SigGAN : Adversarial Model for Learning Signed Relationships in Networks

Roshni Chakraborty  
Indian Institute of Technology  
Patna  
Patna, India  
roshni pcs15@iitp.ac.in

Ritwika Das  
National Institute of Technology  
Durgapur  
Durgapur, India  
ritwikadas1@gmail.com

Joydeep Chandra  
Indian Institute of Technology  
Patna  
India  
joydeep@iitp.ac.in

ABSTRACT

Signed link prediction in graphs is an important problem that has applications in diverse domains. It is a binary classification problem that predicts whether an edge between a pair of nodes is positive or negative. Existing approaches for link prediction in unsigned networks cannot be directly applied for signed link prediction due to their inherent differences. Further, additional structural constraints, like, the structural balance property of the signed networks must be considered for signed link prediction. Recent signed link prediction approaches generate node representations using either generative models or discriminative models. Inspired by the recent success of Generative Adversarial Network (GAN) based models which comprises of a discriminator and generator in several applications, we propose a Generative Adversarial Network (GAN) based model for signed networks, SigGAN. It considers the requirements of signed networks, such as, integration of information from negative edges, high imbalance in number of positive and negative edges and structural balance theory. Comparing the performance with state of the art techniques on several real-world datasets validates the effectiveness of SigGAN.

KEYWORDS

Signed Networks, Generative Adversarial Networks, Link Prediction, Structural Awareness, Structural Balance

1 INTRODUCTION

The huge popularity of online social networks attracts users to share and exchange their views and opinions. Existing research works represent these relationships between the users through signed networks [59, 60] where the edges depict both polarity and strength [17, 63]. Understanding these relationships has many applications [14, 43, 48], like, personalized product recommendation [40] and identification of malicious users [33]. Therefore, the prediction of the signed links in these networks remains an important problem [58]. However, prediction of links in signed networks has different challenges than the conventional unsigned networks [2]. These challenges mainly stem from the differences due to the inclusion of sign in links. For example, several theories of social science, like homophily and social influence [18, 45] does not apply directly to signed networks. While two users who are connected are likely to be similar in unsigned networks, negative link between two users means they are likely to be dissimilar in signed networks whereas positive link between two users signify that they are likely to be similar. So, in signed networks, homophily and social influence does not directly apply just on the basis of the presence of link between a pair of nodes but also depend on the sign of the link. Similarly, the presence of sign affects the nature of transitive relationship between three users [50], like, enemy of my friend is my enemy, friend of my friend is my friend and enemy of my enemy is my friend are common in signed networks whereas in unsigned networks, two nodes connected to a common node has a higher likelihood to be connected among themselves than two nodes who do not share a common node. Therefore, in signed networks, both the presence of sign and the link are important in order to understand the type of transitive relationship. Furthermore, Leskovec et al. [35] highlights that the presence of negative links adds more value as compared to only positive links. However, it is highly challenging to integrate the information from the negative links as the fraction of negative links is very low compared to the positive links.

To mitigate these issues, existing research works propose traditional feature based learning [7, 9, 29, 49] and recently, deep learning approaches. Existing deep learning approaches rely on different forms of node embedding approaches which does not require the feature engineering efforts of traditional feature based learning approaches. These approaches either follow generative or discriminative based models to derive the node embedding. The generative model-based representation learning frameworks rely on the conditional distribution of a node, \(v_a\), being a neighbor of node \(v_b\), given the connectivity preferences of \(v_b\). These frameworks learn the likelihood of edges based on this conditional connectivity distribution, and thereby generate the vertex embedding that maximizes the likelihood of the given network connectivity. Several existing works, like SNE [61], SIDE [31], SigNET [27], sign2Vec [3] follow generative based approach to generate signed network embeddings. Subsequently, several existing research works, like SiNE [56], SHINE [55], DNE-SBP [46], have proposed discriminative models which predict the presence of an edge based on the characteristics of the node pair, \(v_a\) and \(v_b\). However, there are several issues like missing and incomplete data that are quite prevalent in large scale network data that affect the performance of the generative or discriminative based models [21]. GAN based models by combining both generative and discriminative models have shown to be more robust to these issues [21]. Recently, they have been highly successful in several applications, like, information retrieval [39], image generation [15] and collaborative filtering [5, 6]. Recently, Wang et al. [54] have proposed a GAN based model to represent nodes in unsigned networks in which the generator effectively learns the underlying connectivity distribution which is highly effective for link prediction in unsigned networks.

