Tang. "Relevance Measurements in Online Signed Social Networks." KDD 14th International Workshop on Mining and Learning with Graphs (MLG), 2018.

This is partially due to the added complexities associated with the introduction of negative links and also because we now need to differentiate between the "famous" and "infamous" users in a signed network. Though recently there have been some measurements proposed that take into account negative links by extending existing unsigned centrality measurements [24, 50, 51], they have not explored the use of exploiting higher-order network information in signed networks. Thus, in Section 3.2, we develop a signed centrality measurement built upon both prominent triadic social theories defined on networks having both positive and negative links, namely the status and balance theories.

3.1 Node Relevance Measurements in Signed Networks

It is evident from recent research that negative links have significant added value over positive links in various analytical tasks. For example, even a small number of negative links can significantly boost the performance of positive link prediction [3, 4], and can similarly improve recommender systems [5, 6]. Thereby, negative links could offer the potential to help us develop novel relevance measurements for signed networks. There are a few very recent works in designing node similarities for link prediction [52, 53]. However, a general and systematic investigation on signed relevance measurements and their effects on signed network analysis had previously not been explored. Hence, we perform the initial and comprehensive study on the problem of measuring node relevance in signed social networks. Analogous to node relevance research in unsigned networks, we aim to investigate the following: (a) how to make use of both positive and negative links in signed relevance measurements; and (b) what are the effects of these measurements on two signed network mining tasks of sign prediction and signed tie strength prediction.

Data Analysis Discussions: Social theories such as homophily [8] play an important role in building node relevance measurements for unsigned social networks [54]. This stimulates the investigation for one of the most fundamental social theories related to signed social networks, i.e., balance theory [30], that could be helpful in building node relevance measurements in signed social networks.

Table 3.1: Notations regarding node relevance in signed networks.

Notations	Descriptions
$\mathbf{A}$	Adjacency matrix
$\mathbf{A}^+(\mathbf{A}^-)$	Adjacency matrix of only positive(negative) links
$\|\mathbf{A}\|$	Absolute adjacency matrix
$\mathbf{R}$	Node relevance matrix
d_i	Degree of node u_i
$d_i^{in}(d_i^{out})$	Indegree (Outdegree) of node u_i
$d_i^{in+}(d_i^{out+})$	Indegree (Outdegree) of positive links of node u_i
$d_i^{in-}(d_i^{out-})$	Indegree (Outdegree) of negative links of node u_i
N_i	Set of neighbors for node u_i
$N_i^{in}(N_i^{out})$	Set of incoming (outgoing) neighbors for node u_i
$N_i^+(N_i^-)$	Set of positive (negative) neighbors for node u_i
$\mathbf{X}_{ij}$	the (i,j) entry of the matrix $\mathbf{X}$

Based on the analysis performed in Chapter 2.3, properties of negative links are different from positive links, which makes signed social networks be distinct from unsigned social networks. Therefore, though node relevance measurements have been extensively studied, it still needs dedicated efforts to systematically investigate signed relevance measurements. Furthermore, as most of the triads in signed social networks satisfy balance theory, we can use this to guide building novel signed relevance measurements.

Node relevance measurements have been extensively studied in unsigned networks. According to our preliminary data analysis, the availability of negative links makes signed networks unique in many aspects such as properties and balance theory. In this section, analogous to unsigned networks, we develop node relevance measurements for signed networks.

Notations and Definitions: We use $\mathbf{R} \in \mathbb{R}^{N\times N}$ to denote the relevance score matrix, where $\mathbf{R}_{ij}$ represents the node relevance from user u_i to user u_j. Note that node relevance values are not necessarily symmetrical. We summarize the above notations in Table 3.1 where d_i and N_i denote degree and the set of neighbors of u_i in an unsigned network.

Many node relevance measurements have been proposed for unsigned networks. According to the used information, we can roughly divide them to local and global measurements. Local

measurements only use local neighborhood information such as common neighbors; while global measurements utilize the whole structural information such as Random Walk with Restart. Meanwhile, node relevance measurements can be undirected and directed, corresponding to undirected and directed networks. Note that we could use any method that requires a directed network for an undirected network, since undirected networks are simply directed networks where each edge has both directions. In this work, we will group signed relevance measurements as local and global methods.

With node relevance measurements for unsigned networks, there are three strategies to design signed ones. The first is to only use $\mathbf{A}^+$ in the calculation of node relevance scores. This strategy completely ignores the negative links that could result in over-estimation of the impact of positive links [55]. The second strategy would be to convert negative links in the signed network into positive links, thus making the signed network into an unsigned network. Such a network can be represented by the matrix $\tilde{\mathbf{A}}$ where $\tilde{\mathbf{A}}_{ij} = |\mathbf{A}_{ij}|$. Ignoring signs of links not only overlooks the differences between negative and positive links; but also makes balance theory for signed networks not applicable. Our third strategy is to take advantage of negative links and balance theory to develop signed relevance measurements based on unsigned ones. In the following subsections, we will detail how to apply the third strategy to representative unsigned node relevance measurements.

3.1.1 Local Methods

In this subsection, we build local signed relevance measurements based on representative local methods for unsigned networks including common neighbors, Jaccard Index, and Preferential Attachment [56, 17]. For each unsigned measurement, we will first briefly introduce it, then detail how to design the signed one and finally discuss its connection with signed network properties and balance theory.

3.1.1.1 Common Neighbors

Unsigned Common neighbors (UCN): If two nodes share a lot of common friends, they are likely to be relevant. Based on this intuition, UCN defines the relevance score between u_i and u_j as the number of common neighbors, which is formally defined as:

$$\mathbf{R}_{ij} = |N_i \cap N_j| \tag{3.1}$$

where $|x|$ denotes the size of the set x.

Signed Common neighbors (SCN): UCN cannot be directly extended to include negative links. Therefore, we define SCN as follows:

$$\mathbf{R}_{ij} = (|N_i^+ \cap N_j^+| + |N_i^- \cap N_j^-|) - (|N_i^+ \cap N_j^-| + |N_i^- \cap N_j^+|)$$

We can interpret SCN as number of common neighbors of u_i and u_j where they agree on the polarity of the sign ($|N_i^+ \cap N_j^+| + |N_i^- \cap N_j^-|$) and then subtracting the number of neighbors that they disagree on the sign ($|N_i^+ \cap N_j^-| + |N_i^- \cap N_j^+|$).

*C*onnection to Balance Theory: If u_i and u_j agree with the majority of the signs of their neighbors, i.e., $(|N_i^+ \cap N_j^+| + |N_i^- \cap N_j^-|) > (|N_i^+ \cap N_j^-| + |N_i^- \cap N_j^+|)$, then $\mathbf{R}_{ij}$ is positive which will lead to more balanced triads. Otherwise, they have more disagreements on the signs, i.e., $(|N_i^+ \cap N_j^-| + |N_i^- \cap N_j^+|) > (|N_i^+ \cap N_j^+| + |N_i^- \cap N_j^-|)$, then $\mathbf{R}_{ij}$ is negative, which will also result in more balanced triads. Therefore, SCN aims to force more triads with u_i and u_j to be balanced.

3.1.1.2 Jaccard Index

Unsigned Jaccard Index (UJI): UCN only considers the number of common neighbors of u_i and u_j, but it ignores the number of unique neighbors these two users have. Therefore, UCN is likely to give users with large numbers of neighbors high relevance scores. To mitigate such effect, UJI penalizes the UCN scores by the number of unique neighbors two users have as:

$$\mathbf{R}_{ij} = \frac{|N_i \cap N_j|}{|N_i \cup N_j|} \tag{3.2}$$

Signed Jaccard Index (SJI): Similar to from UCN to UJI, SJI is defined as SCN divided by the total number of unique neighbors u_i and u_j have:

$$\mathbf{R}_{ij} = \frac{SCN_{ij}}{|N_i^+ \cup N_i^- \cup N_j^+ \cup N_j^-|} \tag{3.3}$$

*C*onnection to Balance Theory: Similar to SCN, SJI targets to force more triads balanced.

3.1.1.3 Preferential Attachment

Unsigned Preferential Attachment (UPA): One commonly used interpretation behind this method, taken from the finance realm, is that the rich gets richer. In terms of social network analysis, users that already have many friends are more likely to create new friends in the future. Therefore, the node relevance score of UPA is to multiply the degrees of the two users [11].

$$\mathbf{R}_{ij} = d_i \times d_j \tag{3.4}$$

Signed Preferential Attachment (SPA): In the Section 2.3.2, we demonstrate that both positive and negative links follow the power-law distributions. In other words, we observe "the rich getting richer" for both positive and negative links, which paves us a way to define SPA. We first split the network from $\mathbf{A}$ to a positive network $\mathbf{A}+$ and a negative network $\mathbf{A}^-$. Then we can use UPA to calculate relevance scores from the positive and negative networks, separately, since degrees in both networks follow power-law distributions. The relevance score for i and j from $\mathbf{A}^+$ is denoted as UPA_{ij}^+ and similarly we denote the relevance as UPA_{ij}^- from $\mathbf{A}^-$. UPA_{ij}^+ and UPA_{ij}^- are computed as:

$$UPA_{ij}^+ = d_i^+ \times d_j^+, \quad UPA_{ij}^- = d_i^- \times d_j^-$$

Then we define SPA between u_i and u_j as:

$$\mathbf{R}_{ij} = sign(UPA_{ij}^+ - UPA_{ij}^-) f(UPA_{ij}^+, UPA_{ij}^-) \tag{3.5}$$

where $sign(x) = 1$, 0, or -1 if x is larger, equal or smaller than 0. Intuitively, if the positive relevance score UPA_{ij}^+ is larger than the negative one UPA_{ij}^-, the overall $\mathbf{R}_{ij}$ should be positive;

otherwise, $\mathbf{R}_{ij}$ should be negative. Therefore the sign of $\mathbf{R}_{ij}$ is decided by $sign(UPA^+_{ij} - UPA^-_{ij})$. The relevance strength $|\mathbf{R}_{ij}|$ is to aggregate UPA^+_{ij} and UPA^-_{ij} via a function f. A straightforward way is to set $f(UPA^+_{ij}, UPA^-_{ij}) = |UPA^+_{ij} - UPA^-_{ij}|$. It may not work well. For example, when u_i and u_j have both larger positive and negative degrees, positive and negative relevance scores will cancel each other, which contradicts with " the rich getting richer". Actually we empirically find that $f(UPA^+_{ij}, UPA^-_{ij}) = \max(UPA^+_{ij}, UPA^-_{ij})$ works better than $f(UPA^+_{ij}, UPA^-_{ij}) = |UPA^+_{ij} - UPA^-_{ij}|$.

*C*onnection to the signed network property: According to the power-law distributions of positive and negative links, we design SPA, which will allow users with higher degrees to have higher relevance scores with others.

3.1.2 Global Methods

The global methods make use of not only the local neighborhoods, but also allow for the propagation of relevance information to pass through the whole network. Most of the global methods for unsigned networks assume that two users u_i and u_j should have high relevance if they have neighbors with high relevance. In this subsection, we detail how to design global signed relevance measurements based on representative unsigned ones and then connect them to balance theory.

3.1.2.1 Katz

Unsigned Katz (UK) : This method sums over the collection of all paths from i to j and has an exponential decay on the weight associated with the count of paths as the length increases [57]:

$$\mathbf{R}_{ij} = \sum_{l=1}^{\infty} \beta^l \cdot |\text{paths}^l_{i,j}| = \sum_{l=1}^{\infty} \beta^l \mathbf{A}^l \tag{3.6}$$

where $|\text{paths}^l_{i,j}|$ is the count of paths of length l from i to j. Note that we should have $\beta < 1$ so that longer paths will be assigned less weight than shorter paths. This can be formulated recursively as

follows to handle the counting of the paths of varying length:

$$\mathbf{R}_{ij} = \frac{\beta}{d_x} \sum_{k=1}^{N} \mathbf{A}_{ik} \mathbf{R}_{kj} + \delta_{ij} \tag{3.7}$$

Note that δ_{ij} is used to ensure that every node in the network has a high relevance to themselves (i.e., "self-similarity"). It is a diagonal term and is defined as $\delta = \mathbf{I}$. It normalizes the relevance scores from each user u_i based on the degree d_i.

Signed Katz (SK): Balance theory states that a k-cycle in a signed social network is balanced if it contains an even number of negative edges and unbalanced if it contains an odd number of negative edges. With relevance scores from SK, we expect more balanced k-cycles than unbalanced ones involving users i and j. To achieve this, we would therefore need to choose the sign of the node relevance $\mathbf{R}_{ij}$ to be either positive or negative, such that it optimizes over all the cycles involving i and j (i.e., all the paths between i and j). As done in UK, we also can similarly allow the decay of importance on the longer paths. Our formulation and its recurrence relation for the calculation of paths of length l having an even or odd number of negative edges is defined as follows:

$$\mathbf{R} = \sum_{l=1}^{\gamma} \beta^l f(\mathbf{B}_l, \mathbf{U}_l) \tag{3.8}$$

with

$$\mathbf{B}_l = \mathbf{B}_{l-1}\mathbf{A}^+ + \mathbf{U}_{l-1}\mathbf{A}^-$$

$$\mathbf{U}_l = \mathbf{B}_{l-1}\mathbf{A}^- + \mathbf{U}_{l-1}\mathbf{A}^+$$

$$\mathbf{B_1} = \mathbf{A}^+, \quad \mathbf{U_1} = \mathbf{A}^-$$

where $f(\mathbf{B}_l, \mathbf{U}_l)$ is a function to combine the counts of paths with even and odd number of negative links. $\mathbf{B}_l$ and $\mathbf{U}_l$ are the matrices that hold the number of paths with an even and odd number of negative links in paths of length l, respectively. Next we will discuss the inner working of SK. When counting paths of length 1 (i.e., a direct edge connecting the two nodes), we set $\mathbf{B}_1$ as $\mathbf{A}^+$ since having a positive edge is trivially having an even number of negative links in a path of length 1, and similarly reasoned for initializing $\mathbf{A}^-$. We assume that $\mathbf{B}_{l-1}$ and $\mathbf{U}_{l-1}$ represent the paths

of length $l-1$ having an even and odd number of negative edges, respectively, between all pairs of nodes. Adding one positive link ($\mathbf{A}^+$) to a path in $\mathbf{B}_{l-1}$ or adding a negative link ($\mathbf{A}^-$) to a path in $\mathbf{U}_{l-1}$ will result in a path of length l with an even number of negative links. This intuition leads to the update rule of $\mathbf{B}_l = \mathbf{B}_{l-1}\mathbf{A}^+ + \mathbf{U}_{l-1}\mathbf{A}^-$. Similarly, we can obtain the update rule of $\mathbf{U}_l = \mathbf{B}_{l-1}\mathbf{A}^- + \mathbf{U}_{l-1}\mathbf{A}^+$.

Theorem 1. *When we choose* $f(\mathbf{B}_l, \mathbf{U}_l) = (\mathbf{B}_l - \mathbf{U}_l)$ *and* $\mathbf{A} \in \mathbb{R}^{N\times N}$*, where* $\mathbf{A}_{ij} = 1$ *if* u_i *has a positive link to* u_j*,* -1 *if* u_i *creates a negative link to* u_j*, and* 0 *when* u_i *has no link to* u_j*, signed Katz in Eq (3.8) is equivalent to applying unsigned Katz in Eq (3.6) on the signed network adjacency matrix defined as* $\mathbf{A}$*.*

Proof. To prove the theorem, we only need to show that: $\mathbf{B}_l - \mathbf{U}_l = \mathbf{A}^l$. We use mathematical induction as:

Basis: Let $l = 1$, based on our definition of $\mathbf{B}_1$ and $\mathbf{U}_1$, we have $(\mathbf{B}_1 - \mathbf{U}_1) = (\mathbf{A}^+ - \mathbf{A}^-) = \mathbf{A} = \mathbf{A}^l$.

Inductive Hypothesis: Suppose the theorem holds for $l = k$. In other words, $(\mathbf{B}_k - \mathbf{U}_k) = \mathbf{A}^k$.

Inductive Step: Let $l = k+1$. Then our left size is $(\mathbf{B}_{k+1} - \mathbf{U}_{k+1}) = \Big((\mathbf{B}_k\mathbf{A}^+ + \mathbf{U}_k\mathbf{A}^-) - (\mathbf{B}_k\mathbf{A}^- + \mathbf{U}_k\mathbf{A}^+)\Big) = (\mathbf{B}_k - \mathbf{U}_k)(\mathbf{A}^+ - \mathbf{A}^-) = \mathbf{A}^k(\mathbf{A}) = \mathbf{A}^{k+1}$, which completes the proof. □

*C*onnection to Balance Theory: SK is built based on balance theory. SCN and SJI forces more balanced triads (or 3-cycles), while SK pushes more for any l-circles to be balanced. If the majority of paths between i and j have an even number of negative links, according to balance theory, we should have a positive node relevance between them. Similarly, when having an odd number of negative edges, we want to have a negative relevance. Therefore, if we count the number of paths between i and j with an even or odd number of negative edges, then we can subtract the number with an odd number of negative links from the number of paths having an even number of links, since this will give us the optimal choice of sign between i and j as mentioned above. More specifically, if the resulting value is positive, the node relevance between i and j is positive, otherwise negative.

3.1.2.2 Asymmetric Similarity Measure for Weighted Networks

Unsigned Asymmetric Similarity Measure for Weighted Networks (UASCOS++): This method is an enrichment of the ASCOS [58] to handle weighted networks. The formulation of ASCOS is the following:

$$\mathbf{R}_{ij} = \begin{cases} \frac{c}{|N_i^{in}|} \sum\limits_{k \in N_i^{in}} \mathbf{R}_{kj} & i \neq j \\ 1 & i = j \end{cases}$$

Let $\mathbf{P}_{ij} = \frac{\mathbf{A}_{ij}}{d_i^{in}}$ and we can rewrite the formulation as:

$$\mathbf{R} = c\mathbf{P}^\top \mathbf{R} + (1-c)\mathbf{I}$$

It defines the node relevance as the summation of normalized relevance from the incoming neighbors of i to j. The modifications for ASCOS++ were performed to handle weights on the edges. The formulation is shown below:

$$\mathbf{R}_{ij} = \begin{cases} c \sum\limits_{k \in N_i^{in}} \frac{\mathbf{A}_{ik}}{\sum\limits_{q \in N_i^{in}} \mathbf{A}_{iq}} (1 - e^{-A_{ik}}) \mathbf{R}_{kj} & i \neq j \\ 1 & i = j \end{cases} \tag{3.9}$$

The adjustment is that they now normalize each of the edge weights coming into i by the summation of all the incoming weights into i. The term $(1 - e^{-A_{ik}})$ maps the weights to be close to 1 when edge weights are large, and when the weights are small, it maps them close to 0.

Signed ASCOS++ (SASCOS++): ASCOS++ has difficulties to directly adapt to signed networks. Assume that a node i has an even number of incoming edges, where half the edges are positive, while the other half are negative. Therefore, this would lead to an undefined value as the summation over all incoming edges to i $\sum\limits_{q \in N_i^{in}} \mathbf{A}_{iq}$ is zero.

Another issue is if we directly apply ASCOS++, the resulting relevance score could contradict with balance theory. To ease our analysis in the following case, let $\kappa = \sum\limits_{q \in N_i^{in}} \mathbf{A}_{iq}$, $\lambda = \frac{\mathbf{A}_{ik}}{\kappa}$ and $\mu = (1 - e^{-A_{ik}})$. If $\mathbf{A}_{ik} = 1$ and κ is negative, hence λ is negative and μ is positive. Thus, if $\mathbf{R}_{kj}$ is

also positive, then the product of these three terms $\mathbf{R}_{ij}$ is negative and the resulting triad (+, +, −) does not follow balance theory. Similarity, when $\mathbf{R}_{kj}$ is negative, the product is positive and the resulting triad (+,−,+) is also not balanced.

Due to the fact using ASCOS++ with signed networks, could inherently disagree with balance theory, which motivates us to build SASCOS++. We note that when using ASCOS++ with signed networks, μ is equal to approximately 0.63 and -1.72 when $\mathbf{A}_{ik}$ is positive or negative, respectively. Thus, it is providing a stronger push in the similarity (by about three times) when seeing a negative link. Due to the imbalance of the numbers of positive and negative links in signed networks, we leave this μ term as is, but make a change to the normalization (i.e., κ). The formulation for SASCOS++ is shown below:

$$\mathbf{R}_{ij} = \begin{cases} c \sum\limits_{k \in N_i^{in}} \frac{\mathbf{A}_{ik}}{\sum\limits_{q \in N_i^{in}} |\mathbf{A}_{iq}|} (1 - e^{-A_{ik}}) \mathbf{R}_{kj} & i \neq j \\ 1 & i = j \end{cases} \tag{3.10}$$

*C*onnection to Balance Theory: It is easy to verify that SASCOS++ is able to have the relevance measurements aligning with balance theory. In other words, it will push more balanced triads.

3.1.2.3 Random Walk with Restart

Unsigned Random Walk with Restart (URWR): A random walker starting on node i that has a probability of $(1 - c)$ to return to i and with probability c chooses a neighbor of the current node to move to based on a transition matrix $\mathbf{W}$ (where $\mathbf{W}_{ij}$ is the probability that the walker starting at i will end at node j). We define this transition matrix as $\mathbf{W}_{ij} = \frac{1}{d_i}$ if i and j are connected and $\mathbf{W}_{ij} = 0$ otherwise (i.e., no link between i and j). With the intuition, URWR is formulated as [44]:

$$\mathbf{R} = c\mathbf{W}\mathbf{R} + (1 - c)\mathbf{I} = (1 - c)(\mathbf{I} - c\mathbf{W}^{\top})^{-1} \tag{3.11}$$

Signed Random Walk with Restart (SRWR): The transition matrix $\mathbf{W}$ has to be non-negative, thus we cannot directly apply URWR to signed networks. Therefore, we study signed random walk

with restart. Based on balance theory, the relevance score of u_k w.r.t u_i can be useful to infer that of u_j to u_i if there's a link from u_k to u_j. For example, if $\mathbf{A}_{kj} > 0$ (or u_k and u_j are friends), and $\mathbf{R}_{ik} > 0$ (or u_i and u_k are likely to be friends), it may suggest that u_i and u_j are friends (or $\mathbf{R}_{ij} > 0$) because friends' friends are friends. On the contrary, if $\mathbf{A}_{kj} < 0$ (or u_k and u_j are enemies) but $\mathbf{R}_{ik} > 0$ (or u_i and u_k are likely to be friends), it may indicate that u_i and u_j are enemies (or $\mathbf{R}_{ij} < 0$) because friends' enemies are enemies, which is implied from "the enemy of my enemy is my friend". This indicates that (1) u_j's relevance score to u_i can be indicated by these of nodes (e.g., u_k) that have links to u_j; and (2) the estimation also depends on the signs of links from u_k to u_j and the relevance scores from u_i to u_k. These intuitions suggested by balance theory pave us a way to build SRWR. Let $\bar{\mathbf{D}}$ be a diagonal matrix with its diagonal element $\bar{\mathbf{D}}_{ii} = \sum_k |\mathbf{A}_{ik}|$. In this way, $\bar{\mathbf{D}}_{ii}$ is the out degree of u_i considering both positive and negative links. Thus, the normalized weight of the link from u_i to u_k is given as

$$\bar{\mathbf{W}}_{ik} = \frac{|\mathbf{A}_{ik}|}{\bar{\mathbf{D}}_{ii}}$$

According to aforementioned intuitions, $\mathbf{R}_{ik}$ can be used to estimate $\mathbf{R}_{ij}$ with $\mathbf{A}_{kj} \neq 0$. Intuitively the portion of relevance score of u_k contributes to $\mathbf{R}_{ij}$ should be weighted by $\bar{\mathbf{W}}_{ij}$. This is to account for the number of neighbors of u_k. If $\bar{\mathbf{D}}_{ii}$ is large, then $\bar{\mathbf{W}}_{ij}$ is small and the effects of u_i to each of its neighbor is small. Thus, $\mathbf{R}_{ij}$ can be estimated as:

$$\mathbf{R}_{ij} \propto \sum_k sign(\mathbf{A}_{kj}) \bar{\mathbf{W}}_{kj} \mathbf{R}_{ik} \tag{3.12}$$

where $sign(\mathbf{A}_{kj})$ is used to encode the impact of the sign of the links. With sign introduced in the estimation of $\mathbf{R}_{ij}$, the relevance score can be both positive and negative. Two users with negative links can affect each other with negative relevance scores and thus can capture the semantic meanings of signed links.

With the analysis above, we are ready to discuss the details of SRWR. We focus on the relevance score of $u_j, j = 1, \ldots, n, j \neq i$ w.r.t u_i since the relevance scores w.r.t other nodes can be derived similarly. Firstly, $\mathbf{R}_{ij}, j = 1, \ldots, n, j \neq i$, are initialized to 0, which means that the relevance scores of u_j to u_i is unknown; while $\mathbf{R}_{ii}$ is initialized to 1 because u_i should be positively relevant

to itself. Now considering that a random walker starting from u_i. It can iteratively transmit to its neighborhood through positive and negative outgoing links. Each time the walker arrives at a node u_j, it will update $\mathbf{R}_{ij}$ by the relevance scores of nodes that have links to u_j. If the random walker arrives at u_i, then $\mathbf{R}_{ii}$ is updated as

$$\mathbf{R}_{ii} \leftarrow c \sum_k sign(\mathbf{A}_{ki}) \bar{\mathbf{W}}_{ki} \mathbf{R}_{ik} + (1-c) * 1 \tag{3.13}$$

where the first term of the right-hand side of Eq.(3.13) is the relevance score estimated from neighborhood, and the second term is to make sure that $\mathbf{R}_{ii} > 0$, i.e., u_i is relevant to itself. c is a scalar between 0 and 1, which is used to control the contribution of the two parts. If the random walker arrives at u_j, $j \neq i$, $\mathbf{R}_{ij}$ is updated as

$$\mathbf{R}_{ij} \leftarrow c \sum_k sign(\mathbf{A}_{kj}) \bar{\mathbf{W}}_{kj} \mathbf{R}_{ik} \tag{3.14}$$

Combining Eq.(3.12) and Eq.(3.13) together, $\mathbf{R}_{ij}$ is updated as

$$\mathbf{R}_{ij} \leftarrow c \sum_k sign(\mathbf{A}_{kj}) \bar{\mathbf{W}}_{kj} \mathbf{R}_{ik} + (1-c)\mathbb{I}(i,j)$$

where $\mathbb{I}(i,j)$ is a binary indicator function with $\mathbb{I}(i,j) = 1$ if $i = j$ and 0 otherwise. The random walker keeps moving until $\mathbf{R}$ doesn't change, which gives

$$\mathbf{R}_{ij} = c \sum_k sign(\mathbf{A}_{kj}) \bar{\mathbf{W}}_{kj} \mathbf{R}_{ik} + (1-c)\mathbb{I}(i,j) \tag{3.15}$$

By noticing that $sign(\mathbf{A}_{kj})\bar{\mathbf{W}}_{kj} = \frac{\mathbf{A}_{kj}}{\bar{\mathbf{D}}_{kk}}$, we define $\mathbf{S} = \bar{\mathbf{D}}^{-1}\mathbf{A}$ and then Eq.(3.15) can be written in matrix form with $\mathbf{R} = c\mathbf{R}\mathbf{S} + (1-c)\mathbf{I}$ where $\mathbf{I}$ is the identity matrix. The solution to the above equation is given as

$$\mathbf{R} = (1-c)(\mathbf{I} - c\mathbf{S})^{-1} \tag{3.16}$$

*C*orrectness: Here we show that SRWR is correct, i.e., $(\mathbf{I} - c\mathbf{S})^{-1}$ exists. The existence of $(\mathbf{I} - c\mathbf{S})^{-1}$ can be proofed using the following lemma, which is known as Levy-Desplanques theorem [59]. The Levy-Desplanques theorem is stated as follows

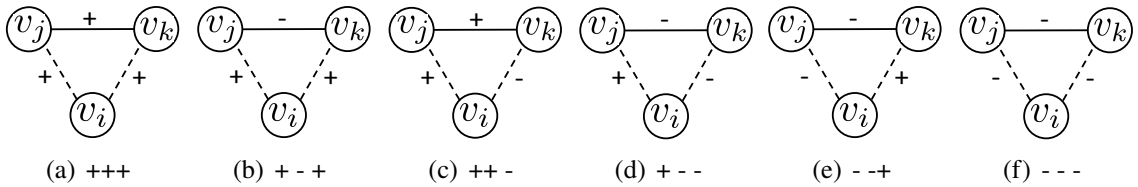


Figure 3.1: Triplets encountered during signed random walk.

Lemma 1. *Let* $\mathbf{P} \in \mathbb{R}^{n\times n}$ *be a square matrix.If* $|\mathbf{P}_{ii}| > \sum_{j\neq i} |\mathbf{P}_{ij}|$ *for all* $i = 1, \ldots, n$*, then* $\mathbf{P}$ *is nonsingular.*

Based on the above lemma, we have

Theorem 2. $\mathbf{I} - c\mathbf{S}$*,* $0 < c < 1$*, is non-singular.*

Proof. Let $\mathbf{P} = \mathbf{I} - c\mathbf{S}$. Since $\mathbf{S}_{ii} = 0$, we have $\mathbf{P}_{ii} = 1$. Also, $\sum_{j\neq i} |\mathbf{S}_{ij}|$ is given as

$$\sum_{j\neq i} |\mathbf{S}_{ij}| = \sum_{j} |\mathbf{S}_{ij}| = \sum_{j} \frac{|\mathbf{A}_{ij}|}{\bar{\mathbf{D}}_{ii}} = \frac{\sum_j |\mathbf{A}_{ij}|}{\bar{\mathbf{D}}_{ii}} = 1. \tag{3.17}$$

which leads to $\sum_{j\neq i} |\mathbf{P}_{ij}| = c \sum_{j\neq i} |\mathbf{S}_{ij}| = c$. Then we have $|\mathbf{P}_{ii}| > \sum_{j\neq i} |\mathbf{P}_{ij}|$ for all $i = 1, \ldots, n$. Thus, $\mathbf{I} - c\mathbf{S}$ is non-singular and $(\mathbf{I} - c\mathbf{S})^{-1}$ exists. □

*C*onnection to balance theory: Figure 3.1 gives representative triplets that will happen during the update process. The solid line with +/- means positive/negative links. The dashed line with +/- means $\mathbf{R}_{ij} > 0/\mathbf{R}_{ij} < 0$. According to the social balance theory [49], the resulting triads in Figures 3.1(a), 3.1(d) and 3.1(e) are balanced while the remaining three are unbalanced. Next we show that SRWR is likely to keep the balanced structures while reducing unbalanced structures during the updating process. For example, in Figure 3.1(a), $\mathbf{R}_{ik}\mathbf{S}_{kj} > 0$ will be added to $\mathbf{R}_{ij}$ according to Eq. (3.14), which increases the positive relevance score $\mathbf{R}_{ij}$. However, in Figure 3.1(b), $\mathbf{R}_{ik}\mathbf{S}_{kj} < 0$ will be added to $\mathbf{R}_{ij}$ that reduces the positive relevance score $\mathbf{R}_{ij} > 0$. $\mathbf{R}_{ij}$ will be consistently reduced until $\mathbf{R}_{ij}$ becomes negative (or the triad becomes balanced). Following a similar process, we can give similar observations for other triads. Thus, SRWR actually tends to learn relevance scores that increase the structural balance of a given signed network.

3.1.3 Experiments

In this section, we investigate the impact of signed relevance measurements on two signed network analysis tasks, i.e., sign prediction and tie strength prediction. We aim to answer the following two questions. As mentioned in the last section, we can have three strategies to adapt unsigned measurements for signed networks – (1) removing negative links; (2) ignoring signs; and (3) building advanced signed versions based on signed network properties and balance theory. Note that in the following subsections, given an unsigned measurement "X", we use "X-R" and "X-I" to denote the corresponding measurements applicable to signed networks by removing negative links and ignoring signs, respectively. For example, "UCN-R" and "UCN-I" denote the strategies of adapting "UCN" to signed networks by removing negative links and ignoring signs, separately. The first question we want to answer is – which strategy leads to better measurements. We have built numerous local and global measurements. The second question is – how they perform in different tasks.

For each of the parameterized measurements, we performed cross validation for the parameter tuning for each of the tasks. Among measurements discussed in the last section, common neighbor (CN), Jaccard Index (JI), and Preferential Attachment (PA)based measurements are designed for undirected networks; while ASCOS and RWR are for directed networks. As mentioned before directed measurements can be naturally applied to undirected ones by considering one undirected link as two directed links. Therefore, we conduct experiments with both undirected and directed settings.

3.1.3.1 Sign Prediction

The problem of sign prediction in signed networks is to predict whether an unlabeled links is positive or negative given knowledge of other link signs in the signed network. A previous study in unsigned networks suggested that good node relevance measurements generally are good for the prediction of links [60]. Therefore, the sign prediction performance can reflect the quality of relevance measurements.

Table 3.2: Performance comparison of link prediction under the undirected setting.

Metrics	Bitcoin-Alpha	Bitcoin-OTC	Slashdot	Epinions
UCN-R	0.500	0.523	0.520	0.520
UCN-I	0.501	0.497	0.508	0.508
SCN	0.671	0.716	0.549	0.629
UJI-R	0.499	0.524	0.513	0.522
UJI-I	0.497	0.489	0.503	0.512
SJI	0.669	0.725	0.550	0.630
UPA-R	0.497	0.587	0.571	0.634
UPA-I	0.481	0.475	0.484	0.498
SPA	0.559	0.628	0.641	0.634
UK-R	0.517	0.587	0.542	0.560
UK-I	0.488	0.482	0.498	0.538
SK	0.730	0.766	**0.693**	0.702
URWR-R	0.531	0.628	0.569	0.566
URWR-I	0.500	0.481	0.494	0.530
SRWR	0.751	**0.775**	0.677	0.703
UASCOS++-R	0.530	0.603	0.554	0.573
UASCOS++-I	0.496	0.484	0.497	0.537
SASCOS++	**0.765**	0.774	0.663	**0.705**

For each dataset, we randomly choose 80% as training, and the remaining as testing. We perform relevance measurements on the training set to get the relevance scores for each pair of users. The signed specific measurements can obtain a relevance score from $[-1, 1]$; hence we directly use the sign of the relevance score to indicate the sign of links. For “X-R" and “X-I", the relevance score is in “[0,1]". From the training data, we search an optimal threshold from the training data, and then if the relevance score is less than threshold, we predict a negative link and positive otherwise. Since positive and negative links are usually imbalanced in real-world signed networks, we use Area Under the Curve (AUC) as the metric to assess the performance of link prediction. For all four datasets, network information is available thus they all can be used in the link prediction experiment. Under the undirected setting, we ignore the directions of links following common practice in [4].

Sign Prediction Performance: The sign prediction comparison results are shown in Table 3.2

Table 3.3: Performance comparison of link prediction under the directed setting.

Metrics	Bitcoin-Alpha	Bitcoin-OTC	Slashdot	Epinions
UASCOS++-R	0.588	0.630	0.524	0.516
UASCOS++-I	0.562	0.639	0.519	0.493
SASCOS++	0.644	0.705	0.578	0.580
URWR-R	0.606	0.644	0.541	0.565
URWR-I	0.556	0.590	0.500	0.563
SRWR	**0.791**	**0.809**	**0.627**	**0.687**

and Table 3.3 for undirected and directed settings, respectively. We note that signed specific relevance measurements perform much better than these that (1) remove negative links and (2) ignore signs. These results suggest the importance of negative links in building node relevance measurements for signed networks. Meanwhile, global signed measurements consistently obtain better sign prediction performance than local signed measurements. We note that global methods consider long circles; while local methods only consider triads. This observation is consistent with that in [61] – long circles contain rich information in helping predict the signs of links. Under the directed setting, SASCOS++ also outperforms the ASCOS++ variants that (1) remove negative links and (2) ignore signs; while the signed RWR (i.e., SRWR) obtains the best performance.

3.1.3.2 Tie Strength Prediction

The relevance score for signed networks not only can indicate the signs of links but also can indicate the connection strengthen. Therefore, another possible application of relevance measurements is tie strength prediction, which aims to assign a weight to a link to indicate the connection strengthen [62, 63, 64]. In other words, the input of a tie strength prediction algorithm is an unweighted (or binary) network and the output is a weighted network.

We have only used the two Bitcoin datasets (Bitcoin-Alpha and Bitcoin-OTC) for this task as they are the only two of the four datasets that have a ground truth strength associated with each edge in the network. Note that we have normalized the two datasets to have their strength in the range [-1,1] to ensure easy mappings from our presented node relevance measurements to the tie

strengths associated with these datasets edges.

We provide the entire binary network as input and then attempt to predict the tie strength associated with each edge of the network. Note that we directly use the relevance scores of signed specific measurements as the predicted tie strength. While for "X-R" and "X-I", we use the similar strategy as sign prediction for tie strength prediction – we search an optimal threshold from the training data to map the relevance scores to [-1,1]. Therefore, we use root-mean-square error (RMSE) as the metric to evaluate the performance of tie strength prediction.

Tie Strength Prediction Performance: The tie strength prediction performance is demonstrated in Table 3.4 and Table 3.5 for undirected and directed settings, respectively. It can be observed from the Table 3.4 for the undirected setting:

The first observation Table 3.4 for the undirected setting is that the random tie strength prediction of picking values uniformly in the range [-1,1] results in the worst performance. Now, given the context of the random baseline performance, we further discuss the results of the relevance measurements. We note that most of the time, signed specific measurements outperform these that (1) remove negative links or (2) ignore signs for tie strength prediction. The overall best measurement in each dataset was a signed specific measurement. This further supports the importance of negative links in signed relevance measurements. Meanwhile, local signed measurements obtain comparable or even better performance than global signed measurements in tie strength prediction. This observation is different from that of sign prediction. To achieve better sign prediction performance, we only need to predict the sign accurately. However, for tie strength prediction, in addition to signs of links, we also need to predict the strength of the relevance correctly. Thus, local information could be good at predicting relevance strength. In fact, most existing tie strength prediction algorithms for unsigned networks only use local information [63, 62]. For the directed setting, we can see that again SRWR is the best performing measurement.

Table 3.4: Performance comparison of tie strength prediction under the undirected setting.

Metrics	Bitcoin-Alpha	Bitcoin-OTC
UCN-R	0.286	0.324
UCN-I	0.291	0.332
SCN	**0.277**	**0.308**
UJI-R	0.286	0.324
UJI-I	0.291	0.332
SJI	**0.277**	**0.308**
UPA-R	0.298	0.333
UPA-I	0.298	0.333
SPA	0.302	0.335
UK-R	0.290	0.326
UK-I	0.295	0.333
SK	0.284	0.320
URWR-R	0.294	0.329
URWR-I	0.296	0.331
SRWR	0.291	0.328
UASCOS++-R	0.292	0.328
UASCOS++-I	0.302	0.345
SASCOS++	0.299	0.334
Random	0.648	0.664

Table 3.5: Performance comparison of tie strength prediction under the directed setting.

Metrics	Bitcoin-Alpha	Bitcoin-OTC
UASCOS++-R	0.321	0.362
UASCOS++-I	0.319	0.364
SASCOS++	0.320	0.364
URWR-R	0.318	0.361
URWR-F	0.319	0.363
SRWR	**0.301**	**0.338**

3.2 Node Centrality Measurement in Signed Networks

Over the years, a large volume of research has focused on the development of centrality measures for networks. These measures seek to define a real-valued function on the nodes of a network, where these values can provide a ranking of the nodes based on how central (i.e., important) they are to the network. Social network analysis has primarily driven these efforts seeking to answer the question – "Who are the most important or central users in a network?". However, most of the literature has only focused on unsigned networks, but today there are many networks that can have positive and negative links (or signed networks), especially in online social media.

There have been recent attempts to define centrality when considering the inclusion of negative links and they can be roughly grouped into the following two categories: 1) separating the positive and negative links into two independent networks, then applying existing unsigned centrality measures to each network, and finally combining the isolated results [65]; and 2) handling positive and negative links simultaneously by treating negative links as either weak positive links or the negation of positive links [66]. Apparently, the separation of the positive and negative links is inherently losing vital information as they fail to capture the interactions between them. It is also evident that negative links have very different properties from positive links and they are not the negation of positive links [67] and so methods in the second group are also insufficient to handle signed networks. Hence, new signed centrality measures are still desired.

Deep learning has been proven to not only be powerful in learning and extracting complex patterns in data [68, 69, 70, 71] but also being able to approximate functions [72, 73]. Given these advantages, deep learning has been used to advance various analytical and mining tasks in complex networks such as learning representations of networks [74, 75, 76, 77], generative network modeling [78, 79, 80, 81] and node classification [82]. In addition, we have seen deep learning's utility in a plethora of other applications [83, 84, 85] and efforts to understand them have seen continued improvements [86]. Furthermore, the use of a deep neural network would allow the incorporation of multiple perspectives in defining a signed centrality measurement including multiple social theories and higher-order structural information, while also having the inductive

properties such that centrality can be calculated across networks (i.e., training the deep model on one signed network and then utilizing it for calculating the centralities of the nodes in another network). Therefore, due to the complexities already inherent in unsigned networks and even more introduced by negative links, along with the previously stated benefits, deep models have the potential to capture the complexities for an advanced node centrality in signed networks.

Therefore, we aim to investigate the problem of developing a dedicated centrality measurement specific for signed networks. We propose a deep framework for learning a signed centrality score for each user guided by status and balance theories.

3.2.1 An Overview of Deep Signed Centrality (DeSCent) Measurement

Node centrality in unsigned networks is to measure the status of users in a network such that a more "central" user has a higher value, while other typical "normal" users have lower values. These measures are based on the network structure and also typically have intuitive physical interpretations as to which users are being targeted to have a higher centrality based on their respective definitions. In signed networks, in addition to differentiating between "normal" and "important" users due to the introduction of negative links, we note a user can be important in either a positive or negative way (e.g., famous or infamous users).

Hence, we need a dedicated way to realize the signed centrality value for each user. However signed networks are inherently very complex due to the fact that users can form relations to other users with both positive and negative links. We therefore propose to use a deep learning framework to learn a signed centrality mapping $f : u_i \rightarrow \mathbf{c}_i$ that projects a user u_i to their learned/corresponding signed centrality value $\mathbf{c}_i$. The benefits of the deep neural network for signed centrality are three fold: 1) has the significant benefit that allows for the deep network parameters learned in one signed network to be utilzed for signed centrality calculation in other (perhaps much larger) networks without needing to train a new model for the other network, 2) the deep network can better capture the complex patterns in the signed network found in the feature input coming from the interactions of both positive and negative links, 3) it also allows us to construct our definition of signed centrality to

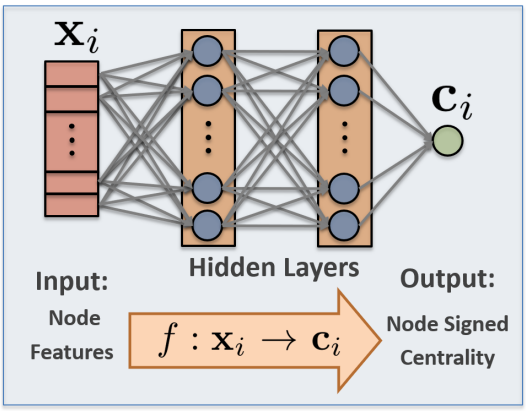


Figure 3.2: An illustration of our deep neural network for learning signed centrality scores.

easily include multiple perspectives including higher-order structures and multiple social theories.

Due to the fact that users status is related to their connections in the network, we choose to represent each user u_i with a feature vector $\mathbf{x}_i$ extracting from their connections. More specifically, we extract a set of node and local neighborhood features. However, other approaches could be taken to construct $\mathbf{x}_i$, such as signed network embedding techniques [87, 88, 89], but and we leave this as one future work. Thus, we can redefine the mapping we wish to discover as $f : \mathbf{x}_i \rightarrow \mathbf{c}_i$.

Figure 3.2 illustrates the deep learning framework for learning the signed centrality score for each user. We let our deep model, parameterized by θ, be represented by f_θ, such that it defines the mapping $f_\theta(\mathbf{x}_i) \rightarrow \mathbf{c}_i$. Note that it is not a supervised task since we do not have the "ground truth" signed centrality scores. Hence, learning the mapping function is challenging. Thus, we seek for discovering the mapping function that can optimize an objective related to centrality.

In the remainder of this section, we develop our Deep Signed Centrality (DeSCent) measurement using the two social theories on signed networks, namely status [2] and balance [30, 29] theories. We first develop our objective function for DeSCent, then discuss some details of the deep neural network and the optimization procedure used to train it.

3.2.2 Signed Centrality Measurement Objective Function

This subsection is organized in four parts: first, we introduce the basic objective for signed centrality based on status theory and eigenvector centrality to capture local information. Thereafter we propose to harness balance theory in our signed centrality to include global information. Finally, two additional constraints are added to our measurement's objective. After introducing the idea for each component of DeSCent, we formalize them into an objective function that our deep neural network can optimize to learn the signed centrality score for each user.

3.2.2.1 Signed Centrality Based on Status Theory

For defining a user's signed centrality, we first want to discuss the usage of status theory [2]. The theory states that a positive (or negative) link from u_j to u_i is implying u_j's opinion that u_i is of a higher (or lower) status (i.e., rank) than user u_j. Therefore, if we want to utilize the collective opinions of other users in the network to describe the ranking of u_i, we can derive the following:

$$\mathbf{c}_i = |N_i^{in+}| - |N_i^{in-}| = \sum_j \mathbf{A}^+_{ji} - \sum_k \mathbf{A}^-_{ki} \tag{3.18}$$

Note that we do not use the ratio of positive and negative links since in our setting, centrality scores can be both positive and negative. However this does not take into account the actual status of the users giving their links to u_i, but instead simply counting the number of links u_i receives of each type. Thus, based on the ideas of Eigenvector centrality [90], we can modify Eq. (3.18) to be the following:

$$\mathbf{c}_i = \sum_{j \in N_i^{in+}} \mathbf{c}_j - \sum_{k \in N_i^{in-}} \mathbf{c}_k = \sum_j \mathbf{c}_j \mathbf{A}^+_{ji} - \sum_k \mathbf{c}_k \mathbf{A}^-_{ki} \tag{3.19}$$

We can see that now, rather than counting the number of positive and negative incoming neighbors, we utilize the centrality of u_i's neighbors to weight these links and construct a recursive definition of signed centrality.

3.2.2.2 Harnessing Balance Theory and Higher-order Structures

Until now our formulation only includes information from single directed signed links. However, it has been shown that not all links are the same (i.e., of equal strength) in social networks and in fact there is a spectrum of strength implicitly associated with every connection. In unsigned networks, one heavily studied heuristic to determine the strength of links (both from a social theoretical standpoint [91] and empirically [62]) is the use of local triangles. Similarly in signed networks we use local triangles with structural balance theory [30, 29] to further differentiate the types of signed triangles.

Balance theory tells us that balanced triangles are more likely to form in social networks as compared to unbalanced ones. A balanced (or unbalanced) triangle is defined as having an even (or odd) number of negative links. The theory implies that any such unbalanced triangles are unstable (due to higher frustration in those social triads) and this is the reason they are less likely to exist. We can utilize these differences and parameterize both types of triangles such that we can differentiate between those adhering to the social theory (i.e., balanced) and those that do not (i.e., unbalanced) while adding this local clustering (i.e., higher-order) information to our signed centrality measurement.

We therefore propose to utilize not only the relationships between users on the single link level (as shown so far in Eq. (3.19)), but also with triangles in an attempt to more accurately define a node's signed centrality. One intuition is that we are more likely to trust the opinion u_j gives to u_i if they have a "stronger" connection (i.e. have more common neighbors). Note that whether the link from u_j to u_i is positive or negative we assume that when these two users are involved in more triangles, they have a better sense at judging the status of one another and therefore their opinion (i.e., directed signed link) should have a higher weight. This provides a principled way for estimating signed edge strength, and can be further understood through the following example: if a user u_i is positively connected to three users u_x, u_y, and u_z, but shares a balanced triad with u_x, an unbalanced triad with u_y, and no triangles with u_z, we might want to infer the strength of the given links to u_i are not equal from all three neighbors. More specifically, we want to parameterize the

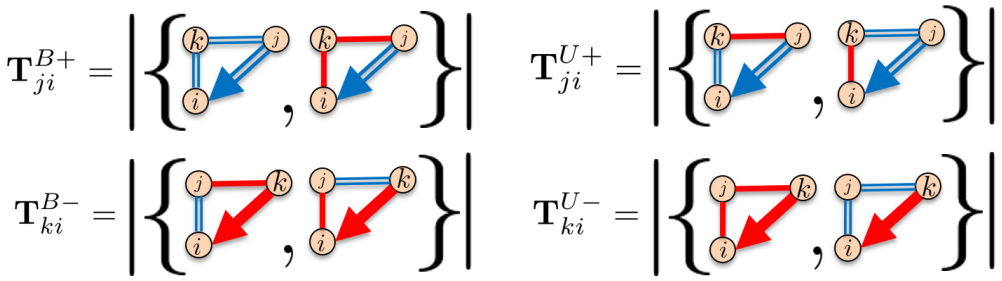


Figure 3.3: An illustration of how we calculate the matrices $\mathbf{T}^{B+}$, $\mathbf{T}^{U+}$, $\mathbf{T}^{B-}$, and $\mathbf{T}^{U-}$.

triangles such that u_i receives their ranking based more on u_x and u_y (however not necessarily the same importance for these two) over u_z. Although this example is given in the context of u_i having three positively linked neighbors, a similar logic applies for negatively linked neighbors.

Here we define the matrix $\mathbf{T}^+$ which will represent the relations between positive linked pairs of users based on the number of triangles they have in common as follows:

$$\mathbf{T}^+_{ji} = |\{(i,j,k)|(j,i) \in \mathcal{E}^+ \text{ and where } u_i, u_j, \text{ and } u_k \text{together form a triangle}\}|$$

Similarly we can define for negatively linked pairs a matrix $\mathbf{T}^-$ as follows:

$$\mathbf{T}^-_{ki} = |\{(i,j,k)|(k,i) \in \mathcal{E}^- \text{ and where } u_i, u_j, \text{ and } u_k \text{together form a triangle}\}|.$$

Next, we further separate the triangles relation matrices based on whether they are balanced (i.e., having an even number of negative links) or unbalanced (i.e., having an odd number of negative links). More specifically, we separate $\mathbf{T}^+$ into $\mathbf{T}^{B+}$ and $\mathbf{T}^{U+}$, for the balanced and unbalanced triangles, respectively, and similarly separate $\mathbf{T}^-$ into $\mathbf{T}^{B-}$ and $\mathbf{T}^{U-}$. Figure 3.3 shows how we define and calculate the matrices $\mathbf{T}^{B+}$, $\mathbf{T}^{U+}$, $\mathbf{T}^{B-}$, and $\mathbf{T}^{U-}$ where blue double (or red single) lines represent the positive (or negative) links. We extend Eq. (3.19) to obtain the below formulation:

$$\mathbf{c}_i = \sum_j \frac{\mathbf{c}_j}{\mathbf{p}_j}\left(\mathbf{A}^+_{ji} + \beta^+\mathbf{T}^{B+}_{ji} + \eta^+\mathbf{T}^{U+}_{ji}\right) - \sum_k \frac{\mathbf{c}_k}{\mathbf{p}_k}\left(\mathbf{A}^-_{ki} + \beta^-\mathbf{T}^{B-}_{ki} + \eta^-\mathbf{T}^{U-}_{ki}\right) \tag{3.20}$$

where β^+ and η^+ are now used to control the contribution of shared balanced and unbalanced triads between u_j and u_i, respectively, and similarly for β^- and η^- with u_k and u_i. We define $\mathbf{p}_z$ for a

user u_z as:

$$\mathbf{p}_z = \sum_y \left(\mathbf{A}^+_{zy} + \mathbf{A}^-_{zy} + \beta^+ \mathbf{T}^{B+}_{zy} + \beta^- \mathbf{T}^{B-}_{zy} + \eta^+ \mathbf{T}^{U+}_{zy} + \eta^- \mathbf{T}^{U-}_{zy} \right)$$

Note that the utilization of the normalizing vector $\mathbf{p}$ is based on the same idea as the normalization in PageRank [92]. More specifically, this is done to prevent a user of very high absolute centrality from distributing their influence too much to other users and also allows the measurement to be less susceptible to malicious users attempting to boost (or shrink) rankings.

We point out that we continue to incorporate the single directed edge information from $\mathbf{A}^+$ and $\mathbf{A}^-$. This is because if there exists no triangle involving a pair of nodes, but they have an existing edge, then the corresponding values in the four triangle matrices would all have zero. However, we seek to include the triangle information to strengthen certain edges in the network, but not wanting to remove existing "weaker" connections. Although we only focus on triangles to include 2-hop information, balance theory can be applied to circles of any length. Thus, we can extend our work to consider longer circles to capture more global information and we will leave it as one future work.

In terms of our objective for DeSCent, if we let $f_\theta(\mathbf{x}_i)$ replace $\mathbf{c}_i$ for a given user u_i, then Eq. (3.20) can be converted into the below objective:

$$\forall u_i \in \mathcal{U} : f_\theta(\mathbf{x}_i) = \left[\sum_j \left(\frac{f_\theta(\mathbf{x}_j)}{\mathbf{p}_j} \left(\mathbf{A}^+_{ji} + \beta^+ \mathbf{T}^{+B}_{ji} + \eta^+ \mathbf{T}^{+U}_{ji} \right) \right) - \sum_k \left(\frac{f_\theta(\mathbf{x}_k)}{\mathbf{p}_k} \left(\mathbf{A}^-_{ki} + \beta^- \mathbf{T}^{-B}_{ki} + \eta^- \mathbf{T}^{-U}_{ki} \right) \right) \right]$$

The above ensures our network correctly maps the user feature vectors to centrality scores that match our recursive definition. This leads to the following minimizing problem:

$$\min_\theta \ \mathcal{L}(\theta) = \sum_{u_i \in \mathcal{U}} \left(f_\theta(\mathbf{x}_i) - \left(\sum_j \frac{f_\theta(\mathbf{x}_j)}{\mathbf{p}_j} \left(\mathbf{A}^+_{ji} + \beta^+ \mathbf{T}^{+B}_{ji} + \eta^+ \mathbf{T}^{+U}_{ji} \right) - \sum_k \frac{f_\theta(\mathbf{x}_k)}{\mathbf{p}_k} \left(\mathbf{A}^-_{ki} + \beta^- \mathbf{T}^{-B}_{ki} + \eta^- \mathbf{T}^{-U}_{ki} \right) \right) \right)^2 + Reg(\theta) \tag{3.21}$$

where the last term $Reg(\theta)$ is a regularization term on the deep neural network parameters.

3.2.2.3 Additional DeSCent Measurement Constraints

We note that there exists a critical problem in the proposed objective function. If a user u_i has no incoming links, but only outgoing links, then their centrality would be zero (or undefined). In fact, this problem actually diffuses throughout the network because for another user u_x, even if u_x has incoming links, if those incoming neighbors are all similar to u_i (in that they themselves have no incoming links) then u_x will also have a centrality of zero (or undefined) [17]. One solution to this issue (and the one we have included in DeSCent) is to define users having no incoming links to have some small constant signed centrality value γ. We can see that this is similar to how PageRank [92] assigns the zapping probability for all nodes. Furthermore, we also want to prevent our network from optimizing a trivial solution where the signed centrality value is zero for all users. Therefore we enforce a constraint having the sum lowerbound on the absolute signed centrality for all users to be $\gamma|\mathcal{U}|$, i.e., the L1-norm of $\mathbf{c}$ should be at least γ times the number of users in the network. In the rest of the paper, we will refer to this constraint as the *sum constraint*. We therefore also seek to minimize $\mathcal{L}(\theta)$ with respect to θ according to the below constraint:

$$||\mathbf{c}||_1 = \sum_{u_i \in \mathcal{U}} |\mathbf{c}_i| \geq \gamma|\mathcal{U}|$$

which can be represented as minimizing the following after substituting the network output $f_\theta(\mathbf{x}_i)$ for $\mathbf{c}_i$:

$$\max\left(0, \gamma|\mathcal{U}| - \sum_{u_i \in \mathcal{U}} |f_\theta(\mathbf{x}_i)|\right) \tag{3.22}$$

The second additional constraint we place on our signed centrality measurement is based on the sign of the centrality values. We note that although we want to utilize our more comprehensive formulation for the signed centrality, we still wish to have the signed centrality scores maintain the correct sign suggested by status theory. This can be performed by requiring the centrality of u_i to be the same sign as suggested by status theory, which can be expressed formally as: $sign(\mathbf{c}_i) = sign(|N_i^{in+}| - |N_i^{in-}|)$. In terms of minimization (and similarly substituting $f_\theta(\mathbf{x}_i)$ for

$\mathbf{c}_i$), this can be rewritten as the following:

$$\sum_{u_i \in \mathcal{U}} \mathbb{I}[i]\big(|f_\theta(\mathbf{x}_i)| + \zeta\big) \tag{3.23}$$

where we add the margin $\zeta = 1$ to force $f_\theta(\mathbf{x}_i)$ to be the same sign by pushing through zero to the correct sign, and $\mathbb{I}[i]$ is an indicator function. The indicator function's purpose is to equal 1 whenever the sign of the centrality does not match the sign suggested by status theory and 0 otherwise. Formally we have defined $\mathbb{I}[i]$ below:

$$\mathbb{I}[i] = \begin{cases} 1 & \text{if } \big(f_\theta(\mathbf{x}_i) \times (|N_i^{in+}| - |N_i^{in-}|)\big) < 0 \\ 0 & \text{otherwise} \end{cases}$$

where we can see that if $f_\theta(\mathbf{x}_i)$ and $(|N_i^{in+}| - |N_i^{in-}|)$ have differing signs then $\big(f_\theta(\mathbf{x}_i) \times (|N_i^{in+}| - |N_i^{in-}|)\big) < 0$ and therefore $\mathbb{I}[i] = 1$ (and $\mathbb{I}[i] = 0$ otherwise). We refer to this term as the *status constraint*.

We can now construct DeSCent's full objective function using Eqs. (3.21), (3.22), and (3.23) as the following:

$$\begin{aligned} \min_\theta \ \mathcal{L}(\theta) = \sum_{u_i \in \mathcal{U}} \Bigg(f_\theta(\mathbf{x}_i) - \Big(\sum_j \frac{f_\theta(\mathbf{x}_j)}{\mathbf{p}_j} \big(\mathbf{A}^+_{ji} + \beta^+ \mathbf{T}^{+B}_{ji} + \eta^+ \mathbf{T}^{+U}_{ji}\big) - \sum_k \frac{f_\theta(\mathbf{x}_k)}{\mathbf{p}_k} \big(\mathbf{A}^-_{ki} + \beta^- \mathbf{T}^{-B}_{ki} + \eta^- \mathbf{T}^{-U}_{ki}\big) \Big) \Bigg)^2 \\ + \lambda_1 \Bigg(\max\big(0, \gamma|\mathcal{U}| - \sum_{u_i \in \mathcal{U}} |f_\theta(\mathbf{x}_i)|\big) \Bigg) + \lambda_2 \Bigg(\sum_{u_i \in \mathcal{U}} \mathbb{I}[i]\big(|f_\theta(\mathbf{x}_i)| + \zeta\big) \Bigg) + Reg(\theta) \end{aligned} \tag{3.24}$$

where λ_1 and λ_2 are introduced to regularize the two terms from our additional constraints.

3.2.3 Overall DeSCent Deep Network Framework

Now, having the objective defined for our deep signed centrality measurement, next we describe the deep network structure, and the optimization procedure used.

To optimize the objective given in Eq. (3.24) we use a 3-layer fully connected network that consisting of 500 hidden neurons per layer and using LeakyReLU [93] activation functions on the hidden layers with negative slope 0.2. We perform batch gradient descent using ADAM [94] with an initial learning rate set to 0.001.

Algorithm 3.1: Optimization procedure for DeSCent.

Input: $\mathcal{G} = (\mathcal{U}, \mathcal{E}^+, \mathcal{E}^-)$
Output: $\mathbf{c}$

1 Respectively create $\mathbf{A}^+$ and $\mathbf{A}^-$ from $\mathcal{E}^+$ and $\mathcal{E}^-$
2 Construct $\mathbf{T}^{B+}$, $\mathbf{T}^{B-}$, $\mathbf{T}^{U+}$, $\mathbf{T}^{U-}$ from $\mathbf{A}^+$ and $\mathbf{A}^-$
3 Use $\mathbf{A}^+$ and $\mathbf{A}^-$ to extract node features $\mathbf{X}$
4 Randomly initialize the neural network parameters θ
5 **while** Not convergent **do**
6 Create constant vector $\mathbf{k}$ where $\mathbf{k}_z \leftarrow f_\theta(\mathbf{x}_z)$ for user u_z
7 Calculate gradient of $\mathcal{L}(\theta)$ using $f_\theta(\mathbf{x}_i)$ after replacing $f_\theta(\mathbf{x}_j)$ and $f_\theta(\mathbf{x}_k)$ with constants $\mathbf{k}_j$ and $\mathbf{k}_k$, respectively
8 Update parameters θ using batch gradient descent
9 Construct signed centrality vector $\mathbf{c}$ where $\mathbf{c}_i \leftarrow f_\theta(\mathbf{x}_i)$

Algorithm 3.1 details the steps for optimizing our model. On line 1 we construct the two adjacency matrices $\mathbf{A}^+$ and $\mathbf{A}^-$, and thereafter on line 2 create the four signed triangle motif matrices (separated based on whether the triangles are balanced or unbalanced). Line 3 extracts node features based on the network structure. Then, on line 4, the parameters θ of the deep neural network are randomly initialized. Lines 5 to 8 loop until convergence and discuss how to calculate the gradient of our objective. We construct a constant vector $\mathbf{k}$ on line 6, which contains DeSCent's current proposed signed centrality values. Next on line 7, to perform the update to θ we replace $f_\theta(\mathbf{x}_j)$ and $f_\theta(\mathbf{x}_k)$ with $\mathbf{k}_j$ and $\mathbf{k}_k$, respectively, and by holding their values, we are effectively treating $\Big(\sum_j \frac{f_\theta(\mathbf{x}_j)}{\mathbf{p}_j} \big(\mathbf{A}^+_{ji} + \beta^+ \mathbf{T}^{+B}_{ji} + \eta^+ \mathbf{T}^{+U}_{ji}\big) - \sum_k \frac{f_\theta(\mathbf{x}_k)}{\mathbf{p}_k} \big(\mathbf{A}^{-B}_{ki} + \beta^- \mathbf{T}^{-B}_{ki} + \eta^- \mathbf{T}^{-U}_{ki}\big) \Big)$ in our objective (i.e., Eq. (3.24)) as a constant when calculating the error and only using the derivative in relation to $f_\theta(\mathbf{x}_i)$. This update procedure is repeated until convergence using batch gradient descent.

3.2.4 Experiments

In this section, we conduct experiments to evaluate the effectiveness of the proposed deep signed centrality measurement (DeSCent). We seek to answer the following three questions: (1) Can DeSCent learn better signed centrality scores than other existing signed centrality measurements? (2) Can the use of deep learning enable centrality cross networks? and (3) How do the parameters of DeSCent affect its performance?

Addressing the first two questions is not straight-forward because we do not have "ground-truth" signed centrality values (i.e., a signed centrality ranking of the users of a signed network). It is observed that both positive and negative links follow the power law distributions [67]; thus the link formation in signed networks is related to node status [95]. Therefore, following the common centrality evaluation in the literature [65], we perform an indirect approach to evaluate the quality of the signed centrality values by utilizing them for the signed link prediction problem. We will further discuss the signed link prediction problem and the results of these experiments later in this section. Then, to answer the second question, we perform further experiments to evaluate how general the learned deep networks are for mapping the node features to their corresponding centrality scores, which is evaluated by training on a single dataset and then utilizing the learned network to calculate the centrality of nodes in the other datasets. To address the third question, we perform a parameter analysis on DeSCent to observe the contribution of balance theory (β^+, β^-, η^+, and η^-) and the *sum* and *status* constraints (controlled by λ_1 and λ_2, respectively).

Extracting Node Features: As noted before, the node features can be extracted manually or learnt automatically via embedding from the network structure. Here we will first try manual extraction and leave automatic embedding as one future work. We propose to extract three groups of features for each node – the given node's signed degree distribution, their neighbors signed degree distribution, and the number of balanced/unbalanced triangles they are involved in. Below we define and discuss each feature extracted.

First we discuss how to extract the node signed degree distribution features for a user u_i. These 4 features are the in/out positive/negative degrees for the given user u_i (i.e., $|N_i^{in+}|$, $|N_i^{in-}|$, $|N_i^{out+}|$, and $|N_i^{out-}|$).

For the group of signed degree distribution of u_i's neighbors, we extract the average in/out positive/negative features. However, we obtain four different sets of these averages based on averaging over neighbors that linked with u_i using one of the four possible directed signed links. This provides an additional 16 features. For example, one of these features would be the average incoming negative degree for the set of neighbors that u_i has given a positive link to. If we were to

denote this example feature as $\mathbf{x}_{i*}$, then more formally this can be defined as follows:

$$\mathbf{x}_{i*} = \frac{1}{|N_i^{out+}|} \sum_{u_j \in N_i^{out+}} |N_j^{in-}| \tag{3.25}$$

Note that the other 15 neighbor based sign distribution features for user u_i can be defined similar to $\mathbf{x}_{i*}$.

Finally the last two features are the number of undirected balanced and unbalanced triangles u_i is involved in. These two features can be easily calculated with vector and matrix operations on $\mathbf{S}^+$ and $\mathbf{S}^-$, which are the undirected symmetric versions of $\mathbf{A}^+$ and $\mathbf{A}^-$. Here we show how to calculate the balanced (i.e., Δ_i^B) and unbalanced (i.e., Δ_i^U) triangles for user u_i as:

$$\Delta_i^B = (\mathbf{S}_i^{+T}\mathbf{S}^+\mathbf{S}_i^+)/2 + (\mathbf{S}_i^{+T}\mathbf{S}^-\mathbf{S}_i^-) + (\mathbf{S}_i^{-T}\mathbf{S}^+\mathbf{S}_i^-)/2 \quad \Delta_i^U = (\mathbf{S}_i^{-T}\mathbf{S}^-\mathbf{S}_i^-)/2 + (\mathbf{S}_i^{-T}\mathbf{S}^+\mathbf{S}_i^+) + (\mathbf{S}_i^{+T}\mathbf{S}^-\mathbf{S}_i^+)/2$$

Signed Link Prediction: Since the link formation in signed networks is related to node status, following the tradition [65], we compare the signed centrality measurements by using them to perform the signed link prediction task. The problem of link prediction in signed networks is to predict new positive and negative links when given an existing signed network [4]. For every user u_i in the signed network, we construct a feature vector $\mathbf{f}_i$ consisting of 5 features based on the computed signed centrality. However, before constructing the feature vectors, we normalize all signed centrality measurements for a fair comparison across signed centrality measurements. The first feature, $\mathbf{f}_{i1}$, is the centrality value $\mathbf{c}_i$ for the user u_i themselves. The other four are the average centrality scores associated with the neighbor sets of u_i when categorized based on incoming/outgoing positive/negative connections. The formulations are as follows:

$$\mathbf{f}_{i2} = \frac{1}{|N_i^{in+}|} \sum_{u_j \in N_i^{in+}} c_j, \quad \mathbf{f}_{i3} = \frac{1}{|N_i^{in-}|} \sum_{u_k \in N_i^{in-}} c_k, \quad \mathbf{f}_{i4} = \frac{1}{|N_i^{out+}|} \sum_{u_j \in N_i^{out+}} c_j, \quad \mathbf{f}_{i5} = \frac{1}{|N_i^{out-}|} \sum_{u_k \in N_i^{out-}} c_k$$

We approach the signed link prediction problem as a classification problem similar to that done in [4]. For every edge e_{ij}, we can construct a feature vector that is the concatenation of feature vector $\mathbf{f}_i$ and $\mathbf{f}_j$ and the label is based on whether the edge e_{ij} was positive or negative. We train a logistic regression model on the training dataset of edges and then predict the signs of unseen links (i.e., those in the testing set).

For the evaluation of the signed link prediction binary classification problem we use Area Under the receiver operating characteristics Curve (AUC), since in real-world datasets the positive and negative links are typically imbalanced (i.e., significantly more positive links than negative links). Note that a higher AUC means a higher probability we rank a randomly selected positive edge higher than a randomly selected negative one and therefore the higher the better the performance. For each dataset, we randomly use 90% as training, and the remaining 10% as testing. Hyperparameter tuning used cross-validation on the training set.

3.2.4.1 Performance Comparison

Here we present some existing signed network centrality measurements such that we can study the effectiveness of our proposed measurement. We have selected baseline methods that were designed for determining the centrality or importance of nodes in signed networks. We note that for succinctness we have selected representative measurements that include recent measurements and those that have shown to perform well. Similarly, for the sake of space, we do not include any comparison against unsigned node ranking measurements, but our experiments when ignoring negative links or treating them equivalent to positive links have shown to perform significantly worse on the signed link prediction task. Below we have categorized the baselines into two groups: 1) Single Network Baselines; and 2) Separate Network Baselines.

Single Network Baselines: those that utilize the positive and negative links together while calculating the signed centrality values.

- Signed Spectral Ranking (SR) [24]: This method computes the dominant left eigenvector of the signed adjacency matrix $\mathbf{A}$.
- Exponential Ranking (ER) [51]: This method simultaneously uses both positive and negative links and is based upon PageRank [92]. It utilizes a heuristic approach on an exponential variation and has a fixed-point solution if the exponential parameter μ is selected appropriately.

- Signed Random Walk with Restart (sRWR) [96]: This model is state-of-the-art for personalized ranking in signed networks based on the unsigned random walk with restart method and incorporates balance theory. We note that this method is not the same as the signed random walk with restart method we proposed in Section 3.1. The centrality $\mathbf{c}_i$ of a user u_i can thus be the summation of personalized rankings for all other users u_j towards u_i.

Separate Network Baselines: those that split the positive and negative links into two independent networks and then at the end combine two separate centrality values that were calculated independently.

- Modified PageRank (MPR) [65]: Here PageRank [92] is performed on the positive only and negative only networks (i.e., $\mathbf{A}^+$ and $\mathbf{A}^-$, respectively) and then the negative centrality scores are subtracted from the positive centrality scores.

- Modified HITS (MHITS) [65]: This method recursively calculates the hubs and authorities scores separately for the positive only and negative only networks. Then centrality is the authority The final signed centrality is the authority score on the positive network minus the authority score on the negative network.

The parameter settings for the baselines were set as follows: 1) for sRWR we use the selected parameter values of $\beta = 0.6$ and $\gamma = 0.9$ for Slashdot, $\beta = 0.5$ and $\gamma = 0.9$ for the Epinions, and the two Bitcoin datasets use the same a Epinions, since these were the selected parameters from a grid search in [96] for Slashdot and Epinions, and since the two Bitcoin datasets have a more similar balance and positive/negative link ratio to Epinions as compared to Slashdot; 2) for MPR and SR we used $\alpha = 0.15$ for the zapping probability as commonly used in practice [24, 65]; 3) for ER we use $\mu = 2$ as this value satisfies their convergence requirement discussed in [51] based on signed edge weights as either -1 or 1. For the parameters of our DeSCent measurement we had performed a grid search over a set of parameter values. More specifically we varied $\lambda_1, \lambda_2, \beta$, and η while fixing $\gamma = 0.1$ and $\zeta = 1$. Note that in our experiments we fixed $\beta^+ = \beta^-$ and $\eta^+ = \eta^-$ and we use β and η to denote these merged parameter values, respectively.

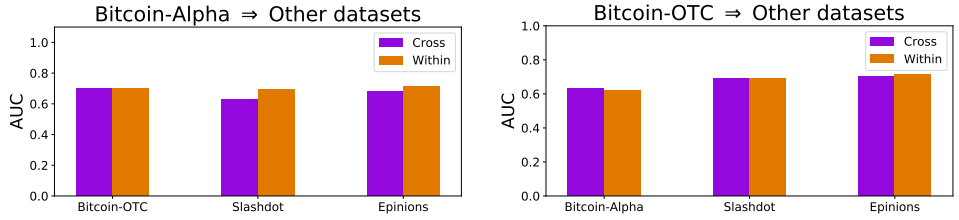
Table 3.6: Signed link prediction results with AUC.

Centrality Measurement	Bitcoin-Alpha	Bitcoin-OTC	Slashdot	Epinions
SR	0.567	0.565	0.570	0.669
ER	0.598	0.619	0.602	0.666
sRWR	0.576	0.570	0.570	0.666
MPR	0.618	0.692	0.593	0.658
MHITS	0.602	0.650	0.680	0.667
DeSCent	**0.622**	**0.702**	**0.692**	**0.713**

Comparison Results: The results across our four signed networks can be found in Table 3.6. We first observe that most of the time, single network baselines obtain worse performance. These observations support that we should not consider negatives links as neither weak positive links nor the negation of positive links and distinguishing positive and negative links is necessary. We also note that our Deep Signed Centrality (DeSCent) measurement has the best AUC across all four datasets. Due to the fact that the AUC metric is more sensitive to incorrectly handling negative relations, we believe the better performance is DeSCent's ability to utilize a deep neural network (which can effectively extract more complex patterns between the positive and negative links) along with the utilization of both status and balance theories along with higher-order relations in our objective function.

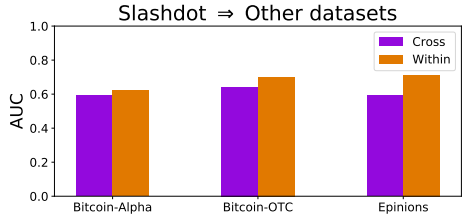
3.2.4.2 Generalization Across Datasets

In this subsection we seek to further quantify the advantages of using deep learning in defining our deep signed centrality measurement. Thus we perform experiments to test how well our DeSCent deep network framework is able to generalize across signed network datasets. More specifically, we perform inductive experiments where we train a deep network on a single dataset and then utilize that learned model to extrapolate the centrality scores in other networks from their respective node features. This for one tests whether the deep framework is learning specific properties nested inside each signed network dataset, or if it is leaning more general patterns that are inherently found in all signed network datasets.

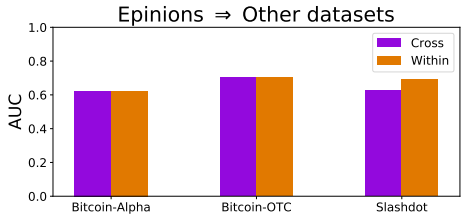


(a) Bitcoin-Alpha

(b) Bitcoin-OTC



(c) Slashdot



(d) Epinions

Figure 3.4: Signed link prediction performance comparison of within versus cross training.

Here we present the AUC for the signed link prediction comparing the performance when their centralities were calculated from a deep network trained on the same network (which we denote as "within") or when utilizing the deep network trained on another dataset and thus performing centrality calculations across signed networks (which we denote as "cross"). This can be seen in Figure 3.4, where we have shown the generalization across datasets when training in each of our four datasets and applying to the other three. The main observation we can make from Figure 3.4 is that indeed the parameters learned for the deep networks are very robust and general, since the performance when trained cross networks is very similar (and in fact sometimes slightly better or identical) to the centralities calculated within the same network. We note that there are many advantages of being able to calculate the centralities across networks efficiently. One such example would be the ability to learn the parameters of DeSCent's deep network from a small network (such as either of the two bitcoin datasets) fast and efficiently to then have the ability to calculate the signed centrality scores in a larger network (such as Epinions or Slashdot) by just feeding their features through the deep network already optimized from the smaller dataset. Thus, these inductive experiments provide even more evidence to the usefulness of harnessing deep learning in

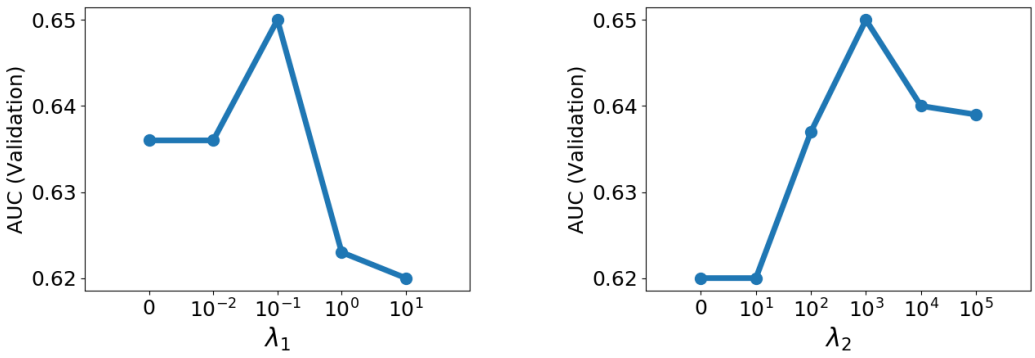
our DeSCent framework to discover signed centrality scores.

3.2.4.3 Parameter Analysis

Here we evaluate the contribution of the *sum constraint* (i.e., Eq. (3.22)) and the *status constraint* (i.e., Eq. (3.23)). We only show results on Bitcoin-Alpha as a representative dataset, since we have similar observations on other datasets.

The parameter λ_1 in this method is used to control the contribution of the *sum constraint*, which ensures our algorithm can avoid convergence towards the trivial zero solution. Here we investigate the change in performance as we vary the value of λ_1 (including setting it to zero, which would fully eliminate any contribution). Note that we keep all of DeSCent's other parameters fixed while we vary the value of λ_1. We present the results for a representative dataset, Bitcoin-Alpha, from the cross validation results in terms of AUC in Figure 3.5(a). We point out that while λ_1 is set to zero the performance is decreased, thus showing the *sum* constraint is able to aid in finding a higher performing solution.

Next we vary the parameter λ_2 while keeping the other parameters fixed. In this method λ_2 had been used for controlling the contribution of the *status constraint*, which was designed to ensure that the solution we find (in a global sense) still adheres to the sign suggested by status theory. Similarly as done for the *sum constraint* we report the AUC found while varying the value of λ_2 on the Bitcoin-Alpha dataset. From the results shown in Figure 3.5(b), we can see similar findings to those discovered for λ_1. More specifically, we notice the usage of the *status constraint* with $\lambda_2 = 1000$ is shown to be quite effective.



(a) *Sum constraint* (i.e., λ_1).

(b) *Status constraint* (i.e., λ_2).

Figure 3.5: Analyzing the signed centrality additional constraints on Bitcoin-Alpha.

CHAPTER 4

MODELING NETWORKS WITH NEGATIVE LINKS

In this chapter[1,2,3], we investigate modeling of networks having negative links. More specifically, we first focus on constructing a generative network model for signed networks including an automated parameter learning framework that we empirically evaluate compared with existing mechanistic signed network models and other baseline models. Thereafter, we study how to extend and model balance theory in signed bipartite networks, followed by an empirical analysis verifying its applicability towards being harnessed for sign link prediction.

Generative network modeling aims to design a model to represent a complex network through a few relatively simple set of equations and/or procedures such that, when provided a network as input, the model can learn a set of parameters to construct another network that is as similar to the input as possible. Ideally this would result in many observable/measurable properties being maintained from the input to the generated output network. In unsigned networks, the typical modeled properties are the power law degree distribution [97, 98, 99, 11], assortativity [100, 101], clustering coefficients [102, 103, 104, 105], and small diameter [98, 103]. Nowadays, more data can be represented as large networks in many real-world applications such as the Web [28, 106], biology [107, 108], and social media [109, 110]. Increasing attention has been attracted in better understanding and modeling networks. Traditionally network modeling has focused on unsigned networks. However, many networks can have positive and negative links (or signed networks [30, 29]), especially in online social media, which then raises the question – whether dedicated efforts

[1]Tyler Derr, Charu Aggarwal, and Jiliang Tang. "Signed Network Modeling Based on Structural Balance Theory." In Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM). 2018.

[2]Tyler Derr, Cassidy Johnson, Yi Chang, and Jiliang Tang. "Balance in Signed Bipartite Networks." In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM). 2019.

[3]Tyler Derr and Jiliang Tang. "Congressional Vote Analysis Using Signed Networks." In Proceedings of the 18th International Conference on Data Mining Workshops (ICDMW). 2018.

are needed to model signed networks in addition to the unsigned techniques. Thus, in Section 4.1 we present our proposed generative signed network model that focuses on maintaining core network properties including the degree distribution and local clustering and in addition those specific to signed networks, namely link sign ratio and signed triangle distributions to ensure they maintain the correct level of balance.

Although we have primarily focused on the development of signed networks, which are a specific type of network that has become increasingly ubiquitous, there are in fact various variants of signed networks. However, previous work and theories for signed networks have primiarly focused on unipartite signed networks, which are networks that have a single node type and signed links are able to connect any two nodes in the network. However, a common form of signed networks that have primarily been overlooked – signed bipartite networks. These networks have two sets of nodes and links are only able to be formed between nodes of different types. Actually, signed bipartite networks appear across multiple domains. For example, in e-commerce, a signed bipartite network can be constructed between buyers and sellers in multi-vendor marketplaces when the users are asked to rate the other after each transaction and helpfulness ratings from users to reviews can be naturally denoted as a signed bipartite network. Signed bipartite networks on the one hand, are commonly found, but have primarily been overlooked. Their complexities of having two node types where signed links can only form across the two sets introduce challenges that prevent most existing literature on unipartite signed and unsigned bipartite networks from being applied. On the other hand, balance theory, a key signed social theory, has been generally defined for cycles of any length and is being used in the form of triangles for numerous unipartite signed network tasks. However, in bipartite networks there are no triangles and furthermore there exist two types of nodes. Therefore, in Section 4.2, we conduct the first comprehensive analysis and validation of balance theory using the smallest cycle in signed bipartite networks - signed butterflies (i.e., cycles of length 4 containing the two node types). Then, to investigate the applicability of balance theory aiding signed bipartite network tasks, we develop multiple sign prediction methods that utilize balance theory in the form of signed butterflies.

4.1 Generative Modeling of Signed Networks

Signed networks are unique from unsigned not only due to the increased complexity added to the network by having a sign associated with every edge, but also (and more importantly) because there are specific principles (or social theories), such as balance theory, that play a key role driving the dynamics and construction of signed networks [2, 4]. For example, in unsigned networks we have the property of transitivity and we see a large amount of local clustering (i.e., formation of triangles). In comparison, with signed networks, not only are their patterns in the network driving local clustering, but also in the distribution of triangles (based on their edge signs) found in the network. Suggested by balance theory [29], some triangles are more likely to be formed (i.e., balanced) than others (i.e., unbalanced) in signed networks. Hence, modeling signed networks requires to preserve not only unique properties of signed networks such as the sign distribution, but also other properties suggested by their principles such as the distribution of formed triangles. However, these mechanisms are not incorporated into unsigned network modeling and unsigned network models are unequipped for signed networks. Thus, there is a need to design network models for signed networks.

Network models have many direct applications and a diverse set of benefits beyond and including the better understanding of the network structure and dynamics. Currently there is a significant push for better anonymization in social media. However, for researchers wanting to further advance their field, it is necessary to utilize the network data for knowledge discovery, mining, and furthermore for testing and benchmarking their methods and algorithms. A generative network model could be utilized for constructing synthetic networks having similar properties as their corresponding real network, but without compromising the user's privacy and allowing further advancements through the use of the synthetic network datasets. Similarly such a model can be used as a null-model for network property significance testing or for constructing synthetic networks of varying network properties to further understand the relationship between the network model and real world networks in terms of their dynamics and construction process. Thus, we propose a novel signed network model, which targets to preserve three key properties of signed networks – (1) degree distribution;

(2) sign distribution and (3) balance/unbalanced triangle distribution suggested by balance theory.

4.1.1 Problem Statement

A signed network $\mathcal{G}$ is composed of a set $V = \{v_1, v_2, \ldots, v_N\}$ of N vertices, a set of M^+ positive links $\mathcal{E}^+$ and a set of M^- negative links $\mathcal{E}^-$. Let $\mathcal{E} = \mathcal{E}^+ \cup \mathcal{E}^-$ represent the set of $M = M^+ + M^-$ edges in the signed network when not considering the sign. Here we focus on undirected and unweighted signed networks and leave modeling directed and weighted signed networks as one future work. We note that unlike defined in Section 2.1 here we use $\mathcal{V}$ to denote the set of vertices in the signed graph, as compared to $\mathcal{U}$ representing the users in the signed network to follow more traditional notations in unsigned generative graph models.

We can formally define the generative signed network modeling problem as follows:
Given a signed network $\mathcal{G}_I = (V_I, \mathcal{E}_I^+, \mathcal{E}_I^-)$ as input, we seek to learn a set of parameters Θ for a given model $\mathcal{M}$ that can retain the network properties found in $\mathcal{G}_I$, such that we can construct synthetic output networks $\mathcal{G}_O = (V_O, \mathcal{E}_O^+, \mathcal{E}_O^-)$, using $\mathcal{M}$ based on Θ, that closely resemble the input network in terms of measured network properties.

Traditionally network modeling has focused on unsigned networks and preserving unsigned network properties. Signed networks have distinct properties from unsigned networks [67]. For example, negative links are available in signed networks and ignoring the negative links can result in over-estimation of the impact of positive links [111]; and most of triangles in signed networks satisfy balance theory [28]. However, these properties cannot be simply captured by unsigned network models. Hence, dedicated efforts are demanded to model signed networks. The notations we will use in defining our proposed signed generative network model are demonstrated in Table 6.4.

4.1.2 An Overview of Balanced Signed Chung-Lu (BSCL) Model

A previous study demonstrated that the node degrees of signed networks also follow power law distributions [67] similar to that of unsigned networks. Hence, we propose to build the signed network model based on the unsigned Chung-Lu model, which can preserve the degree distributions

Table 4.1: Notations regarding signed network generative modeling.

Notations	Descriptions
e_{ij}	undirected edge between vertices v_i and v_j
N_i	set of neighbors for node i
$\mathbf{d}$	degree vector based on $\mathcal{E}_I$ where d_i is the degree of v_i
η	fraction of links being positive in $\mathcal{G}_I$ (i.e., M_I^+/M_I)
Δ_B	fraction of triangles balanced in $\mathcal{G}_I$
$\boldsymbol{\pi}$	sampling vector from degree distribution in $\mathcal{E}_I$
ρ	probability new edge closes a wedge to be a triangle in $\mathcal{G}_o$ via two-hop walk instead of a randomly inserting an edge
α	probability a randomly inserted edge into $\mathcal{G}_o$ is positive
β	probability of closing a wedge to have more balanced triangles in $\mathcal{G}_o$
$\Delta_{ij(B)}^{random}$	approximation to the $\mathbb{E}$[# of (balanced) triangles] that will get created due to randomly inserting e_{ij}
$\Delta_{\mathcal{G}(B)}^{random}$	average of $\Delta_{ij(B)}^{random}$ across all possible edges
$\Delta_{ij(B)}^{triangle}$	approximation to the $\mathbb{E}$[# (balanced) triangles] created when inserting edge e_{ij} via the wedge closing procedure
$\Delta_{\mathcal{G}(B)}^{triangle}$	average of $\Delta_{ij(B)}^{triangle}$ across all possible edges

of the input work. The Chung-Lu (CL) model first takes an unsigned network $\mathcal{G}_I = (V_I, \mathcal{E}_I)$ as input and independently decides whether each of the N^2 edges are placed in the generated network with each edge e_{ij} having probability $\frac{d_i d_j}{2M}$ where d_i is the degree of node v_i and M is the number of edges in the network. It can be shown that the expected degree distribution of the output network $\mathcal{G}_o$ is equivalent to that of $\mathcal{G}_I$. A fast variant of the Chung-Lu model, FCL [112], is proposed to create a vector $\boldsymbol{\pi}$ which consists of $2M$ values, where for each edge both incident vertices are added to the vector. Rather than deciding whether each of the N^2 edges get added to the network (as done in CL), FCL can just randomly sample two vertices from $\boldsymbol{\pi}$ uniformly, since this simulates the degree distribution. Note that FCL ignores self-loops and multi-edges when sampling M edges. However, since most real-world unsigned networks have higher clustering coefficients than those generated by CL and FCL, another CL variant Transitive Chung-Lu (TCL) was introduced in [102] to maintain the transitivity. Rather than always picking two vertices from $\boldsymbol{\pi}$, instead, TCL occasionally picks a single vertex from $\boldsymbol{\pi}$ and then, with a parameter ρ, performs a two-hop walk to select the second vertex. When including this edge, the process is explicitly constructing at least one triangle by

closing off the wedge (i.e., wedge closing procedure) created by the two-hop walk.

The proposed Balanced Signed Chung-Lu model (BSCL) is based on the TCL model, which automatically allows the mechanism for maintaining the degree distribution and also the local clustering coefficient during the construction process. However, as previously mentioned, the distribution of formed triangles is a key property in signed networks and most of these triangles adhere to balance theory. Note that, when performing the wedge closure procedure, we are not only closing the single wedge we explicitly constructed (through our two-hop walk), but there could be other common neighbors between these two vertices. Thus, we introduce a parameter β, which denotes the probability of assigning the edge sign to ensure the majority of the triangles being created by this new edge are balanced. With the introduction of this parameter, our model is able to capture a range of balance in signed networks. This is necessary since not all signed networks are completely balanced, and in fact real-world networks can have a varied percentage of triangles being balanced [4].

Meanwhile, we also want to maintain sign distribution. However, the above process of determining the edge sign for wedge closure is based on balance theory (i.e., local sign perspective) and not on the global sign perspective (i.e., η). This implies that when randomly inserting an edge into the network if we simply choose the sign based on η, then this could lead to our generated networks deviating from the true sign distribution of the input network. Therefore, we introduce α, which is a corrected probability (instead of using η) for a randomly inserted link and is used to correct the bias of positive or negative edges that will be created through the use of β which is from the local sign perspective.

With the introduction of three parameters (i.e., ρ, α and β), the proposed balanced signed Chung-Lu model (BSCL) is shown in Algorithm 4.1. Here we step through the high level processes of BSCL before later discussing both the network generation process and the parameter learning algorithms. On line 1 of Algorithm 4.1, we first construct $\mathcal{E}$. Then, using the degree distribution of $\mathcal{E}$, we can construct the vertex sampling vector $\boldsymbol{\pi}$ as shown on line 2. Next we calculate the properties of the input network we aim to preserve. These include the percentage of positive links

Algorithm 4.1: Balanced Signed Chung-Lu (BSCL) Model

Input: Signed Network $\mathcal{G}_I = (V_I, \mathcal{E}_I^+, \mathcal{E}_I^-)$
Output: Synthetic Signed Network $\mathcal{G}_o = (V_I, \mathcal{E}_o^+, \mathcal{E}_o^-)$
1 $\mathcal{E}_I = \mathcal{E}_I^+ \cup \mathcal{E}_I^-$
2 $\boldsymbol{\pi} \leftarrow$ Sampling vector based on the degree distribution in $\mathcal{E}_I$
3 $\eta \leftarrow \frac{M_I^+}{M_I^+ + M_I^-}$
4 $\mathbf{d} \leftarrow$ Calculate_Degree_Vector$(V_I, \mathcal{E})$
5 $\Delta_B \leftarrow$ Percentage balanced triangles in $\mathcal{G}_I$
6 $\rho, \alpha, \beta \leftarrow$ Parameter_Learning$(\mathcal{E}_I, \eta, \Delta_B, \mathbf{d}, M)$
7 $\mathcal{E}_o^+, \mathcal{E}_o^- \leftarrow$ Network_Generation$(\eta, M, \boldsymbol{\pi}, \rho, \alpha, \beta)$

η, vector of degrees $\mathbf{d}$, and percentage of balanced triads Δ_B from lines 3 to 5. With these values, we will estimate the major parameters of BSCL including ρ, α and β as mentioned on line 6 using our learning algorithms that will be discussed in subsection 4.1.4. Finally, we generate the network based on the learnt parameters on line 7 and then output the constructed synthetic signed network $\mathcal{G}_o$. In the next subsection, we will discuss the details of the network generation process performed by BSCL and then discuss how these parameters can be automatically and efficiently learned.

4.1.3 Network Generation for BSCL

Given the parameter values for ρ, α and β, we show in Algorithm 4.2 how BSCL can generate a synthetic signed network maintaining the key signed network properties. First, on line 1, we use the FCL method for the construction of a set $\mathcal{E}_o$ of M edges, which adheres to the original degree distribution. Then, on line 2, we split the unsigned edges into two sets, $\mathcal{E}_o^+$ and $\mathcal{E}_o^-$, by randomly assigning edge signs, based on η, such that the percentage of positive links matches that of the input network. Next, from lines 3 to 17, we add M new edges to the network, one at a time, while removing the oldest edge in the network for each new edge inserted. The reason for starting with this initial set of edges from FCL on line 1 is due to the fact when performing the wedge closing procedure (from lines 5 to 10), if the starting network is initially too sparse, there will not be many opportunities for two-hop walks to create triangles. We note that after each iteration from lines 3 to 17, $\mathbf{G}_o$ maintains the correct total number of edges, M.

Algorithm 4.2: BSCL_Network_Generation($\eta, M, \pi, \rho, \alpha, \beta$).

```
1  E_o = FCL(M,π)
2  E_o^+, E_o^- ← Randomly partition E_o based on η
3  for 1 to M do
4      v_i = sample from π
5      if wedge_closing_procedure(ρ) then
6          v_j = Perform two-hop walk from v_i through neighbor v_k
7          if close_for_balance(β) then
8              Add e_ij to E_o^+ or E_o^- based on the sign that closes the wedge and other common
                neighbors to have more balanced triangles
9          else
10             Add e_ij to E_o^+ or E_o^- to have more unbalanced triangles
11     else insert a random edge
12         v_j = sample from π
13         if create_positive_edge(α) then
14             E_o^+ ← E_o^+ ∪ {e_ij}
15         else
16             E_o^- ← E_o^- ∪ {e_ij}
17     Remove oldest edge from {E_o^+ ∪ E_o^-} respectively
18 return E_o^+, E_o^-
```

We will either insert an edge by closing a wedge into a triangle and using β to help maintain the balance, or insert a random edge and select its sign based on α to correctly maintain the sign distribution. On line 5, we use our parameter ρ to determine which edge insertion method we will use. Next we will further discuss these two edge insertion procedures.