However, GraphGAN [54] cannot be directly applied to signed networks, due to the inherent differences between signed and unsigned networks as discussed previously. Further, one of the major characteristics of the signed networks is the structural balance theory [4, 24, 26] among the connected node triplets. In a structurally balanced triplet, the sign of the link between a pair of nodes \((v_a, v_b)\) connected to a common neighbor \(v_c\) would be
positive if the sign of the links $(u_a, v_c)$ and $(v_d, v_e)$ are either both positive or negative. Similarly, the sign of link $(u_a, v_b)$ would be negative if the sign of one of the links to $v_c$ is positive, and the other is negative. Therefore, we propose SigGAN, a GAN based model which is adapted to signed networks and integrates structural balance property. The novel contribution of this paper is a GAN based model for generating node representations for signed networks that considers both the structural proximity of the nodes and the structural balance of the network. The proposed framework effectively unifies two models—a binary-class generative model that generates likely positive (or, negative), connected neighbors of a node, $v_a$ and a discriminative model that identifies whether the generated node pair with the corresponding sign is one from the generated samples or the actual dataset—to learn individual parameters for each model to minimize a combined loss function. Validation on real world datasets, like Slashdot, Epinions, Reddit, Bitcoin and Wiki-RFA indicate that the SigGAN can ensure $3 - 11\%$ higher micro F1-score than the existing works in both link prediction and handling of sparsity. The organization of the paper is as follows. We discuss the existing research works in Section 2. A formal definition of the problem is provided in Section 3 followed by the details of the SigGAN in Section 4. We discuss the experimental settings in Section 5 and the details of our observations and results are given in Section 6. We finally draw our conclusions in Section 7.

2 RELATED WORKS

Existing research works which predict the sign of a link can be categorized into feature based approaches and representation learning based approaches. We discuss the works that belong to either of these categories next.

2.1 Feature based Machine Learning Approaches

Existing feature based machine learning approaches use either local, global or a combination of both to predict the sign of the link. For example, the local attributes include information about common neighbours, structural balance [35, 36], node features [49], etc. Existing approaches have explored the role of user interactions [1], similarity in user characteristics [11], neighborhood information [25], etc., to predict the sign of the missing link. While the utilization of the local attributes incur low computational cost, it fails in sparse networks. While global attributes, like, social hierarchy [38], individual trust [23, 58], structural balance theory [4] and community based ranking [44] can handle the sparsity, they are computationally expensive [30]. To mitigate this computation cost, several research works explore meso attributes, like, community structures [12, 13, 30], edge dual properties [62], Katz similarity-based walk [47] and, clustering-based approaches [28]. However, utilization of only one type of attribute affects the prediction accuracy of the sign of the link. Therefore, recent works explore a combination of local and global attributes [8, 25, 29]. Though a combination of both attributes can ensure higher accuracy, the extraction of global features for all nodes will lead to high computational cost. Further, these existing works ignore the high variance in the network characteristics of the edges that may provide vital cues in the prediction of the sign of an edge [7]. This knowledge can aid in the development of an adaptive system that applies local, global, or a combination of both based on the characteristics of the edge [7].

Further, these approaches are highly dependent on the selection of the features and therefore, deep learning approaches which can inherently integrate both local and global node attributes are more successful.

2.2 Node Embedding Based Approaches

Recently, node embedding has been proven to be very useful in understanding the structure of the network, link prediction, node classification, etc. Several existing research works [22, 42, 51–53] have explored different aspects of network topology of a node to map each node into a low-dimensional vector space. However, these approaches can not be directly applied to signed networks due to the inherent differences between signed and unsigned networks. Existing node embedding approaches for signed networks are either matrix factorization based or, deep learning frameworks. The deep learning frameworks can further be segregated into generative and discriminative based representation learning, where generative models focus on learning the underlying connectivity distribution in the graph and discriminative models predict the probability of an edge based on the attributes of the pair of nodes. Existing generative models rely on representation learning by random walk based approaches and the discriminative models rely on representation learning by adjacency relationship-based embedding which we discuss next.

Existing representation learning by random walk based approaches consider the nodes present in randomly selected paths [61] or, the co-occurrence of nodes from randomly selected paths [31]. Further, to handle the high computational cost to incorporate all the possible edges, Kim et al. [31] propose a variant of negative sampling in signed networks. However, none of these works consider the higher-order proximity or the signed relationships. Bhownick et al. [3] address this issue by a trust-based random walk that considers both higher-order neighborhood and signed (trust) relationships. However, as none of these works explicitly consider structural balance into the generated embeddings. Therefore, Islam et al. [27] introduce the integration of structural balance into random walks. Subsequently, discriminator based frameworks generate low dimension representation of nodes, like, Shen et al. [46] explicitly considers structural balance information while generating the embedding vector of a node. Additionally, Lu et al. [37] consider both status and structural balance and Chen et al. [10] consider bridge edges, status and structural balance for generating the node embedding. However, the major problem with all these methods is that they suffer from poor compatibility for high dimensional data. Further, they are prone to poor training in the face of missing edges or smaller sample sizes [21], which are major issues in large scale networks generated from real data.