The wedge closing procedure is selected with probability ρ on line 5, but starts on line 4 with the selection of v_i uniformly at random from π. Then, on line 6, we perform a two-hop walk from v_i through a neighbor v_k to land on v_j. We have just selected the wedge consisting of edges e_{ik} and e_{kj} to close into a triangle. We note that although we are explicitly constructing the triangle composed of vertices v_i, v_k and v_j that edge e_{ij} would also implicitly be closing wedges to form triangles with any other common neighbors that v_i and v_j might have. Hence, we use our learned parameter β for determining if we should introduce more balanced or unbalanced triangles into the network based on the total balance in the input signed network (i.e., Δ_B). Therefore, on line 7, with probability β, we choose to select the edge sign of e_{ij} such that the majority of the triangles

being created (both those explicitly through the two-hop walk and implicitly through other common neighbors) will adhere to balance theory. As mentioned on line 8, depending on whether balance theory would suggest e_{ij} to being positive or negative, we will add the edge to the set $\mathcal{E}_o^+$ or $\mathcal{E}_o^-$, respectively. Similarly, with probability (1-β), the sign of e_{ij} will be selected to introduce more unbalanced triangles into the generated network.

If not performing the wedge closing procedure, BSCL will instead insert a random edge with probability $(1 - \rho)$. This process starts similarly as line 4 by selecting the first vertex v_i. Then, on line 12, a second vertex is sampled from $\boldsymbol{\pi}$ such that we can then insert edge e_{ij} into the network. However, since we desire our generated network to maintain the correct sign distribution, the sign for the edge e_{ij} needs to carefully be determined. As previously discussed, the wedge closing procedure will disrupt the global sign distribution and therefore rather than using η for the sign selection, we use our learned parameter α. We again note that α will be learnt such that it incorporates the bias induced from the local sign selections made during the wedge closing procedure controlled by β. Therefore, with probability α, on line 13, we choose to go to line 14 and insert e_{ij} as a positive link and add it to the set $\mathcal{E}_o^+$. On the other hand, with probability $(1 - \alpha)$, we go to line 16 and select e_{ij} to be negative and therefore add it to the set of negative edges $\mathcal{E}_o^-$.

After edge insertion, the next step is to remove the oldest edge in the generated network $\mathcal{G}_o$ such that it maintains M edges. Line 17 shows that we select the oldest edge from the union of the positive and negative edge sets (i.e., $\mathcal{E}_o^+ \cup \mathcal{E}_o^-$) and then respectively remove it from the edge set it was selected from. After performing this loop from lines 3 to 17 M times, all the initial edges from FCL will have been removed and the network generator can return the resulting positive and negative edge sets $\mathcal{E}_o^+$ and $\mathcal{E}_o^-$, respectively.

One step we did not mention in Algorithm 4.2 for ease of description is that we also make use of a queue for when having collisions (i.e, selecting to insert an edge that already exists in the network or a self-loop). For every time we have such a collision, the vertices are added to the queue. Then, before each time selecting an edge from $\boldsymbol{\pi}$ (i.e. on lines 4 and 12), the queue is checked. If the queue is empty, then we proceed to sample from $\boldsymbol{\pi}$. However, if the queue is non-empty, then we

instead take from the front of the queue. Similarly, we utilize the queue if unable to perform a two hop walk from vertex v_i. Next we will discuss how we can learn the parameters ρ, α, and β.

4.1.4 Parameter Learning for BSCL

In the last subsections, we have introduced the BSCL model and network generation process based on the parameters ρ, α, and β, here we discuss how to learn these parameters from the input signed network. We notice that these parameters are related to each other. For example, when constructing triangles to be balanced or unbalanced (based on β), this will disrupt the global sign distribution since these decisions are only based on the local sign perspective. Similarly, when inserting a random edge with a sign based on α, this has the potential to disrupt the distribution of triangles and the percentage of triangles that are balanced in the network. This is because the decision for the sign of a random edge insertion is based solely on the global sign perspective and ignores the local perspective of whether triangles are being created via this inserted edge to be balanced or unbalanced. Hence, next we discuss the proposed algorithm for learning these parameters alternatively and iteratively.

4.1.4.1 Learning ρ

For the parameter ρ, we make use of the Expectation-Maximization (EM) learning method following a similar process in the TCL model [102]. The general idea is that it can be learned after defining a hidden variable associated with each edge, which determine whether the edge was added to the network randomly or through a wedge being closed into a triangle. More specifically, let $z_{ij} \in Z$ be the latent variable assigned to each edge e_{ij}. These latent variables can be equal to 1 or 0, where $z_{ij} = 0$ indicates that the edge was created via random sampling from $\boldsymbol{\pi}$ and $z_{ij} = 1$ suggests that the edge e_{ij} was created through the two-hop walk wedge closing procedure.

Let $\boldsymbol{\pi}_i$ represent the probability of selecting v_i from the sampling vector $\boldsymbol{\pi}_i$, $\mathbb{I}[v_j \in N_k]$ as an indicator function to be 1 if v_j is in the neighbor set of v_k and 0 otherwise, and ρ^t denote the value of ρ at iteration t during the EM process. Next we analyze the two procedures of wedge

closing or random insertion given a starting node (i.e., first selected node) v_i. We can calculate the probabilities based on the following: (1) for the random insertions with probability $(1-\rho)$ and selecting v_j as the second node with probability π_i; (2) the wedge closing with probability ρ and the probability we were able to perform a two-hop walk to v_j is based on first having v_k that is a mutual neighbor of v_i and v_j (i.e., $v_k \in N_i$ and $v_k \in N_j$) and then the walk continues to v_j (i.e., selecting v_j from the d_k neighbors of v_k) once arriving at the mutual neighbor v_k. Therefore, we can formulate the conditional probabilities for placing the edge e_{ij} given ρ, the starting node v_i and the method for either the random insertion or wedge closing procedure (that is represented with z_{ij}) are as follows, respectively:

$$P(e_{ij}|z_{ij}=0, v_i, \rho^t) = (1-\rho^t)(\pi_i)$$
$$P(e_{ij}|z_{ij}=1, v_i, \rho^t) = \rho^t \sum_{v_k \in N_i} \Big(\frac{\mathbb{I}[v_j \in N_k]}{d_i}\Big)\Big(\frac{1}{d_k}\Big)$$

For the calculation of the expectation of z_{ij} given ρ^t, e_ij, and the starting node v_i, the conditional probability of z_{ij} can be defined using the Bayes' Rule as follows:

$$P(z_{ij}=1|e_{ij}, v_i, \rho^t) = \frac{P(e_{ij}|z_{ij}=1, v_i, \rho^t)}{P(e_{ij}|z_{ij}=1, v_i, \rho^t) + P(e_{ij}|z_{ij}=0, v_i, \rho^t)} \tag{4.1}$$

which calculates the probability of z_{ij} being 1 based on the probability of the edge being created by wedge closure over the probability the edge e_{ij} is expected to get created. This leads to the expectation of z_{ij} to $\mathbb{E}[z_{ij}|\rho^t] = P(z_{ij}=1|e_{ij}, v_i, \rho^t)$. Furthermore, the maximization for the expectation can be calculated via sampling a set of edges $\mathbb{S}$ uniformly from $\mathcal{E}$. Then, due to the fact z_{ij} is conditionally independent, we can individually calculate the expectation of z_{ij} for each edge in $\mathbb{S}$ and then take the average across the set of edges sampled as:

$$p^{t+1} = \frac{1}{\mathbb{S}} \sum_{e_{ij} \in \mathbb{S}} \mathbb{E}[z_{ij}|\rho^t] \tag{4.2}$$

4.1.4.2 Learning β

Note that we have calculated Δ_B from the input network that denotes the percentage of triangles that were adhering to balance theory. We seek to approximate the expected number of triangles

BSCL will construct through the wedge closure and the random edge insertion methods on average for each edge added to the network. Let us denote the values we calculate for these two methods as $\Delta_{\mathcal{G}}^{triangle}$ and $\Delta_{\mathcal{G}}^{random}$, respectively, which will be calculated with respect to both ρ and α. Furthermore, we will calculate what percent of these we expect to be balanced as $\Delta_{\mathcal{G}B}^{triangle}$ and $\Delta_{\mathcal{G}B}^{triangle}$. Details of estimating $\Delta_{\mathcal{G}}^{triangle}$, $\Delta_{\mathcal{G}}^{random}$, and $\Delta_{\mathcal{G}B}^{random}$ will be discussed later. To correctly maintain the percentage of triangles being balanced in the synthetic network, we desire the following:

$$\Delta_B = \frac{\Delta_{\mathcal{G}B}^{triangle} + \Delta_{\mathcal{G}B}^{random}}{\Delta_{\mathcal{G}}^{triangle} + \Delta_{\mathcal{G}}^{random}}$$

which simply states the combined balanced percentage from the two methods should be the balanced percentage of the input network. Then, we can calculate the above mentioned values and if we let $\Delta_{\mathcal{G}B}^{triangle} = \beta\Delta_{\mathcal{G}}^{triangle}$, which denotes that β percent of the triangles we close via the wedge closing procedure are balanced, then we can solve β and obtain the below:

$$\beta = \frac{\Delta_B\left(\Delta_{\mathcal{G}}^{triangle} + \Delta_{\mathcal{G}}^{random}\right) - \Delta_{\mathcal{G}B}^{random}}{\Delta_{\mathcal{G}}^{triangle}} \tag{4.3}$$

Next, we discuss how to estimate $\Delta_{\mathcal{G}}^{random}$, $\Delta_{\mathcal{G}B}^{random}$ and $\Delta_{\mathcal{G}}^{triangle}$.

Estimating $\Delta_{\mathcal{G}}^{\mathbf{random}}$: We note that the starting set of edges are constructed with the FCL method and edge signs randomly assigned to them. Furthermore, each edge will have been added into the network with probability $p_{ij} = \frac{d_i d_j}{2M}$. We note that the expected number of common neighbors between two vertices v_i and v_j would be equivalent to the number of triangles that get created if the edge e_{ij} was inserted into the network where we denote this number of triangles to be Δ_{ij}^{random}.

To obtain the number of common neighbors for v_i and v_j, we calculate the probability that $v_l \in V_I \backslash \{v_i, v_j\}$ is a common neighbor based on the probability there exists an edge from v_l to both v_i and v_j. Note that after having the probability of the existence for the first edge e_{il}, we must subtract 1 from d_l, since we have already conditioned on the existence of the first edge e_{il}, thus

causing v_l to have one less opportunity to connect to v_j. We formulate this idea as the following:

$$\begin{aligned}\Delta_{ij}^{random} &= \sum_{v_l \in V \setminus \{v_i, v_j\}} \Big(\frac{d_i d_l}{2M}\Big)\Big(\frac{d_j(d_l-1)}{2M}\Big) \\ &= \Big(\frac{d_i d_j}{2M}\Big) \sum_{v_l \in V \setminus \{v_i, v_j\}} \Big(\frac{d_l(d_l-1)}{2M}\Big)\end{aligned}$$

Next we present the average value of Δ_{ij}^{random} across all possible unordered pairs of vertices as follows:

$$\Delta_{\mathcal{G}}^{random} = \frac{1}{\frac{1}{2}N(N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \Delta_{ij}^{random}$$

where $\Delta_{\mathcal{G}}^{random}$ is used to denote the average triangles constructed by a randomly inserted edge in the model. We note that the above would require $O(N^3)$ time to compute, but using the fact that $2M = N * avg(d)$, we can use the following approximation if we treat the summation such that it includes v_i and v_j instead of excluding them. We use $avg(d)$ to denotes the average degree and $avg(d^2)$ represents the average value of squared degrees. Then we have:

$$\begin{aligned}\sum_{v_l \in V \setminus \{v_i, v_j\}} \Big(\frac{d_l(d_l-1)}{2M}\Big) &\approx \sum_{v_l \in V} \Big(\frac{d_l(d_l-1)}{2M}\Big) \\ &= \frac{(d_1^2 + d_2^2 + \cdots + d_N^2) - (d_1 + \cdots + d_N)}{2M} \\ &= \frac{\big(avg(d^2)N\big) - \big(avg(d)N\big)}{N * avg(d)} \\ &= \Big(\frac{avg(d^2) - avg(d)}{avg(d)}\Big) \qquad (4.4)\end{aligned}$$

We therefore can rewrite $\Delta_{\mathcal{G}}^{random}$ as follows:

$$\Delta_{\mathcal{G}}^{random} \approx \frac{avg(d^2) - avg(d)}{avg(d)MN(N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d_i d_j$$

First we note that we only need to compute $avg(d)$ and $avg(d^2)$ once (which can be performed in $O(N)$ time). Second, for $\sum_{i=1}^{N-1} d_i \sum_{j=i+1}^{N} d_j$, rather than iterating over the nested sum of $j = i+1$ to N, we can instead use dynamic programming to construct a vector $\mathbf{s}$ where s_i represents

$\sum_{j=i+1}^{N} d_j$. We can construct this vector $\mathbf{s}$ starting with $s_i = d_N$ when $i = N-1$ and then recursively filling in the vector using $s_{i-1} = s_i + d_i$, which can be performed in linear time in relation to the number of vertices N. Therefore the below approximation for $\Delta_{\mathcal{G}}^{random}$ can be performed in $O(N)$ time instead of $O(N^3)$.

$$\Delta_{\mathcal{G}}^{random} \approx \frac{avg(d^2) - avg(d)}{avg(d)MN(N-1)} \sum_{i=1}^{N-1} d_i s_i \tag{4.5}$$

Estimating $\Delta_{\mathcal{G}B}^{random}$: We further analyze beyond our calculation of $\Delta_{\mathcal{G}}^{random}$ by examining the wedges (i.e., common neighbors) to be one of the following formations: ($\{+,+\}$, $\{+,-\}$, $\{-,+\}$, $\{-,-\}$), where $\{+,-\}$ is used to represent the wedge formed by edges e_{il} and e_{lj} and their signs are positive and negative, respectively. Note that we first initialize our model with edges from FCL and select edge signs to perfectly match the sign distribution of the original network, and then we attempt to correctly maintain this distribution with the parameter α; hence we assume that all wedges were created by the original sign distribution (where η is the probability of a link being positive). Below, we use $\Delta_{ij}^{random+-}$ to represent the number of wedges that would be closed into triangles when adding the edge e_{ij} and were formed with a wedge of type $\{+,-\}$. The definitions for all the wedge types are:

$$\begin{aligned}
\Delta_{ij}^{random++} &= \eta\eta\Delta_{ij}^{random} \\
\Delta_{ij}^{random+-} &= \Delta_{ij}^{random-+} = \eta(1-\eta)\Delta_{ij}^{random} \\
\Delta_{ij}^{random--} &= (1-\eta)(1-\eta)\Delta_{ij}^{random}
\end{aligned}$$

The expected number of balanced triangles that would be created if the edge e_{ij} is inserted randomly, Δ_{ijB}^{random}, can be obtained via the expected number of wedges of different types and the corrected positive link probability α as:

$$\begin{aligned}
\Delta_{ijB}^{random} &= \alpha\Delta_{ij}^{random++} + (1-\alpha)\Delta_{ij}^{random+-} \\
&\quad +(1-\alpha)\Delta_{ij}^{random-+} + \alpha\Delta_{ij}^{random--}
\end{aligned} \tag{4.6}$$

where for a wedge with two existing edges to close to a balanced triangle, the added third edge would need to have a sign such that there are an even number of negative links in the resulting

triangle, according to balance theory. This can also be extended for the calculation of $\Delta_{\mathcal{G}B}^{random}$ by averaging across all edges.

Estimating $\Delta_{\mathcal{G}}^{triangle}$: Similarly, we will calculate the expected total number of triangles and the balanced percentage when using the wedge closing procedure. The main idea for this wedge closure is that we are guaranteed to select vertices such that we have at least one triangle being created each time. We then need to also add the expected number of triangles that would be created randomly by other common neighbors of v_i and v_j (similar to the random edge insertion case above). Note that we must however discount the degree of v_i and v_j by 1, since in this method, we have already explicitly used one of the links coming from both to discover this one common neighbor for the wedge closure edge insertion. Let us denote the selected common neighbor as v_c, which forms a wedge with the edges e_{ic} and e_{cj}.

$$\begin{aligned}\Delta_{ij}^{triangle} &= 1 + \sum_{v_l \in V \setminus \{v_i, v_j, v_c\}} \Big(\frac{(d_i - 1)d_l}{2M}\Big)\Big(\frac{(d_j - 1)(d_l - 1)}{2M}\Big) \\ &= 1 + \Big(\frac{d_i d_j - d_i - d_j + 1}{2M}\Big) \sum_{v_l \in V \setminus \{v_i, v_j, v_c\}} \Big(\frac{d_l(d_l - 1)}{2M}\Big)\end{aligned}$$

Similarly, as the approximation in Eq. (4.4), we can simplify the formulation of $\Delta_{ij}^{triangle}$ as:

$$\Delta_{ij}^{triangle} \approx 1 + \Big(\frac{d_i d_j - d_i - d_j + 1}{2M}\Big)\Big(\frac{avg(d^2) - avg(d)}{avg(d)}\Big)$$

Then we can also calculate $\Delta_{\mathcal{G}}^{triangle}$ similar to $\Delta_{\mathcal{G}}^{random}$, which results in the following:

$$\Delta_{\mathcal{G}}^{triangle} \approx 1 + \frac{avg(d^2) - avg(d)}{avg(d)MN(N-1)} \sum_{i=1}^{N-1} \big((d_i - 1)(s_i - N + i)\big)$$

4.1.4.3 Learning α

Here we want to estimate the percentage of positive edges. Therefore, we will examine this percentage for both the wedge closing and the random edge insertion, which are denoted as $\eta^{triangle}$ and η^{random}, respectively. Then, if we have these two values, we can correctly maintain

the percentage of positive links in the synthetic network as the following:

$$\eta = \rho\eta^{triangle} + (1-\rho)\eta^{random} \tag{4.7}$$

The above is due to the fact that the wedge closing procedure will insert ρ percent of the links into the generated network while the random insertion method will construct $(1-\rho)$ percent.

For the wedge closure, we examine the probability of the four types of wedges: {+,+}, {+,-}, {-,+} and {-,-}. Then we define their probabilities of existing in the network to be P({+,+}), P({+,-}), P({-,+}) and P({-,-}). Next we note that the wedge {+,+} and {-,-} would result in a positive edge being created with probability β, while wedge {+,-} and {+,-} would only provide a positive edge with probability $(1-\beta)$. The reason for this is due to the fact that balance theory would be controlling the third edge sign and while the first two require a positive link to adhere to balance theory; while it would be when we construct more unbalanced triangles that the later two wedge types would result in a positive link insertion. Therefore, we denote the probability of inserting a positive edge with the wedge closure to be the following:

$$\begin{aligned}\eta^{triangle} = \rho\Big(&\beta\big(P(\{+,+\}) + P(\{-,-\})\big) \\ &(1-\beta)\big(P(\{+,-\}) + P(\{-,+\})\big)\Big)\end{aligned} \tag{4.8}$$

We note that ρ is the probability of performing the wedge closing procedure. If we assume that α and β are correctly solved, then the expected probability for the wedges is based on η and $(1-\eta)$. We can therefore rewrite Eq. 4.8 as:

$$\begin{aligned}\eta^{triangle} = \rho\Big(&\beta\big((\eta\eta) + (1-\eta)(1-\eta)\big) \\ &(1-\beta)\big(\eta(1-\eta) + (1-\eta)\eta\big)\Big)\end{aligned} \tag{4.9}$$

Next, we show how to calculate η^{random}. We notice that α will be the percentage of links we construct to be positive and this process will happen $(1-\rho)$ percent of the time. Thus, $\eta^{random} = (1-\rho)\alpha$. We can now substitute $\eta^{triangle}$ and η^{random} into Eq. 4.7 and solve for α as

follows:

$$\alpha = \frac{1}{(1-\rho)}\Bigg(\eta - \Big(\rho\beta\big((\eta\eta) + (1-\eta)(1-\eta)\big) + (1-\beta)\big(\eta(1-\eta) + (1-\eta)\eta\big)\Big)\Bigg) \tag{4.10}$$

4.1.5 Time Complexity of BSCL

We first discuss the running time of the learning algorithm and then the time needed for the network generation process. The preprocessing needed for the learning algorithm is to determine the probability of edges being positive, η, and the probability of triangles in the network being balanced and adhering to balance theory, Δ_B, of the original network. We can determine η trivially in $O(M)$. However, Δ_B can be reduced the complexity of triangle listing algorithms, which can be easily performed using classical methods in $O(min(M^{3/2}, Md_{max}))$, where d_{max} is the maximum degree in the network[113]. The learning process for the parameter ρ is $O(sI)$ where I is the number of iterations in the EM method and $s = |\mathbb{S}|$ is the number of edges sampled for each iteration. The running time of β and α can be determined as follows. We initially calculate the expected number of triangles added by each of the processes of BSCL, which takes $O(N)$ instead of $O(N^3)$ due to the approximation used and dynamic programming approach. Then the update equations (i.e., Eqs. (4.3) and (4.10)) can both be performed in $O(1)$ time. Thus when allowing for I' maximum iterations of the alternating update process between α and β (which empirically only takes a small constant number of iterations to converge), we have the overall learning time complexity for BSCL as $O(min(m^{3/2}, md_{max}) + N^2 + sI + I')$. The generation process of BSCL, is built upon the fact that the running time for TCL is shown to be $O(N+M)$ in [102]. The triangle closing process of determining the best sign selection based on the set of triangles being closed, is reduced to the complexity of common neighbors between two vertices, which is known to be $O(d_{max}^2)$. Thus the generation process of BSCL is $O(N + Md_{max}^2)$.

Table 4.2: Statistics of three signed social networks for generative modeling.

Network	N	(E, η)	Δ_B
Bitcoin-Alpha	3,784	(14,145 , 0.915)	0.862
Bitcoin-OTC	5,901	(21,522 , 0.867)	0.869
Epinions	131,580	(711,210 , 0.830)	0.892

4.1.6 Experiments

In this section, we conduct experiments to evaluate the effectiveness of the proposed signed network model. In particular, we try to answer two questions via our experiments - (1) can the proposed model, BSCL, effectively maintain signed network properties? and (2) is the parameter learning algorithm able to learn the appropriate parameter values from the input signed network?

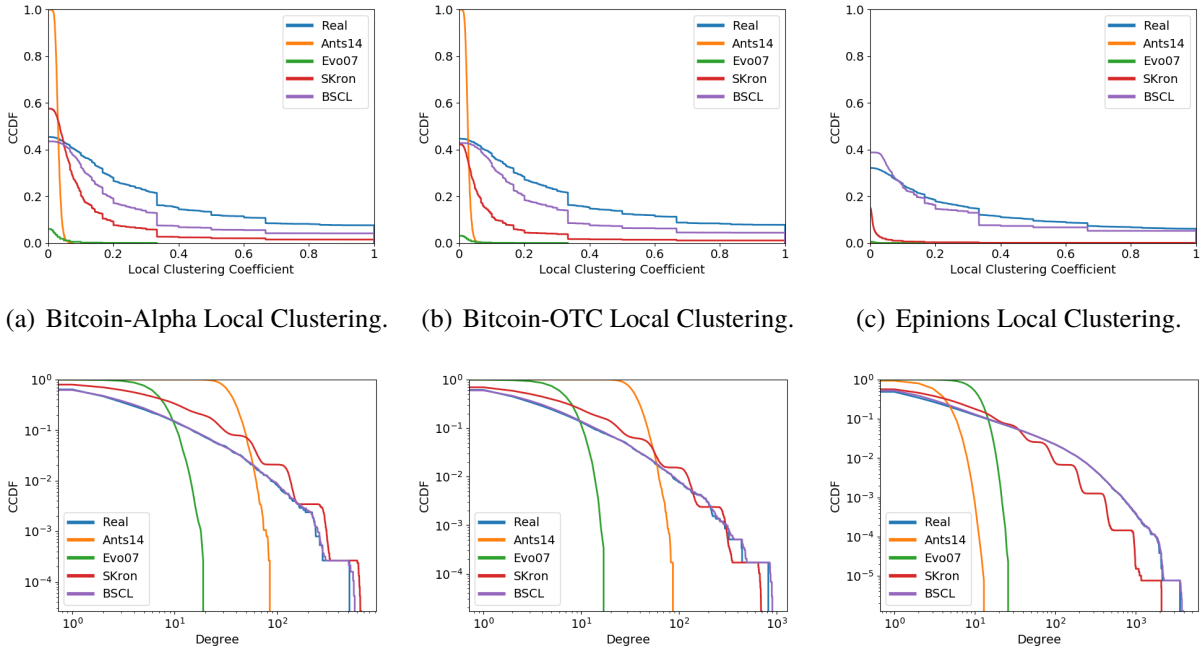
For our study of signed network modeling, we utilize three signed network datasets, i.e., Bitcoin-Alpha, Bitcoin-OTC, and Epinions. We provide more details of the datasets in Table 4.2. Note that, since we focus on undirected and unweighted signed networks, we ignore the directions of signed links in these datasets.

4.1.6.1 Network Generation Experiment

The first set of experiments are to compare the network properties of the resulting generated networks from our model and the baselines. These properties will be used as a metric to determine how well the models are able to capture the underlying dynamics of signed networks. More specifically, we will focus on the three key signed network properties - (1) degree distribution; (2) positive/negative link ratio and (3) proportion of balance/unbalanced triangles suggested by balance theory. Note that we also present the local clustering coefficient distribution and the triangle distribution (in relation to the edge signs in the triangles). Our results are the averaged results of 10 generated networks for each of the methods on each dataset.

The first group of two baselines are existing signed network models:

- Ants14: This method is an interaction-based model for signed networks based on using ants



(a) Bitcoin-Alpha Local Clustering. (b) Bitcoin-OTC Local Clustering. (c) Epinions Local Clustering.

(d) Bitcoin-Alpha Degree Dist. (e) Bitcoin-OTC Degree Dist. (f) Epinions Degree Dist.

Figure 4.1: Visualization of the degree distributions and local clustering coefficients.

to lay pheromone on edges [114].

- Evo07: This method is an evolutionary model for signed networks that had a "friendliness" index that controls the probability of positive or negative links and also a parameter that controls the maximum amount of unbalance [115].

Note that for Ants14, we perform a grid search on the parameter space for its 6 parameters according to the values reported in [114]. Similarly, for Evo07, a grid search was performed for the two parameters.

The next two baselines are built upon two popular unsigned generative models. We first convert the network to unsigned by ignoring the links, run the baseline model, and then randomly assign signs to the edges such that the global sign distribution is maintained using η.

- STCL: This method is from the unsigned TCL model [102].

- SKron: This method is from the unsigned Kronecker Product model [99].

Table 4.3: Positive/negative link sign distribution.

Links Positive	Real	Ants14	Evo07	BSCL
Bitcoin-Alpha	0.915	0.741	0.917	0.912
Bitcoin-OTC	0.867	0.740	0.869	0.860
Epinions	0.830	0.930	0.830	0.808
Absolute Difference		0.401	0.004	0.031

Table 4.4: Proportion of triangles balanced in generated signed networks.

Percent Balanced	Real	STCL/SKron	Ants14	Evo07	BSCL
Bitcoin-Alpha	0.862	0.786	0.787	0.750	0.802
Bitcoin-OTC	0.869	0.698	0.817	0.677	0.752
Epinions	0.892	0.644	0.939	0.639	0.748
Absolute Difference		0.484	0.174	0.557	0.321

The results of the properties that are in common with unsigned networks (i.e., the degree distribution and the local clustering coefficient) can be can be seen in Figure 4.1. We see that BSCL and STCL both perform near identically on the degree distribution as they are both based on the Transitive Chung-Lu model and therefore can very closely maintain the degree distribution. However, it can be seen that the two signed network baselines, Ants14 and Evo07, perform very poorly and do not even appear to follow a power-law distribution. We mention that SKron is not able to exactly model the degree distribution, but does not perform as poorly as the two existing signed network baselines. For the two existing signed network models similar poor findings can be found for the local clustering coefficient. Our proposed model BSCL along with STCL perform the best. The SKron model has some clustering, but not near that of the original input network.

In Table 4.3, we show the positive/negative link ratio; while in Table 4.4, we present the proportion balance/unbalance triangles. We make a comparison to the two existing signed network methods, and also then follow with a comparison of BSCL to STCL and SKron which are the two modified unsigned network models.

In Bitcoin-Alpha dataset, our model BSCL is able to achieve the closest proportion of balance triangles. Then in the Bitcoin-OTC dataset, the Ants14 performs the best in terms of the proportion of balance in the network. We further show a fine-grained comparison by separating the four types of triads on the Bitcoin-Alpha, Bitcoin-OTC, and Epinions datasets in Table 4.5, Table 4.6, and

Table 4.5: Distribution of signed triangle types in the Bitcoin-Alpha dataset.

Triad Type	Real	STCL/SKron	Ants14	Evo07	BSCL
{+,+,+}	0.793	0.766	0.446	0.750	0.804
{+,+,-}	0.134	0.213	0.200	0.250	0.174
{+,-,-}	0.069	0.020	0.341	0.000	0.021
{-,-,-}	0.004	0.001	0.013	0.000	0.001
Absolute Difference		0.158	0.694	0.232	0.102

Table 4.6: Distribution of signed triangle types in the Bitcoin-OTC dataset.

Triad Type	Real	STCL/SKron	Ants14	Evo07	BSCL
{+,+,+}	0.724	0.652	0.445	0.613	0.707
{+,+,-}	0.122	0.300	0.170	0.323	0.246
{+,-,-}	0.145	0.046	0.372	0.064	0.045
{-,-,-}	0.009	0.002	0.013	0.000	0.002
Absolute Difference		0.356	0.558	0.401	0.248

Table 4.7: Distribution of signed triangle types in the Epinions dataset.

Triad Type	Real	STCL/SKron	Ants14	Evo07	BSCL
{+,+,+}	0.808	0.572	0.929	0.587	0.698
{+,+,-}	0.096	0.351	0.061	0.349	0.250
{+,-,-}	0.084	0.072	0.010	0.052	0.050
{-,-,-}	0.013	0.005	0.000	0.012	0.002
Absolute Difference		0.511	0.243	0.507	0.309

Table 4.7, respectively. We notice that Ants14 achieves this by drastically changing the distribution among the four triangle types. However, our model performs the best overall in terms of the triangle distribution. Similarly we notice a drastically low overall clustering in the Ants14 output networks as seen in the local clustering coefficient plot in Figure 4.1(b). Furthermore, although the Ants14 method more closely resembles the percentage of triangles being balanced across the three signed networks (having a smaller absolute difference than BSCL), we can observe that this comes at a tradeoff of having a very inconsistent percentage of links being positive in the network as compared to the input signed network, which can be seen in the absolute difference row of Table 4.3. Note that in both of the Bitcoin datasets, our model BSCL is able to achieve better performance than the baselines in terms of the triangle distributions, while only at the expense of sacrificing $< 1\%$ in terms of matching the correct positive link percentage, η, of the input networks. The results are

similar in the Epinions dataset.

When comparing BSCL with STCL and SKron we mention that by design these other two models will always have the exact percentage of positive links and their expected triangle distribution can be calculated as: $(\eta\eta\eta)$, $(3\times(\eta\eta(1-\eta)))$, $(3\times(\eta(1-\eta)(1-\eta)))$, and $((1-\eta)(1-\eta)(1-\eta))$ for the {+,+,+}, {+,+,-}, {+,-,-}, and {-,-,-} triangle types, respectively. We can observe that in terms of the absolute difference between the percentages in each of these triangle types, the BSCL method performs much better having an absolute difference of only 0.248 in the Bitcoin-OTC dataset while the straight-forward modification to the unsigned network models of maintaining the positive link percentage correctly results in an absolute difference of 0.356. Similarly, when looking at the percentage of triangles adhering to balance theory across the three signed network datasets BSCL has a value of only 0.321 while STCL and SKron share an absolute difference of 0.484.

Overall we can see that the Ants14 model favors capturing the percentage of triangles adhering to balance theory, but at the sacrifice of the triangle distribution (in regards to the triangle edge signs) and performing just as poorly at maintaining the positive/negative link ratio. On the other hand, the Evo07 model is able to correctly maintain the positive/negative link ratio, but struggles to maintain reasonable percentage of balanced triangles when examining across the three datasets. Our model BSCL overall outperforms the two previous signed network models as seen in the Tables and Figures. Furthermore, we can see BSCL out performs STCL and SKron in terms of maintaining the percentage of balanced triangles and triangle distribution. We note that BSCL finds this improvement in other signed network properties while only losing about 1% on average across each the three datasets in maintaining the positive/negative link ratio.

4.1.6.2 Parameter Learning Experiment

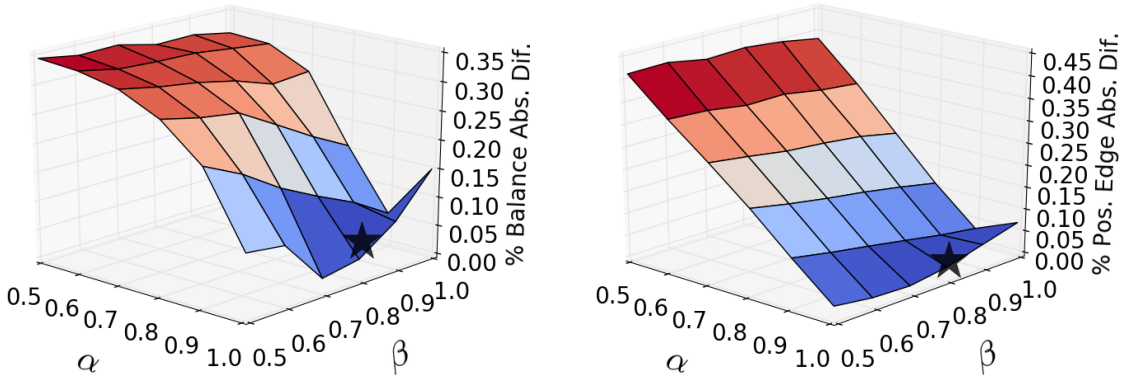
The second set of experiments are designed to test the learning algorithm we have proposed in determining appropriate parameters for BSCL. Here we first utilized natural and intuitive heuristics for setting the parameter values of α and β to further evaluate the effectiveness of our parameter learning algorithms. More specifically, the value of η (i.e., the real network's percentage of positive

Table 4.8: Absolute difference from the generated networks to the real signed networks averaged over the three datasets for each respective property.

	BSCL (learning α & β)	BSCL ($\alpha = \eta$ & $\beta = \Delta_B$)
Sign Distribution	0.031	0.040
Proportion Balanced Triangles	0.107	0.164
Distribution Triangle Types	0.220	0.351

links) is used as a natural choice for the value of α (i.e., the percentage of the time a randomly inserted edge is positive) when not utilizing the proposed learning algorithm described in Section 4.1.4 for BSCL. Similarly, the value Δ_B (i.e., the real network's percentage of triangles that are balanced) can be chosen as the value for β (i.e., the percentage of time we explicitly close a wedge into a balanced triangle) if we were to not use our proposed learning algorithm. Table 4.8 contains the performance comparison (in relation to the link sign distribution, proportion of triangles balanced, and distribution of triangle types) between the proposed parameter learning algorithms selected values against the above mentioned heuristically picked values for α and β. Note that these absolute difference values are averaged across the three real-world signed networks. We can observe that in all three properties our parameter learning algorithm significantly out performs the most natural heuristically picked values and thus providing further evidence our model can learn parameters to more accurately generate synthetic variants of the real input signed network.

For a more detailed analysis we also perform a grid search across a reasonable area of the parameter space for α and β to obtain optimal parameters. Then we compare the performance of the learnt parameters and the searched optimal parameters to demonstrate the ability of the proposed parameter learning algorithm. We only present the results in Figures 6.4 in terms of percentage of balanced triangles and positive/negative link ratio for the Bitcion Alpha dataset, since we have similar observations for Bitcion OTC and Epinions with other settings. Note that the z-axis is the absolute difference away from the true input networks value (where lower is better). The "stars"



(a) Bitcoin-Alpha % balanced triangles results. (b) Bitcoin-Alpha positive/negative ratio results.

Figure 4.2: BSCL parameter learning analysis.

in the figures are the coordinates along the x- and y-axis for the learned parameter. From the figures, it looks convincing that indeed our parameter learning algorithm is able to find appropriate parameters for the input network.

4.2 Balance in Signed Bipartite Networks

Signed bipartite networks have begun to appear more commonly especially online. For example, many online rating systems, such as Netflix and YouTube, adopt “thumbs-up" or “thumbs-down" rating that can also be formulated as signed bipartite networks. Another real-world example is from the political science domain, more specifically, we observe that indeed the United States Congress is inherently a signed bipartite network formed from the representatives and the bills they have voted on (where the "Yea" and "Nay" votes can be represented as positive and negative links, respectively) [116, 117], which we present in more detail in this specific application later in Section 6.2).

Although there have been works focused on unsigned bipartite networks, these methods are lacking the capability to handle the further complexities of negative links. Similarly, methods developed for unipartite signed networks might not be applicable when having the two node types or limiting the possible connections in the network. For example, a fundamental theory that explains the social phenomena of the link structure in signed network analysis is balance theory [29, 30]. It suggests that a cycle in signed networks with an even number of negative links

Table 4.9: Notations regarding signed bipartite networks.

Notations	Descriptions
$\mathbf{B}$	Undirected signed bidjacency matrix
$\mathbf{P}_B$	Adjacency matrix for the $\mathcal{U}_B$-projection network
$\mathbf{P}_S$	Adjacency matrix for the $\mathcal{U}_S$-projection network
$\mathbf{A}$	Adjacency matrix constructed from $\mathbf{B}, \mathbf{P}_B$, and $\mathbf{P}_S$
$\mathbf{U}$	Low-dimensional representation of nodes in $\mathcal{U}_B$
$\mathbf{V}$	Low-dimensional representation of nodes in $\mathcal{U}_S$

is balanced, which is typically stated as "a friend of my friend is my friend" while an "enemy of my friend is my enemy". In unipartite signed networks balance theory has been extensively applied on signed triangles (i.e., the smallest undirected cycle) across various real-world networks to obtain better performance across modeling [115, 114, 118], measuring [51, 119, 120, 65], and mining applications [23, 61, 53, 121]. However, in signed bipartite networks it is fundamentally impossible to have any triangles while having the two different types of nodes. Therefore it is important to understand balance theory in signed bipartite networks and its possibility to enhance applications, due to the prevalence of signed bipartite networks. Thus, dedicated efforts are desired for signed bipartite networks in additional to unipartite signed networks and unsigned bipartite networks. Here we present a comprehensive analysis and validation of balance theory using signed butterflies and then show their effectiveness on improving the performance of link sign prediction in signed bipartite networks. Furthermore, the empirical findings in sign prediction paves the way for improvements in other signed bipartite network analysis tasks.

4.2.1 Balance Theory in Signed Bipartite Networks

In this section, we will introduce the signed bipartite network datasets we have collected for this study. Thereafter we discuss balance theory from a general signed network perspective, then we validate its applicability in signed bipartite networks, and perform a preliminary analysis on our datasets; but first, we introduce the definitions and notations.

Consider an undirected signed bipartite network, $\mathcal{G} = (\mathcal{U}_B, \mathcal{U}_S, \mathcal{E}^+, \mathcal{E}^-)$, where $\mathcal{U}_B =$

Table 4.10: Statistics on signed bipartite networks.

	Bonanza	U.S. Senate	U.S. House
$n_B = \lvert\mathcal{U}_B\rvert$	7,919	1,056	1,281
$n_S = \lvert\mathcal{U}_S\rvert$	1,973	145	515
$\lvert\mathcal{E}\rvert = \lvert\mathcal{E}^+\rvert + \lvert\mathcal{E}^-\rvert$	36,543	27,083	114,378
% Links Positive	97.98%	55.31%	53.96%
% Links Negative	2.02%	44.69%	46.04%
Density of $\mathbf{B}$	2.339×10^{-3}	0.1769	0.1734

$\{b_1, b_2, \ldots, b_{n_B}\}$ and $\mathcal{U}_S = \{s_1, s_2, \ldots, s_{n_S}\}$ represent two mutually exclusive sets of homogeneous nodes with n_B and n_S representing the number of nodes for each set, respectively. $\mathcal{E}^+ \subset \mathcal{U}_B \times \mathcal{U}_S$ and $\mathcal{E}^- \subset \mathcal{U}_B \times \mathcal{U}_S$ represent the sets of positive and negative edges, respectively, between the two sets of nodes $\mathcal{U}_B$ and $\mathcal{U}_S$. We let $\mathcal{E} = \mathcal{E}^+ \cup \mathcal{E}^-$ be the set of all edges where $\mathcal{E}^+ \cap \mathcal{E}^- = \emptyset$, in other words, two nodes cannot have both a positive and negative edge between them. We use $\mathbf{B} \in \mathbb{R}^{n_B \times n_S}$ to represent the undirected signed bipartite biadjacency matrix of $\mathcal{G}$, where $\mathbf{B}_{ij} = 1, -1$, or 0, when there exists a positive, negative, or no link between b_i and s_j. We further summarize the major notations used throughout this section in Table 6.4.

4.2.1.1 Signed Bipartite Network Datasets

We have collected three signed bipartite networks for this study. The first signed bipartite network is from the e-commerce website Bonanza[4]. Bonanza is similar to eBay[5] and Amazon Marketplace[6] in that users create an account for which they can buy or sell various goods. After a buyer purchases a product from a seller, both are able to provide a rating about the other along with a short comment. At the time of collection, Bonanza was using a rating scale of "Positive", "Neutral", and "Negative" to rate another user after a transaction. For representing the buyers and sellers, we use $\mathcal{U}_B$ and $\mathcal{U}_S$, respectively.

The next two datasets are representing the role call votes combined from the 1st to 10th United

[4]http://www.bonanza.com
[5]http://www.ebay.com
[6]http://www.amazon.com

States Congress. More specifically, we collected two separate datasets[7]; one for the U.S. Senate and the other for the U.S. House of Representatives (which we will refer to as U.S. House). In each of these datasets we represent the bills that were voted by the set $\mathcal{U}_B$ and the senators or representatives by $\mathcal{U}_S$. If a congressperson voted "Yea" or "Nay" for the bill, then we represent these as positive or negative links between them, respectively, and leave the connection missing otherwise.

Note that for simplicity throughout the rest of this section we will refer to the nodes in $\mathcal{U}_B$ as "buyers" and those in $\mathcal{U}_S$ as "sellers". In Table 4.10 we report some basic statistics of our three collected datasets. We note that in the Bonanza dataset there is a significant imbalance between the number of positive and negative links as compared to the two U.S. Congress datasets. Although these datasets are representing vastly different real-world social structures, we next investigate balance theory [30, 29] to the signed bipartite network setting.

4.2.1.2 Signed Butterflies in Signed Bipartite Networks

In signed networks one of the most fundamentally studied social theories is balance theory [30, 29], which discusses the settings in signed networks that are socially "balanced" (i.e., stable), and those that are more likely to change (to be balanced) due to the social tensions involved in maintaining "unbalanced" and seemingly unnatural connections. In recent signed network analysis works balance theory is usually investigated and then applied towards many tasks [74, 2, 55], but almost always in the form of triangles (or cycles of length 3) in a unipartite signed network. As seen in Figure 2.2, there are four possible configurations between the three nodes. We can further observe in Figure 2.2 that triangles (a) and (b) are balanced (due to having an even number of negative links), while (c) and (d) are unbalanced. Nevertheless, as previously mentioned, since there are no triangles in signed bipartite networks and they have two different node types, it is unknown whether balance theory is still applicable towards a bipartite setting.

Next, we will therefore introduce how we plan to extend the usage of balance theory to the

[7] https://www.govtrack.us/data/

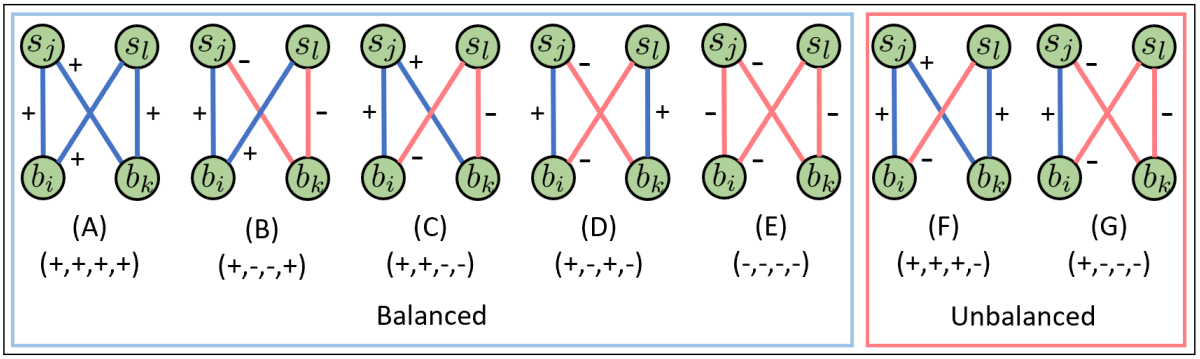


Figure 4.3: Undirected signed butterfly isomorphism classes.

smallest signed cycles (i.e., butterflies) in undirected signed bipartite networks. Thereafter we investigate and present our initial analysis of these signed butterflies in three real-world signed bipartite networks.

4.2.1.3 Signed Butterfly Isomorphism Classes

In unsigned bipartite networks, one commonly investigated structure is that of a "butterfly" [122, 123], which is a cycle of length 4. More formally, a butterfly is the simplest cohesive higher-order structure and also a complete biclique. Thus, this provides the most natural structure to investigate as a possible extension for balance theory in signed bipartite networks.

Just as there are different types of signed triangles, there are different types of signed butterflies. In Figure 4.3 we present the 7 non-isomorphic undirected signed butterflies. Note that there are five that adhere to balance theory while only two are categorized as unbalanced. We use the notation $(*,*,*,*)$ to denote a signed butterfly isomorphism class that represents the links between the buyers and sellers (b_i, s_j, b_k, s_l) (in that order with the last sign connecting s_l and b_i). The simplest of types are $(+,+,+,+)$ and $(-,-,-,-)$, which denote the classes having all positive or all negative links, respectively, and both are balanced due to having an even number of negative links (and can be seen in Figures 4.3(A) and 4.3(E), respectively). We can interpret the (+,+,+,+) class as the situations where two buyers have bought from the same two sellers and the sentiment amongst them across the four purchases was positive. Next, we have $(+,+,+,-)$ and $(+,-,-,-)$, which

are the two unbalanced classes of signed butterflies (since they have an odd number of negative links). In Figure 4.3(F) we have the signed butterfly isomorphism class that encompasses all the signed butterflies with a single negative link. We can observe that no matter where this single negative link is placed, we always have one buyer with two positive links, one buyer with a positive and negative link, and similar structure for the two sellers. The isomorphism class $(+,-,-,-)$ can be seen as the complement (if defined as swapping link signs in a signed network) of the class $(+,+,+,-)$ and defined in a similar way, but with swapping the positive and negative links in the definition. This leaves the signed butterflies having two positive and two negative links, of which we have three isomorphism classes. In Figure 4.3(D) we see the class $(+,-,+,-)$ is used to represent signed butterflies where all buyers and sellers have one positive and one negative link in their cycle. When one of the buyers has two positive links, while the other buyer has two positive links, we observe in Figure 4.3(B) that both sellers have a single positive and single negative link, and define the isomorphism class of $(+,-,-,+)$. Finally, the last type of signed butterfly has both buyers connected positively to one seller, and negatively to the other, which we represent as the class $(+,+,-,-)$ shown in Figure 4.3(C).

4.2.1.4 Signed Butterfly Analysis

In Table 4.11 we report our analysis after counting the number of signed butterflies for each isomorphism class as shown in Figure 4.3. We further calculated the percentage each isomorphism class takes up of the total signed butterfly count in each dataset (given in column "%"). Next, we analyzed the significance of these signed butterflies being found in signed bipartite networks and wanted to test whether they are overrepresented or underrepresented. Remember, balance theory would suggest that balanced isomorphism classes (A) through (E) should appear frequently while (F) and (G) (being unbalanced) should appear less frequently. To quantify this, extending the approach taken in [2], we calculate "$E\%$" as the expected percentage of total signed butterflies to fall into the given isomorphism class when randomly reassigning the positive and negative signs to the signed bipartite network. In other words, for example, "$E\%$" for the isomorphism class

Table 4.11: Signed butterfly statistics on signed bipartite networks.

Signed Butterfly Isomorphism Classes	**Bonanza**				**U.S. Senate**				**U.S. House of Representatives**			
	Count	%	E%	s	Count	%	E%	s	Count	%	E%	s
(A) $(+,+,+,+)$	2554388	0.986	0.922	386	13404168	0.262	0.094	4142	227660420	0.244	0.085	17459
(B) $(+,-,-,+)$	3830	0.001	7.8e-04	40	5595440	0.110	0.122	-277	103731010	0.111	0.123	-1137
(C) $(+,+,-,-)$	726	2.8e-04	7.8e-04	-29	9404006	0.184	0.122	1349	173875858	0.186	0.123	5843
(D) $(+,-,+,-)$	456	1.7e-04	7.8e-04	-35	5537080	0.108	0.122	-302	101409932	0.109	0.123	-1368
(E) $(-,-,-,-)$	20	7.7e-06	1.7e-07	30	6815324	0.133	0.040	3414	137478104	0.147	0.045	15104
Balanced	2559420	**0.988**	0.924		40756018	**0.797**	0.500		744155324	**0.797**	0.500	
(F) $(+,+,+,-)$	30685	0.012	0.076	-390	6225745	0.122	0.302	-2811	109763190	0.118	0.289	-11565
(G) $(+,-,-,-)$	100	3.9e-05	3.2e-05	2	4118075	0.081	0.197	-2099	79053742	0.085	0.210	-9430
Unbalanced	30785	**0.012**	0.076		10343820	**0.203**	0.500		188816932	**0.203**	0.500	

$(+,-,-,-)$ is calculated by $\binom{4}{1}\Big((|\mathcal{E}^+|/|\mathcal{E}|) \times (|\mathcal{E}^-|/|\mathcal{E}|)^3\Big)$, since there are 4 permutations of having a single positive link in a signed butterfly in class $(+,-,-,-)$ and the probability of each link appearing in a signed network with randomly assigned link signs would be the independent probabilities of having a single positive link (i.e., $|\mathcal{E}^+|/|\mathcal{E}|$) and three negative links (i.e., $|\mathcal{E}^-|/|\mathcal{E}|$). Finally, the value "$s$" is used to denote the number of standard deviations the actual count differs from our calculated expected number (based on "$E\%$") for each signed butterfly type and just as in [2], a positive (or negative) "s" value signifies appearing significantly more (or less) than expected.

We first observe that the large majority of signed butterflies in our three signed bipartite networks are indeed balanced. Furthermore, they are significantly more balanced than expected based on the link sign ratio in the given network (i.e., comparing columns "%" and $E\%$). The second observation is that all unbalanced signed butterflies across the three datasets are significantly underrepresented, except for the $(+,-,-,-)$ butterflies in Bonanza, where it shows a minimal over representation. Similarly, across all datasets the $(+,+,+,+)$ and $(-,-,-,-)$ signed butterflies are significantly overrepresented, further strengthening the applicability of balance theory in signed bipartite networks. However, the isomorphism classes involving two positive and two negative links appear to not always be found overrepresented. For example, the class where all buyers and sellers have one positive and one negative link, i.e., $(+,-,+,-)$, is less commonly found than expected across all three datasets.

In summary, our findings suggest that: 1) we can use signed butterflies to extend balance theory for signed bipartite networks; and 2) signed bipartite networks adhere to balance theory when defined in terms of signed butterflies, thus making them applicable to advance numerous tasks in signed bipartite networks.

4.2.1.5 Signed Caterpillars in Bipartite Networks

A "signed caterpillar" we define as paths of length 3 that are missing just one link to becoming a signed butterfly. Therefore, a signed caterpillar can take on one of eight different forms, since

it is composed of three links being ether positive or negative. Note that all caterpillar types have the potential to be transformed into a signed butterfly (i.e., closed into a cycle of length 4) that is either balanced or unbalanced. If a signed caterpillar contains an even number of negative links, we refer this as a "balanced path" and balance theory would suggest a positive (or negative) link transforming it into a balanced (or unbalanced) signed butterfly. Similarly, we define an a signed caterpillar as an "unbalanced path" when having an odd number of negative links and balance theory would suggest a negative (or positive) link to close into a balanced (or unbalanced) signed butterfly.

4.2.2 Sign Prediction for Signed Bipartite Networks

With the aforementioned definitions and notations, we formally define the problem of sign prediction in undirected signed bipartite networks as the following:

Given an undirected signed bipartite network $\mathcal{G} = (\mathcal{U}_B, \mathcal{U}_S, \mathcal{E}^+, \mathcal{E}^-)$ *represented as a bi-adjacency matrix* $\mathbf{B} \in \mathbb{R}^{|\mathcal{U}_B| \times |\mathcal{U}_S|}$*, we seek to predict the signs of no link pairs* $(b_i, s_j) \in \{\mathcal{U}_B \times \mathcal{U}_S\} \backslash \{\mathcal{E}^+ \cup \mathcal{E}^-\}$.

Sign prediction in signed networks has been previously studied [61, 124, 125, 126, 4]. However, in the signed bipartite setting, many of these methods are no longer applicable, since there are no triangles. In Section 4.2.1.2, we validated that the large majority of signed butterflies in signed bipartite networks are balanced. Methods for predicting link signs in unipartite signed networks can be categorized into three main groups: 1) supervised methods; 2) low-rank approximation methods; and 3) propagation based methods. Therefore we develop a representative sign prediction method specific to signed bipartite networks from each group. More specifically, we propose: 1) a supervised classification method that uses signed caterpillars/butterflies; 2) extend a low-rank modeling method to ensure the predicted signs favor creating more balanced signed butterflies; and 3) a random walk based approach that integrates one-mode projection networks for $\mathcal{U}_B$ and $\mathcal{U}_S$ constructed using balance theory.

4.2.2.1 Signed Caterpillars Based Classifier

One common approach towards predicting links or link signs in both signed and unsigned networks is to frame the task in terms of a supervised classification problem [4, 61, 127, 128]. Here we extend the idea to the signed bipartite setting by formulating the problem of predicting the sign between a buyer b_i and a seller s_j by extracting features from either the individuals (i.e., their positive and negative degrees) or local neighborhood features based on balance theory (i.e., signed caterpillars).

To train our model we construct a training dataset consisting of known signed links (between a buyer and seller). Then, after having a trained model, we can extrapolate what we learned from the training data to predict a positive or negative sign for an unknown buyer and seller pair. More specifically, we use a logistic regression model following the prediction on directed signed unipartite networks work in [4].

Feature Extraction. The two different sets of features we evaluate are either based on the two nodes degree distributions or information about how many signed caterpillars they are the two endpoints of (i.e., they would be the buyer and seller connection transforming the signed caterpillar to a balanced or unbalanced signed butterfly). Thus, the feature vector $\mathbf{x}_{ij}^{d}$ for the pair (b_i, s_j) includes the the positive and negative degrees for both b_i and s_j. In comparison, $\mathbf{x}_{ij}^{sc}$ contains the counts for each of the 8 possible signed caterpillars that have b_i and s_i as the endpoints. The expectation is that the features $\mathbf{x}_{ij}^{sc}$ will be more informative than those of $\mathbf{x}_{ij}^{d}$ because they would provide a vast amount of informaiton as to whether their link sign is likely to be positive or negative according to balance theory when considering the types of signed butterflies that would be constructed. This is in comparison to only using the degrees in a method similar in nature to a signed preferential attachment model with $\mathbf{x}^{d}$. We denote the supervised classifiers that use $\mathbf{x}_{ij}^{d}$ and $\mathbf{x}_{ij}^{sc}$ as SCd and SCsc, respectively.

4.2.2.2 Low-Rank Sign Prediction

In recent years the low-rank matrix factorization approaches have been gaining popularity for numerous applications involving link related network predictions [129, 130, 131]. Although some of these works have focused on signed networks [131, 132, 130], none are structured to select link signs that would explicitly push towards more signed butterflies being balanced in signed bipartite networks. Thus, we first introduce a basic matrix factorization approach to model the signed bipartite network using the biadjacency matrix $\mathbf{B}$. Then we introduce how we can successfully modify this model through the inclusion of additional pairs of buyers and sellers derived from suggested implicit signed links that would construct the most balanced signed butterflies with the suggested link sign.

Basic Matrix Factorization Model: The set of existing edges in $\mathbf{B}$ are denoted in the set $\mathcal{E} = \{(b_i, s_j)|\mathbf{B} \neq 0\}$. In terms of the link sign prediction task we would like to discover two latent matrices $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_{n_B}] \in \mathbb{R}^{d \times n_B}$ and $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{n_S}] \in \mathbb{R}^{d \times n_S}$ of dimension d for the set of buyers and sellers, respectively, to solve the following optimization problem:

$$\min_{\mathbf{U},\mathbf{V}} \sum_{(b_i,s_j)\in\mathcal{E}} \max\left(0, 1 - \mathbf{B}_{ij}(\mathbf{u}_i^\top \mathbf{v}_j)\right)^2 + \lambda\left(|\mathbf{U}|_F^2 + |\mathbf{V}|_F^2\right) \tag{4.11}$$

where $\mathbf{u}_i^\top \mathbf{v}_j$ is used to model the link sign between buyer b_i to seller s_j. Note that when the real link sign (i.e., $\mathbf{B}_{ij}$) and the predicted link sign (i.e.,$\mathbf{u}_i^\top \mathbf{v}_j$) are of the same sign (i.e., both positive or both negative) then $\mathbf{B}_{ij}(\mathbf{u}_i^\top \mathbf{v}_j)$ is positive, and if over 1 then there is no loss. However, when the real and predicted values have differing signs then there is a higher loss value associated to drive the minimization during the training process. Following the work in [131] we use Stochastic Gradient Descent (SGD) to minimize the objective in Eq. (4.11).

This allows us to then utilize the learned low-dimensional representations for each buyer and seller to predict the sign of unknown buyer and seller pairs. However, although this model is effectively learning a representation that can accurately predict the existing links, it does not explicitly control whether the signs of non-existing links are actually going to predict link signs that adhere to balance theory (i.e., having more signed butterflies balanced than unbalanced). Therefore we

denote this method simply as MF. Next, we will present an extension to this basic framework to further ensure more signed butterflies between the missing links are balanced.

Matrix Factorization with Balance Theory: As previously discussed, the aforementioned basic matrix factorization approach given in Eq. (4.11) does not explicitly enforce the non-existing link signs to favor balanced relationships. Instead it can only focus on learning low-dimensional representations for each buyer and seller such that the model minimizes the error on predicting the existing link signs. The approach we have selected is to further encourage the model learning link signs for buyer and seller pairs that currently do not exist in the signed bipartite network, but would convert many signed caterpillars into balanced signed butterflies if they were to exist.

The first step is calculating whether balance theory would suggest a positive or negative link for each buyer and seller pair (b_i,s_j), that currently do not have a link between them, based on the types of signed caterpillars they're jointly involved in and the endpoints of.

Theorem 3. *Given a signed undirected biadjacency matrix* $\mathbf{B}$*, then the matrix* $\hat{\mathbf{S}} = \mathbf{B}\mathbf{B}^T\mathbf{B} \odot \overline{\mathbf{B}}$ *is such that* $sign(\hat{\mathbf{S}}_{ij})$ *suggests the sign of a non-existent link in* $\mathbf{B}$ *that would result in a net gain of* $|\hat{\mathbf{S}}|$ *additional balanced signed butterflies created (after subtracting the number of potential unbalanced signed butterflies created simultaneously) if the suggested signed link were to be added between* b_i *and* s_j*, where we define* $\overline{\mathbf{B}}$ *as* $\overline{\mathbf{B}}_{ij} = 0$ *if* $\mathbf{B}_{ij} \neq 0$ *and* $\overline{\mathbf{B}}_{ij} = 1$ *when* $\mathbf{B}_{ij} = 0$*.*

Proof. If we let $\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{0} \end{bmatrix}$ be the adjacency matrix in $\mathbb{R}^{|\mathcal{U}|\times|\mathcal{U}|}$. We can observe that $\mathbf{A}^3 = \begin{bmatrix} \mathbf{0} & \mathbf{B}\mathbf{B}^T\mathbf{B} \\ \mathbf{B}^T\mathbf{B}\mathbf{B}^T & \mathbf{0} \end{bmatrix}$. We note that in [133] it has been shown $\mathbf{A}^l = \mathbf{M}_B^l - \mathbf{M}_U^l$, where $\mathbf{M}_B^l, \mathbf{M}_U^l \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{U}|}$ store the number of balanced and unbalanced paths of length l, respectively, between all pairs of nodes in a signed network represented as $\mathbf{A}$. Thus, since $\mathbf{A}^3_{ij} = \left[\mathbf{B}\mathbf{B}^T\mathbf{B}\right]_{ij}$ for some buyer b_i and seller s_j, we observe that this represents the number of of balanced paths of length 3 subtracted by the number of unbalanced paths of length 3. By definition of a signed caterpillar, if one is a balanced path, then it would suggest a positive link to close to be a balanced signed butterfly, but if it was formed by an unbalanced path it would require the closing link to be negative to form a balanced butterfly. Therefore, it follows that $sign(\left[\mathbf{B}\mathbf{B}^T\mathbf{B}\right]) = sign(\mathbf{M}_B^l - \mathbf{M}_U^l)$ indeed represents

the sign that would promote the creation of more balanced signed butterflies, and similarly for the net gain of balanced butterflies being formed equaling the absolute value of their difference ($i.e., |\mathbf{M}_B^l - \mathbf{M}_U^l|$). It is then easy to extend to only the buyer and seller pairs b_i and s_j in $\left[\mathbf{B}\mathbf{B}^T\mathbf{B}\right] \odot \overline{\mathbf{B}}$ after taking the element-wise product with $\overline{\mathbf{B}}$ that zeros out the pairs that have an existing link. □

Note that $\hat{\mathbf{S}}$ can also be calculated (sometimes more efficiently) using the following:

$$\hat{\mathbf{S}}_{ij} = \begin{cases} \left[\mathbf{B}\mathbf{B}^\top\mathbf{B}\right]_{ij} & \text{if } \mathbf{B}_{ij} = 0 \\ 0 & \text{otherwise} \end{cases}$$

to avoid using the potentially very dense matrix $\overline{\mathbf{B}}$ for sparse signed bipartite networks.

Using Theorem 3 we can construct additional sets $\mathcal{E}_i^+$ and $\mathcal{E}_i^-$ of implicit positive and negative links, respectively, suggested by balance theory that would create the highest net gain of balanced signed butterflies in the signed bipartite network. We define these sets as follows:

$$\begin{aligned} \hat{\mathcal{E}}_i^+ &= \{(b_i, s_j) \,|\, \hat{\mathbf{S}}_{ij} > 0 \text{ and } \hat{\mathbf{S}}_{ij} \in top_k(\hat{\mathbf{S}})\} \\ \hat{\mathcal{E}}_i^- &= \{(b_i, s_j) \,|\, \hat{\mathbf{S}}_{ij} < 0 \text{ and } \hat{\mathbf{S}}_{ij} \in bottom_k(\hat{\mathbf{S}})\} \end{aligned} \tag{4.12}$$

where $top_k(\hat{\mathbf{S}})$ and $bottom_k(\hat{\mathbf{S}})$ are used to denote the k largest and smallest values, respectively, in $\hat{\mathbf{S}}$.

We formulate our object that incorporates balance theory as follows:

$$\begin{aligned} &\min_{\mathbf{U},\mathbf{V}} \sum_{(b_i,s_j)\in\mathcal{E}} \max\left(0, 1 - \mathbf{B}_{ij}(\mathbf{u}_i^\top \mathbf{v}_j)\right)^2 + \lambda\left(|\mathbf{U}|_F^2 + |\mathbf{V}|_F^2\right) \\ &+ \alpha \sum_{(b_i,s_j)\in\hat{\mathcal{E}}_i^+} \max\left(0, 1 - \hat{\mathbf{S}}_{ij}(\mathbf{u}_i^\top \mathbf{v}_j)\right)^2 + \beta \sum_{(b_i,s_j)\in\hat{\mathcal{E}}_i^-} \max\left(0, 1 - \hat{\mathbf{S}}_{ij}(\mathbf{u}_i^\top \mathbf{v}_j)\right)^2 \end{aligned} \tag{4.13}$$

where α and β are used to control the level at which we incorporate the modeling of signed butterflies through the inclusion of the implicit positive and negative links, respectively. We again note that these implicit positive and negative links are implied by balance theory by using $\hat{\mathbf{S}}$, which effectively counts for each node pair (b_i, s_j) what the net gain of total balanced signed butterflies

would be once including the link with the suggested sign (according to the majority count of signed caterpillars being of balanced or unbalanced paths of length 3). We denote this matrix factorization method using balance theory as MFwBT.

4.2.2.3 Random Walk Based Sign Prediction

Typical propagation based methods, such as the random walk with restart [44] have seen many variants and been applied to solve link prediction and ranking related tasks in unsigned unipartite networks. However, signed bipartite networks pose multiple challenges that prevent them from directly using the typical methods. One such problem is that bipartite networks do not have a stationary distribution and thus do not converge [134]. One way of handling this problem in unsigned bipartite networks is considered a "lazy" random walk, where the walker will probabilistically stay at the same node. We will later use this method as a comparison against our proposed random walk based method. Furthermore as seen in previous sign prediction methods for unipartite signed networks, balance theory is the key component towards obtaining higher performance when predicting the sign of unknown links. Thus, due to our analysis of the signed butterflies, indeed signed bipartite networks are showing high levels of balance and therefore we should also be using balance theory to guide the random walk based method for signed bipartite networks towards a solution having more balanced relations.

Here we present a random walk based approach that integrates the $\mathcal{U}_B$ and $\mathcal{U}_S$ one-mode projection adjacency matrices, which are constructed using balance theory, to aid in handling the issues faced with the bipartite setting, and develop a signed random walk based approach to not only allow a proper transition matrix, but to furthermore have the random walker be promoting balance theory. The first step will be the construction of a signed adjacency matrix $\mathbf{A}$ based on balance theory, followed by defining a signed transition matrix that can further promote and propagate balanced relations throughout the network.

Constructing the one-mode adjacency matrices: In unsigned bipartite network analysis one-mode projections are typically used for both analysis and aiding to solve various tasks [135, 136,

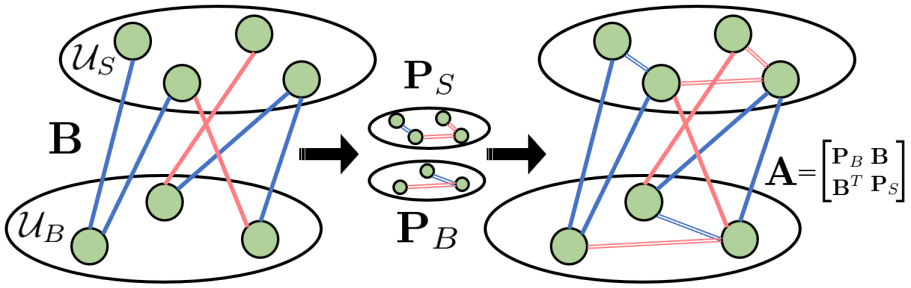


Figure 4.4: High-level overview of how we construct $\mathbf{A}$ from $\mathbf{B}$, $\mathbf{P}_S$ and $\mathbf{P}_B$.

137]. They are constructed by creating a projection network that creates implicit connections between nodes of the same type. In terms of our definitions, two one-mode projection networks can be performed, one that connects the buyers in $\mathcal{U}_B$ together amongst themselves and the other for the sellers in $\mathcal{U}_S$ by constructing seller to seller links; these relations can be represented in the adjacency matrices $\mathbf{P}_B \in \mathbb{R}^{|\mathcal{U}_B|\times|\mathcal{U}_B|}$ and $\mathbf{P}_S \in \mathbb{R}^{|\mathcal{U}_S|\times|\mathcal{U}_S|}$. A visual example can be seen in Figure 4.4 when going from $\mathbf{B}$ to $\mathbf{P}_S$ and $\mathbf{P}_B$ from left to right through the first arrow.

We note that there is not just one way to discover these implicit connections between pairs of users in the same set, and in fact there are many possible methods for one-mode projections [136, 137]. It has also been studied that using different methods to construct the projection networks can cause drastic changes to the usability and performance [138]. In wanting to carefully construct these projection networks, we choose to utilize balance theory in the form of signed triangles. Next we will discuss the formation of the adjacency matrix $\mathbf{P}_B$, and a similar process can be followed for constructing $\mathbf{P}_S$ (although we only discuss $\mathbf{P}_B$ here).

Based on the ideas of common neighbor similarity in unsigned networks, we will possibly connect two buyers b_i and b_j if they have at least one seller in common they are linked to. Let the number of common sellers that b_i and b_j agree upon (in terms of link sign) be denoted as ns_{ij}^A. Similarly let ns_{ij}^D denote the number of sellers these two buyers disagree on in terms of link sign. Then we define $\mathbf{P}_{Bij} = \mathbf{P}_{Bji} = ns_{ij}^A - ns_{ij}^D$, which we can see is taking the number of sellers they agree upon in terms of signed connections (i.e., both either negatively or positively connected to that seller) and subtracting the number of sellers they disagree on (i.e., the sellers where one buyer

has a positive link while the second buyer has a negative connection with that seller). We can now further see the connection between $\mathbf{P}_{Bij}$ and the common neighbor similarity method. It is easy to verify that out of all the triangles formed between b_i, b_j, and the sellers s_k they are commonly linked to, that using the links $\mathbf{B}_{ik}$, $\mathbf{B}_{jk}$ and $\mathbf{P}_{Bij}$, we see that the majority will adhere to balance theory. This is by design since if $na_{ij}^A > na_{ij}^D$ then $sign(\mathbf{P}_{Bij})$ is positive and closing the ns_{ij}^A triangles to be balanced, while the lesser number of ns_{ij}^D will close to be unbalanced. Note that a similar argument can be given when $ns_{ij}^A < ns_{ij}^D$ and if $ns_{ij}^A = ns_{ij}^D$ then $\mathbf{P}_{Bij} = 0$ and no signed triangles are formed. Ultimately, we construct a parameterized version as follows:

$$\mathbf{P}_{Bij} = \begin{cases} 0 & \delta_n < ns_{ij}^A - ns_{ij}^D < \delta_p \\ ns_{ij}^A - ns_{ij}^D & \text{otherwise} \end{cases} \tag{4.14}$$

where δ_p and δ_n are used to define thresholds for the necessary magnitude of $ns_{ij}^A - ns_{ij}^D$ to have a non-zero value in $\mathbf{P}_{Bij}$. This allows us to ignore adding smaller values (e.g., 2), since in some settings having such a small value might not be very significant and thus we might not want to construct a link between b_i and b_j. Note that for simplicity we allow δ_p and δ_n to be shared for constructing both $\mathbf{P}_B$ and $\mathbf{P}_S$.

Performing the random walk: Now having the two projection adjacency matrices $\mathbf{P}_B$ and $\mathbf{P}_S$, we can use them to construct an adjacency matrix $\mathbf{A} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{U}|}$, which will be the unibipartite signed network we perform our random walk on, where $\mathcal{U} = \{b_1, \ldots b_{n_B}, s_1, \ldots s_{n_S}\}$. In Figure 4.4 we show the high-level intuition of how to construct $\mathbf{A}$. First we denote $\hat{\mathbf{B}}$ as the row normalized biadjacency matrix where $\hat{\mathbf{B}}_{ij} = \mathbf{B}_{ij} / \sum_k |\mathbf{B}_{ik}|$. We similarly construct row normalized adjacency matrices $\hat{\mathbf{P}}_B$ and $\hat{\mathbf{P}}_S$. Now we can formulate $\mathbf{A}$ as follows:

$$\mathbf{A} = \begin{bmatrix} \hat{\mathbf{P}}_B & \omega\hat{\mathbf{B}} \\ \omega\hat{\mathbf{B}}^T & \hat{\mathbf{P}}_S \end{bmatrix} \tag{4.15}$$

where ω is a parameter that can be used to bias the random walker to favor the real links in our signed bipartite network as compared to the implicit links we obtained through the $\mathcal{U}_B$ and $\mathcal{U}_S$ one-mode projection networks. Next we construct a similar row normalized adjacency matrix $\hat{\mathbf{A}}$ where $\hat{\mathbf{A}}_{ij} = \mathbf{A}_{ij} / \sum_k |\mathbf{A}_{ik}|$.

Finally, we utilize $\hat{\mathbf{A}}$ in a random walk propagation model where we define $\mathbf{Y}$ to be the matrix holding the inferred link signs as follows:

$$\mathbf{Y}_{ij} = \sum_{k} \hat{\mathbf{A}}_{ik} \mathbf{Y}_{kj} \tag{4.16}$$

Next we describe how the above adheres to balance theory in terms of triangles of the adjacency matrix $\mathbf{A}$. This is because for some k if $\hat{\mathbf{A}}_{ik}\mathbf{Y}_{kj} > 0$ it increases $\mathbf{Y}_{ij}$ ensuring it to be positive, which would be a triangle consisting of either three positives, or two negatives and a positive. Similarly when $\hat{\mathbf{A}}_{ik}\mathbf{Y}_{kj} < 0$ we are decreasing $\mathbf{Y}_{ij}$ and encouraging it to be negative, thus also following balance theory. The closed form solution that includes the restart capability, with probability $(1 - c)$, is given to be the following:

$$\mathbf{Y} = (1 - c)(\mathbf{I} - c\hat{\mathbf{A}})^{-1} \tag{4.17}$$

Note that each signed butterfly involving b_i, s_j, b_k, and s_l in the original network $\mathbf{B}$ now consists of up to $\binom{4}{3}$ triangles in $\hat{\mathbf{A}}$. Thus, when we are encouraging balanced triangles here in $\mathbf{Y}$ this correlates to having balanced signed butterflies in the upper right corner of $\mathbf{Y}$, which is where we obtain the link sign predictions (i.e., when predicting the sign between b_i and s_j we have use $\hat{\mathbf{B}}_{ij'}$ where $j' = (n_B + j)$. We denote this method as Signed Bipartite Random Walk (SBRW).

For comparison, if we set the two one-mode projection matrices to the identity matrix (i.e., $\mathbf{P}_B = \mathbf{P}_S = \mathbf{I}$) and set $\omega = 1$ then Eq. 4.16 becomes the equation for a lazy random walk method, which we denote as LazyRW.

4.2.3 Experiments

In this section, we empirically evaluate our proposed sign prediction methods for signed bipartite networks that harness balance theory. We seek to answer the following: (1) Does the extended balance theory to signed butterflies in the bipartite setting provide an increase in performance for sign prediction? and (2) How do the proposed methods work/compare? To address these questions we perform experiments to measure the performance for each of the proposed sign prediction

methods across three real-world signed bipartite networks. To better understand our methods and the contribution of balance theory, we also follow-up with a parameter sensitivity analysis for the major parameters of our methods.

Experimental Settings: Here we discuss the settings used for our experiments on sign prediction in signed bipartite networks. As previously discussed in Section 4.2.1.1 we have collected three signed bipartite networks for this study, namely, Bonanza, U.S. Senate, and U.S. House. For our sign prediction experiments we have randomly selected 10% of the links as test, utilized a random 5% for validation purposes of tuning the hyperparameters of our models, and the remaining 85% as training for each of our datasets. More specifically, each method is only given access to the signed bipartite network induced from the training links, then, for each edge in the testing set, we compare the ground truth link sign with the link sign the specific method suggests for that undirected pair. For evaluation we use both F1 and Area Under the receiver operating characteristic Curve (AUC), since the positive and negative links are unbalanced especially in the Bonanza dataset. To the best of our knowledge this is the first study of predicting link signs in signed bipartite networks; hence other existing methods either for unipartite signed or unsigned bipartite networks are likely not applicable. The main investigation is two-fold. First, we want to test the applicability of balance theory (based on signed butterflies) to aid in sign prediction. Second, we want to provide insights to guide practical usage of sign predictors with different types of signed bipartite networks. Thus, we only provide a comparison against the methods we have presented in this dissertation.

4.2.3.1 Comparison Results

The results across our three signed bipartite networks in terms of AUC and F1 can be found in Table 4.12 and the first observation we make is that there is not one proposed method that outperforms the others across all the datasets.

The second observation we make is that the three methods SCsc, MFwBT, and SBRW, which receive aid in prediction from balance theory when defined using signed butterflies, always perform better than their respective baseline method (i.e., SCd, MF, LazyRW) that only use generic signed

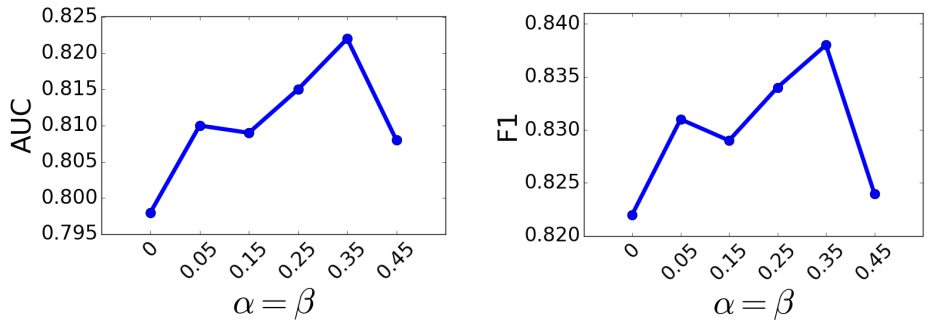
Table 4.12: Link sign prediction results in terms of (AUC,F1).

Sign Prediction Method	Bonanza	U.S. Senate	U.S. House
SCd	(0.553 , 0.959)	(0.638 , 0.654)	(0.625 , 0.635)
SCsc	(0.664 , 0.674)	(0.812 , 0.823)	(0.827 , 0.837)
MF	(0.593 , 0.903)	(0.792 , 0.812)	(0.831 , 0.846)
MFwBT	(0.608 , 0.905)	(0.814 , 0.827)	(0.834 , 0.848)
LazyRW	(0.547 , 0.979)	(0.808 , 0.821)	(0.815 , 0.827)
SBRW	(0.582 , 0.949)	(0.836 , 0.849)	(0.846 , 0.858)

network information in terms of AUC and only in two cases the F1 is worse. In the Bonanza dataset we have the SCd and LazyRW outperforming SCsc and SBRW, respectively, in terms of F1 (although performing worse in AUC). The reason for this is the heavy imbalance between the positive and negative links in this dataset, more specifically, almost 98% of the links are positive, which is generally a setting where the AUC measurement is preferred to understand the performance better. Therefore we can see that to better detect the few negative links comes at the sacrifice of misclassifying some of the positive links, which is why the F1 of SCsc and SBRW is less than SCd and LazyRW, but comes with a significant increase in AUC. In general we observe that in fact the usage of signed butterflies for sign prediction in signed bipartite networks provides a very significant improvement in almost all cases. This fact suggests that we can give a positive answer to our first question – the usage of balance theory in the form of signed butterflies for sign prediction in signed bipartite networks indeed provides an empirically verifiable improvement.

In the U.S. Senate and U.S. House datasets, for the methods constructed based on intuitions of how to correctly ensure more balanced signed butterflies are being created when predicting missing link signs (i.e, SCsc, MFwBT, and SBRW), we see the low-rank model outperforms the the supervised classifier approach, while the random walk method performs the best (for both AUC and F1).

However, unlike the two U.S. Congress datasets, in the Bonanza dataset we actually observe the complete opposite behavior (in terms of AUC) for the ranking of methods that utilize the signed butterfly based balance theory. We hypothesize this is due to the heavy class imbalance between

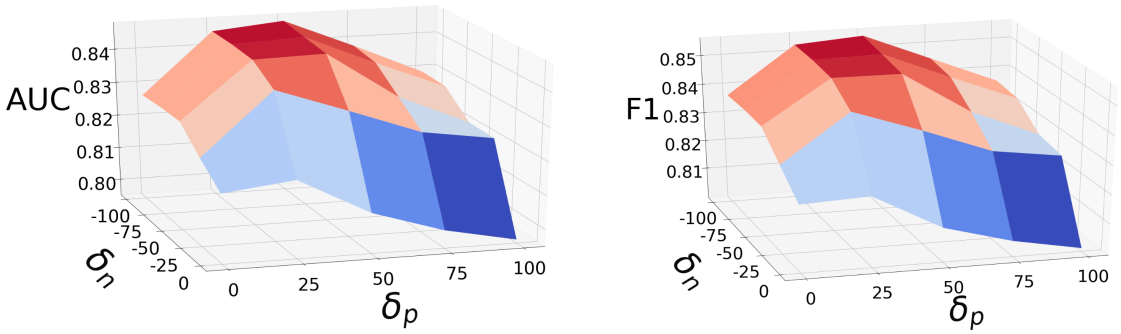


(a) MFwBT (AUC)

(b) MFwBT (F1)

Figure 4.5: Parameter sensitivity on α and β in MFwBT on the U.S. Senate dataset.

the positive and negative links. With this imbalance the SBRW method might be unable to directly handle this setting as the parameters only focus on separating real/implicit and balance/unbalance through ω and δ_p/δ_n. Futhermore, if most negative links are involved in balance relationships then actually this would cause even more positive links to be constructed in the two one-mode projection matrices (since two negatives would result in a positive link being created). In comparison, MFwBT is able to more accurately control the ratio of positive to negative implicit links being used in the training procedure (through selecting the size of both $\hat{\mathcal{E}}_i^+$ and $\hat{\mathcal{E}}_i^-$) when extracting them from investigating which links would cause the most signed caterpillars to turn into balance signed butterflies. Also, we note that in our study we fixed $\alpha = \beta$, but this mechanism would further allow MFwBT to balance the contribution of implicit positive and negative links towards learning the most effective representations. Finally, although we see a drastic improvement in terms of AUC for the SCsc method, we also observe this comes at great cost to the F1 measure, and thus this method is just discovering a trade-off of predicting more negative links. This is because we have tuned our logistic regression model to use weights on each training example inversely proportional to the frequency of that link type.



(a) SBRW (AUC)

(b) SBRW (F1)

Figure 4.6: Parameter sensitivity on δ_p and δ_n in SBRW on the U.S. House dataset.

4.2.3.2 Parameter Analysis

Among our three proposed sign prediction methods, the low-rank modeling with balance theory (MFwBT) and random walk (SBRW) methods contain interesting hyperparameters from the perspective of wanting to further understand balance theory in signed bipartite networks.

In our MFwBT method, we discussed that we can control the number of suggested implicit positive and negative links from signed butteflies being included in $\mathcal{E}_i^+$ and $\mathcal{E}_i^-$, respectively. We performed a grid search for both the size of $\mathcal{E}_i^+$ and $\mathcal{E}_i^-$, in the set {0,1000,10000}. We discovered that the best setting when considering across the three datasets was having $|\mathcal{E}_i^+| = 1000$ and $|\mathcal{E}_i^-| = 10000$, which we suspect is due to the class imbalance and having more explicit positive links than negative links. Furthermore, the values of α and β were used to control the contribution of training on both positive and negative links suggested based on signed butterflies (i.e., links in $\mathcal{E}_i^+$ and $\mathcal{E}_i^-$), respectively. For simplicity of our analysis we set $\alpha = \beta$ and report the performance on our validation set for the U.S. Senate dataset in Figure 4.5. We observe that updating the node representations using suggested signed links (that were selected since they would close the most signed caterpillars into balanced signed butterflies) provides an improvement over not taking balance theory into account (which is when $\alpha = \beta = 0$), but care should be taken to not put too much focus on these implicit links. We observe similar findings in our other datasets.

For our SBRW method, there are two main sets of parameters ω and the threshold pair δ_p

with δ_n. We varied ω at a large granularity in the set $\{1, 2, 3, 5\}$, and observed there was not as much significant difference as found in varying δ_p and δ_n, thus we selected the best on average across the three datasets of $\omega = 2$ and fixed this value to investigate the impact of δ_p and δ_n on the performance for predicting the missing link signs.

In Figure 4.6, for the U.S. House dataset, we varied both δ_p in the set {0,25,50,75,100} and similarly for δ_n in the set {0,-25,-50, -75, -100}. Note that although we saw similar trends across the three datasets, the specific magnitude of δ_n and δ_p we needed to tune separately for each dataset due to the average magnitude of ns^A_{ij} and ns^D_{ij} (i.e., number of common sellers b_i and b_j agree or disagree on, respectively) for constructing $\mathbf{P}_B$ and similarly for $\mathbf{P}_S$, although we fixed δ_p and δ_n for constructing both one-mode projection matrices. We observe in terms of both AUC and F1 from Figure 4.6 that indeed using these two thresholds to avoid implicit links that do not have a significant amount of information (i.e., low magnitude of $|ns^A_{ij} - ns^D_{ij}|$) provides great improvement to our method. It appears that implicit positive links that have low support are helpful to include. However, it seems better to avoid inferred negative links, which we obtained based on balance theory in the form of signed triangles between two buyers and a seller (similarly for the case of two sellers and a buyer). Although including a few is helpful, ones that have low amount of balance theory support from the network are better left out of the propagation process.

CHAPTER 5

MINING NETWORKS WITH NEGATIVE LINKS

In this chapter[1,2], we focus on the development of signed network mining methods. In traditional network analysis there are two major directions for network mining, which focuses on utilizing data mining techniques for graph data. More specifically, they can be categorized into either link-oriented tasks and node-oriented tasks.

For signed network analysis the objective of link-oriented tasks are to reveal fine-grained and comprehensive understanding of positive and negative links. The availability of negative links not only enriches the existing link-oriented tasks for unsigned networks such as tie strength prediction [62, 63, 64], but also encourages novel link-oriented tasks specific to signed networks such as sign prediction and negative link prediction. While positive/negative link prediction [3, 4, 61, 131] and sign prediction [26, 139, 140] have been extensively explored, research on signed tie strength prediction is rather limited. We note that the signed node relevance measures we presented in Section 3.1 were evaluated on these link-oriented mining tasks of sign and tie-strength prediction in signed networks.

Node-oriented mining tasks provide necessary means in order to better understand nodes in networks. Hence, for signed networks, the major node-oriented tasks include community detection, node classification, and node embedding, among which community detection [141, 142] is the most extensively studied, while signed network embedding is in the earliest of stages compared to the others. Hence, in Section 5.1 we first seek to develop state-of-the-art node embeddings by combining signed social theories with the modern techniques of graph neural networks [143, 144, 22, 145, 75, 146] that are a class of deep learning methods specifically designed for graph-structured

[1] Tyler Derr, Yao Ma, and Jiliang Tang. "Signed Graph Convolutional Networks." In Proceedings of the 18th International Conference on Data Mining (ICDM). 2018.

[2] Amin Javari, Tyler Derr, Pouya Esmalian, Jiliang Tang, and Kevin Chen-Chuan Chang. "ROSE: Role-based Signed Network Embedding." In Proceedings of the 29th International Conference on The World Wide Web (WWW). 2020.

data. Then, thereafter we seek to develop a novel transformation-based signed network embedding methodology that is able to first transform the network based on node roles associated with directed labeled edges to take advantage of traditional network embeddings models and then aggregate the learned embeddings back to the original signed network.

5.1 Signed Graph Convolutional Networks

Recently there has been a large and growing interest of generalizing neural network models to structured data, with one of the most prevalent structures being graphs (such as those found in social media). The idea of generalizing neural network models to graph structures, namely graph neural networks (GNNs) [143, 144, 22, 145, 75, 146, 147, 148], has lately started to become more developed by overcoming the difficulties and trade-offs previously associated with fast heuristics compared to slow and more principled approaches. One particular type of GNN are graph convolutional networks (GCNs) which are modeled after the classical convolutional neural networks [70]. The first GCN introduced for learning representations at the node level was in [22], where they utilized GCNs for the semi-supervised node classification problem. Furthermore, learning low-dimensional node representations have been previously proven to be useful in many network analysis tasks including node classification [82, 77], such as link prediction [149, 150], community detection [151, 152], and visualization [153, 149].

Previous work has mostly focused on using GCNs for unsigned graphs (or graphs consisting of only positive links). However, especially with the ever growing popularity of online social media, signed graphs are becoming increasingly ubiquitous. This naturally leads the question as to whether unsigned GCNs are suitable to be used on signed networks. Unfortunately, there are many reasons as to why unsigned GCNs are not capable of learning meaningful node representations in signed networks. First, it is unclear how they would handle the availability of negative links in signed networks, and furthermore, negative links invalidate some of the underlying key assumptions of GCNs. For example, GCNs designed for unsigned networks learn a node representation using the fundamental social theory homophily [8], which states users having connections are more

Table 5.1: Notations in regards to signed graph convolutional networks.

Notations	Descriptions
$\mathbf{Z}$	Low-dimensional representation of signed network $\mathcal{G}$
$B_i(l)$ $(U_i(l))$	The set of users that can be reached from u_i along a (un)balanced path of length l.
$B(l)$ $(U(l))$	The aggregator responsible for incorporating the information from the set of users $B_i(l)(U_i(l))$
$\mathbf{z}_i$	The final embedding of user u_i
$\mathcal{N}_i^+$ $(\mathcal{N}_i^-)$	Set of positive (negative) neighbors of u_i
$\mathbf{h}_i^{B(l)}$ $(\mathbf{h}_i^{U(l)})$	The (un)balanced representation of u_i at the l^{th} layer
$\mathbf{W}^{B(l)}$ $(\mathbf{W}^{U(l)})$	Weight matrices used for learning how to propagate (un)balanced information in the l^{th} layer

likely to be similar than those without links. Hence, the aggregation processes of GCNs use local neighborhood information when constructing the low-dimensional embedding for each node. However, homophily may not be applicable to signed networks [67]. Instead, in signed networks, there are specific social theories and principles defined in the context of having both positive and negative links. Therefore dedicated efforts are needed for redesigning GCNs specifically for signed networks.

Although it is now clear that GCNs will need to be specifically redesigned to provide the same fruitful performance as previously shown in unsigned networks when applied to signed networks, there are still tremendous challenges to overcome. When designing signed GCNs the primary challenges are: (1) how to correctly handle negative links, since their properties are inherently different than those of positive links; and (2) how to combine the positive and negative links into a single coherent model to learn effective node representations. Thus, we turn our attention towards social theories specific to signed networks (similarly to how the unsigned models were constructed using unsigned theories like homophily). More specifically, one fundamental signed network social theory that had been developed in social psychology is balance theory [29, 30]. Thus, we seek to harness this signed network social theory to solve these two challenges of applying graph convolutional networks to signed graphs.

5.1.1 Problem Statement

With the aforementioned notations and definitions presented in previous chapters and those summarized in Table 6.4, we can formally define the problem of signed network embedding as follows: *Given a signed network* $\mathcal{G} = (\mathcal{U}, \mathcal{E}^+, \mathcal{E}^-)$ *represented as an adjacency matrix* $\mathbf{A} \in \mathbb{R}^{n \times n}$*, we seek to discover a low-dimensional vector for each node as*

$$F : \mathbf{A} \rightarrow \mathbf{Z} \tag{5.1}$$

where F is a learned transformation function that maps the signed network's adjacency matrix $\mathbf{A}$ to a d-dimensional representation $\mathbf{Z} \in \mathbb{R}^{n \times d}$ for the n nodes of the signed network.

5.1.2 The Proposed Signed Graph Convolutional Network Framework

Graph convolutional neural networks have recently started to become more developed and have already shown their superiority in extracting and aggregating information from graph data. Their use cases spread over the vast field of network analysis, but one such domain that has shown to be very influential recently is network embedding. The discovery of representative low-dimensional features for each node in the network has previously shown to enhance many tasks from link prediction and node classification, to community detection and visualization. However, previous work has mostly focused on constructing GCNs for unsigned networks. Due to the inherent differences between unsigned and signed networks, this leaves a gap that we seek to bridge with the development of a signed graph convolutional network (SGCN).

Even with dedicated efforts towards the construction of a GCN specific to signed networks, there are still tremendous challenges we must face and overcome. The first of which is figuring out how we can correctly incorporate negative links during the aggregation process. We cannot simply treat the negative links the same as positive links, since their properties and semantic meaning vastly differ. The second challenge is how we can combine the two sets of links (i.e., positive and negative) into a single coherent model. This combination is essential because certainly positive and negative links interact in the network structure in complex ways and indeed are not segregated and isolated from each other.

In this work we propose to go to the roots of signed network analysis and utilize one of the most fundamental and indispensable signed social theories developed in social psychology, balance theory [30, 29]. We harness balance theory to construct a bridge to connect the gap between the ongoing development of GCNs for unsigned networks and signed networks. In the remainder of this section we will first briefly discuss a general GCN framework in the unsigned network setting and discuss the relationships of this framework to the structure of signed networks. Then we introduce balance theory and how we can use this signed social theory to correctly capture both positive and negative links simultaneously during the aggregation process. Thereafter, we present how to learn the parameters of our SGCN – first through the construction of an objective function designed to effectively learn the node representations in signed networks, and finally discussing the optimization procedure taken to optimize our proposed objective.

5.1.2.1 Unsigned Graph Convolutional Networks

Currently, most GCNs have a similar structure in that they utilize a convolutional operator that can share weights across all locations in the graph. The benefits of this neural network structure in graphs as compared to the cumbersome fully connected models are at least three fold: 1) it avoids the parameter explosion associated with fully connected layers (especially when handling larger graphs); 2) it allows for parameter sharing across the network to avoid overfitting; and 3) a single GCN is capable of handling as input graphs of varying structures and even sizes (in terms of the number of nodes and edges).

Typically the architecture of an unsigned GCN for learning node representations is of the form shown in Algorithm 5.1. In the process of generating the d^{out}-dimensional embedding matrix $\mathbf{Z} \in \mathbb{R}^{x \times d^{out}}$, they make use of the unsigned adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ and a feature matrix $\mathbf{X} \in \mathbb{R}^{n \times d^{in}}$, where d^{in} is the length of feature vector $\mathbf{x}_i$ for user u_i. The matrices $\mathbf{H}^{(l)} \in \mathbb{R}^{n \times d^{out}}$ for $l \in \{1, \ldots, L\}$ represent the hidden representations for each of the n nodes of the graph at each layer l of the GCN. On line 1 we set the initial representation $\mathbf{H}^{(0)}$ equal to $\mathbf{X}$ to ease the notations in the remainder of the algorithm. Then, on line 2 we loop updating the parameters of the GCN

until convergent. Inside this loop, for each update iteration we propagate the graph features through the L layers of the GCN using the unsigned adjacency matrix $\mathbf{A}$ and neighborhood aggregation function $f()$. Note that the function $f()$ is where the variations of GCNs primarily differ. Finally, after the model converges, the embedding is taken as the last layer's representation matrix $\mathbf{H}^{(L)}$.

Limitations of unsigned GCN for signed networks: Given the above discussion on the unsigned GCN framework, we note that in relation to signed networks this would be similar to applying the unsigned GCN on the positive only adjacency matrix $\mathbf{A}^+$ where $\mathbf{A}^+_{ij} = 1$ if there exists a positive link between users u_i and u_j, and 0 otherwise (i.e., when there exists either a negative link or no link between them). However, this would ignore the negative links.

Initially, our thoughts may lead to some naïve approaches of handling the negative links by either ignoring them, treating them the same or the negation of positive links, or separately applying the GCN framework to first the positive network, and then the negative network with finally combining them at the end stage. However, each of these methods is either based on incorrect assumptions or ignoring parts of the rich information awaiting to be extracted from the complex network structure of signed networks towards the learning of an advantageous low-dimensional representation. For example, trivially treating the negative links the same as the positive links would be an incorrect assumption, since negative links have been shown to have different principles and semantically represent vastly different meanings. Similarly, treating negative links as the negation of positive links is likely an incorrect assumption [67]. This leaves the last two initial thoughts of ignoring the negative links or applying an unsigned GCN separately on the positive only and negative only networks, but intuitively the first choice is certainly ignoring a large amount of information, and based on signed social theories [2, 29, 30], there exist complex relations between the positive and negative links that if extracted, can provide fruitful results [4, 24]. Therefore, next we will discuss one such signed social theory, balance theory [29, 30] and how we propose to harness it for capturing both the positive and negative links coherently together during the aggregation process.

Algorithm 5.1: Typical unsigned GCN framework.

Input: An unsigned network adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$; a feature matrix $\mathbf{X} \in \mathbb{R}^{n \times d^{in}}$; number of aggregation layers L; neighborhood aggregation function $f()$

Output: Low-dimensional representation matrix $\mathbf{Z} \in \mathbb{R}^{n \times d^{out}}$

$\mathbf{H}^{(0)} \leftarrow \mathbf{X}$
while not convergent **do**
  for $l \in \{0, \ldots, L-1\}$ **do**
    $\mathbf{H}^{(l+1)} \leftarrow f(\mathbf{H}^{(l)}, \mathbf{A})$
  Update GCN parameters based on $loss(\mathbf{H}^{(L)})$
$\mathbf{Z} \leftarrow \mathbf{H}^{(L)}$

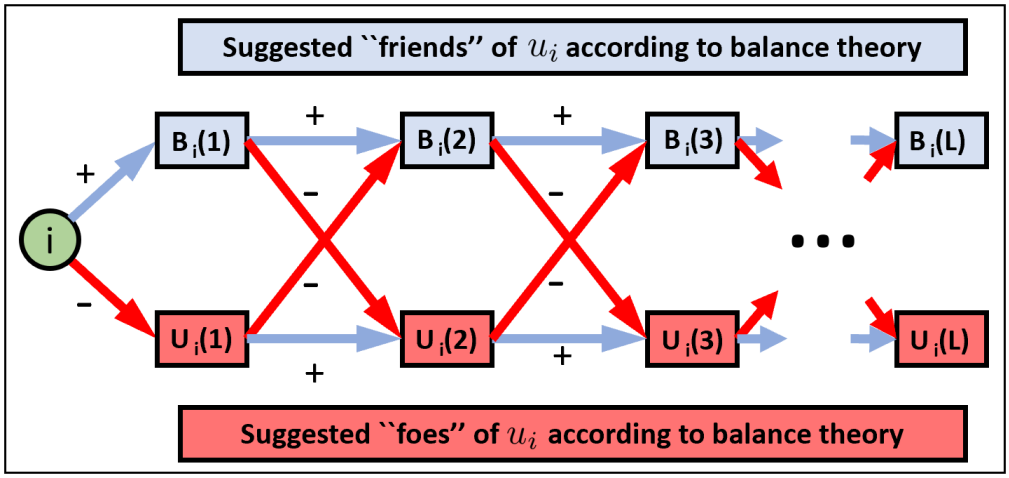


Figure 5.1: An illustration of the aggregation paths according to balanced and unbalanced paths.

5.1.2.2 Aggregation Paths with Positive and Negative Links

Balance theory dates back to the early seminal work in [29] and later generalized in [30] having a graph theoretical foundation. In general, balance theory implies "the friend of my friend is my friend" and "the enemy of my friend is my enemy". The theory classifies cycles in a signed network as being either *balanced* or *unbalanced*, where a balanced cycle consist of an even number of negative links while a cycle having an odd number of negative links is considered unbalanced. More details on balance theory and the four possible cycles that can be formed in a signed network can be found in Section 2.3.3.1, where in Figure 2.2 we can see that triangles (a) and (b) are balanced, while (c) and (d) are unbalanced. We propose to denote a *balanced path* as one that

consists of an even number of negative links, and similarly an *unbalanced path* being one that has an odd number of negative links. With these definitions, along with balance theory, we can see that if we had a path of length l from u_i to u_j that had an even number of negative links, then balance theory would suggest a positive link between u_i an u_j. An example of a balanced path can be seen in Figure 2.2 triangle (b), where the path of length two from u_i to u_j (through u_k) consists of two negative links and thus balance theory would suggest a positive link connecting u_i and u_j (to result in a balanced cycle between users u_i, u_k, and u_j). From the context of user u_i we would then place u_j into the set $B_i(2)$, which we use to denote the set of users that can be reached from user u_i along a balanced path of length 2. In the general case, users that can be reached from u_i along a balanced (or unbalanced) path of length l we place in the set $B_i(l)$ (or $U_i(l)$). In Figure 5.1 we provide an illustration of how all the signed paths of a given length would place users along paths from u_i into their respective sets. Note that the arrows are only used to aid the illustration and that our definition is based on the more general undirected setting.

Before continuing, let us define $\mathcal{N}_i^+$ to be the set of positive neighbors of a user u_i, i.e., $u_j \in \mathcal{N}_i^+$ if $\mathbf{A}_{ij} = 1$. We similarly denote the set of negative neighbors for user u_i as $\mathcal{N}_i^-$, where $u_j \in \mathcal{N}_i^-$ when $\mathbf{A}_{ij} = -1$. In Figure 5.1 we can see that when having a balanced path of length l from u_i to some user u_k (i.e., $u_k \in B_i(l)$), then all the positively linked neighbors of u_k (which we denoted as the set $\mathcal{N}_k^+$) would be placed in $B_i(l+1)$. This is because adding a positive link to a balanced path (i.e., a path consisting of an even number of negative links) still results in a balanced path, but just of additional length. Similarly when adding a negative link to a balanced path, we obtain an unbalanced path.

Another key observation from Figure 5.1 is how we can obtain the balanced and unbalanced sets $B_i(l+1)$ and $U_i(l+1)$ of length $l+1$, respectively for user u_i, from the sets $B_i(l)$ and $U_i(l)$ of length l. Below we provide a recursive definition for calculating the balanced and unbalanced sets from the perspective of user u_i as follows:

When $l = 1$

$$B_i(1) = \{u_j \mid u_j \in \mathcal{N}_i^+\}, \qquad U_i(1) = \{u_j \mid u_j \in \mathcal{N}_i^-\}$$

For $l > 1$

$$B_i(l+1) = \{u_j \mid u_k \in B_i(l) \text{ and } u_j \in \mathcal{N}_k^+\} \cup \{u_j \mid u_k \in U_i(l) \text{ and } u_j \in \mathcal{N}_k^-\} \tag{5.2}$$
$$U_i(l+1) = \{u_j \mid u_k \in U_i(l) \text{ and } u_j \in \mathcal{N}_k^+\} \cup \{u_j \mid u_k \in B_i(l) \text{ and } u_j \in \mathcal{N}_k^-\}$$

Given the above definition, we again note that the users in the balanced sets (which are reached along balanced paths) for a user u_i are those that either: 1) have a positive link directly to u_i; or 2) those that balance theory would suggest a positive link between them since they have an even number of negative links along the path connecting them. For the unbalanced sets the definition is similar, except with direct/suggested negative links. We note that these definitions, based upon balance theory, now allow us a principled way of aggregating and propagating information in signed networks using balanced and unbalanced paths/sets. Next we will propose aggregation functions for our signed GCN and follow with the rest of the details of our framework.

5.1.2.3 Signed Graph Convolutional Network

Before formalizing our signed graph convolutional network, we provide some insights and intuitions behind the construction in light of balanced and unbalanced sets and paths. The first insight is that in unsigned GCNs, when constructing a node representation, they aggregate their immediate local neighbors' information into a single representation and then through the use of multiple layers, propagate this in the network allowing a node to incorporate information from a multi-hop neighborhood (where the number layers in the GCN denotes the number of hops away information is being aggregated from). However, in signed networks, we cannot categorize all users the same. This is because semantically users that are connected through positive links to u_i are thought of as their "friends" while neighbors across negative links are their "foes". Similarly, for users in u_i's balanced sets, balance theory would suggest they are their "friends" (even though they are not directly linked) and those in u_i's unbalanced sets are suggested to be their "foes" based on this social theory. This phenomenon can be visualized in Figure 5.1. Therefore, we propose rather than maintaining a single representation for each node, we keep a representation of both their "friends"

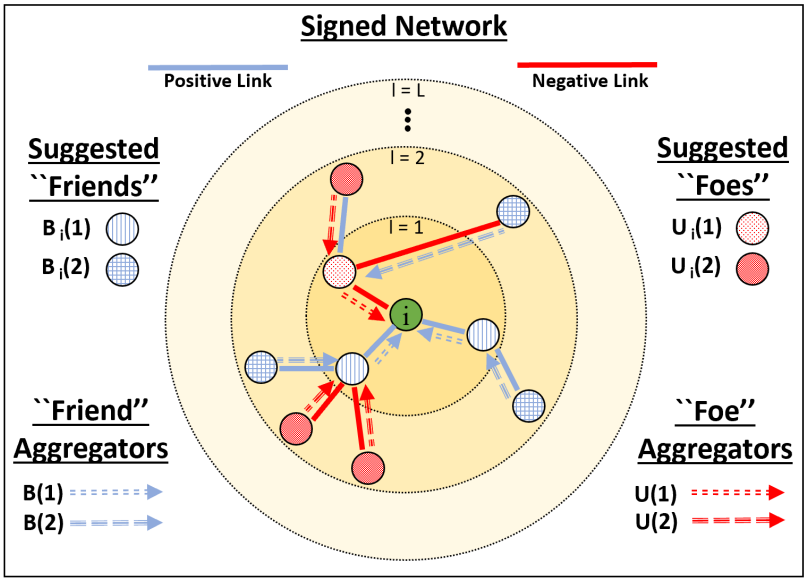


Figure 5.2: An illustration of how SGCN aggregates neighbor information in a signed network.

and "foes", which successfully incorporates both the positive and negative links and gives a more thorough representation of a given user.

In Figure 5.2 we provide an illustration of how we plan to aggregate and propagate information in a signed networks. Note that the circles labeled $l = 1, 2, \ldots, L$ are used to denote how many hops away the user is from u_i and simultaneously denotes at which layer in our signed GCN that user's information will be incorporated into the two learned representations for user u_i. We can observe that we could have a separate aggregator responsible for incorporating the information from each respective balanced and unbalanced sets. For example, in the first layer of Figure 5.2 we can see that the two positive neighbors of u_i will be incorporated into the level one "friend" representation through the use of aggregator $B(1)$. Similarly u_i's single negatively linked neighbor is used for learning the level one "foes" representation. Then, through the use of a second layer in our GCN, we can incorporate the two-hop neighbors. However, the crucial step here is that we must aggregate the information of these neighbors correctly to adhere to balance theory according to our defined balanced and unbalanced paths/sets. Therefore we employ a second set of aggregators, namely $B(2)$ and $U(2)$ which will help propagate the information from users in sets $B_i(2)$ and $U_i(2)$, respectively. Notice that just as shown in Figure 5.1 users being included by the $B(2)$

aggregator are the users who are along a path of two consecutive positive links, or two consecutive negative links, because they are both suggested as "friends" according to balance theory. On the other hand, aggregator $U(2)$ (which is gathering information from users in the set $U_i(2)$) seeks to utilize the information from users along paths that consist of one positive and one negative link (in either ordering, since both fall into set $U_i(2)$). Now we can more formally discuss the aggregation functions used by our proposed SGCN.

While aggregating and propagating information in our SGCN, we will maintain two representations at each layer, one for the corresponding balanced set of users (i.e., suggested "friends"), and one for the users in the respective unbalanced set (i.e., suggested "foes"). Similar to the unsigned GCN, we use $\mathbf{h}_i^{(0)} \in \mathbb{R}^{d^{in}}$ to represent the initial d^{in} node features for user u_i. Thus, for the first aggregation layer (i.e, when $l = 1$), we utilize the following:

$$\mathbf{h}_i^{B(1)} = \sigma\left(\mathbf{W}^{B(1)}\Big[\sum_{j \in \mathcal{N}_i^+} \frac{\mathbf{h}_j^{(0)}}{|\mathcal{N}_i^+|}, \mathbf{h}_i^{(0)}\Big]\right) \tag{5.3}$$

$$\mathbf{h}_i^{U(1)} = \sigma\left(\mathbf{W}^{U(1)}\Big[\sum_{k \in \mathcal{N}_i^-} \frac{\mathbf{h}_k^{(0)}}{|\mathcal{N}_i^-|}, \mathbf{h}_i^{(0)}\Big]\right) \tag{5.4}$$

where $\sigma()$ is a non-linear activation function, $\mathbf{W}^{B(1)}, \mathbf{W}^{U(1)} \in \mathbb{R}^{d^{out} \times 2d^{in}}$ are the linear transformation matrices responsible for the "friends" and "foes" coming from sets $B_i(1)$ and $U_i(1)$, respectively, and d^{out} is the length of the two internal hidden representations. More specifically, for determining the hidden representation $\mathbf{h}_i^{B(1)}$ we also concatenate the hidden representation of user u_i (i.e., $\mathbf{h}_i^{(0)}$) along with the mean of the users in set $B_i(1)$. In all subsequent layers, the aggregation is more complex, just as the definition of $B_i(l)$ an $U_i(l)$ were more complex when $l > 1$ in Eq. (5.2). This is similarly due to the cross linking of negative links as seen in Figure 5.1. The aggregations for $l > 1$ are defined as follows:

$$\mathbf{h}_i^{B(l)} = \sigma\left(\mathbf{W}^{B(l)}\Big[\sum_{j \in \mathcal{N}_i^+} \frac{\mathbf{h}_j^{B(l-1)}}{|\mathcal{N}_i^+|}, \sum_{k \in \mathcal{N}_i^-} \frac{\mathbf{h}_k^{U(l-1)}}{|\mathcal{N}_i^-|}, \mathbf{h}_i^{B(l-1)}\Big]\right) \tag{5.5}$$

$$\mathbf{h}_i^{U(l)} = \sigma\left(\mathbf{W}^{U(l)}\Big[\sum_{j\in\mathcal{N}_i^+}\frac{\mathbf{h}_j^{U(l-1)}}{|\mathcal{N}_i^+|}, \sum_{k\in\mathcal{N}_i^-}\frac{\mathbf{h}_k^{B(l-1)}}{|\mathcal{N}_i^-|}, \mathbf{h}_i^{U(l-1)}\Big]\right) \tag{5.6}$$

where $\mathbf{W}^{B(l)}, \mathbf{W}^{B(l)} \in \mathbb{R}^{d^{out}\times 3d^{out}}$ for $l > 1$. Note that we are utilizing the same logic here as when defining the sets $B_i(l)$ and $U_i(l)$. When gathering user u_i's "friend" representation (i.e., $\mathbf{h}_i^{B(l)}$) at layer l (when $l > 1$) it is based upon aggregating the "friend" representation at layer $(l-1)$ (i.e., $\mathbf{h}_k^{B(l-1)}$) for all positively linked neighbors $u_j \in \mathcal{N}_i^+$ while simultaneously collecting the average amongst the "foes" level $(l-1)$ (i.e, $\mathbf{h}_j^{U(l-1)}$) information from all negatively linked neighbors $u_k \in \mathcal{N}_i^-$. Thus, for the case when $l = 2$ we can see the "friend" representation is in fact gathering information from not only their direct friends (i.e., positively linked neighbors), but also (at the two hop level) friends of friends', and foes of foes'. Similarly, in the case of $l = 2$ our hidden representation $\mathbf{h}_i^{U(l)}$ (i.e., user u_i's "foes" representation), the first layer would have gathered direct negatively linked neighbor information, but in the second layer, we are gathering from u_i's friends' foes and their foes' friends.

With the above discussed aggregation methods, we can now present the entire framework of SGCN. First, in Algorithm 5.2 we discuss how to obtain the embedding for each user u_i in the signed network. On line 1, we set $\mathbf{h}_i^{(0)}$ equal to $\mathbf{x}_i$ for ease in defining the rest of the algorithm. Then on lines 2 through 5 we show the first layers aggregation process. Next, if the total number of layers in the SGCN is greater than one (i.e, $L > 1$), then we perform the subsequent aggregations according to the defined higher level aggregation functions we designed based on balance theory. Finally, on line 14 the last step is concatenating the two hidden representations for user u_i, namely $\mathbf{h}_i^{B(L)}$ and $\mathbf{h}_i^{U(L)}$ together into a single low-dimensional representation.

Next we design an objective function to learn the parameters of SGCN. The objective function for SGCN is based upon two components, both of which are based on the goal that we would like the representations to be able to understand the relationships between pairs of users in the signed network's embedded space. The first term incorporates an additional layer for performing a weighted multinomial logistic regression (MLG) classifier. Here we wish to classify whether a

Algorithm 5.2: Signed Graph Convolutional Network (SGCN) embedding generation.

Input: $\mathcal{G} = (\mathcal{U}, \mathcal{E}^+, \mathcal{E}^-)$; an initial seed node representation $\{\mathbf{x}_i, \forall u_i \in \mathcal{U}\}$; number of aggregation layers L; weight matrices $\mathbf{W}^{B(l)}$ and $\mathbf{W}^{U(l)}$, $\forall l \in \{1, \ldots, L\}$; non-linear function σ

Output: Low-dimensional representations $\mathbf{z}_i, \forall u_i \in \mathcal{U}$

$\mathbf{h}_i^{(0)} \leftarrow \mathbf{x}_i, \forall u_i \in \mathcal{U}$

for $u_i \in \mathcal{U}$ **do**

  $\mathbf{h}_i^{B(1)} \leftarrow \sigma\left(\mathbf{W}^{B(1)}\left[\sum_{j \in \mathcal{N}_i^+} \frac{\mathbf{h}_j^{(0)}}{|\mathcal{N}_i^+|}, \mathbf{h}_i^{(0)}\right]\right)$

  $\mathbf{h}_i^{U(1)} \leftarrow \sigma\left(\mathbf{W}^{U(1)}\left[\sum_{k \in \mathcal{N}_i^-} \frac{\mathbf{h}_k^{(0)}}{|\mathcal{N}_i^-|}, \mathbf{h}_i^{(0)}\right]\right)$

if L > 1 **then**

  for $l = 2 \ldots L$ **do**

    for $u_i \in \mathcal{U}$ **do**

      $\mathbf{h}_i^{B(l)} = \sigma\left(\mathbf{W}^{B(l)}\left[\sum_{j \in \mathcal{N}_i^+} \frac{\mathbf{h}_j^{B(l-1)}}{|\mathcal{N}_i^+|}, \sum_{k \in \mathcal{N}_i^-} \frac{\mathbf{h}_k^{U(l-1)}}{|\mathcal{N}_i^-|}, \mathbf{h}_i^{B(l-1)}\right]\right)$

      $\mathbf{h}_i^{U(l)} = \sigma\left(\mathbf{W}^{U(l)}\left[\sum_{j \in \mathcal{N}_i^+} \frac{\mathbf{h}_j^{U(l-1)}}{|\mathcal{N}_i^+|}, \sum_{k \in \mathcal{N}_i^-} \frac{\mathbf{h}_k^{B(l-1)}}{|\mathcal{N}_i^-|}, \mathbf{h}_i^{U(l-1)}\right]\right)$

$\mathbf{z}_i \leftarrow [\mathbf{h}_i^{B(L)}, \mathbf{h}_i^{U(L)}], \forall u_i \in \mathcal{U}$

pair of node embeddings are from users with a positive, negative, or no link between them. More specifically, we construct a mini-batch of users and then a set $\mathcal{M}$, which contains triplets of the form (u_i, u_j, s) which denotes the pair of users (u_i,u_j) along with $s \in \{+, -, ?\}$ for denoting whether there was a positive, negative, or no link between the pair of users. For input into the classifier, we use the final embeddings for users u_i and u_j concatenated together (i.e,. $[\mathbf{z}_i, \mathbf{z}_j]$). We use ω_s to denote the weight associated with class s. We introduce a second term that is founded on extended structural balance theory. This term is controlled by λ to balance the contribution towards the overall objective. The goal of this second term is to have positively linked users closer in the embedded space than the no link pairs, and the no link paired users should be closer than users

having a negative link between them. The overall objective is formalized in the following:

$$\mathcal{L}(\theta^W, \theta^{MLG}) = -\frac{1}{\mathcal{M}} \sum_{(u_i,u_j,s)\in\mathcal{M}} \omega_s \log \frac{\exp\left([\mathbf{z}_i, \mathbf{z}_j]\theta_s^{MLG}\right)}{\sum_{q\in\{+,-,?\}} \exp\left([\mathbf{z}_i, \mathbf{z}_j]\theta_q^{MLG}\right)}$$
$$+ \lambda \Bigg[\frac{1}{|\mathcal{M}_{(+,?)}|} \sum_{\substack{(u_i,u_j,u_k)\\ \in\mathcal{M}_{(+,?)}}} \max\left(0, (||\mathbf{z}_i - \mathbf{z}_j||_2^2 - ||\mathbf{z}_i - \mathbf{z}_k||_2^2)\right)$$
$$+ \frac{1}{|\mathcal{M}_{(-,?)}|} \sum_{\substack{(u_i,u_j,u_k)\\ \in\mathcal{M}_{(-,?)}}} \max\left(0, (||\mathbf{z}_i - \mathbf{z}_k||_2^2 - ||\mathbf{z}_i - \mathbf{z}_j||_2^2)\right) \Bigg]$$
$$+ Reg(\theta^W, \theta^{MLG}) \tag{5.7}$$

θ^W represents the weight matrices used in the layers of our SGCN, θ^{MLG} denotes the parameters of the MLG classifier, ω_s is used for the weight associated with the class s (with $s \in \{+, -, ?\}$ for the positive, negative, and no link classes), $\mathcal{M}_{(+,?)}$ and $\mathcal{M}_{(-,?)}$ are the sets for the pairs of positive and negatively linked users, respectively, where for every linked pair (u_i, u_j) we further sample another user u_k randomly (and different in each epoch) that has no link to u_i. The term $Reg(\theta^W, \theta^{MLG})$ we use for regularization on the parameters of our model. For updating the parameters, we utilize the same SGD style updating as presented in [75], since it has been show to effectively update the parameters of a GCN using a mini-batch setting (as compared to previous work such as in [22] that performed batch gradient descent).

5.1.3 Experiments

In this section, we experimentally evaluate the effectiveness of the proposed signed graph convolutional network (SGCN) in learning node representations. We seek to answer the following questions: (1) Is SGCN capable of learning meaningful low-dimensional representations? and (2) Does the introduction of balance theory into the aggregation process along with longer path information provide a performance increase in learning the node embeddings?

To address the first question, we conduct experiments to measure the learned embedding quality by performing the most fundamental signed network analysis task, namely link sign prediction [4],

and compare against the signed network embedding state-of-the-art baseline methods. To answer the second question, we investigate variants of our framework that do not exploit the longer paths (i.e., only performing a single aggregation step) or that do not make use of balance theory (i.e., the fundamental signed social network theory).

Experimental Settings: Next, we note the datasets used, the link sign prediction problem, and the metrics used for evaluation. For our study of learning representations using signed graph convolutional networks, we conduct our experiments on four real-world signed network datasets, i.e., Bitcoin-Alpha, Bitcoin-OTC, Slashdot, and Epinions. We note that for each of these datasets we perform our experiments on the undirected signed networks and have further filtered out users randomly from the two larger networks (Slashdot and Epinions) that had very few links. We summarize the new variants of the Slashdot and Epinions datasets in Table 5.2 with some basic statistics, while Bitcoin-Alpha and Bitcoin-OTC are the same as previously presented in Table 4.2.

The problem of predicting the signs of links [4] is that given a set of existing links in the signed network that had been held out of the training set, we wish to predict their signs being positive or negative between those pairs of users. Thus, a binary classifier is used to predict the sign based on a set of input features from the pair of users (more specifically we employ a logistic regression model). In our case we concatenate the final embeddings of the two users together as the set of features. The model is trained using the labeled edges from the training data. For evaluation, since the positive and negative links are unbalanced (i.e., there are many more positive links than negative links), we utilize both F1 and Area Under the receiver operating characteristic Curve (AUC). We note that higher F1 and AUC both mean better performance. For each dataset, we randomly choose 20% of the data as test, and the remaining 80% as training. Note that we used a grid search along with cross validation on the training data to tune the hyperparameters of our model.

5.1.3.1 Performance Comparison

Here we present some existing state-of-the-art signed network embeddings methods such that we can study the effectiveness of our signed GCN (SGCN) in learning node representations in signed

networks. For succinctness we do not include unsigned methods since previous signed network embedding work has shown their superiority over the non-dedicated efforts towards signed network embeddings. The baselines are as follows:

- Signed Spectral Embedding (SSE) [153]: A spectral clustering algorithm based on the proposed signed version of the Laplacian matrix. We utilize the top-d^{out} eigenvectors corresponding to the smallest eigenvalues as the embedding vectors for each node.
- SiNE [87]: This method is a deep learning framework that utilized extended structural balance theory.
- SIDE [89]: A random walk based method, utilizing balance theory, is used to obtain indirect connections for a likelihood formulation.

Furthermore, we propose to evaluate the following two variants of our model:

- SGCN-1: This method only makes use of the first single aggregation layer and therefore only separates the positive from the negative links (i.e, does not yet make use of balance theory and our defined balanced paths).
- SGCN-1+: This method similar to SGCN-1 does not make use of balance theory, instead it performs the naïve aggregation of the first layer, but twice. In other words, the final representation for each user is based on propagating information along the positive links twice, and the negative links twice, separately.

Some final notes are the following: 1) in our experiments we do not have node attributes, therefore instead we use the final embedding of the SSE model as the input feature matrix (i.e., X) to all our SGCN variants; 2) for all embedding methods we fixed the final low-dimensional representation to be 64; 3) We used the authors released code for SiNE[3] and use their suggested hyperparameters [87] for our experiments; 4) For SIDE, we use the authors implementation[4] and

[3]http://www.public.asu.edu/ swang187/codes/SiNE.zip

[4]https://datalab.snu.ac.kr/side/resources/side.zip

Table 5.2: Statistics of two signed network dataset variants for SGCN.

Network	# Users	# Positive Links	# Negative Links
Slashdot	33,586	295,201	100,802
Epinions	16,992	276,309	50,918

Table 5.3: Link sign prediction results with AUC.

Embedding Method	Bitcoin-Alpha	Bitcoin-OTC	Slashdot	Epinions
SSE	0.764	0.803	0.769	0.822
SiNE	0.778	0.814	0.792	0.849
SIDE	0.630	0.618	0.547	0.571
SGCN-1	0.780	0.818	0.784	0.663
SGCN-1+	0.785	0.817	**0.804**	0.722
SGCN-2	**0.796**	**0.823**	**0.804**	**0.864**

the suggested hyperparameter settings from [89], but for the unsuggested parameters we used a grid search around their code's default settings; and 5) for our models we set $\lambda = 5$ and the "friend" and "foe" hidden representations were each set to 32, such that the final embeddings were of size 64.

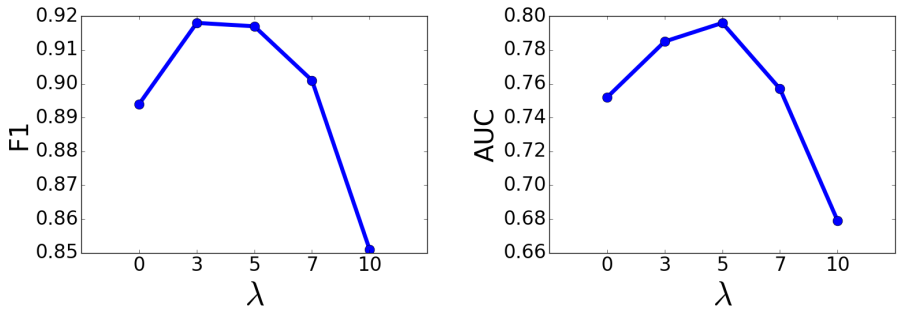
Table 5.4: Link sign prediction results with F1.

Embedding Method	Bitcoin-Alpha	Bitcoin-OTC	Slashdot	Epinions
SSE	0.898	0.923	0.820	0.901
SiNE	0.888	0.878	0.854	0.914
SIDE	0.738	0.750	0.646	0.711
SGCN-1	0.910	0.918	0.853	0.851
SGCN-1+	0.912	0.923	**0.865**	0.893
SGCN-2	**0.917**	**0.925**	0.864	**0.933**

Comparison Results:

The comparison results in terms of AUC and F1 are demonstrated in Tables 5.3 and 5.4, respectively. For the tables, we make the following observations:

- SGCN-1 with only one step aggregation from positive and negative links obtains comparable performance with the best performance from the baselines. This observation suggests that it



(a) SGCN-2 (F1)

(b) SGCN-2 (AUC)

Figure 5.3: Parameter sensitivity when varying the parameter λ on the Bitcoin-Alpha dataset.

is necessary to separate positive and negative links.

- SGCN-1+ outperforms SGCN-1. The results indicate that propagating multiple steps during the aggregation can help improve the performance.
- Most of the time, SGCN-2 outperforms SGCN-1 and SGCN-1+. Aggregation following the longer balance and unbalanced paths can boost the performance.

5.1.3.2 Parameter Analysis

The proposed signed GCN has one major hyperparameter, λ (besides the number of layers and the aggregation types which we have already investigated with our variants of SGCN). The parameter λ is used to control the balance between the two terms in our objective function as given in Eq. (5.7). More specifically, the first term introduced the multinomial logistic regression term in an attempt to guide the learned node embedding to be separable such that pairs of user that have positive, negative, and no link can be positioned such that the classifier can distinguish their relationship. The second term we utilized for discovering node embeddings that adhere to extended structural balance theory [154]. With its contribution controlled by λ, this term forced pairs of users that have positive links to be closer in the low-dimensional embedding space than to other users they had no link with, and further also sought to have users with negative links pulled further apart by wanting no linked pairs closer together than the negative pairs.

In Figures 5.3(a) and 5.3(b) we report the results when varying λ for one of the signed network datasets, namely Bitcoin-Alpha. We do not show results of other settings since we can have similar observations. As we can see from these two figures $\lambda = 5$ seems to be a good balance between the AUC and F1 performance. The second observation is that when setting λ equal to zero we have a drastic decrease in performance. Note that we saw similar results across all datasets in that the contribution of the second term, based on balance theory, was able to provide an improvement.

5.2 Role-based Signed Network Embedding

Most existing network embedding methods have been designed for networks with only a single edge type [155] and where relations between two nodes implies closeness. Hence, they primarily try to encode an unsigned network in a way that neighboring nodes are closer in the embedding space [155]. However, real-world networks might have more than one link type, i.e., a network can have K types of links where each type represents a different quality of relation between the nodes. Signed networks are an important class of such networks, having two types of links: positive and negative [55, 7].

A variety of social media sites, such as Amazon, Wikipedia, and Epinions can be represented as a signed network where positive signs represent trust, agreement or friendship while negative ones may show distrust, disagreement or enmity. The underlying principles of signed networks can be quite different from those of classic networks due to having both positive and negative links. Therefore, network embedding for signed networks cannot be carried out by simply applying classic embedding models. While embedding of signed networks is challenging, it has the potential to greatly advance network analysis tasks such as link/sign prediction [87].

Recently, signed network embedding has attracted increasing attention [156, 157, 158, 118]. Similar to many embedding models for unsigned networks, these models try to embed the network through finding similarities between nodes assuming connecting paths represent closeness. However, in signed networks path-based similarities are challenging since signed paths can indicate either closeness or distantness. Existing works [118, 156, 157] solve this challenge by relying on

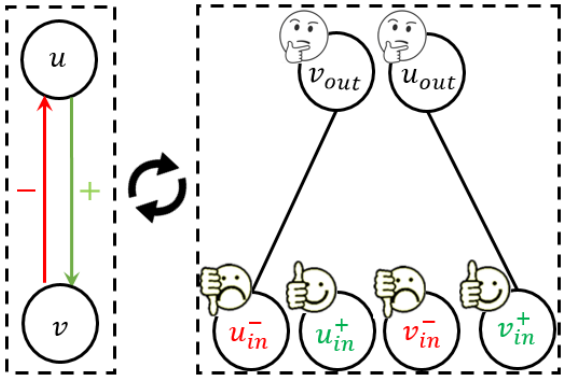


Figure 5.4: Transformation of a signed network with two nodes to an unsigned bipartite network of role-nodes.

two signed social theories, namely balance theory [29, 30] and status theory [2]. However, the general architecture on the way they define similarity between two nodes and the use of social theories is associated with two major challenges. First, social theories are incomplete in explaining signed network structure, so models built on them are affected and sometimes result in lower quality embeddings and depend how closely the networks align to these theories. Second, classic embedding models aim to capture presence/absence of links while existing signed embedding models only use two of the possible interaction states: positive and negative links. Thus, since they ignore link absence (the third interaction state), they can not reconstruct the presence/absence of links well, resulting in low performance in link prediction.

To address these shortcomings, we lay out a new perspective for network embedding denoted as network transformation based embedding: if embedding the original network is challenging, it can be transformed into another network for which the embedding task has lower complexity. The transformation can be done by mapping each node in the original network to multiple nodes in the transformed network. Next, the transformed network can be embedded. Finally, the embedding vectors obtained from the transformed network can be aggregated to encode the original network. More specifically, to embed signed networks, we introduce a ROle based Signed network Embedding (ROSE) that bypasses the aforementioned challenges. The underlying idea is to transform the signed network into a bipartite network where each node takes both "user" and "item" roles for which they are the giver and receiver of signed links, respectively. Therefore, each node of a signed network

can be modeled by a set of roles, denoted as role-nodes, where the relations between role-nodes can be fully captured using unsigned links. Then, ultimately this transformed network can utilize the state of the art unsigned embedding technique. Figure 5.4 is a toy example of the transformation process

Each role-node captures a certain aspect of a node in the original network. Hence, a comprehensive embedding of a node can be obtained by aggregating the embeddings of their role-nodes. We introduce two aggregation methods, denoted as fixed aggregation and target-aware aggregation. The fixed aggregation simply concatenates all the role-node embeddings together. The target-aware aggregation is based on a recent deep learning based recommendation model that introduced a model for target dependent encodings of users [159]. Based on this idea, we propose an attention mechanism based model to aggregate the embeddings of role-nodes in which attendance weights are obtained with respect to the target entity. To the best of our knowledge, this is the first work to build target aware embeddings of nodes in a signed network.

5.2.1 Problem Statement

Let a graph be defined as $G(V, E)$ with a link type mapping function $\varphi : E \rightarrow A$ where V represents the nodes, E represents the links, and each link $e \in E$ belongs to a link type $\varphi(e) \in A = +, -$. In unsigned networks A has only one values, but signed networks have two values: positive and negative. Given the graph G, the task of node encoding is to learn a function $f : V \longrightarrow |d|$ that maps each node v to a d-dimensional embedding vector which can be parametrized by the Matrix W with size $|V| * d$.

5.2.1.1 Unsigned Network Embedding

Embedding models can be described as an encoding-decoding framework [160] having four components: 1) A pairwise node similarity function. 2) An encoder function to create embeddings from the similarity function. 3) A decoding function to recover the pairwise node similarities from their embeddings. 4) A loss function that evaluates the reconstructed similarity values. The

primary difference in the literature is how the embedding methods define node similarity. However, the shared principle in unsigned similarity functions is that an unsigned path between two nodes indicates their closeness (e.g., as in [161, 150, 149]).

5.2.1.2 Signed Network Embedding

The unsigned similarity functions cannot be directly applied to signed networks because negative edges do not represent closeness. Thus, the challenge in embedding signed networks is how to involve negative edges without hindering positive proximity. Existing methods have sought to capture node similarity using the paths between them at length one, two, or higher order paths.

Single-length paths: A trivial approach to embed signed networks while involving negative links is to embed a node based on its immediate neighbors [162, 88]. This, however, has limited effectiveness because it cannot capture the higher order proximities between nodes. However, capturing global structures in signed networks is challenging, e.g., given a path containing m positive links and n negative links, how can the similarity of the nodes can be defined? Does the path indicate closeness or distantness between the nodes? To aid this previous works used the two signed social theories, namely balance and status.

Paths of length two: As the above social theories are formed on triangle structures, existing methods [87, 157] have used them to determine whether the signed path is representing closeness or distantness between two nodes. However, these methods do not go beyond paths of length two. As such, they have limited power in capturing global structures.

Longer paths: More recent work has tried to capture longer cycle paths in their embedding process mainly by relying on the extended version of social theories. [156, 163] both run a random walk on signed networks similar to node2vec algorithm, [133] applied to node relevance and personalized ranking [164] using random walks. Then, a graph convolutional network method for embedding signed networks has been introduced which relies on balance theory [118].

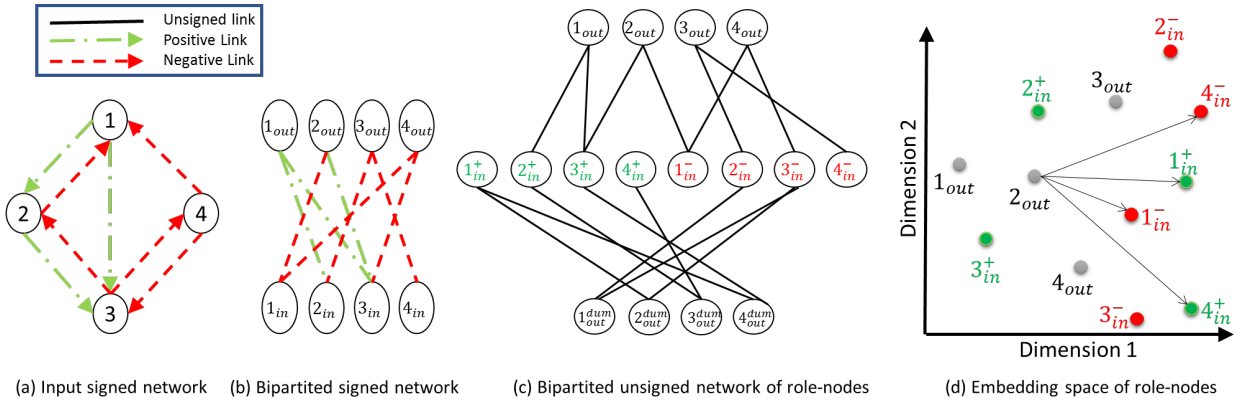
In all, the shared strategy of all these works is that they embed nodes by analyzing the paths between them. If a path indicates closeness, they embed the nodes closer and distance them

otherwise. However, to interpret if a path indicates closeness or distantness, they exploit some strong assumptions which naturally induce noise to the embedding process. Also, this strategy does not use a principled way to distant nodes based on the absence of links/paths between them, i.e., it only focuses on capturing positive/negative paths.

5.2.2 Role-based Signed Network Embedding

In this section, we describe the structure of ROSE. Based on the drawbacks of the previous works, we outline the following requirements: an effective universal network embedding model should be able to 1) capture the higher order connectivity between nodes, 2) take into account the link labels as well as the link structures (presence or absence of links), and 3) do not make assumptions about the origin of the network.

Network transformation based embedding: To address the requirements, we introduce the general notion of network transformation based embedding. Rather than directly finding the similarities of the nodes in the input network, it can be transformed to another network in which we do not encounter the embedding challenges present in the input network. One possible way to do transformation is to define different roles for a node, denoted as "role-nodes", and build a network of role-nodes in a way that the similarities between role-nodes can be determined by adopting the classic similarity functions. Since each role-node captures a certain aspect of an original node, the embedding vector of a target node can be derived by aggregating the embeddings of the corresponding role-node. In sum, a network transformation based embedding model can be described in three main steps: 1) Network transformation. 2) Embedding the transformed network. 3) Embedding the original network by aggregating the embeddings of the transformed network. By relying on the general idea of network transformation, we propose ROSE. In the following, ROSE is described based on the aforementioned three-step architecture. We then illustrate how ROSE addresses the requirements of the problem.



(a) Input signed network (b) Bipartited signed network (c) Bipartited unsigned network of role-nodes (d) Embedding space of role-nodes

Figure 5.5: Transformation process of the input signed network to the network of role-node.

5.2.2.1 Network Transformation

The way the role nodes are defined is fundamental to the effectiveness of ROSE. We aim to define the transformation such that similarities between role-nodes can be obtained using classical methods. Note that there can be multiple ways to define the transformation process. Various embedding techniques have been introduced based on different similarity measures. Similarly, different embedding methods can be developed based on the idea of transformation based embedding by creating different transformations. Our transformation idea is inspired by recommender systems. Traditionally, user-item interactions in recommender systems are modeled by a user-item bipartite network. A signed all-to-all connected network can also be viewed as a bipartite network where each node plays a "user" role for the links it creates and plays an "item" role for the links it receives. Based on this analogy, we capture user/item roles of a node separately through a transformation process.

Step 1: Transformation to a bipartite network. Based on user-item analogy, each node is mapped to two role-nodes, i.e., the node u is mapped to the role-nodes u_{out} and u_{in} where a link from u to v in the original network is modeled as an undirected link between u_{out} and v_{in}. As it can be seen in Figure 5.5(a), the input network is transformed to a signed bipartite network [165] with two types of nodes "in" and "out". However, applying a classic similarity measure on the transformed network is still a challenge due to presence of positive and negative links.

Step 2: Transformation to an unsigned network. We transform the network into an unsigned

network by defining new role-nodes. A role-node of type "in" v_{in} is mapped into two role-nodes: v_{in}^{+} (or v_{in}^{-}) representing its role when positive (or negative) links point toward it. Accordingly, a link from u_{out} to v_{in} with label l is modeled as an unlabeled, undirected link between u_{out} and v_{in}^{l}. This enables us to use the well-established similarity functions to determine the similarities of role-nodes.

Step 3: Augmenting the network. Our strategy is to encode the original network by embedding the transformed network. However, some of the role-nodes may have a very low degree. In particular, role-nodes of type "in-" tend to have a very low degree due to the fact that the number of negative links is often under-represented compared to positive links. According to our results, this can dramatically hinder the accurate embedding of such role-nodes. To solve this, we leverage implicit knowledge about the problem domain. If node u_{out} has connections towards both v_{in}^{+} and w_{in}^{-}, not only it reflects the adjacency between these two role-nodes but also it implies dependence between their opposite role-nodes: v_{in}^{-} and w_{in}^{+}. To bring this knowledge into our embedding process, which can attenuate the sparsity problem, we augment our unsigned network with a set of dummy nodes of type "out", i.e., for each node of type "out" in the unsigned network with the set of connections $\{v1_{in}^{l_1}, v2_{in}^{l_2}, ..., vn_{in}^{l_m}\}$, we add a node of type "out-dummy" with the set of connections $\{v1_{in}^{l'_1}, v2_{in}^{l'_2}, ..., vn_{in}^{l'_m}\}$ where l' is the inverse of l, i.e., if l is "-", l' is "+" and vice versa.

Summary of transformation: In sum, $G(V, E)$ is transformed to a bipartite unsigned graph $G_u(V_u, E_u)$ where $|V| = 4|V_u|$, $|E| = 2|E_u|$, i.e., a node $u \in V$ is mapped to four role-nodes in V_r: 1) u^{out} which initiates a link, 2) u_{out}^{dummy} which initiates a dummy link, 3) u_{in}^{+} which receives a positive link, and 4) u_{in}^{-} which receives a negative link. And a link $e_{u,v}$ with the label l is transformed to links $e_{u_{out},v_{in}^{l}}$ and $e_{u_{out}^{dummy},v_{in}^{l'}}$. Figure 5.5 depicts a toy example describing the process. It should be noted that the transformation is lossless, i.e., $G(V, E)$ can be fully reconstructed from $G_u(V_u, E_u)$.

5.2.2.2 Embedding the Original Network

Analogous to unsigned networks, the links between role-nodes indicates their closeness. Hence, a classic embedding model can be used to embed role-nodes. We employ node2vec [150]. Note that more advanced embedding models [155, 166] can be used/designed for this purpose. However, in this paper, our focus is on introducing the structure of ROSE. Once having the embeddings of role-nodes, a node's embedding is created by aggregating the embeddings of the corresponding role-nodes. In the following, we introduce two different aggregation models.

Fixed Aggregation: Each of the roles represents a certain perspective/role of a node in the original network. In general, if there are multiple representations of an entity, we can concatenate them or linearly combine them to build a unified representation. Accordingly, a straight forward way to build a comprehensive and unified embedding of a node is to concatenate the embedding vectors of the corresponding role-nodes. As such, the fixed representation of node u can be defined as $W_u = W_{u_{out}} || W_{u^+_{in}} || W_{u^-_{in}}$ in which $||$ represents concatenation. Note that dummy role-nodes are not used in the aggregation process. In fact, "out-dummy" role nodes are inverses of the role nodes of type "out" and do not add extra knowledge about the representations of the original nodes.

Target Aware Aggregation: One important application of graph embedding is to use the embedding vectors to predict the pairwise interactions of nodes, e.g., link prediction. Intuitively, for such tasks, it is more accurate to encode the given initiator node with respect to the target entity. In fact, the idea of target-aware profiling is the basis for most of the recommendation models. For example, in item-based collaborative filtering, to predict the rating of a user towards an item, her previous ratings are aggregated in a weighted way because not all of her interactions are equally important in reflecting her taste towards the item [167]. Typically, the weight of a rating is determined based on the similarity of the corresponding item to the target item [159].

Inspired by recommender systems, we introduce a target-aware embedding technique by proposing a target-aware aggregation model. To the best of our knowledge, the existing techniques build only fixed embeddings. In our framework, intuitively predicting the pairwise interaction from u to v depends on the "out" role node of u and "in" role nodes of v. We propose that "out" role-node of

u can be embedded according to v which is denoted by $W_{u^v_{out}}$. Having $W_{u^v_{out}}$, the target dependent embedding of u w.r.t. v is defined as $W^v_u = W^v_{u_{out}}||W_u$. Indeed, we concatenate the fixed embedding of u with a component that depends on the target entity to build its target-aware embedding. To build $W_{u^v_{out}}$, we design an attention mechanism. We suggest that $W_{u^v_{out}}$ can be obtained by attending to the neighbors of u_{out} based on their relevancy to the target entity v. More formally, $W_{u^v_{out}}$ is defined as:

$$W_{u^v_{out}} = \sum_{s^l_{in} \in N(u_{out})} [e(s^l_{in}, v) W_{s^l_{in}}],$$

where $e(s^l_{in}, v)$ is the importance weight of s_{in} w.r.t. v and $N(u_{out})$ is the set of role-nodes connected to u_{out}. To estimate the importance weights, we introduce an unsupervised attention model. The intuition behind the model is that the "in" role-nodes of two nodes are more related if the are more tightly connected in the network. Note that we do not take into account the labels of connections to find the relevancy of two nodes. To systematically implement the idea, given the target signed network, we assume the links are unsigned and transform it to a bipartite network where the obtained network has two types of role-nodes: "in" and "out". Next, the transformed network is embedded using node2vec. Finally, $att(s^l_{in}, v)$ is defined as follows:

$$att(s^l_{in}, v) = \sigma(W_{s_{in}}, W_{v_{in}}) = \frac{1}{1 + exp(-W_{s_{in}}.W_{v_{in}})}.$$

In fact, this weight determines how tightly s and v are connected in terms of the nodes that rated them regardless of the rating values.

5.2.2.3 Model Justification

We outline three main requirements for an effective signed embedding technique. ROSE fulfills the requirements. 1) Unlike existing models, ROSE does not rely on any assumption about the origin of the network. 2) To obtain the embedding of role-nodes, we use a random walk based model to ensure the obtained embeddings capture the higher order proximities. 3) The model preserves both link labels and link structures. To address the sign/link prediction tasks the embeddings from ROSE can be fed to a nonlinear function trained by a method like MLP to determine the target

interaction state. In fact, the embeddings of role-nodes contain major patterns that can aid to fully reconstruct the graph. We encode the role-nodes in a way that if a link with label l exists from u to v, $W_{u_{out}}$ has higher proximity to $W_{v^l_{in}}$ than $W_{v^{l'}_{in}}$. And if there is no link from from u to v, $W_{u_{out}}$ is expected to have low proximities to both $W_{v^l_{in}}$ and $W_{v^{l'}_{in}}$. As such, having a function to find the similarities of embeddings, the label of the link from u to v is expected to be l if the proximity of $W_{u_{out}}$ to $W_{v^l_{in}}$ is greater than it's proximity to $W_{v^{l'}_{in}}$. And if $W_{u_{out}}$ has low similarity to both $W_{v^+_{in}}$ and $W_{v^-_{in}}$ it indicates absence of link. Moreover, we observe other interesting patterns in our experiments, which will be discussed in the experiments section. In addition to addressing the requirements of the problem, the proposed framework creates an avenue to make a connection between the recommender systems and signed networks contexts. Lastly, the proposed model is quite generalizable. Since the model does not rely on any assumption specific for signed networks, it can be generalized for networks with multi-type of links.

5.2.3 Experiments

We conducted experiments to verify the effectiveness of the proposed framework and the ideas behind the model. The experiments are focused on answering two key questions:

- How do the proposed embedding frameworks perform when compared to the state of the art models in terms of link-label prediction and link prediction tasks?
- What is the interpretation of the embeddings obtained from the network of role-nodes?

Datasets: Three real-world datasets were used in the experiments: Epinions [3], WikiElection [168], and Slashdot [169] which have been used in previous works [55]. *WikiElection:* In Wikipedia election, users may give positive or negative votes for the promotion of other users as administrator. WikiElection dataset is the signed network obtained from users' votes for elections of administrators. *Epinions:* Epinions was an online product review site. Users can express positive or negative votes to other users regarding the trustworthiness of their reviews; this dataset is from the positive/negative votes between users. *Slashdot*: Slashdot dataset is also obtained from an online service (technology

Table 5.5: Statistics of three signed network dataset variants for ROSE.

	Nodes	Edges	Positive Edges	Negative Edges
WikiElection	7118	103747	78.7%	21.2%
Epinions	119217	841200	85.0%	15.0%
Slashdot	82144	549202	77.4%	22.6%

news website) where users can share comments and flag each other as friend or foe. The flags indicate approval or disapproval of comments. Analogous to Epinions, Slashdot dataset models the interactions of users using a signed network. Statistics of the datasets are given in Table 5.5.

5.2.3.1 Performance Comparison

In this experiment, we compared the performance of ROSE with four recently introduced signed network embedding models on two tasks: sign prediction and link prediction. Moreover, we compared the target dependent variant of ROSE denoted as ROSE-UAT with the fixed variant represented by ROSE on both of the tasks. AUC (Area Under the Curve) was used as the evaluation metric.

The following is the list of the models used in this experiment. SIDE is a random walk based approach that aims to capture global structures in the embedding process [156]. BESIDE aims to use both balance and status theories in a complementary manner to encode signed networks[157]. SiNE is a deep learning based framework that performs based on undirected networks. The main principle behind the model is that "users should sit closer to their friends than their foes" [87]. SIGNet is also a random-walk based model that maintains structural balance using targeted negative sampling. [88].

Evaluation: In our datasets, the number of negative edges is much smaller than the positive links. Thus, comparing methods based on their original test set accuracy could be misleading, especially for sign prediction. Thus, analogous to previous works [4, 157], we balanced the datasets by randomly removing positive links and used 5-fold cross-validation for our experiments. The baselines were evaluated based on the source-codes released by their authors. The embedding dimension for all of the models was set to 30.

Table 5.6: AUC of the proposed model (ROSE) and the baseline methods on the WikiElection, Slashdot and Epinions datasets.

	Sign Prediction			Link Prediction		
Model	WikiElection	Slashdot	Epinions	WikiElection	Slashdot	Epinions
SIDE	0.7986	0.8815	0.8672	0.9184	0.9342	0.9314
BESIDE	0.8953	0.9012	0.9342	0.9092	0.9265	0.9397
SiNE	0.8632	0.8680	0.8543	0.5833	0.5983	0.6488
SIGNET	0.8943	0.8997	0.9181	0.9099	0.8862	0.9205
ROSE	0.9091	0.9082	0.9533	0.9418	0.9357	0.9403
ROSE-UAT	**0.9116**	**0.9095**	**0.9547**	**0.9426**	**0.9391**	**0.9444**

Sign Prediction: The problem of sign prediction [3, 124] is the major task that has been used to evaluate encoding models in previous works. The Table 5.6 shows the AUC of the models on three datasets. First, we notice ROSE-UAT and ROSE outperform all the baselines. For example, ROSE-UAT outperforms BESIDE by 1.6%, 0.8%, and 2% in terms of AUC on WikiElection, Slashdot, and Epinions datasets, respectively. The higher accuracy of ROSE can be attributed to its effectiveness in addressing the requirements of the problem. Additionally, we observe ROSE-UAT perform better than ROSE. In fact, encoding the nodes with respect to a target entity helps to better analyze the interactions of the node and the entity.

Link Prediction: Although link prediction is an important task in network mining [54], previous works have not evaluated their models based on link prediction task. To evaluate the models for the link prediction task, we first fed the training graph to the models and obtained the node encodings. Next, we created training and test sets. Each data instance in the training/test sets is the concatenation of the encoding vectors of a node pair (W_u, W_v) and the label of the instance is 1 if there is a link from u to v and 0 otherwise. In both training and test sets, 50% of instances have label 1 and 50% of them have label 0. The node pairs with 0 label were randomly selected. The training set obtained from each embedding method was fed to a multi-layer perceptron classifier, and the AUC of the trained model was obtained based on the test set. Table 5.6 shows the results of the experiments. As it can be seen, ROSE has superior performance than the baseline models on all but one of the datasets since the model systematically differentiates the three different interaction-states between nodes in its embedding process.

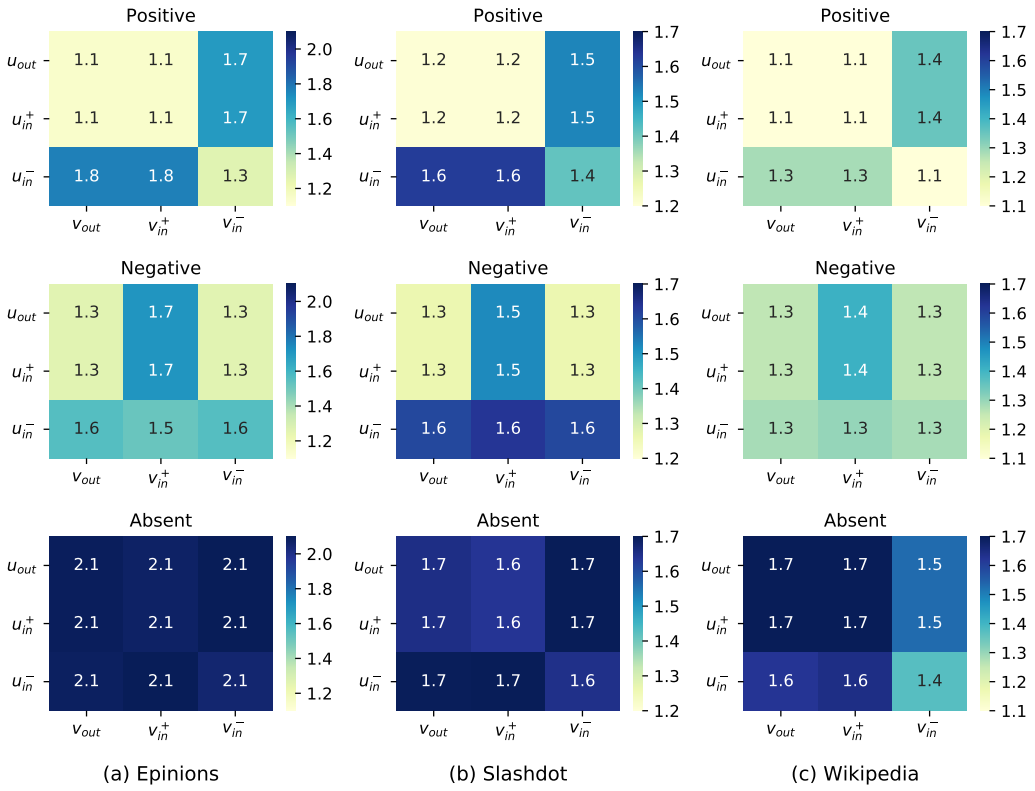


Figure 5.6: The average pairwise distance of the encoding vectors of the role-nodes of a node pair (u, v) for different interaction-types between them: positive link, negative link, and absence of a link.

5.2.3.2 Interpretation of the Encodings of Role-nodes:

Given nodes u as initiator and v as receiver, three interaction-states can be considered between them: absence of link, positive link and negative link. We introduced the major pattern extracted from the distance/similarity of the encoding vectors of u and v that can aid to determine the interaction type between them. We investigated the existence of such patterns. Figure 5.6, shows the average distance of different encoding components of a node pair (u, v) as a function of the interaction-state between them. For example, in Slashdot dataset if the link is positive the average distance of $W_{u_{out}}$ from $W_{v_{in}^{+}}$ denoted as $d_{avg}(u_{out}, v_{in}^{+})$ is 1.2. The average distances of the components are consistent with the introduced major patterns. If the link from u to v is positive, $d(u_{out}, v_{in}^{+})$ is expected to be smaller than $d(u_{out}, v_{in}^{-})$. For example, for a positive link in Epinions, $d_{avg}(u_{out}, v_{in}^{+})$ is 1.1 while $d_{avg}(u_{out}, v_{in}^{-})$ is 1.7. Moreover, we observe that if there is a link

from u to v, $d_{avg}(u_{out}, v_{in}^{-}) + d_{avg}(u_{out}, v_{in}^{+})$ is smaller than the case when there is no link. For example, in Slashdot, $d_{avg}(u_{out}, v_{in}^{-}) + d_{avg}(u_{out}, v_{in}^{+})$ is 3.3 if there is no link from u to v and it is 2.75 if there is a link. In addition to the major pattern, we observe four other patterns. We name these patterns as implicit patterns because our model has not targeted to extract them.

First, if the sign of the link from u to v is positive, *similar* nodes rate them similarly and if it is negative, *similar* nodes rate them with different signs. In fact, the smaller distance between the embeddings of the role-nodes of type "in+" and of type "in-" of two nodes means that they were rated by similar nodes similarly. For example, in Epinions dataset $d_{avg}(u_{in}^{+}, v_{in}^{+})$ and $d_{avg}(u_{in}^{-}, v_{in}^{-})$ are 1.1 and 1.3 respectively when the sign of the edge from u to v is positive while those distances are 1.7 and 1.6 respectively when the edge sign is negative. It can be said this pattern is aligned with balance theory, i.e., the triangle structures described in balance theory can be regarded as a special case of this pattern [170]. Second, u and v rate similar nodes more similarly when there is a positive link between them than when there is a negative a link connecting them. The smaller distance values between the embeddings of the role-nodes of type "out" of two nodes indicate that they have rated similar nodes similarly. As it can be seen, $d_{avg}(u_{out}, v_{out})$ is smaller when there is a positive link from u to v. Again, balance theory can be regarded as the special case of this pattern. Third, the signs of the link between two nodes in different directions are correlated. $d_{avg}(v_{out}, u_{in}^{+})$ - $d_{avg}(v_{out}, u_{in}^{-})$ is smaller when there is a positive link from u to v than when there is a negative link. This pattern is contradictory to status theory. Fourth, the average distance between the embeddings of the role-nodes of two nodes is quite larger when there is no link between them than when there is a link. A large distance between the embeddings of two nodes implies they are not tightly connected and belong to different clusters.

CHAPTER 6

SIGNED NETWORK APPLICATIONS

In this chapter [1,2,3] , we investigate applying signed networks to understanding and solving real-world application problems. In traditional unsigned network analysis there are many data mining applications that are improved through harnessing these networks, such as information propagation [171] and recommendation [172]. Similarly, signed networks have also been used towards improving these traditional applications in information propagation [111, 173, 174] and recommendation [175, 132, 130]. In addition, although recently there have been algorithms for both the tasks of predicting interaction polarity scores [176, 177, 178, 179] and link signs between users [61, 131, 4, 180]. However, there are still some challenges associated with signed networks for predicting interaction/link polarities, such as the cold-start problem [181, 179], which is defined when users first join the system and have not logged many interactions/links yet. Furthermore, most of these methods them have tackled these two tasks independently. This naturally does not utilize the linkage between the two if the content (i.e., "items") were generated by the users themselves. Thus, in Section 6.1 we present a joint model that is able to both predict the interaction polarity scores between users and content, but simultaneously predict the polarity directly between users.

In addition to the traditional applications, there are also a plethora tasks in other domains that can benefit from signed networks. More specifically, signed networks in chemistry (i.e., Möbius graphs) are used with studying molecular systems [182]; in ecology for analyzing community structure [183]; in physics for modeling frustration in spin glasses [184]; and in political science

[1]Tyler Derr, Zhiwei Wang, Jamell Dacon, and Jiliang Tang. "Link and Interaction Polarity Predictions in Signed Networks." Social Network Analysis and Mining. 2020.

[2]Tyler Derr, Zhiwei Wang, and Jiliang Tang. "Opinions Power Opinions: Joint Link and Interaction Polarity Predictions in Signed Networks." In Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). 2018.

[3]Tyler Derr*, Hamid Karimi*, Aaron Brookhouse, and Jiliang Tang. "Multi-Factor Congressional Vote Prediction." In Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). 2019.

for analyzing balance and polarization [185]. In addition, other domain areas such as health (e.g., weight loss prediction using social networks [186]), education (e.g., utilizing social networks to better understand teachers [187, 188]) Thus, in Section 6.2 we present a comprehensive congressional vote prediction framework built around modeling both ideological and social factors, with the latter being modeled as a signed bipartite network.

6.1 Link and Interaction Polarity Prediction

Online social media has become an increasingly popular place for people to share and exchange their opinions. These opinions among users can be expressed in two main ways, namely either directly or indirectly. More specifically, in signed networks, users can directly specify opinions to others via establishing positive or negative links; and they also can give opinions to content generated by others via a variety of social interactions such as commenting and rating. Intuitively these two types of opinions should be related. For example, users are likely to give positive (or negative) opinions to content from those they have established positive (or negative) links; and users tend to create positive (or negative) links with those that they frequently positively (or negatively) interact with. Therefore we can leverage one type of opinions to power the other by capturing the correlation between these two types of opinions. Hence, a joint framework has the potential to mitigate the data sparsity and cold-start problems for both tasks. Meanwhile, they can enrich each other that can help mitigate the data sparsity and cold-start problems in the corresponding predictive tasks – link and interaction polarity predictions, respectively.

6.1.1 Problem Statement

Let $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$ denote the set of n users. We represent signed links between users in an adjacency matrix, $\mathbf{T} \in \mathbb{R}^{n \times n}$, where $\mathbf{T}_{ij} = 1$ if u_i creates a positive link to u_j, -1 if u_i creates a negative link to u_j, and 0 otherwise (i.e., when u_i has shown no link to u_j). Let $\mathcal{R} = \{r_1, r_2, \ldots, r_m\}$ be the set of m content items generated by $\mathcal{U}$. We use $\mathbf{A} \in \mathbb{R}^{n \times m}$ to denote the authorship matrix where $\mathbf{A}_{ij} = 1$ if u_i creates r_j and $\mathbf{A}_{ij} = 0$ otherwise. Social media provides multiple ways for

Table 6.1: Extended epinions dataset statistics.

# of Users	233,429
# of Positive Links	717,667
# of Negative Links	123,705
Density of $\mathbf{T}$	7.75×10^{-5}
# of Reviews	755,722
# of Positive Interactions	12,581,553
# of Negative Interactions	1,086,551
Density of $\mathbf{H}$	1.54×10^{-5}

users to express their opinions to content items generate by other users. For example, Facebook and Twitter allow their users to comment on content; Youtube provides thumbs-up and -down buttons; and Epinions enables its users to rate the helpfulness of the content with scores from 1 to 6. We use $\mathbf{H} \in \mathbb{R}^{n \times m}$ to denote opinions expressed by $\mathcal{U}$ to $\mathcal{R}$, where $\mathbf{H}_{ik} = 1 (or -1)$ if u_i gives a positive (or negative) opinion to r_k and we use $\mathbf{H}_{ik} = 0$ to indicate no explicit opinion is expressed from u_i to r_k. Note that in this paper, we define positive (or negative) interactions between u_i and u_j as u_i giving positive (or negative) opinions to content items generated by u_j. In other words, an interaction between users is defined as a triplet (u_i, r_k, u_j) where u_i expresses opinions to r_k that was generated by u_j.

With the above notations and definitions, our problem is stated as follows: *given the signed relations **T**, the authorship matrix **A** and the user-item opinion matrix **H**, we aim to learn a predictor that can infer signed links and interaction polarities simultaneously by leveraging **T**, **A**, and **H**.*

Note that when the content of the item is available, we also can utilize the content of $\mathcal{R}$. However, in this paper, we focus on leveraging $\mathbf{T}$, $\mathbf{A}$, and $\mathbf{H}$ and would like to leave the problem of exploiting content as one future work.

6.1.2 Signed Network User Opinion Data Analysis

In this section, we conduct preliminary analysis on the correlation between signed links and interaction polarities. We begin by introducing an extended version of our Epinions dataset as previously described in Table 4.2.

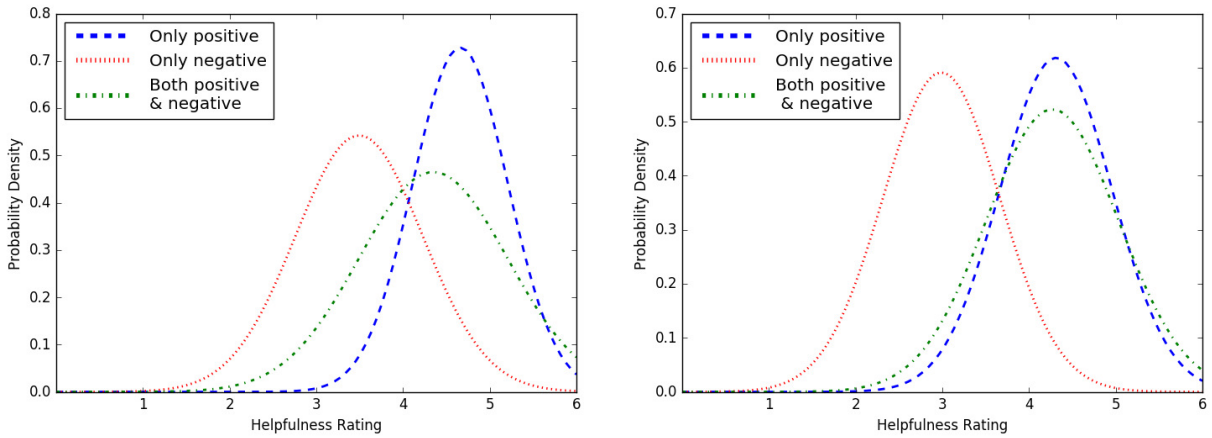
6.1.2.1 Extended Epinions Dataset

We collected an extended dataset from Epinions for this investigation. Epinions users can give positive and negative links to each other, which we use to construct the $\mathbf{T}$ matrix. They also can write reviews and we use this data to construct the authorship matrix $\mathbf{A}$. For each review others can use scores from 1 to 6 to indicate the helpfulness of the given reviews and that we use these to construct the matrix $\mathbf{H}$. We define positive and negative helpfulness ratings to be $\{4, 5, 6\}$ and $\{1, 2, 3\}$, respectively. Some statistics of the dataset are shown in Table 6.1. From the table, we can observe that (1) there are more positive links (or interactions) than negative ones; and (2) both links and interactions are very sparse. The task of creating (or receiving) a signed link to others can be thought of as an explicit form of expressing one's opinion of (or from) others. In contrast, when a user interacts with the content authored by others, they are implicitly marking their opinion towards others in these interactions. Therefore, it is reasonable to assume that the implicit and explicit opinions among users are correlated, so we investigate these correlations from both global and local perspectives.

6.1.2.2 Correlated User Opinions: A Global Perspective

From a global perspective, we want to examine the correlations between these explicit and implicit opinions from one user. In particular, we aim to answer the following questions – (1) is a user, giving more positive (or negative) links, likely to give more positively (or negatively) on content from others? and (2) is a user, receiving more positive (or negative) links, likely to receive more positive (or negative) opinions on his/her content? In this work, we refer to giving links or opinions on content as giving behaviors; while receiving links or opinions on content as receiving behaviors.

To answer the first question, we group users into three classes based upon their outgoing links as follows: (1) users who only have positive outgoing links (76,819 users); (2) users having only negative outgoing links (7,138 users); and (3) users who have both positive and negative outgoing links (11,361 users). Then, we calculate the opinions (or helpfulness ratings) they gave to content from others for each group and we plot kernel smoothing density estimation for each group in



(a) Giving behaviors.

(b) Receiving behaviors.

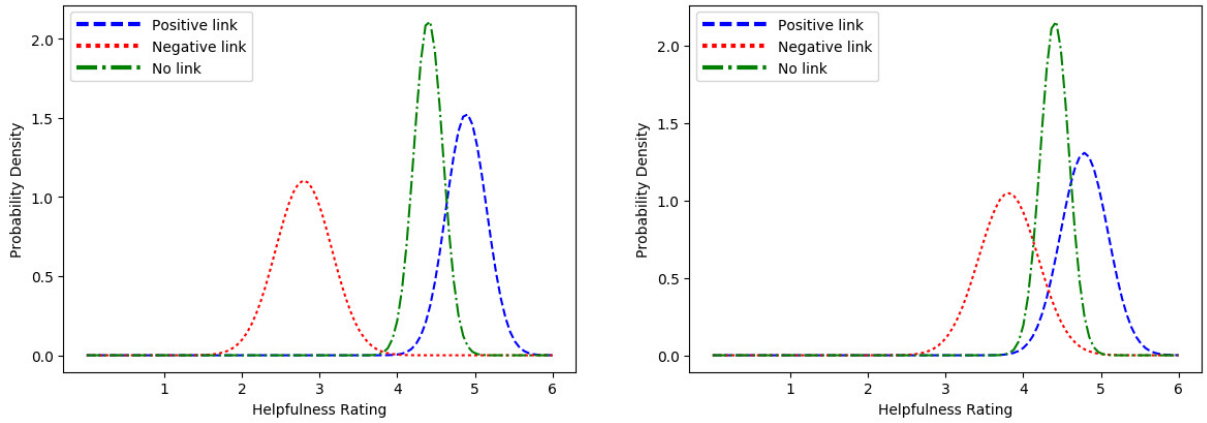
Figure 6.1: Giving and receiving behaviors from the global perspective on opinion correlations.

Figure 6.1(a). We note that on average users who only create positive links also tend to interact more positively with the content generated by other users as compared to users who only create negative links. Furthermore, users who create both positive and negative links show a higher variance than the only positive and only negative classes, and thus are more likely to express both positive and negative behaviors in their interactions.

To answer the second question, we divide users into three groups based upon their incoming links as follows: (1) users who only have positive incoming links (52,810 users); (2) users having only negative incoming links (14,701 users); and (3) users who have both positive and negative incoming links (17,090 users). Following the similar procedure, we plot kernel smoothing density estimation of receiving behaviors for each group in Figure 6.1(b). From the figures, 6.1(a) and 6.1(b), we can make very similar observations for receiving behaviors as giving behaviors, which lead to a positive answer to the second question – users, receiving more positive (or negative) links, are likely to obtain more positive (or negative) opinions on their content.

6.1.2.3 Correlated User Opinions: A Local Perspective

The global perspective in Subsection 6.1.2.2 focuses on correlations between one user and the remaining network. In this subsection, we focus on a pair of users and we want to investigate



(a) Giving behaviors.

(b) Receiving behaviors.

Figure 6.2: Giving and receiving behaviors from the local perspective on opinion correlations.

whether the existence of a positive (or negative) link for a pair of users makes a difference on how they give (or receive) opinions on each other's content. In particular, for a pair of users u_i to u_j, we aim to answer – (1) if u_i gives a positive (or negative) link to u_j, is u_i likely to give positive (or negative) opinions to content from u_j? ; and (2) if u_j receives a positive (or negative) link from u_i, is u_j likely to give positive (or negative) opinions to the content from u_i? Note that in this work, we use $u_i + u_j$, $u_i - u_j$ and $u_i ? u_j$ to denote a positive, negative and no link from u_i to u_j.

To answer the first question, we divide all pairs of users into three groups – (a) positive pairs $u_i + u_j$; (b) negative pairs $u_i - u_j$; and no-link pairs $u_i ? u_j$. For each pair in each group, we calculate the average opinion (or helpfulness ratings) from u_i to the content of u_j. We apply kernel smoothing density estimation for each group and the distributions are shown in Figure 6.2(a). From this figure, we note that on average positive pairs have higher helpfulness scores than no-link pairs, which have higher scores than negative pairs. Hence, it is quite evident from the figure that if u_i gives a positive (or negative) link to u_j, u_i is likely to give positive (or negative) opinions to the content from u_j.

Intuitively, if u_j receives a positive link from u_i, u_j is likely to be friendly to u_i, and as a consequence, u_j is likely to give positive opinions to the content of u_i. On the other hand, if u_j receives a negative link from u_i, u_j could do revenge back and give negative opinions to the content

of u_i. We follow a similar procedure of answering the first question for the second question. The results are demonstrated in Figure 6.2(b). From this figure, we observe that (1) on average, u_j mostly gives positive opinions to the content from those who give positive links to u_j; while u_j mostly gives negative opinions to the content from those who give negative links to u_j. These observations support that if u_j receives a positive (or negative) link from u_i, then u_j is likely to give opinions being more positive (or negative) to the content from u_i.

6.1.3 The Joint Link and Interaction Polarity Prediction (LIP) Framework

In Section 6.1.2, we validated that there exist correlations between a user's opinion of other users in regards to the links they form in signed social networks and the polarities of the interactions between them. Thus, these findings naturally lead us to the question of whether this knowledge can benefit the two prediction tasks that are found in the two domains; link and interaction polarity prediction. In this section, we first briefly discuss a basic framework to solve the two tasks of link and interaction polarity predictions individually. We then discuss how to model the opinion correlations that enable us to have the opinions in one task power the other. Finally we present our proposed framework LIP, which directly incorporates these correlations into a joint optimization algorithm that can infer the polarities of links and interactions jointly.

6.1.3.1 Basic Link and Interaction Polarity Prediction Models

The low-rank matrix factorization approach has gained popularity recently and is now being used across various applications such as link prediction [189, 131] and recommender systems [132, 179]. In this work, we choose to build the basic prediction models based on the low-rank matrix factorization approach.

Link Prediction: Let $\mathcal{T} = \{(u_i, u_j) \mid \mathbf{T}_{ij} \neq 0\}$ be the set of pairs with links. In terms of the link prediction task, we would like to find two latent matrices $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n] \in \mathbb{R}^{K_L \times n}$ and $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n] \in \mathbb{R}^{K_L \times n}$,with K_L being the number of latent dimensions, by solving the

following optimization problem:

$$\min_{\mathbf{U},\mathbf{V}} \ \frac{1}{2} \sum_{(u_i,u_j)\in\mathcal{T}} (\mathbf{T}_{ij} - \mathbf{u}_i^\top \mathbf{v}_j)^2 + \frac{\beta_1}{2}\Big(\|\mathbf{U}\|_F^2 + \|\mathbf{V}\|_F^2 \Big) \tag{6.1}$$

where $\mathbf{u}_i$ and $\mathbf{v}_i$ are the user latent vectors representing giving and receiving link behaviors of u_i, respectively. Thus, $\mathbf{u}_i^\top \mathbf{v}_j$ models the sign of a link from u_i to u_j, and therefore after optimizing the above formulation, we can use such inner products as a prediction for unknown user-user signed links in the network. Note that $\|\mathbf{U}\|_F^2$ denotes the Frobenius norm of $\mathbf{U}$ and is used as a regularization term to prevent overfitting, similarly for $\mathbf{V}$, and both are controlled by the hyperparameter β_1.

Interaction Polarity Prediction: Let $\mathcal{H} = \{(u_i, r_k, u_j) \mid \mathbf{H}_{ik} \neq 0, \mathbf{A}_{jk} \neq 0\}$ be the set of interaction triplets and $\mathbf{H}_{ik}$ denotes the opinion from u_i to the content r_k authored by u_j. The main difference between the basic model for this task from traditional matrix factorization based recommender systems is that we now have a third piece of information, the author. Thus, rather than taking the typical user-item formulation, we instead want to formulate the model so that we can include information about the author of the content.

In this problem, we wish to find three latent matrices $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n] \in \mathbb{R}^{K_I \times n}$, $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n] \in \mathbb{R}^{K_I \times n}$, and $\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_m] \in \mathbb{R}^{K_I \times m}$, where $\mathbf{p}_i$ and $\mathbf{q}_i$ respectively denote the giving and receiving interaction behaviors of u_i, and $\mathbf{s}_k$ is the latent vector for content r_k. One way to represent this would be to ignore the author and want $\mathbf{p}_i^\top \mathbf{s}_k$ to model the interaction between user u_i on content r_k that was authored by u_j. Similarly, we could ignore the content and only use the author, i.e., $\mathbf{p}_i^\top \mathbf{q}_j$), but each of these are lacking information. Hence, we propose to use $\mathbf{p}_i^\top (\mathbf{q}_j + \mathbf{s}_k$, which includes both the context of the author and the content itself. These three matrices can be obtained via solving the following optimization problem:

$$\min_{\mathbf{P},\mathbf{Q},\mathbf{S}} \ \frac{1}{2} \sum_{(u_i,r_k,u_j)\in\mathcal{H}} (\mathbf{H}_{ik} - \mathbf{p}_i^\top (\mathbf{q}_j + \mathbf{s}_k))^2 + \frac{\beta_2}{2}\Big(\|\mathbf{P}\|_F^2 + \|\mathbf{Q}\|_F^2 + \|\mathbf{S}\|_F^2 \Big) \tag{6.2}$$

the term $\big(\|\mathbf{P}\|_F^2 + \|\mathbf{Q}\|_F^2 + \|\mathbf{S}\|_F^2 \big)$ is introduced to avoid over-fitting, which is controlled by β_2. Note that another way of modeling could be to linearly combine the author and content representation. In that way we could define $\mathbf{M} \in \mathbb{R}^{K_I \times 2K_I}$ with $\mathbf{p}_i(\mathbf{M}(\mathbf{q}_j || \mathbf{s}_k))$, where $||$ is used

to denote concatenation. However, this would add extra complexity by needing to learn $\mathbf{M}$, so we use $\mathbf{p}_i^\top(\mathbf{q}_j + \mathbf{s}_k)$, and leave other formulations as future work. Next we will discuss how to capture correlations based on the two aforementioned basic models.

6.1.3.2 Modeling User Opinion Correlations

In Section 6.1.2, we found that the giving (or receiving) behaviors in terms of links and interactions are correlated. In the basic models from Subsection 6.1.3.1, we use $\mathbf{u}_i$ and $\mathbf{v}_i$ to denote users' behaviors when giving and receiving links, respectively. While we use $\mathbf{p}_i$ and $\mathbf{q}_i$ to respectively indicate users' behaviors when giving and receiving interactions, separately. Therefore, we can capture the opinion correlations by bridging the two giving behaviors via $\mathbf{u}_i$ and $\mathbf{p}_i$, and the two receiving behaviors via $\mathbf{v}_i$ and $\mathbf{q}_i$.

Since the two giving behaviors are correlated, we can find a linear mapping matrix $\mathbf{W}_O \in \mathbb{R}^{K_I \times K_L}$ that can map u_i's latent vector $\mathbf{u}_i$, which denotes his/her underlying behavior on how to create links, to the latent vector $\mathbf{p}_i$, which captures their behavior towards how they give opinions to the content authored by other users in the network. Given a set of latent vectors for all users $u_i \in \mathcal{U}$, it can then be easily seen that the linear mapping between them would be a solution to the following optimization problem:

$$\min_{\mathbf{W_O}} \sum_{u_i \in \mathcal{U}} \|\mathbf{W}_O \mathbf{u}_i - \mathbf{p}_i\|_2^2 \tag{6.3}$$

Similarly, we seek to find a matrix $\mathbf{W}_I \in \mathbb{R}^{K_I \times K_L}$ to represent the mapping between the user u_j's latent vectors $\mathbf{v}_j$, and $\mathbf{q}_j$, which denote their receiving behaviors of receiving links and interactions, respectively. The mapping $\mathbf{W}_I$ can be learned as follows:

$$\min_{\mathbf{W_I}} \sum_{u_j \in \mathcal{U}} \left\|\mathbf{W}_I \mathbf{v}_j - \mathbf{p}_j\right\|_2^2 \tag{6.4}$$

Eqs. (6.3) and (6.4) can capture opinion correlations for links and interactions. They also allow us to bridge the two basic models for link and interaction polarity predictions together. Next we will introduce the proposed joint framework.

6.1.3.3 The Proposed Joint Framework

Now we have formulated a model on how to optimize a linear mapping between both the giving and receiving behaviors in the two tasks. Next we show how these mappings can be used as two additional terms in our joint matrix factorization framework, LIP, for the purpose of joint link and interaction polarity prediction. LIP solves the following optimization problem:

$$
\begin{aligned}
&\min_{\substack{\mathbf{U},\mathbf{V},\mathbf{P},\mathbf{Q},\\ \mathbf{S},\mathbf{W}_I,\mathbf{W}_O}} \quad \mathcal{L}(\mathbf{U},\mathbf{V},\mathbf{P},\mathbf{Q},\mathbf{S},\mathbf{W}_I,\mathbf{W}_O) \\
&= \frac{1}{2}\sum_{(u_i,u_j)\in\mathcal{T}} (\mathbf{T}_{ij} - \mathbf{u}_i^\top \mathbf{v}_j)^2 \\
&+ \frac{\eta}{2}\sum_{(u_i,r_k,u_j)\in\mathcal{H}} (\mathbf{H}_{ik} - \mathbf{p}_i^\top(\mathbf{q}_j + \mathbf{s}_k))^2 \\
&+ \frac{\gamma}{2}\Big(\sum_{u_i\in\mathcal{U}} \|\mathbf{W}_O\mathbf{u}_i - \mathbf{p}_i\|_2^2 + \sum_{u_j\in\mathcal{U}} \|\mathbf{W}_I\mathbf{v}_j - \mathbf{q}_j\|_2^2\Big) \\
&+ \frac{\beta_1}{2}\Big(\|\mathbf{U}\|_F^2 + \|\mathbf{V}\|_F^2\Big) + \frac{\beta_2}{2}\Big(\|\mathbf{P}\|_F^2 + \|\mathbf{Q}\|_F^2 + \|\mathbf{S}\|_F^2\Big) \\
&+ \frac{\beta_3}{2}\Big(\|\mathbf{W}_I\|_F^2 + \|\mathbf{W}_O\|_F^2\Big)
\end{aligned} \tag{6.5}
$$

where the first term is a standard user-user matrix factorization model (as discussed in Subsection 6.1.3.1) for the link prediction problem. The second term is a modification to the user-review matrix factorization model that also incorporates the additional vector $\mathbf{q}_j \ \ \forall u_j \in \mathcal{U}$ to represent the influence of the author u_j in the prediction of u_i's opinion on r_k, when r_k was written by u_j. The third and fourth terms capture the correlations of giving and receiving behaviors, respectively, and their contributions are controlled by a hyperparameter γ. Other terms in Eq. (6.5) are added to avoid overfitting.

We note that the balance between optimizing for the two tasks (sign link prediction and user interactions polarities) is balanced by the hyperparameter η, where a small increase in this value will result in an increase to the importance of the user interaction polarity prediction task, and similarly towards the link prediction task when decreasing its value. Also, this transfer of information between problems is done by the linear mapping used in LIP (more specifically the terms controlled by γ

in Eq. (6.5)). If a user u_i has no link information, they are deemed a cold-start user in the link prediction task. Thus there is no way to learn $\mathbf{u}_i$ and $\mathbf{v}_i$ in the basic model and we fail to do link prediction for u_i. However, if u_i has had some interactions with other users in the network, we can learn $\mathbf{p}_i$ and $\mathbf{q}_i$ from his/her interaction data. Thus, the proposed framework LIP can also learn $\mathbf{u}_i$ and $\mathbf{v}_i$ via the model components of capturing giving and receiving correlations via the third and fourth terms in Eq. (6.5). Similarly, LIP can also help when u_i has no interaction data but has link information. Via the above analysis, we note that LIP has the potential to mitigate the data sparsity and cold-start problems in either link prediction or interaction polarity prediction.

6.1.3.4 An Optimization Method for LIP

Given the the optimization objective shown above, we now present how to solve this problem. We have chosen to use stochastic gradient descent (SGD) due to the non-convexity of the joint optimization formulation. First, we compute the partial derivatives with respect to each of the parameters (i.e., $\mathbf{u}_i, \mathbf{v}_j, \mathbf{p}_i, \mathbf{q}_j, \mathbf{s}_k, \mathbf{W}_O$, and $\mathbf{W}_I$) and then iteratively update them using SGD until convergence. We use the combined training data $\mathcal{X} = \{\mathcal{T} \cup \mathcal{H}\}$, where $\mathcal{T}$ and $\mathcal{H}$ are the link and interaction training data, respectively.

For simplicity in the below, let $e_{\mathbf{T}_{ij}} = (\mathbf{T}_{ij} - \mathbf{u}_i^\top \mathbf{v}_j)$ be the error of estimating the link (which in some social networks, such as Epinions, can represent trust-distrust) from user u_i to user u_j, $e_{\mathbf{H}_{ikj}} = (\mathbf{H}_{ik} - \mathbf{p}_i^\top(\mathbf{q}_j + \mathbf{s}_k))$ be the error of estimating the interaction value user u_i gave to content r_k that had been authored by user u_j, $e_O = (\mathbf{W}_O \mathbf{u}_i - \mathbf{p}_i)$ be the error for our linear mapping from user u_i's latent vector $\mathbf{u}_i$ (representing the way they give links) to their latent vector $\mathbf{p}_i$ (representing how they interact with content created by others), and finally, we denote $e_I = (\mathbf{W}_I \mathbf{v}_j - \mathbf{q}_j)$ be the error for our linear mapping from user u_j's latent vector $\mathbf{v}_j$ (representing the way they receive links) to their latent vector $\mathbf{q}_j$ (representing how the content they had authored receives interactions).

Gradients of $\mathcal{L}$ with respect to U and V: The gradients of Eq. (6.5) w.r.t. $\mathbf{u}_i$ and $\mathbf{v}_j$ are as follows, respectively:

$$\frac{\partial \mathcal{L}(\mathbf{U})}{\partial \mathbf{u}_i} = \sum_{\{j \mid (u_i,u_j)\in\mathcal{T}\}} \Big(- e_{\mathbf{T}_{ij}} \mathbf{v}_j \Big) + \gamma \mathbf{W}_O^\top e_O + \beta_1 \mathbf{u}_i$$

$$\frac{\partial \mathcal{L}(\mathbf{V})}{\partial \mathbf{v}_j} = \sum_{\{i \mid (u_i,u_j)\in\mathcal{T}\}} \Big(- e_{\mathbf{T}_{ij}} \mathbf{u}_i \Big) + \gamma \mathbf{W}_I^\top e_I + \beta_1 \mathbf{v}_j$$

Gradients of $\mathcal{L}$ with respect to P, Q, and S: The gradients of Eq. (6.5) w.r.t. $\mathbf{p}_i$, $\mathbf{q}_j$ and $\mathbf{s}_k$ are the following, respectively:

$$\frac{\partial \mathcal{L}(\mathbf{P})}{\partial \mathbf{p}_i} = \sum_{\{(k,j) \mid (u_i,r_k,u_j)\in\mathcal{H}\}} \Big(- \eta e_{\mathbf{H}_{ikj}} (\mathbf{q}_j + \mathbf{s}_k) \Big) - \gamma e_O + \beta_2 \mathbf{p}_i$$

$$\frac{\partial \mathcal{L}(\mathbf{Q})}{\partial \mathbf{q}_j} = \sum_{\{i \mid (u_i,r_k,u_j)\in\mathcal{H}\}} \Big(- \eta e_{\mathbf{H}_{ikj}} \mathbf{p}_i \Big) - \gamma e_I + \beta_2 \mathbf{q}_j$$

$$\frac{\partial \mathcal{L}(\mathbf{S})}{\partial \mathbf{s}_k} = \sum_{\{i \mid (u_i,r_k,u_j)\in\mathcal{H}\}} \Big(- \eta e_{\mathbf{H}_{ikj}} \mathbf{p}_i \Big) + \beta_2 \mathbf{s}_k$$

Gradients of $\mathcal{L}$ with respect to $\mathbf{W}_O$ and $\mathbf{W}_I$: Finally, we present the gradients of Eq. (6.5) w.r.t. $\mathbf{W}_O$ and $\mathbf{W}_I$, which are shown below, in respective order.

$$\frac{\partial \mathcal{L}(\mathbf{W_O})}{\partial \mathbf{W}_O} = \sum_{u_i\in\mathcal{U}} \Big(\gamma e_O \mathbf{u}_i^\top \Big) + \beta_3 \mathbf{W}_O$$

$$\frac{\partial \mathcal{L}(\mathbf{W}_I)}{\partial \mathbf{W}_I} = \sum_{u_j\in\mathcal{U}} \Big(\gamma e_I \mathbf{v}_j^\top \Big) + \beta_3 \mathbf{W}_I$$

With update rules to optimize Eq. (6.5), we use SGD to optimize the framework using the combined training data $\mathcal{X} = \{\mathcal{T} \cup \mathcal{H}\}$, where $\mathcal{T}$ and $\mathcal{H}$ are the link and interaction training data, respectively.

Algorithm 6.1: The optimization method for the proposed LIP framework.

Input: $\mathcal{T} = \{(u_i, u_j)|\mathbf{T}_{ij} \neq 0\}$ be the set of pairs with links and
$\mathcal{H} = \{(u_i, r_k, u_j)|\mathbf{H}_{ik} \neq 0, \mathbf{A}_{jk} \neq 0\}$ be the set of interaction triplets

Output: $\mathbf{U}$ and $\mathbf{V}$ for link predictions; and $\mathbf{P}$, $\mathbf{Q}$, and $\mathbf{S}$ for interaction polarity predictions

1 Randomly initialize $\mathbf{U}, \mathbf{V}, \mathbf{P}, \mathbf{Q}, \mathbf{S}, \mathbf{W}_O, \mathbf{W}_I$
2 Construct the learning data set $\mathcal{X} = \{\mathcal{T} \cup \mathcal{H}\}$
3 **while** Not convergent **do**
4 Shuffle($\mathcal{X}$)
5 **foreach** $x \in \mathcal{X}$ **do**
6 **if** $x \in \mathcal{T}$ **then**
7 Calculate gradients of $\mathcal{L}(\mathbf{U}, \mathbf{V}, \mathbf{P}, \mathbf{Q}, \mathbf{S}, \mathbf{W}_I, \mathbf{W}_O)$ w.r.t. $\mathbf{u}_i, \mathbf{v}_j, \mathbf{W}_I$, and $\mathbf{W}_O$
8 **if** $x \in \mathcal{H}$ **then**
9 Calculate gradients of $\mathcal{L}(\mathbf{U}, \mathbf{V}, \mathbf{P}, \mathbf{Q}, \mathbf{S}, \mathbf{W}_I, \mathbf{W}_O)$ w.r.t. $\mathbf{p}_i, \mathbf{q}_j, \mathbf{s}_k, \mathbf{W}_I$, and $\mathbf{W}_O$
10 Update the respective parameters using gradient descent methods

Note that although there are additional methods for optimizing matrix factorization based methods, SGD has been shown to be both efficient and easy to tune, e.g., adaptive learning rates.

With gradients calculated above to optimize Eq. (6.5), the detailed optimization algorithm is presented in Algorithm 6.1. Next we briefly introduce Algorithm 6.1. In line 1, we randomly initialize model parameters. In line 2, the learning data includes links and interactions. From line 3 to line 14, we use stochastic gradient decent to optimize the framework. In particular, for each iteration, we first shuffle the data in line 4; and then update model parameters using gradient decent methods from line 5 to line 12. When having a signed user-user link training example, the algorithm utilizes lines 6 through 8 to calculate the gradients, as compared to when having an interaction training example, lines 9 through 11 are used. Then, on line 12, the model parameters for the respective part of the problem (based on whether we are updating on a signed link or interaction) can be updated using a gradient based method.

6.1.4 Experiments

In this section, we conduct experiments to answer the following two questions: (1) Can our joint model help alleviate the sparsity problem in these two prediction tasks? (2) Do the terms based

upon correlated user opinions/behaviors in LIP provide a transfer of information between the two problems? To address the first question, we perform experiments in which we increase the sparsity of the training data and compare the performance with representative baselines. We address the second question by examining if our algorithm is robust to handle some cold-start users. In the next subsection we will further introduce our dataset and how it was used, the metric used in evaluating the two prediction tasks, then we introduce the experimental settings for the two types of experiments we have performed.

Experimental Settings: As mentioned in Section 6.1.2, we have collected a dataset from Epinions for these experiments. Note that for the purpose of this study, we have filtered our collected Epinions dataset to form more dense user-user and user-content matrices. The first step is to pre-process the data such that we have the appropriate training, validation, and testing sets from our dataset.

The filtering we perform only keeps users that have both given and received a link, and also requires the users to have given at least one helpfulness rating and have also authored at least one review that has received at least one helpfulness rating. For all selected users to be filtered out, we remove all their user links, reviews they had written, and helpfulness ratings associated with that user. The reason for this filtering is that it will allow us to later remove portions of the data to artificially create training sets that have a varying percentage of cold start users and also different levels of sparsity and therefore seemingly becoming more similar to the raw dataset.

The initial extended Epinions dataset had contained 233,429 users, 841,373 user-user links, and 13,668,105 helpfulness ratings. After the above mentioned filtering process, we were left with 29,901 users, 600,976 user-user links, and 11,555,599 helpfulness ratings. The dataset has been randomly split into 70% for training, 10% for validation, and 20% for testing. Note that we then balanced our testing dataset to be 50% positive and 50% negative similar to that done in [4].

For all the models that required hyperparameters to be tuned, we used the validation set to obtain the best hyperparameters for each respective model. Also, the hyperparameter settings for each experiment was fixed (e.g., all LIP results for the five varying cold-start experiment were selected

based on a commonly "good" set of hyperparameters for all five percentages, and not separately for each of the five). However, between the two experiments, we allowed for different hyperparameters as the dynamics of cold-start users and varying the amount of induced sparsity required a different set of hyperparameters for our model and similarly for the baselines. To evaluate and compare the performance of LIP we present the F1 measure for the interaction polarity and the link prediction tasks. Note that the higher the value, the better the performance.

6.1.4.1 Sparsity Experiments

To answer the first question, we compare the proposed framework, LIP, with existing interaction polarity and link prediction methods. We first present the baselines for the interaction polarity prediction task followed by those for the link prediction task.

We choose the following representative interaction polarity prediction baselines for comparison:

- *u*CF: User-based collaborative filtering approach where we used the five most similar users (in terms of cosine similarity) based on their helpfulness rating history for making the predictions. For details on collaborative filtering please see [190]. We use the user-based collaborative filtering approach as our first baseline for predicting the user interaction polarities. Here we present the results where we used the five most similar users (in terms of cosine similarity) based on their helpfulness rating history for making the predictions.

- *M*F: Our low-rank matrix factorization method as shown in Eq. (6.2). Here a comparison is made with the low-rank matrix factorization method, that attempts to find a lower dimensional representation of the user-review matrix. Note this follows the same formulation as that in Eq.(6.2) where we use the matrices $\mathbf{P}, \mathbf{Q}, \mathbf{S}$, and $\mathbf{H}$ equivalently as they are in LIP for the predictions.

For link prediction, the representative baselines are presented below and details of the methods can be found in their respective cited work.

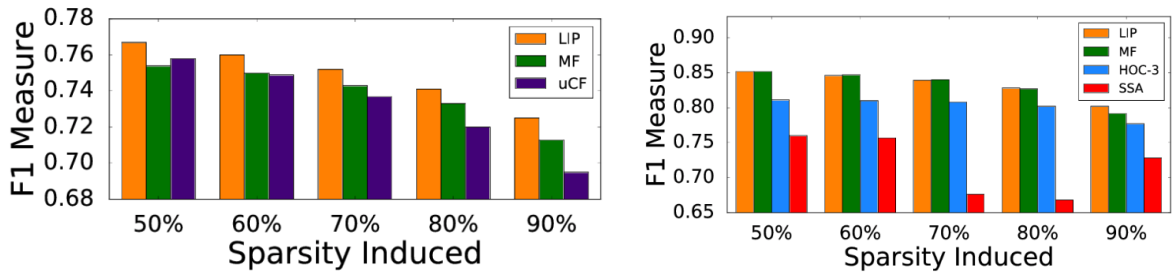
- *S*SA: A spectral based method using the signed Laplacian matrix [153] and regularized Laplacian kernel [191] is used. Due to the fact this method was presented for undirected networks, we convert the directed link information by making **T** symmetric, thus resulting in an undirected network, we use the undirected version of the dataset by removing the directions of the links, but keep the testing set the same.

- *H*OC-3: It is an approach that was based on the social balance and status theories [2]. Features for a supervised approach are extracted from triads and also node features (e.g., number of incoming positive edges). A total of 23 features are created based on 16 possible directed triad configurations, and 7 node features. The details of this method can be found in [4].

- *M*F: Low-rank matrix factorization method as shown in Eq. (6.1), which was first introduced in [131]. The final comparison is with the low-rank matrix factorization method, which was first introduced for this problem in [131]. This is the natural baseline predictor for our model since LIP is built upon this MF technique. This method optimizes the squared error, has the regularization hyperparameter β, and uses SGD. We note that it is formulated just as seen in Eq.(6.1) and the matrices **U**, **V**, and **T** are used equivalently to those found in LIP.

In the first experiment, we are able to simulate a ranging sparsity across each user, since we have already limited our attention to a subset of the data that is denser than the original dataset. We remove x% of the links and interactions for each user and vary x in {50, 60, 70, 80, 90}. we are able to simulate a ranging sparsity across each user. We vary the sparsity of the dataset by removing 50% to 90% of the data, in increments of 10%.

Experimental Results: The interaction polarity prediction results can be found in Figure 6.3(a). Most of the time, we see that the baseline MF method outperforms the user-based collaborative filtering method. Similarly, we have LIP finding significant gains over MF across the levels of sparsity induced. *A*nother thing to mention is that since we had first increased the density of the user-review matrix , it is not until the 80% sparsity that the density of the network drops below that



(a) Interaction Polarity Prediction.

(b) Link Prediction.

Figure 6.3: Experimental results with varied sparsity settings.

of the original matrix **H**. Therefore in fact at 80% sparsity the density of this induced sparse network is quite similar to that of the original network. We report the results of the sparsity experiments for the link prediction in Figure 6.3(b). LIP and MF obtain much better performance than SSA and HOC-3. We are able to observe that LIP performs comparable to the MF method for the lower sparsity settings, but upon reaching the higher sparsity level, LIP achieves better performance than MF.

From the results in the sparsity experiment, we have seen LIP's ability to help alleviate the sparsity problem found in the interaction polarity and link prediction tasks; thus providing evidence that our joint framework is able to partially alleviate the sparsity problem inherent in signed networks. More specifically, we see a significant improvement in the interaction polarity predictions, and increasing improvement for the link prediction with the increase of the sparsity.

6.1.4.2 Cold-Start Experiments

Note that one of the main contributions of this work is the ability of the framework to handle not just the data sparsity problem, but also to help alleviate issues that are commonly faced with cold-start users in signed networks, which are quite common characteristics in these datasets. Therefore to answer the second question, we compare LIP with existing algorithms that are able to handle cold-start users in both of the two prediction tasks.

For this experiment we want to empirically evaluate the robustness of LIP when faced with

networks having cold-start users. Note that this is a very difficult problem to overcome due to the fact if there is no knowledge about a user in a certain domain, then it becomes difficult, if not impossible, to make reasonable predictions involving them. However, since LIP is jointly predicting the signed links and user interaction polarities, the opinions formulated in one task can power those in the other task and simultaneously they should be able to gain information for users that previously had none in one of the tasks.

For the cold-start setting, we choose the following user interaction polarity prediction baselines:

- *R*G: The random guessing method for user interactions first calculates the class distributions, and then selects randomly based on that distribution to make predictions for unknown values.

- *A*vgG: The average guessing method (AvgG), first calculates the average interaction value found in the entire training set, next it predicts that value for all missing values, and then it predicts that same value for all other edges in the network that have yet to be assigned.

- *M*FwRG: We note that the typical matrix factorization method would not be applicable in this experiment, since if we have no training information for a given user, then the latent vectors of such users would never be updated. Thus, this would leave the predicted value to be assigned based on the dot product of two randomly initialized vectors. So instead we modify *M*F by adding the condition that if either of the two users' vectors have not been updated (i.e., they had no data in the training set and thus are a cold-start user), then instead of using the dot product as we normally would with *M*F for predicting links, we instead use the *R*G method for the given link.

We note that the typical matrix factorization method would not be applicable in this experiment, since if we have no interaction information for a given user, then the latent vectors of such users would never be updated. This would leave the predicted value to be assigned based on the inner product of two randomly initialized vectors. Thus, we modified *M*F by adding the condition that if either of the two users' vectors have not been updated (i.e., they had no training interaction data

Table 6.2: Interaction polarity prediction cold-start results.

Method	Induced Percent Cold Start Users				
	5%	10%	15%	20%	25%
RG	0.655	0.655	0.655	0.655	0.655
AvgG	0.667	0.667	0.667	0.667	0.667
MFwRG	0.769	0.764	0.754	0.746	0.739
LIP	0.773	0.771	0.769	0.766	0.763

Table 6.3: Link prediction cold-start results.

Method	Induced Percent Cold Start Users				
	5%	10%	15%	20%	25%
RG	0.641	0.641	0.641	0.641	0.640
MFwRG	0.848	0.837	0.825	0.813	0.797
LIP	0.860	0.858	0.853	0.848	0.839

and are therefore a cold-start user), then instead of using the inner product as we normally would with *M*F for predicting links, we instead use the *R*G method for that given link.

We compare the proposed framework LIP with the following link prediction baselines:

- *R*G: Randomly guess missing links to be positive or negative based on training data class distribution.
- *M*FwRG: This method has the identical extension for the cold-start users as described in *M*FwRG for the interaction polarity prediction task.

For these experiments, we vary the percentage of users that become cold start users in a given task, but do not modify the testing set. We randomly select x% of the users and remove all their links, then randomly select x% of the users (who we have not already selected) and remove their interaction information while varying vary x in {5, 10, 15, 20, 25} i.e. the number of cold-start users from 5% of the training dataset users to 25%, in intervals of 5%, thus making 5 data subsets.

Experimental Results: Table 6.2 holds the results of the cold-start experiments for the interaction polarity prediction task when varying the number of cold start users. The very naive baseline RG is just shown to provide a reference for the F1 measure, but the *MFwRG* is expected to perform quite well. In this table, we are able to observe LIP's superiority over the baseline methods

when observing cold-start users. We also see that LIP's performance as compared to the baselines drastically increases as the number of cold-start users increases, which is extremely intuitive based upon the use of the correlation terms. This is because even if a user has no current helpfulness rating information, LIP is able to transfer information (i.e., their opinions) through the linear mapping matrices $\mathbf{W}_O$ and $\mathbf{W}_I$ and use information that the user had from their link information.

In Table 6.3, we present the link prediction results when varying the amount of cold-start users in the training set. Upon seeing these results the advantages of LIP over the other baseline methods become even more obvious. We note that whenever *MFwRG* has the ability to learn a low dimensional representation for a user, it can then perform the prediction using it's learned low dimensional latent vectors. But when there is no link information for a given user, then the user must resort to randomly guessing. Similarly to the interaction polarity prediction task, as the percentage of cold-start users increases, the performance gap in terms of F1 becomes larger in favor of LIP having the best prediction.

6.1.4.3 Experiment Discussions

This leads us back to our second question, where we set out to determine if the linking terms based upon the correlated user opinions in LIP are able to provide a transfer of information between the two tasks that ultimately have a user's opinions in one task power the other. Based upon the results presented in this section, for both the sparsity and the cold-start experiments, we have shown that indeed LIP is able to utilize the inherent correlations behind the opinions expressed in the two tasks to boost the performance in both the prediction tasks simultaneously. Next we present our analysis on the hyperparameters of LIP. We seek to not only to gain a better understanding of the relation between these two prediction tasks (i.e., η), but perhaps even more important in this study, is the focus on γ, since it controlled the amount of opinion information to be transferred from one prediction task to the other; specifically the ones that control the correlation terms and the balance between optimizing the interaction polarity prediction task along with the link prediction task.

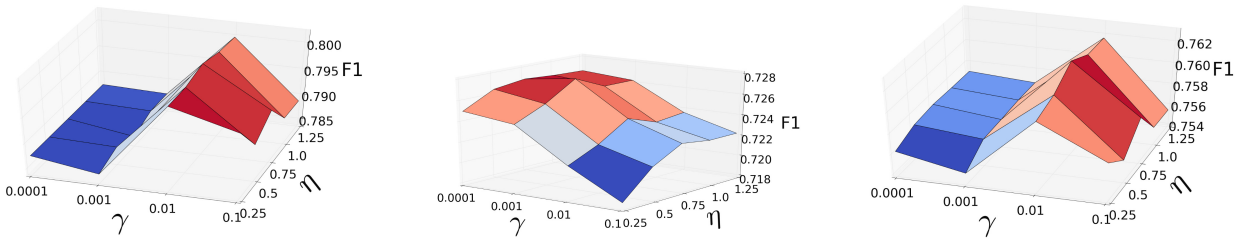
Based on the above experimental results we have successfully verified our claim that our joint

matrix factorization model using additional terms for modeling the fact that user's in social networks express their opinions in correlated ways across tasks when faced with sparse datasets. However, the most obvious claim we are now able to express is that LIP does indeed help alleviate the cold-start problem over the baseline MF method and the other baselines. In the next subsection we perform a hyperparameter analysis to gain a better understanding to not only the relation between these two prediction tasks (i.e., η), but perhaps more importantly in this study is the focus on γ, since it controlled the amount of opinion information to be transferred from one prediction task to the other.

6.1.4.4 Parameter Analysis

First we will discuss the hyperparameters used in LIP. Thereafter we discuss an analysis on some of the important hyperparameters in our model.

In this work β_1, β_2, and β_3 are used as the typical regularization hyperparameters and we noticed they behave normally. In fact, they could be collapsed into a single regularization hyperparameter β without much change to the performance (as compared to splitting them into three separate hyperparameters). The other hyperparameters are quite necessary and typical for joint modeling (and similarly for cross-domain recommendation problems). For η this is used to balance between the two tasks, which is assumed to result in large changes in performance when varying this hyperparameter greatly. This is because it controls to what extend the optimization is favoring higher performance (perhaps at the cost of the other) for one of the two problems over the other. As for γ, we have introduced this as a Lagrange multiplier used to solve this challenging optimization problem. In other words, based on our analysis, it appears there should be a transformation between the two domains of links and interactions, and to solve this problem we have relaxed this constraint of finding such a mapping to instead find a mapping with minimal error (since we also assume the data is noisy). Hence, we introduce the hyperparameter γ to solve the optimization problem. Finally, we have K_L and K_I that denote the length of the representations in the link and interaction domains, respectively. These are the typical hyperparameters for embedding based methods, and



(a) Link prediction.

(b) Interaction polarity prediction.

(c) Trade-off between the two tasks (i.e., mean of 6.4(a) and 6.4(b)).

Figure 6.4: Performance variations of LIP on the 90% data sparsity experiment w.r.t. η and γ.

we have observed similar results as other methods that vary the embedding, i.e., the performance starts to increase, but then drops once the embedding becomes too large. Next we will discuss an analysis on η and γ as these are the most interesting hyperparameters of LIP.

The hyperparameters η and γ control the balance between optimizing the link prediction and user interaction polarity tasks, and how strongly to keep the two tasks low dimensional representations correlated, respectively. In this subsection, we perform an analysis on how changing these two hyperparameters affects the performance of LIP. We first fix all other hyperparameters (i.e., the regularization hyperparameters β_1, β_2, and β_3 and dimension sizes K_L and K_I) based upon the best hyperparameters found against our validation set when performing a grid search over the hyperparameter space. We evaluate the performance on all paired (η, γ) values while we vary the value of η as $\{0.25, 0.5\ 0.75, 1.0, 1.25\}$ and γ as $\{0.0001, 0.001, 0.01, 0.1\}$, providing us with 20 possible combinations for running the grid search. Although the best hyperparameter settings varied between the two above mentioned experiments, we only display one representative from the sparsity user experiment, since we have similar observations in every other experimental setting. We present the analysis on the 90% sparsity experiment since it had the most variation in performance across the different settings.

In Figure 6.4, we have shown the 3D surfaces for the mentioned combination of hyperparameters. In Figure 6.4(a), we can see that $\gamma = 0.01$ is shown to clearly be a good region for this hyperparameter, as both to the left and right the performance in terms of F1 drops for the link

prediction. However, there is little to no significant difference between the link predictions when varying η in the range provided. It can also be noticed that for the interaction polarity prediction task (seen in Figure 6.4(b)) the larger η leads to much better performance, which intuitively makes sense because a larger η relates to increasing the weight of how much we were to optimize the interaction polarity prediction as compared to the link prediction task. Unlike what we observed in the link prediction task, the interaction polarity prediction performs better with a smaller γ; meaning the two tasks have a different preferred weight to be associated with the correlation between the user latent vectors.

Finally, in Figure 6.4(c) shows that there is a drastic trade-off between the two tasks. Where if one of the tasks has a large increase in F1, then the other task becomes slightly worse. Thus to obtain better performance in both tasks, we would want to choose a hyperparameter setting such that the trade-off between the two tasks is balanced. Based on our analysis such a point would have $\gamma = 0.01$, but as for the value of η, there is not a decisive value to choose. Thus, we have shown that the balance between optimizing the two tasks is not very sensitive, although from the figure it appears choosing $\eta = 0.75$ has a slight advantage in both of the two tasks.

6.2 Congressional Vote Prediction

Recently there has been an enormous interest in computational approaches to solve political science related problems, especially in relation to political elections and congressional voting. With the seemingly ever-growing tension between the two dominant political parties in the U.S. [185], congressional representatives are receiving immense social pressure towards blindly following their political party and associated leaders. However, due to the nature of some representatives refusing to give up their beliefs and ethical grounds, they sometimes vote against their party or cast no vote; thus resulting in a highly complex system.

Although knowing the voting behaviors in the congressional system are undoubtedly complicated, we remain diligent towards the goal of being able to predict and understand them. If we can construct better vote prediction models, we could utilize this information to better inform the

public of the real intentions of those running for re-election on upcoming critical issues. Similarly, congressional leadership could utilize these models for specifically targeting potential swing voters.

We recognize and identify two sets of effective factors. The first set being ideological factors, which are well recognized to play an important role in the U.S. congress [192, 193] and come from both the congressional representatives as well as the ideology of the bills, whose values and beliefs are woven deep into the content of the bills. The second set of influential factors are social factors and are in relation to 1) the party affiliations of representatives, and 2) how their past voting recording intertwines with other representatives. In relation to the first social factor, it is well known that representatives in the U.S. Congress are polarized [194, 195, 196, 197, 185]; and thus likely to follow their political affiliation when casting their votes (although not all the time). As for the second social factor, we propose the voting records to be modeled as a signed bipartite social network (i.e., contains both positive and negative connections) between the representative and the bills [116], which opens the door to extracting a plethora of novel predictive features.

We propose an end-to-end framework Multi-Factor Congressional Vote Prediction (MFCVP) that first utilizes Wikipedia[4] pages of the representatives to learn an embedding that encodes ideological information associated with each representative. Then, for the bills, we use their texts to directly learn an embedding that encodes their semantic ideological information. Next, we utilize signed network analysis to first construct a bipartite voting network between the representatives and the bills, followed by harnessing powerful signed social theories to construct novel features. Finally, all the extracted features coming from multiple factors are combined to be utilized for vote prediction.

6.2.1 Problem Statement

To introduce the problem, we first denote the set of n representatives as $\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$. We let $\mathcal{B} = [b_1, b_2, \ldots, b_m]$ denote the sequence of bills associated with the past m roll-call votes for which we know the voting outcomes. These voting outcomes are denoted in the set

[4]https://www.wikipedia.org/

Table 6.4: Notations regarding congressional vote prediction.

Notations	Descriptions
$\mathcal{R}$	The set of representatives.
$\mathcal{B}$	The set of past roll-call votes and their bills.
$\mathcal{V}$	The set of past votes $\mathcal{R}$ gave on $\mathcal{B}$.
$\tilde{\mathcal{B}}$	The set of future roll-call votes and their bills.
$\tilde{\mathcal{V}}$	The future votes we seek to predict.
t_i	The representative r_i when they are voting.
s_j	The sponsor of bill b_j.
c_j	The set of cosponsors for the bill b_j.
$v_{ij}(\tilde{v}_{ij})$	The vote associated with voter t_i on bill b_j ($\tilde{b}_j$).

$\mathcal{V} = \{v_{ij} | r_i \textit{ voted on bill } b_j\}$ and $v_{ij} \in \{+, -, o\}$, which denotes a "yea", "nay", or "present"/"no vote", respectively. Furthermore, we have the sequence of $\tilde{m}$ future roll-call bills denoted as $\tilde{\mathcal{B}} = [\tilde{b}_1, \tilde{b}_2, \ldots \tilde{b}_{\tilde{m}}]$. The sequence $\tilde{\mathcal{B}}$ has corresponding votes $\tilde{\mathcal{V}} = \{\tilde{v}_{ij} | r_i \textit{ will vote on bill } \tilde{b}_j\}$ which we seek to predict . We then denote any additional contextual feature or those extracted from the past votes as the set $\mathcal{X}$. Note that these notations and others used throughout the paper can be found in Table 6.4. Finally, we can formally define the congressional vote prediction problem as follows:

Given a set of congressional representatives $\mathcal{R}$, a sequence of past roll-call votes on the bills $\mathcal{B}$ having associated votes $\mathcal{V}$, features $\mathcal{X}$, and a future sequence of the upcoming roll-call votes on the bills $\tilde{\mathcal{B}}$, we seek to learn a model F as follows:

$$F : \{\mathcal{R}, \mathcal{B}, \mathcal{V}, \mathcal{X}, \tilde{\mathcal{B}}\} \rightarrow \tilde{\mathcal{V}} \tag{6.6}$$

6.2.2 Overview of Multi-factor Congressional Vote Prediction (MFCVP)

For congressional vote prediction, we must overcome the challenges of how to represent the underlying factors influencing the voting system and how to handle this added complexity introduced by incorporating multiple factors. To address these we propose the end-to-end framework Multi-Factor Congressional Vote Prediction demonstrated in Figure 6.5. More specifically, MFCVP will utilize ideological factors and social factors, which the latter consists of factors coming from both

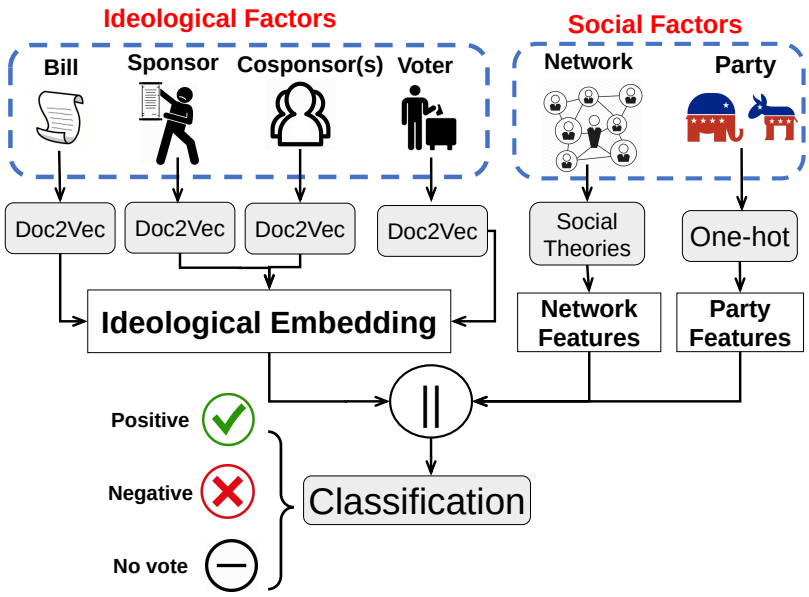


Figure 6.5: The proposed Multi-Factor Congressional Vote Prediction (MFCVP) framework.

a network and political party affiliation perspective. We first explain how these different factors are represented through both learning embeddings and constructing novel hand-crafted features. Thereafter, we discuss how the representations of different factors are combined and used for the vote classification.

6.2.3 Ideology Factors of MFCVP

The first set of factors is ideology factors. It is without a doubt that representatives' ideology and ideological information reflected in a bill are influencing how a voter will vote on a bill. To effectively and comprehensively represent ideology factors in our framework, we recognize and propose the use of two other entities (besides a bill and a voting representative) which are associated with ideology factors, namely the sponsor and possible cosponsor(s). These two entities are essentially representatives who construct and promote a bill. Hence, we seek to learn representations about the beliefs and values of the voters, sponsors, and cosponsors, along with those that are present in the bills.

To represent the representatives, many previous works focused on ideal point models [192, 198, 199]. Nevertheless, ideal point methods require many assumptions about voter behaviors

which are inherently highly complex, so instead, it seems more natural and reasonable to extract a vector representation from the raw data [195, 193] (e.g., Wikipedia pages that are collectively written about the representative from the large online community). Furthermore, extracting vector representations are practically more feasible than attempting to compute ideal points [192] which are also open for biases in their human construction. Given the more recently developed deep models for extracting meaningful representations for text documents, we propose to utilize doc2vec [200] as an efficient embedding method to represent the ideological factors. Doc2vec has shown significant improvement in many approaches [201, 202].

We use Wikipedia pages to learn a representation for each of the congressional representatives using doc2vec as illustrated in Figure 6.5. We combine all textual information about a representative from their Wikipedia profile page as a single document. Then, we train a doc2vec model which learns a compact embedding about each entire document (i.e., a representative's Wikipedia page) encoding the semantic information about a representative including their political ideology. Due to the fact that voters, sponsors, and cosponsors are all representatives, we utilize the same representation obtained through the learned embeddings of our trained doc2vec model (i.e., the doc2vec model fed with Wikipedia pages of representatives) as the ideology factors for the representative in all three roles. We should emphasize that in our experiments we utilize historical Wikipedia pages to ensure there is no data leakage.

The usefulness of Wikipedia is that this ideological perspective is less susceptible to biases or falsehoods since it is maintained by a large community. However, other data sources could be used to obtain the ideological representation, such as the generated content of voters on social media (e.g., their tweets on Twitter[5] or their campaign financial information as to which organizations are supporting them. We leave connecting other sources of data about the congressional representatives as one future work. Finally, we let E_{t_i}, E_{s_j}, and E_{c_j} denote embeddings of the voter $t_i \in \mathcal{R}$, the sponsor $s_j \in \mathcal{R}$ sponsoring the bill $b_j \in \mathcal{B} \cup \tilde{\mathcal{B}}$, and the cosponsor $s_j \in \mathcal{R}$ cosponsoring the bill $b_j \in \mathcal{B} \cup \tilde{\mathcal{B}}$ for the votes $v_{ij} \in \mathcal{V} \cup \tilde{\mathcal{V}}$.

[5]http://www.twitter.com

The textual content of the bill offers very essential information. In fact, the text of a bill reflects both the conscious and sometimes even subconsciously instilled ideologies of the sponsor and cosponsors who prepared it. Therefore, it is of great importance to effectively represent the semantic information about a bill in a compact and efficient way. To achieve this, similar to our embeddings for the representatives, we utilize a doc2vec model to represent the bills, where each bill's textual data (after some preprocessing) is considered as a document. Let E_{b_j} be the learned embedding of the bill $b_j \in \mathcal{B} \cup \tilde{\mathcal{B}}$. Note that we train the bill doc2vec model on $\mathcal{B}$.

We can now succinctly represent the set of embedded ideological features that we will utilize when considering the relation between a voter t_i and a bill b_j (along with their sponsor and cosponsor(s), s_j and c_j, respectively) as $\mathcal{E}_{ij} = \{E_{t_i}, E_{b_j}, E_{s_j}, E_{c_j}\}$.

6.2.4 Social Factors of MFCVP

Having discussed the ideological factors that get incorporated into MFCVP, here we discuss the more novel social factors (with an emphasis on the network features) that have been commonly overlooked by previous methodologies and analyses in relation to the predictions and understanding of congressional votes. We propose to categorize these social factors into two main groups as follows: 1) political party affiliation features, and 2) features coming from the network constructed from the past voting records. Next we discuss these two feature categories.

6.2.4.1 Party Features

The inspiration of these features for our proposed framework comes from the fact that sometimes there is an influence coming from voters of a political party to cast their votes aligned with the party's interest.

Given a single vote v_{ij} made by voter t_i on bill b_j that was sponsored by s_j and cosponsored by the set of representatives c_j, we construct the corresponding features P_{t_i}, P_{s_j}, and P_{c_j} to represent their party affiliations, respectively. More specifically, P_{t_i} and P_{s_j} are one-hot vectors indicating the affiliated party of the voter and sponsor, respectively. Then for the set of cosponsors c_j we

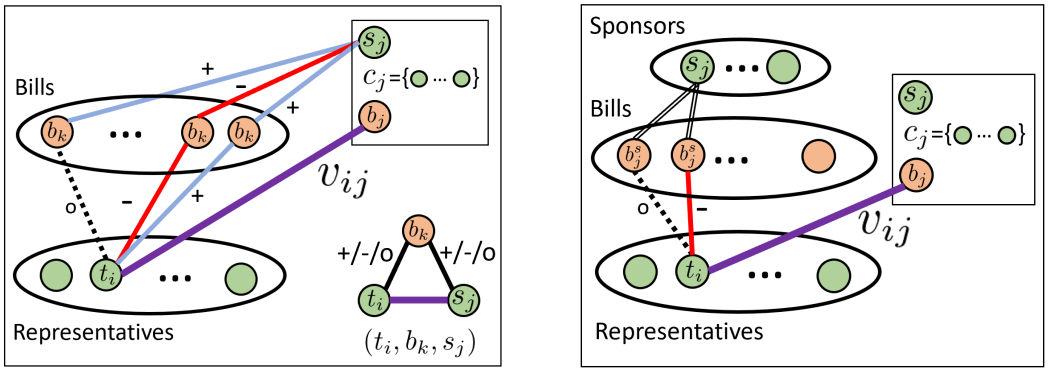
obtain the distribution of the cosponsors across the party affiliations. Note that if there are no cosponsors, we simply use a vector of zeros for P_{c_j}. These three features are represented in the set of features $\mathcal{P}_{ij} = \{P_{t_i}, P_{s_j}, P_{c_j}\}$.

6.2.4.2 Signed Bipartite Network Features

Typical network representations that are used for congressional voting records are the two one-mode networks coming from a bipartite network which ultimately separates and/or condenses the "yea" and "nay" votes [203]. However, this is inherently destined to lose drastic amounts of vital information that could have perhaps been extracted if using network analysis techniques that incorporate the "yea" and "nay" votes simultaneously. Therefore, we propose a more advanced representation - signed bipartite network.

Let $\mathcal{G} = \{\{\mathcal{R} \cup \mathcal{B}\}, \mathcal{V}\}$ denote the signed bipartite network that is constructed using the set $\{\mathcal{R} \cup \mathcal{B}\}$ of $n+m$ nodes (i.e., the representatives and bills), and set of links (i.e., votes $\mathcal{V}$) between them where we treat "yea", "nay", and "no vote" as a positive, negative, and non-existent link in the signed network. Now, given that we have modeled the voting history in the form of a signed network, we can utilize signed social theories to extract insightful features. More specifically, we utilize balance theory, which colloquially can be summarized as "a friend of a friend is a friend" while "an enemy of a friend is an enemy" [30, 29].

The first set of features we construct when considering the relationship between a voter t_i and a bill b_j can be seen in Figure 6.6(a). We can observe that we want to extract information on how the voter t_i and the sponsor s_j have interacted together on other bills b_k to gain information on how t_i might vote on the current bill b_j. We note that there can be 9 possible situations when considering the triplet (t_i, b_k, s_j), since both t_i and s_j can have either a positive, negative, or no link to the other bills $b_k \in \mathcal{B}\backslash\{b_j\}$. We utilize this information to construct a feature vector $N^{\mathcal{B}}_{t_i s_j}$ that represents the distribution over the nine aforementioned possibilities where the triangles involving an even number of negative links are adhering to balance theory. The distribution over the number of balanced and unbalanced triangles along with the number of open structures (i.e.,



(a) $N^B_{t_i s_j}$ (b) $N^S_{t_i s_j}$

Figure 6.6: Illustrations of the signed bipartite network features.

those involving at least one "no link") should provide great insight for our model to discover the patterns related to this fundamental social theory. Signed triangle distributions have also recently been used in benchmarking generative signed network models [204], since they hold such rich information about a signed network.

We note that these features are similar to the ones utilized in the seminal work [4] that focused on building a supervised model to predict the missing sign between t_i an s_j, but here we use s_j as a proxy for their introduced bill b_j. This relates to balance theory because the signed social theory would suggest that if t_i has voted equally to s_j (i.e., $v_{ik} = v_{jk}$), then it is likely that t_i should think positively towards b_j. Similarly, we construct a feature vector $N^B_{t_i c_j}$ where instead of using s_j, we obtain the average over the cosponsors in the set c_j.

We furthermore extract the second type of feature from our constructed signed network. In the first network feature (described before), we sought to discover how the overall distribution of balance between the votes from the voter t_i and the current sponsors and cosponsors (i.e, s_j and c_j) towards the rest of the bills b_k. However, unlike the first features, here we want to directly observe how t_i has interacted on the bills sponsored by s_j or sponsored by someone in c_j (i.e., a more personalized set of social features), which is related to the polarity of their interactions in the signed network [205]. In Figure 6.6(b) we show an illustration for how we construct the feature

vector $N^S_{t_i s_j}$ having length 3. Given the fact that we want to extract information about how t_i might vote on b_j, we observe the distribution over the three possible votes (i.e., positive, negative, or no link in terms of the signed network) that t_i has given to all other bills b^s_j that were also sponsored by s_j. Similarly, we construct the feature vector $N^S_{t_i c_j}$, but rather than observing the vote distribution over the set of bills b^s_j, instead, we average over b^c_j, which denotes the set of bills sponsored by the cosponsors c_j (who has cosponsored b_j).

Finally, we construct the full set of network features $\mathcal{N}_{ij} = \{N^B_{t_i s_j}, N^B_{t_i c_j}, N^S_{t_i s_j}, N^S_{t_i c_j}\}$, where $|\mathcal{N}_{ij}| = 24$. Note that these network features are in fact general and if given additional context (e.g., the connections between the voters, sponsors, and cosponsors on Twitter), we could easily extend these ideas to obtain a larger social context between the representatives; we leave this as future work along with the use of advanced signed network embeddings [206].

6.2.5 Classification Details of MFCVP

Now that we have discussed all the features coming from multiple factors, we next discuss how we can utilize them together for training a model for congressional vote prediction. We note that our framework is flexible in that the choice of the classifier is not fixed and can be chosen based on the desired outcome. One choice is to utilize a random forest [207] since it is typically an easy off-the-shelf model to train and also have the added benefit of being interpretable. More specifically, feature importance can be calculated from this model that can give insight into which features are more important for the correct classification of the votes (this will be shown in Section 6.2.6.5). Another choice could be made to utilize the power of deep learning [208] for obtaining perhaps better performance in prediction, but losing the ease of interpretation (although we note that interpreting deep neural networks is a current hot topic field in itself). In this work, we utilize both random forest and a deep neural network as classifiers.

Table 6.5: US Congress dataset statistics.

113^{th} House of Representatives	Total Dataset	Train (80%)	Dev. (10%)	Test (10%)
# roll-call votes	499	400	49	50
# total "Yea" votes	137,926	110,882	12,407	14,637
# total "Nay" votes	68,487	54,874	7,790	5,823
# total "Present"/No votes	8,929	6,934	902	1,093

6.2.6 Experiments

To evaluate the performance of the proposed framework MFCVP, we conduct a set of experiments for predicting individual representative votes and the overall outcome of the roll-call vote for a set of new incoming bills when giving a training set of historical information. Through the conducted experiments, we seek to answer the following research questions:

- *Q1:* How does the proposed framework perform on congressional vote prediction?
- *Q2:* How different factors contribute to the congressional vote prediction?

Next, we describe the dataset followed by experimental setting. Then, we describe the baselines methods and comparison results. We conclude this section by presenting experiments and discussions on factor analysis.

6.2.6.1 Dataset and Data Collection

For our experiments, we have focused on the 113^{th} U.S. Congress House of Representatives. We collected the roll-call vote data along with the sponsor, cosponsor, and party affiliation from the Govtrack database[6]. After obtaining this dataset, we filtered out the roll-call votes not associated with a bill, joint resolution, concurrent resolution, or a simple resolution; for example, roll-call votes related to amendments are not included in our dataset. We obtained ideological embeddings

[6]https://www.govtrack.us

for each of the bills based on the bill's text, which we obtained from the Library of Congress[7]. Ultimately, we split the dataset chronologically into three sets i.e., a train set, a dev set, and a test set as shown in Table 6.5. The training set is constructed with roughly the first 80% of the roll-call votes and all happened before March 5, 2014. Thus, as we mentioned before, to ensure no data leakage, we searched the historical Wikipedia profile pages for each of the representatives to find the date closest to but before March 5, 2014; this data was then collected and used to obtain our ideological embeddings.

6.2.6.2 Experimental Settings

First, we obtain the results for the prediction of individual representative votes. Next, we utilize these individual vote predictions to get the aggregated prediction as to whether the roll-call vote will pass or fail (which is the overall outcome of the roll-call vote). Since our MFCVP framework is flexible in utilizing different classifiers, we utilize a random forest and a deep neural network. For random forest we utilize the scikit-learn library and we used the PyTorch library for our neural network implementation. We denote these two as variants of our framework as MFCVP_RF, and MFCVP_NN, respectively. For the random forest, we use the library default settings. For the deep neural network model (see Figure 6.5), we employ a multi-layer fully connected network with Leaky ReLU (Rectified Linear Unit) [94] as the non-linear activation function. Hyperparameters are set by the grid search via evaluating the framework on the dev set. Using the grid search, the number of layers is set to 5 with 100 hidden units and no regularization is utilized. We utilize ADAM [94] as the optimization algorithm whose learning rate starts from 0.01 and is adjusted dynamically every 100 optimization steps with the decay rate of 0.9. Each simulation is run 2000 steps with the batch size of 100 votes at each step. The embedding size of doc2vec model is set to 50. We repeat each simulation five times and report the average F1 score and accuracy in regards to the test set.

Baselines Methods: To show the effectiveness of our proposed framework MFCVP, we present

[7]https://www.congress.gov

a set of baseline congressional vote prediction methods and discuss why we have selected these baselines from a political standpoint.

- Random Guess: This method performs a random guess when presented with a vote $\tilde{v}_{ij} \in \tilde{\mathcal{V}}$ to predict for voter t_i on a future bill $\tilde{b}_j$. The random guess is based on the class distribution of "yea", "nay", and "no vote" from the set of past votes $\mathcal{V}$. This method is selected to just give context into how difficult this problem is as compared to the most naïve approach.

- Personalized Random Guess: Extending the Random Guess method, here rather than a global class distribution, we extract a personalized class distribution for each voter. In other words, to guess the vote $\tilde{v}_{ij} \in \tilde{\mathcal{V}}$ we extract the class distribution from the set $\{v_{ik} | \forall b_k \in \mathcal{B} \text{ and } v_{ik} \in \mathcal{V}\}$. This method is used to test if indeed individual voters have their own unique patterns in terms of their vote distribution (e.g., one representative might abstain and not vote significantly more often than another).

- Party Voter: This method forces all the representatives to vote aligned with the political parties. More specifically, for predicting a vote $\tilde{v}_{ij} \in \tilde{\mathcal{V}}$ if the voter t_i has the same party affiliation of the sponsor $\tilde{s}_j$ of bill $\tilde{b}_j$, then we predict "yea" and otherwise we predict "nay".

- Sponsor Biased Voter: Given a vote $\tilde{v}_{ij} \in \tilde{\mathcal{V}}$ to be predicted, first the sponsor $\tilde{s}_j$ is obtained from $\tilde{b}_j$, and then we obtain the set of all past votes $\{v_{ik} | v_{ik} \in \mathcal{V} \text{ and } \tilde{s}_j \text{ is the sponsor of } b_k \in \mathcal{B}\}$. This represents the votes that voter t_i has given on past bills b_k that were also sponsored by $\tilde{s}_j$ and we choose the highest vote type over the class distribution. The Sponsor Biased Voter does not necessarily adhere to the political affiliation when voting, but they base their vote on their past experiences with the sponsor of the current bill. In other words, if they have liked (i.e., voted "yea") the past bills of this sponsor, then they will again vote "yea", having similar reasoning for voting "nay" or "no vote" on a bill.

- Top-K Bills: When seeking to predict the vote $\tilde{v}_{ij} \in \tilde{\mathcal{V}}$ this method first obtains the ideological bill embedding E_{b_j} and then finds the closest K bills $b_k \in \mathcal{B}_k$ based on their

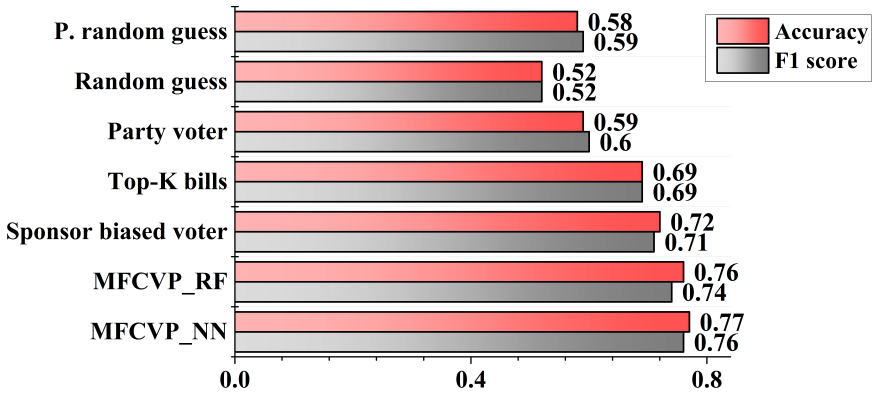


Figure 6.7: Performance evaluation of MFCVP predicting individual representative votes.

embeddings E_{b_k}. Top-K Bills method solely bases their vote on the ideological factors of the proposed bills text. That is to say, predicting the votes using the Top-K Bills method ignores all direct or indirect party affiliations and allows the voter to cast their vote only based on their ideologies. To select hyperparameter K, we varied the value of K in the set $\{1, 3, 5, 8, 10, 20, 30\}$ while predicting on the dev set; which resulted in $K = 8$ being the best performing value. We utilized the Euclidean distance for determining the closest K bills based on their embeddings.

To answer the research *Q1*, we compare the proposed framework MFCVP with the representative baselines for both the local individual representative vote level and also for the global overall roll-call vote. Similar to MFCVP variants, we repeat the Random Guess, Personalized Random Guess methods 5 times and report the average F1 score and accuracy (since they are non-deterministic methods).

6.2.6.3 Individual Representative Vote Predictions

The results are shown in Figure 6.7. Based on the results presented in this figure, we make the following observations:

- Among the baselines methods, sponsor voter approach outperforms the others. This shows the fact that the historical relations between a voter and sponsor have a significant impact

on determining the vote status of a voter for an upcoming bill. Further, as described before, our proposed framework, unlike sponsor voter method, incorporates these relations in a sophisticated way by extracting more principled features from the constructed signed network.

- Comparing Top-K bills method with party voter, we can note that the content of a bill is more important than blindly voting based on a bill's sponsor party. In fact, the low performance of party voter method supports the argument that despite the polarized voting behavior of the U.S. Congress, some representatives adhere to their prior beliefs and ideology instead of merely always voting with or against a proposed bill based on the sponsor's political affiliation.

- Personalized random guess outperforms the random guess. This is not surprising, as personalized random guess incorporates, not effectively though, the prior history of how a representative voted on past bills.

- The variants of the proposed framework MFCVP outperform all baselines methods and in some cases very significantly. This framework, in a comprehensive and sophisticated manner, incorporates various influencing political factors associated with congressional voting. Although MFCVP_NN achieves slightly better performance than MFCVP_RF, we opt to use the random forest for the rest of experiments since it provides with more interpretable insights into the proposed factors.

Therefore, from a local congressional vote perspective, this shows that MFCVP can be utilized as a reliable congressional vote prediction framework. Next, we investigate the global predictions as to whether MFCVP can accurately detect when a proposed bill will pass or fail.

6.2.6.4 Overall Roll-call Vote Outcome Predictions

Here, we utilize the predictions from the local level (i.e., the individual representative vote predictions) to obtain the overall global roll-call vote outcome of whether the bill will pass or fail. The

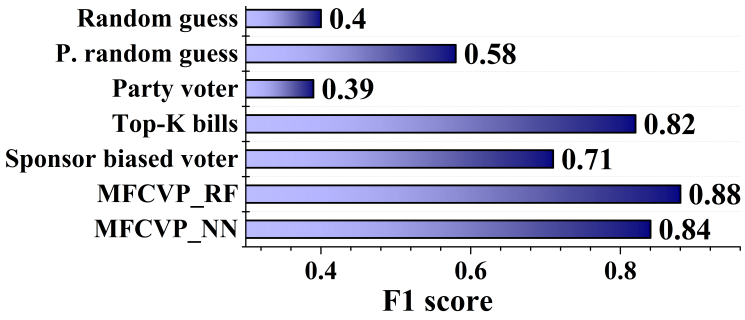


Figure 6.8: Performance evaluation of MFCVP predicting the overall roll-call vote outcome.

results are shown in Figure 6.8. Based on the results presented in this figure, we make the following observations:

- The first observation is that although the personalized random guess performed worse than the party voter for determining individual representative votes, here it significantly outperforms the party voter method. This means that for individual representative votes the better predictor is based on their political party. However, when aggregating all representative votes to the prediction of whether the bill with pass/fail, using the representatives previous voting patterns is better than just considering their party.

- Next we observe a similar swap in that the Top-K bills is now outperforming the Sponsor Voter model. This is interesting since it implies that the overall pass/fail decisions for roll-call votes are happening more likely due to the correlation the voted upon bill has with past similar bills as compared to the relationship all the voters have with the sponsor of the bill. More specifically, this indicates two phenomena: 1) the representatives are quite stable in their ideologies; and 2) when averaged out, the prediction of whether a bill will pass or fail is better predicted through the representatives history according to the bill content as compared to their relation to the sponsor of the proposed bill.

- Although the MFCVP_ NN is better able to predict the local individual representative votes better, it is likely to have slightly overfit the training data (since the models were trained on the local voting patterns) and thus cannot generalize as well when aggregated to the global

level. However, when pairing the random forest model with our MFCVP framework (i.e., MFCVP_ RF), we see it enjoys a better generalization over the neural network variant.

Therefore, based on the results for both the local individual representative predictions as well as the global pass or fail aggregated predictions for the proposed bills, it is clear that our MFCVP framework is a superior and effective methodology for predicting the congressional votes.

6.2.6.5 Political Factor Analysis

The research question *Q2* is concerned with the contribution of political factors for congressional vote prediction. To answer this question, we conduct some experiments for the local individual representative congressional vote prediction. We focus our attention on using the random forest (i.e., MFCVP_RF) as it provides us with feature importance values in an explainable manner.

First, we compute the importance of the three essential factors in our framework i.e., ideological, network and party factors (the latter two are social factors) using the Gini importance [209]. Figure 6.9 shows the importance of these three factors where *Embeddings* indicate the contribution of ideological factors. Based on this figure, we make the following observations:

- Embeddings (i.e., ideological factors) are the most important features in individual vote prediction. This shows that ideological factors play a central role in determining a vote cast on a bill and many representatives adhere to their ethics and beliefs.

- Quite interestingly, network features turn out to be very important. This indicates 1) the interactions and connections among U.S. House of Congress representatives have a significant bearing on the voter's voting behavior, and 2) any political vote prediction should not merely focus on ideological factors and avoid overlooking the role of social networks established among representatives and their historical votes, since it is quite effective in vote prediction.

- In line with the performance of the Party Voter baseline (see Figure 6.7), party features have an inconsequential effect on individual vote prediction. This is politically reassuring

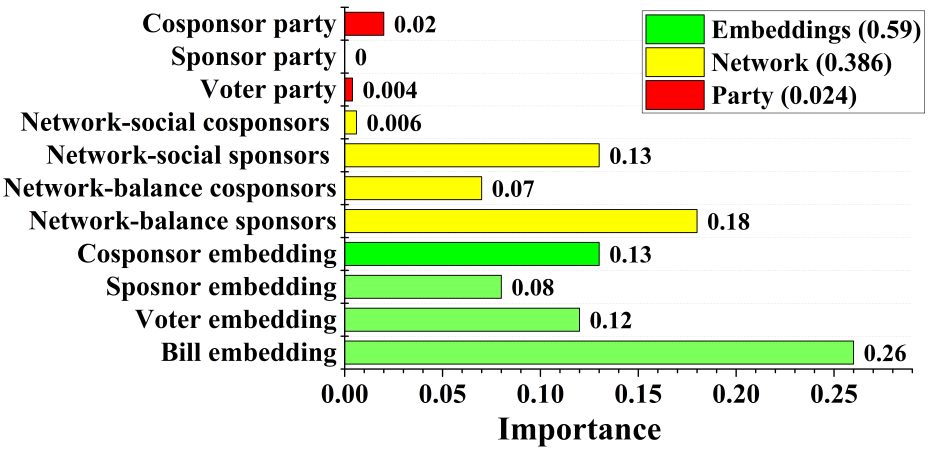


Figure 6.9: Feature analysis using the feature importance values from MFCVP_RF.

as representatives do not submissively follow the inclination of the political party of a bill's sponsor. In the future, we will follow up more on this line of research to investigate if such a phenomenon persists at other points in time throughout history in the U.S. House and Senate or even in other country's political systems.

Now, we narrow down the feature analysis illustrated in Figure 6.9 to investigate contributing features in each of the three overall factors in more detail. We make the following observations according to the results shown in this figure:

- Among the ideological factors, the bill embedding has the highest contribution. This seems reasonable since, after all, it is a bill that is being voted on.

- Interestingly and somehow surprisingly, the embeddings associated with cosponsors are more effective than those of sponsors. It is known that over half of bills being introduced into U.S. Congress are cosponsored [210]. Therefore, based on this fact, it allows our model to categorize whether a given bill has received no cosponsors, or when aggregated across all cosponsors the average representative embedding can provide insight into whether it has received bipartisan support, or only from a single party. This is due to the fact that the embeddings of the representatives are designed such that they hold their ideology and thus likely easily separable in the embedded space for our model.

- Almost the entire contribution of the party features (though very insignificant compared to other factors) stems from the cosponsors party of bills. Similar to ideological factors, this indicates that cosponsors play an important role whose even party affiliation should be taken into account, but as expected, the learned embeddings about the representative's ideology are significantly more important than just knowing the political party they are associated with.

- However, for the network features, we observe the opposite as compared to the embedding and party features in that here the sponsors have a stronger signal. This is likely because aggregating over all the votes a representative has given to a bill proposed by any of the current cosponsors (which tends to be a very large set of votes) results in a more noisy signal as compared to the party or embeddings features (which is just on the order of the number of cosponsors) that can retain most of the information.

- When comparing between the two network features, we observe that the features related to balance theory were more insightful than the social ones that looked at how a voters behavior was with past proposed bills by the same (co)sponsor. One reasoning for this is that the balance theory based features are more principled and looking at a more global view, as compared to the local social features that only look at bills previously proposed by that (co)sponsor. Also, this is likely due to the fact our balance related features are based on pseudo-triangles we extracted from our constructed signed bipartite network (that we note naturally does not contain triangles) and are related to the features extracted in [4] where they were observed to be well suited for predicting whether the sign of a missing link would be positive or negative.

CHAPTER 7

CONCLUSION AND FUTURE DIRECTIONS

In this chapter[1], we provide a summary of our research result and present promising future research directions.

7.1 Summary

In this dissertation, we proposed novel research in the four major directions of network analysis with negative links - (1) signed network measuring; (2) signed network modeling; (3) signed network mining; and (4) signed network applications.

For measuring signed networks, we first performed an analysis of the properties and theories in signed networks taking into consideration both positive and negative links both individually, and holistically together. Next, we performed an initial and comprehensive study of node relevance measurements in signed networks. We built numerous local and global measurements guided by signed network properties and balance theory. We further study the impact of signed relevance measurements on two signed network analysis tasks, i.e., link sign prediction and tie strength prediction. Experimental results demonstrate that (1) dedicated efforts are necessary to build signed relevance measurements with negative links; (2) global methods significantly outperform local methods for link prediction. Thereafter, we developed a Deep Signed Centrality (DeSCent) measure that allowed us to harness the power of deep learning to extract out and learn these complex patterns between positive and negative links while identifying a signed centrality score for each user. Furthermore, the deep framework allows for centrality to be calculated across networks. In other words, training a model in one signed network learns a general enough mapping such that it can be efficiently applied to extract out the centrality of users in other signed networks. Thus, we developed a novel objective for DeSCent based on status theory and balance theory that

[1]Tyler Derr. "Network Analysis with Negative Links." In Proceedings of the 13th ACM International Conference on Web Search and Data Mining (WSDM). 2020.

utilizes higher-order structures. Then, to evaluate the effectiveness of our approach, we conducted experiments using the signed centrality scores for signed link prediction. These experimental results on four real-world signed networks have shown the superiority of DeSCent over other recently proposed signed centrality measures. Furthermore, experiments were performed to validate the usefulness of the deep framework for learning parameters in one network that are general enough to extract meaningful centrality scores in other signed networks, which further strengthens the applicability of the proposed approach.

For modeling signed networks, we proposed our Balanced Signed Chung-Lu model (BSCL), which was the first signed generative network model. BSCL was designed with the objective of preserving three key properties of signed networks - (1) degree distribution; (2) local clustering coefficient; (2) positive/negative link ratio and (3) proportion of balance/unbalanced triangles suggested by balance theory. To achieve this, we introduced a triangle balancing parameter and a sign balancing parameter to control the distribution of formed triangles and signed links, respectively. An automated estimation approach for the two parameters paired with another parameter for controlling how often to create triangles versus inserting a random edge allows BSCL to take as input a signed network, learn appropriate parameters needed to model the key properties, and then output a similar network maintaining the desired properties. In addition, we also provided an initial investigation of balance theory in signed bipartite networks that: (1) extends the definition in the form of signed butterfly isomorphism classes; (2) validated that indeed balanced signed butterflies are found significantly more often as compared to unbalanced in signed bipartite networks; (3) leveraged balance theory for the construction of multiple sign prediction methods; and (4) performed experiments on three real-world signed bipartite networks to provide insight into both balance theory and sign prediction in signed bipartite networks.

For mining signed networks, we proposed the first signed graph convolutional network (SGCN) and examined since unsigned graph convolutional networks (GCNs) are built on the assumption of homophily, that to apply the local neighborhood aggregation the technique could not easily be reused with the introduction of negative links. Hence, we built a novel aggregation scheme for

SGCN built on balance theory and how the positive and negative links interact with each other along the balanced/unbalanced paths we defined. This allowed us to bridge the gap between the recent advances in unsigned GCNs and the domain of signed network analysis. Using our constructed signed graph convolutional network, we performed empirical evaluations through experiments on four real-world signed networks. Comparing against the state-of-the-art signed network embedding algorithms, we had shown the superiority of the SGCNs when performing the classical link sign prediction task. Thereafter, seeking to develop a more universal signed network embedding method, we proposed the idea for network transformation based embedding, namely role-based signed network embedding (ROSE). Essentially the idea is to transform the initial signed network into nodes being transformed to a network where they appear multiple times (one instance of the node per role, e.g., in/out perspective). This led to the creation of an unsigned network consisting of multiple viewpoints for each node from the original network, and after performing a traditional unsigned network embedding we are able to aggregate back the multiple roles/viewpoints to construct an embedding for each of the nodes in the original signed network. Empirically we discovered this is surprisingly very effective and achieved state-of-the-art performance on the link and sign prediction tasks while not relying on social theories.

For applying signed networks, we proposed LIP the joint model that can predict both the polarity between users as well as the interaction polarity scores between users and content generated by other users. The framework is built on harnessing the opinions from both problems, and since we show these opinions are correlated, we were empirically able to help alleviate the cold-start problem that resides in seeking to predict the polarity of both the user-user and user-content links in a real-world dataset. In addition, we presented a comprehensive congressional vote prediction framework MFCVP, that is capable of harnessing both ideological and social factors. We modeled the historical votes in congress between the members of congress and the bills they are to vote on as a signed bipartite network and then extracted network features in addition to other ideological features coming from document embeddings and finally party affiliation features. We furthermore, were able to discover the most influential attributes in congressional vote prediction while discovering

simply predicting based on party affiliations have significantly worse performance. Ultimately, we showed that the proposed signed network features while being informative are having significant role at the congressional vote prediction.

7.2 Future Directions

In this section we present some possible future directions across the major areas of signed network analysis in relation to each of the major directions of measuring, modeling, mining, and applying.

- **Tie Strength Prediction in Signed Networks:** For measuring, in Chapter 2 we presented a set of local and global signed node relevance measurements and performed an empirical evaluation on both the sign prediction and tie strength prediction problems. However, this initial effort just scratched the surface of what can be done in this direction with dedicated efforts. For example, this task can be performed with using only the network structure, or with the added assumption of having additional side information associated with the links and/or nodes, such as link creation timestamps, chat logs between users, comments associated with the links, account creation time, etc.

- **Deep Generative Modeling of Signed Networks:** Recently, there has been a rapid development of generative graph models for unsigned or node attributed networks that utilize deep learning on graph techniques ranging from generative adversarial networks (i.e., GAN-based) [211], recurrent neural networks (i.e., RNN-based) [79], graph recurrent attention network (i.e., GRAN-based) [212] variational autoencoders (VAE-based) [78], and self-attention-based [213]. Thus, to improve upon our BSCL method presented in Chapter 4, the development of a deep generative modeling would be of interest. We refer the readers to a recent survey on this general topic [214].

- **Attack and Defense Methodologies in Signed Networks:** Although there has been some efforts created to both attack and defend against adversarial attacks on traditional network

embedding and graph neural network models (e.g., [215]) this area is recently developed and still evolving. With the given the increased polarization online and social media networks being one of the major areas of concerns for such attacks, both the attacks and defenses are likely to be improved by utilizing this additional information of negative links, such as blocked users, unfollowings, etc. We refer the readers to a recent survey on this general topic [216].

- **Understanding and Predicting Unfollower Relations in Online Social Media:** One important aspect of signed network anlaysis is that it provides a more realistic viewpoint of the underlying system we seek to model with a network (i.e., set of nodes and edges linking them together). However, most signed network analysis thus far has not been able to focus on many mainstream social media sites (such as Twitter), give the lack of data available. For example, Epinions was a product review website that although having negative ties, only later released this this data anatomized for research purposes. Hence, to push the frontier forward in signed social network analysis it is important to understand and characterize which types of interactions online can lead to the modeling of both direct (e.g., unfollowing [217]) and indirect (e.g., negatively commenting on another user's post [218]) negative links in social media.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Jiliang Tang and Huan Liu. Trust in social computing. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 207–208. ACM, 2014.

[2] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. Signed networks in social media. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 1361–1370. ACM, 2010.

[3] Ramanthan Guha, Ravi Kumar, Prabhakar Raghavan, and Andrew Tomkins. Propagation of trust and distrust. In *Proceedings of the 13th International Conference on World Wide Web*, pages 403–412. ACM, 2004.

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

[5] Patricia Victor, Chris Cornelis, Martine De Cock, and Ankur Teredesai. Trust-and distrust-based recommendations for controversial reviews. In *Web Science Conference (WebSci'09: Society On-Line)*, 2009.

[6] Hao Ma, Michael R Lyu, and Irwin King. Learning to recommend with trust and distrust relationships. In *Proceedings of the third ACM conference on Recommender systems*, pages 189–196. ACM, 2009.

[7] Tyler Derr. Network analysis with negative links. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 917–918, 2020.

[8] Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*, pages 415–444, 2001.

[9] Anatol Rapoport and William J Horvath. A study of a large sociogram. *Behavioral science*, 6(4):279–291, 1961.

[10] Derek J De Solla Price. Networks of scientific papers. *Science*, pages 510–515, 1965.

[11] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[12] John Scott. *Social network analysis*. Sage, 2012.

[13] Jundong Li, Jiliang Tang, Yilin Wang, Yali Wan, Yi Chang, and Huan Liu. Understanding and predicting delay in reciprocal relations. *arXiv preprint arXiv:1703.01393*, 2017.

[14] Irawati Karmarkar Karve and Yashwant Bhaskar Damle. *Group relations in village community*, volume 24. Published by SM Katre for the Deccan College Post-graduate and Research, 1963.

[15] Leo Katz and James H Powell. Measurement of the tendency toward reciprocation of choice. *Sociometry*, 18(4):403–409, 1955.

[16] A Ramachandra Rao and Suraj Bandyopadhyay. Measures of reciprocity in a social network. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 141–188, 1987.

[17] Mark Newman. *Networks: An Introduction*. Oxford University Press, Inc., New York, NY, USA, 2010.

[18] R Duncan Luce and Albert D Perry. A method of matrix analysis of group structure. *Psychometrika*, 14(2):95–116, 1949.

[19] Paul W Holland and Samuel Leinhardt. Holland and leinhardt reply: some evidence on the transitivity of positive interpersonal sentiment, 1972.

[20] PF Lazarsfeld and RK. Merton. Friendship as a social process: a substantive and methodological analysis interpersonal sentiment, 1954.

[21] Kibae Kim and Jörn Altmann. Effect of homophily on network formation. *Communications in Nonlinear Science and Numerical Simulation*, 44:482–494, 2017.

[22] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2016.

[23] Srijan Kumar, Francesca Spezzano, VS Subrahmanian, and Christos Faloutsos. Edge weight prediction in weighted signed networks. In *ICDM*, pages 221–230, 2016.

[24] Jérôme Kunegis, Andreas Lommatzsch, and Christian Bauckhage. The slashdot zoo: mining a social network with negative edges. In *Proceedings of the 18th International conference on World wide web*, pages 741–750. ACM, 2009.

[25] Paolo Massa and Paolo Avesani. Controversial users demand local trust metrics: An experimental study on epinions. com community. In *Proceedings of the National Conference on artificial Intelligence*, volume 20, page 121. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

[26] Shuang-Hong Yang, Alexander J Smola, Bo Long, Hongyuan Zha, and Yi Chang. Friend or frenemy?: predicting signed ties in social networks. In *Proceedings of the 35th International ACM SIGIR conference on Research and development in information retrieval*, pages 555–564. ACM, 2012.

[27] Jiliang Tang, Shiyu Chang, Charu Aggarwal, and Huan Liu. Negative link prediction in social media. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 87–96. ACM, 2015.

[28] Michael Szell, Renaud Lambiotte, and Stefan Thurner. Multirelational organization of large-scale social networks in an online world. *Proceedings of the National Academy of Sciences*, 107(31):13636–13641, 2010.

[29] Fritz Heider. Attitudes and cognitive organization. *The Journal of psychology*, 21(1):107–112, 1946.

[30] Dorwin Cartwright and Frank Harary. Structural balance: a generalization of heider's theory. *Psychological review*, 63(5):277, 1956.

[31] Frank Harary et al. On the notion of balance of a signed graph. *The Michigan Mathematical Journal*, 2(2):143–146, 1953.

[32] Dorwin Cartwright and Terry C Gleason. The number of paths and cycles in a digraph. *Psychometrika*, 31(2):179–199, 1966.

[33] Nancy M Henley, Robert B Horsfall, and Clinton B De Soto. Goodness of figure and social structure. *Psychological Review*, 76(2):194, 1969.

[34] James A Davis. Clustering and structural balance in graphs. *Human relations*, 1967.

[35] Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004.

[36] Giuseppe Facchetti, Giovanni Iacono, and Claudio Altafini. Computing global structural balance in large-scale signed social networks. *Proceedings of the National Academy of Sciences*, 108(52):20953–20958, 2011.

[37] Evimaria Terzi and Marco Winkler. A spectral algorithm for computing social balance. In *Algorithms and models for the web graph*, pages 1–13. Springer, 2011.

[38] Lars Backstrom and Jure Leskovec. Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the fourth ACM International conference on Web search and data mining*, pages 635–644. ACM, 2011.

[39] Zhijun Yin, Manish Gupta, Tim Weninger, and Jiawei Han. A unified framework for link recommendation using random walks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on*, pages 152–159. IEEE, 2010.

[40] Smriti Bhagat, Graham Cormode, and S Muthukrishnan. Node classification in social networks. In *Social network data analytics*, pages 115–148. Springer, 2011.

[41] Lei Tang and Huan Liu. Community detection and mining in social media. *Synthesis Lectures on Data Mining and Knowledge Discovery*, 2(1):1–137, 2010.

[42] Hongzhi Yin, Bin Cui, Jing Li, Junjie Yao, and Chen Chen. Challenging the long tail recommendation. *Proc. VLDB Endow.*, 5(9):896–907, May 2012.

[43] Lada A Adamic and Eytan Adar. Friends and neighbors on the web. *Social networks*, 25(3):211–230, 2003.

[44] Hanghang Tong, Christos Faloutsos, and Jia-yu Pan. Fast random walk with restart and its applications. In *Data Mining, 2006. ICDM'06. Sixth International Conference on*, pages 613–622. IEEE, 2006.

[45] Ranjay Gulati. Network location and learning: The influence of network resources and firm capabilities on alliance formation. *Strategic management journal*, 20(5):397–420, 1999.

[46] Arzucan Özgür, Thuy Vu, Güneş Erkan, and Dragomir R Radev. Identifying gene-disease associations using centrality on a literature mined gene-interaction network. *Bioinformatics*, 24(13):i277–i285, 2008.

[47] Paolo Crucitti, Vito Latora, and Sergio Porta. Centrality measures in spatial networks of urban streets. *Physical Review E*, 73(3):036125, 2006.

[48] Roger Guimera, Stefano Mossa, Adrian Turtschi, and LA Nunes Amaral. The worldwide air transportation network: Anomalous centrality, community structure, and cities' global roles. *Proceedings of the National Academy of Sciences*, 102(22):7794–7799, 2005.

[49] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[50] Kiyana Zolfaghar and Abdollah Aghaie. Mining trust and distrust relationships in social web applications. In *Intelligent Computer Communication and Processing (ICCP), 2010 IEEE International Conference on*, pages 73–80. IEEE, 2010.

[51] Vincent Traag, Yurii Nesterov, and Paul Van Dooren. Exponential ranking: Taking into account negative links. *Social Informatics*, pages 192–202, 2010.

[52] Panagiotis Symeonidis and Eleftherios Tiakas. Transitive node similarity: predicting and recommending links in signed social networks. *World Wide Web*, 17(4):743–776, 2014.

[53] Jinhong Jung, Woojeong Jin, Lee Sael, and U. Kang. Personalized ranking in signed networks using signed random walk with restart. In *IEEE 16th International Conference on Data Mining, ICDM 2016, December 12-15, 2016, Barcelona, Spain*, pages 973–978, 2016.

[54] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

[55] Jiliang Tang, Yi Chang, Charu Aggarwal, and Huan Liu. A survey of signed network mining in social media. *ACM Computing Surveys (CSUR)*, 49(3):42, 2016.

[56] Francois Lorrain and Harrison C White. Structural equivalence of individuals in social networks. *The Journal of mathematical sociology*, 1(1):49–80, 1971.

[57] Leo Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

[58] Hung-Hsuan Chen and C Lee Giles. Ascos: an asymmetric network structure context similarity measure. In *Advances in Social Networks Analysis and Mining (ASONAM), 2013 IEEE/ACM International Conference on*, pages 442–449. IEEE, 2013.

[59] Roger A. Horn and Charles R. Johnson. *Matrix analysis*. Cambridge University Press, Cambridge, second edition, 2013.

[60] Linyuan Lü and Tao Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150–1170, 2011.

[61] Kai-Yang Chiang, Nagarajan Natarajan, Ambuj Tewari, and Inderjit S Dhillon. Exploiting longer cycles for link prediction in signed networks. In *Proceedings of the 20th ACM International conference on Information and knowledge management*, pages 1157–1162. ACM, 2011.

[62] Eric Gilbert and Karrie Karahalios. Predicting tie strength with social media. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 211–220. ACM, 2009.

[63] Rongjing Xiang, Jennifer Neville, and Monica Rogati. Modeling relationship strength in online social networks. In *Proceedings of the 19th International conference on World wide web*, pages 981–990. ACM, 2010.

[64] Indika Kahanda and Jennifer Neville. Using transactional information to predict link strength in online social networks. In *Third International AAAI Conference on Weblogs and Social Media*, 2009.

[65] Moshen Shahriari and Mahdi Jalili. Ranking nodes in signed social networks. *Social Network Analysis and Mining*, 4(1):172, 2014.

[66] Cristobald de Kerchove and Paul Van Dooren. The pagetrust algorithm: How to rank web pages when negative links are allowed? In *Proceedings of the 2008 SIAM International Conference on Data Mining*, pages 346–352. SIAM, 2008.

[67] Jiliang Tang, Xia Hu, and Huan Liu. Is distrust the negation of trust?: the value of distrust in social media. In *Proceedings of the 25th ACM conference on Hypertext and social media*, pages 148–157. ACM, 2014.

[68] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[69] Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[70] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[71] Brian D Ripley. *Pattern recognition and neural networks*. Cambridge university press, 2007.

[72] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[73] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[74] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. Community preserving network embedding. In *AAAI*, pages 7203–209, 2017.

[75] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1025–1035, 2017.

[76] Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C Aggarwal, and Thomas S Huang. Heterogeneous network embedding via deep architectures. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 119–128. ACM, 2015.

[77] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 701–710. ACM, 2014.

[78] Aditya Grover, Aaron Zweig, and Stefano Ermon. Graphite: Iterative generative modeling of graphs. In *International Conference on Machine Learning*, pages 2434–2444, 2019.

[79] Jiaxuan You, Rex Ying, Xiang Ren, William L Hamilton, and Jure Leskovec. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *ICML*, 2018.

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

[81] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *In NIPS Workshop on Bayesian Deep Learning*, 2016.

[82] Suhang Wang, Jiliang Tang, Charu Aggarwal, and Huan Liu. Linked document embedding for classification. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 115–124. ACM, 2016.

[83] Wenqi Fan, Tyler Derr, Yao Ma, Jianping Wang, Jiliang Tang, and Qing Li. Deep adversarial social recommendation. *arXiv preprint arXiv:1905.13160*, 2019.

[84] Tyler Derr, Hamid Karimi, Xiaorui Liu, Jiejun Xu, and Jiliang Tang. Deep adversarial network alignment. *arXiv preprint arXiv:1902.10307*, 2019.

[85] Haochen Liu, Tyler Derr, Zitao Liu, and Jiliang Tang. Say what i want: Towards the dark side of neural dialogue models. *arXiv preprint arXiv:1909.06044*, 2019.

[86] Hamid Karimi, Tyler Derr, and Jiliang Tang. Characterizing the decision boundary of deep neural networks. *arXiv preprint arXiv:1912.11460*, 2019.

[87] Suhang Wang, Jiliang Tang, Charu Aggarwal, Yi Chang, and Huan Liu. Signed network embedding in social media. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 327–335. SIAM, 2017.

[88] Mohammad Raihanul Islam, B Aditya Prakash, and Naren Ramakrishnan. Signet: scalable embeddings for signed networks. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 157–169. Springer, 2018.

[89] Ji-Eun Lee Junghwan Kim, Haekyu Park and U Kang. Side: Representation learning in signed directed networks. In *Proceedings of the 27th International Conference on World Wide Web (The Web Conference)*, 2018.

[90] Phillip Bonacich. Power and centrality: A family of measures. *American journal of sociology*, 92(5):1170–1182, 1987.

[91] David Easley and Jon Kleinberg. Strong and weak ties. In *Networks, Crowds, and Markets: Reasoning about a Highly Connected World*, pages 47–84, 2010.

[92] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[93] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. ICML*, 2013.

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

[95] Mark EJ Newman. Clustering and preferential attachment in growing networks. *Physical review E*, 64(2):025102, 2001.

[96] Jinhong Jung, Woojeong Jin, Lee Sael, and U Kang. Personalized ranking in signed networks using signed random walk with restart. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, pages 973–978. IEEE, 2016.

[97] Fan Chung and Linyuan Lu. The average distances in random graphs with given expected degrees. *Proceedings of the National Academy of Sciences*, 99(25):15879–15882, 2002.

[98] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Proceedings of the eleventh ACM SIGKDD International conference on Knowledge discovery in data mining*, pages 177–187. ACM, 2005.

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

[100] Stephen Mussmann, John Moore, Joseph J. Pfeiffer, III, and Jennifer Neville. Incorporating assortativity and degree dependence into scalable network models. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 238–246. AAAI Press, 2015.

[101] Mark EJ Newman. Assortative mixing in networks. *Physical review letters*, 89(20):208701, 2002.

[102] Joseph J Pfeiffer, Timothy La Fond, Sebastian Moreno, and Jennifer Neville. Fast generation of large scale social networks while incorporating transitive closures. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, pages 154–165. IEEE, 2012.

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

[104] Comandur Seshadhri, Tamara G Kolda, and Ali Pinar. Community structure and scale-free collections of erdős-rényi graphs. *Physical Review E*, 85(5):056109, 2012.

[105] Petter Holme and Beom Jun Kim. Growing scale-free networks with tunable clustering. *Physical review E*, 65(2):026107, 2002.

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

[107] Albert-Laszlo Barabasi and Zoltan N Oltvai. Network biology: understanding the cell's functional organization. *Nature reviews genetics*, 5(2):101, 2004.

[108] Erik Volz and Lauren Ancel Meyers. Susceptible–infected–recovered epidemics in dynamic contact networks. *Proceedings of the Royal Society B: Biological Sciences*, 274(1628):2925–2934, 2007.

[109] Mohammad Malekzadeh, M Fazli, P Jalali Khalilabadi, H Rabiee, and M Safari. Social balance and signed network formation games. 2011.

[110] Jure Leskovec and Julian J Mcauley. Learning to discover social circles in ego networks. In *Advances in neural information processing systems*, pages 539–547, 2012.

[111] Yanhua Li, Wei Chen, Yajun Wang, and Zhi-Li Zhang. Influence diffusion dynamics and influence maximization in social networks with friend and foe relationships. In *Proceedings of the sixth ACM International conference on Web search and Data Mining*, pages 657–666. ACM, 2013.

[112] Ali Pinar, Comandur Seshadhri, and Tamara G Kolda. The similarity between stochastic kronecker and chung-lu graph models. In *Proceedings of the 2012 SIAM International Conference on Data Mining*, pages 1071–1082. SIAM, 2012.

[113] Thomas Schank and Dorothea Wagner. Finding, counting and listing all triangles in large graphs, an experimental study. In *WEA*. Springer, 2005.

[114] Vida Vukašinović, Jurij Šilc, and Risth Škrekovski. Modeling acquaintance networks based on balance theory. *International Journal of Applied Mathematics and Computer Science*, 24(3):683–696, 2014.

[115] Mark Ludwig and Peter Abell. An evolutionary model of social networks. *European Physical Journal B–Condensed Matter*, 58(1), 2007.

[116] Tyler Derr and Jiliang Tang. Congressional vote analysis using signed networks. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 1501–1502. IEEE, 2018.

[117] Tyler Derr, Hamid Karimi, Aaron Brookhouse, and Jiliang Tang. Multi-factor congressional vote prediction. In *Advances in Social Networks Analysis and Mining (ASONAM), 2019 IEEE/ACM International Conference on*. IEEE, 2019.

[118] Tyler Derr, Yao Ma, and Jiliang Tang. Signed graph convolutional networks. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 929–934. IEEE, 2018.

[119] Phillip Bonacich and Paulette Lloyd. Calculating status with negative relations. *Social Networks*, 26(4):331–338, 2004.

[120] Zhaoming Wu, Charu C Aggarwal, and Jimeng Sun. The troll-trust model for ranking in signed networks. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 447–456. ACM, 2016.

[121] Pranay Anchuri and Malik Magdon-Ismail. Communities and balance in signed networks: A spectral approach. In *Advances in Social Networks Analysis and Mining (ASONAM), 2012 IEEE/ACM International Conference on*, pages 235–242. IEEE, 2012.

[122] Sinan G Aksoy, Tamara G Kolda, and Ali Pinar. Measuring and modeling bipartite graphs with community structure. *Journal of Complex Networks*, 5(4):581–603, 2017.

[123] Seyed-Vahid Sanei-Mehri, Ahmet Erdem Sariyuce, and Srikanta Tirthapura. Butterfly counting in bipartite networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2150–2159. ACM, 2018.

[124] Amin Javari and Mahdi Jalili. Cluster-based collaborative filtering for sign prediction in social networks with positive and negative links. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(2):24, 2014.

[125] Tongda Zhang, Haomiao Jiang, Zhouxiao Bao, and Yingfeng Zhang. Characterization and edge sign prediction in signed networks. *Journal of Industrial and Intelligent Information Vol*, 1(1), 2013.

[126] Athanasios Papaoikonomou, Magdalini Kardara, Konstantinos Tserpes, and Dora Varvarigou. Edge sign prediction in social networks via frequent subgraph discovery. *IEEE Internet Computing*, 2014.

[127] Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed Zaki. Link prediction using supervised learning. In *SDM06: workshop on link analysis, counter-terrorism and security*, 2006.

[128] Zhengdong Lu, Berkant Savas, Wei Tang, and Inderjit S Dhillon. Supervised link prediction using multiple sources. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 923–928. IEEE, 2010.

[129] Aditya Krishna Menon and Charles Elkan. Link prediction via matrix factorization. In *Machine Learning and Knowledge Discovery in Databases*, pages 437–452. Springer, 2011.

[130] Jiliang Tang, Charu Aggarwal, and Huan Liu. Recommendations in signed social networks. In *Proceedings of the 25th International Conference on World Wide Web*, pages 31–40. International World Wide Web Conferences Steering Committee, 2016.

[131] Cho-Jui Hsieh, Kai-Yang Chiang, and Inderjit S Dhillon. Low rank modeling of signed networks. In *Proceedings of the 18th ACM SIGKDD International conference on Knowledge discovery and Data Mining*, pages 507–515. ACM, 2012.

[132] Rana Forsati, Iman Barjasteh, Farzan Masrour, Abdol-Hossein Esfahanian, and Hayder Radha. Pushtrust: An efficient recommendation algorithm by leveraging trust and distrust relations. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 51–58. ACM, 2015.

[133] Tyler Derr, Chenxing Wang, Suhang Wang, and Jiliang Tang. Relevance measurements in online signed social networks. In *Proceedings of the 14th International Workshop on Mining and Learning with Graphs (MLG)*, 2018.

[134] László Lovász. Random walks on graphs: A survey. *Combinatorics, Paul erdos is eighty*, 2(1):1–46, 1993.

[135] Man Gao, Ling Chen, Bin Li, Yun Li, Wei Liu, and Yong-cheng Xu. Projection-based link prediction in a bipartite network. *Information Sciences*, 376:158–171, 2017.

[136] Tao Zhou, Jie Ren, Matúš Medo, and Yi-Cheng Zhang. Bipartite network projection and personal recommendation. *Physical Review E*, 76(4):046115, 2007.

[137] Katharina Anna Zweig and Michael Kaufmann. A systematic approach to the one-mode projection of bipartite graphs. *Social Network Analysis and Mining*, 1(3):187–218, 2011.

[138] Muhammed A Yildirim and Michele Coscia. Using random walks to generate associations between objects. *PloS one*, 9(8):e104813, 2014.

[139] Jie Tang, Tiancheng Lou, and Jon Kleinberg. Inferring social ties across heterogenous networks. In *Proceedings of the fifth ACM International conference on Web search and Data Mining*, pages 743–752. ACM, 2012.

[140] Jihang Ye, Hong Cheng, Zhe Zhu, and Minghua Chen. Predicting positive and negative links in signed social networks by transfer learning. In *Proceedings of the 22nd International conference on World Wide Web*, pages 1477–1488. International World Wide Web Conferences Steering Committee, 2013.

[141] Ahmed Hassan, Amjad Abu-Jbara, and Dragomir Radev. Detecting subgroups in online discussions by modeling positive and negative relations among participants. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 59–70. Association for Computational Linguistics, 2012.

[142] Kai-Yang Chiang, Joyce Jiyoung Whang, and Inderjit S Dhillon. Scalable clustering of signed networks using balance normalized cut. In *Proceedings of the 21st ACM International conference on Information and knowledge management*, pages 615–624. ACM, 2012.

[143] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *International Conference on Machine Learning (ICML)*, pages 2014–2023, 2016.

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

[145] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2224–2232, 2015.

[146] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. Spectral networks and locally connected networks on graphs. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

[147] Tyler Derr, Yao Ma, Wenqi Fan, Xiaorui Liu, Charu Aggarwal, and Jiliang Tang. Epidemic graph convolutional network. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 160–168, 2020.

[148] Wei Jin, Tyler Derr, Haochen Liu, Yiqi Wang, Suhang Wang, Zitao Liu, and Jiliang Tang. Self-supervised learning on graphs: Deep insights and new direction. *arXiv preprint arXiv:2006.10141*, 2020.

[149] Daixin Wang, Peng Cui, and Wenwu Zhu. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD International conference on Knowledge Discovery and Data Mining*, pages 1225–1234. ACM, 2016.

[150] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International conference on Knowledge Discovery and Data Mining*, pages 855–864. ACM, 2016.

[151] Shaosheng Cao, Wei Lu, and Qiongkai Xu. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM)*, pages 891–900. ACM, 2015.

[152] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. Community preserving network embedding. In *AAAI*, pages 7203–209, 2017.

[153] Jérôme Kunegis, Stephan Schmidt, Andreas Lommatzsch, Jürgen Lerner, Ernesto W De Luca, and Sahin Albayrak. Spectral analysis of signed graphs for clustering, prediction and visualization. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, pages 559–570. SIAM, 2010.

[154] Marek Cygan, Marcin Pilipczuk, Michał Pilipczuk, and Jakub Onufry Wojtaszczyk. Sitting closer to friends than enemies, revisited. *Theory of computing systems*, 56(2):394–405, 2015.

[155] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering*, 31(5):833–852, 2018.

[156] Junghwan Kim, Haekyu Park, Ji-Eun Lee, and U Kang. Side: Representation learning in signed directed networks. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 509–518. International World Wide Web Conferences Steering Committee, 2018.

[157] Yiqi Chen, Tieyun Qian, Huan Liu, and Ke Sun. Bridge: Enhanced signed directed network embedding. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 773–782. ACM, 2018.

[158] Suhang Wang, Charu Aggarwal, Jiliang Tang, and Huan Liu. Attributed signed network embedding. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 137–146. ACM, 2017.

[159] Feng Xue, Xiangnan He, Xiang Wang, Jiandong Xu, Kai Liu, and Richang Hong. Deep item-based collaborative filtering for top-n recommendation. *ACM Transactions on Information Systems (TOIS)*, 37(3):33, 2019.

[160] William L Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*, 2017.

[161] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. Asymmetric transitivity preserving graph embedding. In *Proceedings of the 22nd ACM SIGKDD International conference on Knowledge Discovery and Data Mining*, pages 1105–1114. ACM, 2016.

[162] Hongwei Wang, Fuzheng Zhang, Min Hou, Xing Xie, Minyi Guo, and Qi Liu. Shine: signed heterogeneous information network embedding for sentiment link prediction. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 592–600. ACM, 2018.

[163] Shuhan Yuan, Xintao Wu, and Yang Xiang. Sne: signed network embedding. In *Pacific-Asia conference on knowledge Discovery and Data Mining*, pages 183–195. Springer, 2017.

[164] Jinhong Jung, Woojeong Jin, Lee Sael, and U Kang. Personalized ranking in signed networks using signed random walk with restart. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 973–978. IEEE, 2016.

[165] Tyler Derr, Cassidy Johnson, Yi Chang, and Jiliang Tang. Balance in signed bipartite networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1221–1230. ACM, 2019.

[166] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596*, 2019.

[167] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)*, 52(1):5, 2019.

[168] Moira Burke and Robert Kraut. Mopping up: modeling wikipedia promotion decisions. In *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, pages 27–36, 2008.

[169] Cliff AC Lampe, Erik Johnston, and Paul Resnick. Follow the reader: filtering comments on slashdot. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1253–1262, 2007.

[170] Alec Kirkley, George T Cantwell, and MEJ Newman. Balance in signed networks. *Physical Review E*, 99(1):012320, 2019.

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

[172] Jiliang Tang, Xia Hu, and Huan Liu. Social recommendation: a review. *Social Network Analysis and Mining*, 3(4):1113–1133, 2013.

[173] Wei Li, Pengyi Fan, Pei Li, Hui Wang, and Yiguang Pan. An opinion spreading model in signed networks. *Modern Physics Letters B*, 27(12), 2013.

[174] Yanhua Li, Wei Chen, Yajun Wang, and Zhi-Li Zhang. Voter model on signed social networks. *Internet Mathematics*, 2014.

[175] Rana Forsati, Mehrdad Mahdavi, Mehrnoush Shamsfard, and Mohamed Sarwat. Matrix factorization with explicit trust and distrust side information for improved social recommendation. *ACM Transactions on Information Systems (TOIS)*, 32(4):17, 2014.

[176] Samaneh Moghaddam, Mohsen Jamali, and Martin Ester. Etf: extended tensor factorization model for personalizing prediction of review helpfulness. In *Proceedings of the fifth ACM International conference on Web search and data mining*, pages 163–172. ACM, 2012.

[177] Samaneh Moghaddam, Mohsen Jamali, and Martin Ester. Review recommendation: personalized prediction of the quality of online reviews. In *Proceedings of the 20th ACM International conference on Information and knowledge management*, pages 2249–2252. ACM, 2011.

[178] Jiliang Tang, Huiji Gao, Xia Hu, and Huan Liu. Context-aware review helpfulness rating prediction. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 1–8. ACM, 2013.

[179] Suhang Wang, Jiliang Tang, and Huan Liu. Toward dual roles of users in recommender systems. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1651–1660. ACM, 2015.

[180] Panagiotis Symeonidis and Nikolaos Mantas. Spectral clustering for link prediction in social networks with positive and negative links. *Social Network Analysis and Mining*, 3(4):1433–1447, 2013.

[181] Kai-Yang Chiang, Cho-Jui Hsieh, Nagarajan Natarajan, Inderjit S Dhillon, and Ambuj Tewari. Prediction and clustering in signed networks: a local to global perspective. *Journal of Machine Learning Research*, 15(1):1177–1213, 2014.

[182] Nenad Trinajstic. *Chemical graph theory*. Routledge, 2018.

[183] Jeffrey M Dambacher, Hiram W Li, and Philippe A Rossignol. Relevance of community structure in assessing indeterminacy of ecological predictions. *Ecology*, 83(5):1372–1385, 2002.

[184] G Toulouse. Theory of the frustration effect in spin glasses: I. *Spin Glass Theory and Beyond: An Introduction to the Replica Method and Its Applications*, 9:99, 1987.

[185] Zachary P Neal. A sign of the times? weak and strong polarization in the us congress, 1973–2016. *Social Networks*, 2018.

[186] Zhiwei Wang, Tyler Derr, Dawei Yin, and Jiliang Tang. Understanding and predicting weight loss with mobile social networking data. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1269–1278. ACM, 2017.

[187] Hamid Karimi, Tyler Derr, Kaitlin T Torphy, Kenneth A Frank, and Jiliang Tang. Towards improving sample representativeness of teachers on online social media: A case study on pinterest. In *International Conference on Artificial Intelligence in Education*, pages 130–134. Springer, 2020.

[188] Hamid Karimi, Kaitlin T Torphy, Tyler Derr, Kenneth A Frank, and Jiliang Tang. Characterizing teacher connections in online social media: A case study on pinterest. In *Proceedings of the Seventh ACM Conference on Learning@ Scale*, pages 249–252, 2020.

[189] Priyanka Agrawal, Vikas K Garg, and Ramasuri Narayanam. Link label prediction in signed social networks. In *Proceedings of the Twenty-Third International joint conference on Artificial Intelligence*, pages 2591–2597. AAAI Press, 2013.

[190] Xiaoyuan Su and Taghi M Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009:4, 2009.

[191] Takahiko Ito, Masashi Shimbo, Taku Kudo, and Yuji Matsumoto. Application of kernels to link analysis. In *Proceedings of the eleventh ACM SIGKDD International conference on Knowledge discovery in data mining*, pages 586–592. ACM, 2005.

[192] Simon Jackman. Multidimensional analysis of roll call data via bayesian simulation: Identification, estimation, inference, and model checking. *Political Analysis*, 9(3):227–241, 2001.

[193] Peter Kraft, Hirsh Jain, and Alexander M Rush. An embedding model for predicting roll-call votes. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2066–2070, 2016.

[194] Nolan McCarty, Keith T Poole, and Howard Rosenthal. *Polarized America: The dance of ideology and unequal riches*. 2016.

[195] Joshua Clinton, Simon Jackman, and Douglas Rivers. The statistical analysis of roll call data. *American Political Science Review*, 98(2):355–370, 2004.

[196] Mason A Porter, Peter J Mucha, Mark EJ Newman, and Casey M Warmbrand. A network analysis of committees in the us house of representatives. *Proceedings of the National Academy of Sciences*, 102(20):7057–7062, 2005.

[197] Carlo Dal Maso, Gabriele Pompa, Michelangelo Puliga, Gianni Riotta, and Alessandro Chessa. Voting behavior, coalitions and government strength through a complex network analysis. *PloS one*, 9(12):e116046, 2014.

[198] Sean Gerrish and David M Blei. Predicting legislative roll calls from text. In *International Conference on Machine Learning*, pages 489–496, 2011.

[199] In Song Kim, John Londregan, and Marc Ratkovic. Voting, speechmaking, and the dimensions of conflict in the us senate. In *Annual Meeting of the Midwest Political Science Association*, 2014.

[200] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196, 2014.

[201] Jey Han Lau and Timothy Baldwin. An empirical evaluation of doc2vec with practical insights into document embedding generation. *arXiv preprint arXiv:1607.05368*, 2016.

[202] Hamid Karimi, Courtland VanDam, Liyang Ye, and Jiliang Tang. End-to-end compromised account detection. In *International Conference on Advances in Social Networks Analysis and Mining*, pages 314–321. IEEE, 2018.

[203] Clio Andris, David Lee, Marcus J Hamilton, Mauro Martino, Christian E Gunning, and John Armistead Selden. The rise of partisanship and super-cooperators in the us house of representatives. *PloS one*, 10(4):e0123507, 2015.

[204] Tyler Derr, Charu Aggarwal, and Jiliang Tang. Signed network modeling based on structural balance theory. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 557–566. ACM, 2018.

[205] Tyler Derr, Zhiwei Wang, and Jiliang Tang. Opinions power opinions: Joint link and interaction polarity predictions in signed networks. In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 363–366. IEEE, 2018.

[206] Tyler Derr, Yao Ma, and Jiliang Tang. Signed graph convolutional networks. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 929–934. IEEE, 2018.

[207] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[208] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[209] L. Breiman, J. Friedman, C.J. Stone, and R.A. Olshen. *Classification and Regression Trees*. The Wadsworth and Brooks-Cole statistics-probability series. Taylor & Francis, 1984.

[210] Rick K Wilson and Cheryl D Young. Cosponsorship in the us congress. *Legislative Studies Quarterly*, pages 25–43, 1997.

[211] Aleksandar Bojchevski, Oleksandr Shchur, Daniel Zügner, and Stephan Günnemann. Netgan: Generating graphs via random walks. In *International Conference on Machine Learning*, pages 610–619, 2018.

[212] Renjie Liao, Yujia Li, Yang Song, Shenlong Wang, Will Hamilton, David K Duvenaud, Raquel Urtasun, and Richard Zemel. Efficient graph generation with graph recurrent attention networks. In *Advances in Neural Information Processing Systems*, pages 4255–4265, 2019.

[213] Sohil Atul Shah and Vladlen Koltun. Auto-decoding graphs. *arXiv preprint arXiv:2006.02879*, 2020.

[214] Xiaojie Guo and Liang Zhao. A systematic survey on deep generative models for graph generation. *arXiv preprint arXiv:2007.06686*, 2020.

[215] Yao Ma, Suhang Wang, Tyler Derr, Lingfei Wu, and Jiliang Tang. Attacking graph convolutional networks via rewiring. *arXiv preprint arXiv:1906.03750*, 2019.

[216] Wei Jin, Yaxin Li, Han Xu, Yiqi Wang, and Jiliang Tang. Adversarial attacks and defenses on graphs: A review and empirical study. *arXiv preprint arXiv:2003.00653*, 2020.

[217] Haewoon Kwak, Hyunwoo Chun, and Sue Moon. Fragile online relationship: a first look at unfollow dynamics in twitter. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1091–1100, 2011.

[218] Betty van Aken, Julian Risch, Ralf Krestel, and Alexander Löser. Challenges for toxic comment classification: An in-depth error analysis. *arXiv preprint arXiv:1809.07572*, 2018.