On the other hand generative adversarial networks are more robust to these issues [20] and hence, have been used extensively in several applications [64] including predicting links in unsigned networks [54]. Unlike unsigned networks, where the node representations mainly deal with presence or absence of links, for signed networks, the problem is identifying the sign of the link. Further, for signed networks, the GAN model should be able to handle specific properties of signed networks, like the high imbalance in positive and negative links and structural balance. In this paper, we propose a GAN model for generating node representations in signed networks that can handle these issues. We next provide a formal definition of the problem and subsequently, discuss the SigGAN.
3 PROBLEM DEFINITION

Let, \( G = (V, E, S) \) denote a signed network, where \( V \) denotes
the set of vertices (nodes) and \( E \) represents the set of edges which
has either positive (\( s \)) or negative (\( s' \)) sign. \( S = \{s, s'\} \) denotes
the set of signs considered for the edges. We represent the signs as \( s \)
and \( s' \) rather than \(+\) or \(-\). We assume that we have information
about certain edges along with their signs for a network. Thus,
for a given node \( v_i \), the objective of the problem is to predict the
possible sign \( s \) or \( s' \) of the missing links from \( v_i \). Therefore,
for a given node \( v_i \) we determine the probability \( p(\phi_i|v_j) \),
for all nodes \( v_i (i \neq j) \), where \( \phi_i \) represents a random variable
that takes values in \( S \). For example, \( p(\phi_i = s|v_j) \) denotes the
probability that \( v_i \) is positively connected to \( v_j \). We can further
extend this problem to directed networks, i.e., an edge from \( v_i 
\) to \( v_j \) and \( p(\phi_i|v_j) \) may not be equal to \( p(\phi_j|v_i) \). However,
this would also involve implementing suitable discriminator models
that also consider the directions of the edges. Since the objective
of this paper is to show the applicability of GAN in generating
node representations in signed networks, we avoid dealing with the
issue of the direction of the edges in the network. The formation of
the signed network strictly depends on the application.

For example, for a social news site, like Slashdot, the signed
network comprises of users as nodes and the edges are \((-1, \) or
\(+1\) depending on the interaction of the users in comments. The
prediction task can be considered as determining the probably
signed rating of one user towards another. We discuss SigGAN
next.

4 PROPOSED APPROACH

SigGAN relies on a GAN based framework that combine a gen-
erative and discriminative model using a minimax game [54]
to generate node embedding. The generative model in GAN either
outputs a labeled data from the true dataset or generates a fake
labeled data based on the node embedding, whereas the discrimi-

ator model attempts to classify the generated data as true or
fake. In SigGAN, we explicitly consider the characteristics of
signed networks, like structural balance theory property [27],
signed homophily and the usual requirements that have been
considered in GraphGAN [54], such as, structural awareness and
low computational complexity.

SigGAN learns the underlying signed adjacency distribution of each node, \( v_i \) across all nodes in \( V \), i.e., \( p_{\text{true}}(\phi_i|v_j) \), where
\( \phi_i \in S \) represents the sign of the link from \( v_j \) to \( v_i \). SigGAN
uses a generator and discriminator model that operates at tan-

dem to improve the prediction accuracy of the system. The task
of the generator function \( J(\phi_i|v_j; \theta_J) \) is to approximate the true
positive and negative connections of \( v_i \), that is used to select
it’s likely neighbors with a given sign from the underlying dis-


tribution \( p_{\text{true}}(\phi_i|v_j) \). The proposed generator model has the fol-

lowing key properties:

(1) It generates a probability of the sign of the edges from \( V_j 
\) to other nodes.

(2) It is structurally aware, where the probability of the exis-
tence of an edge to a far away node tends to zero.

(3) It maintains the structural balance theory.

(4) The generator model is computationally efficient.

The discriminator function, \( D(v_i, v_j, \phi_i; \theta_D) \) uses a MLP based
model that outputs a single scalar value that represents how
likely an edge \( (v_i, v_j) \) is of a given sign \( \phi_i \). We train the discrimi-

ator by an iterative selection of negative and positive edges. How-

ever, Shen et al. [46] shows that the number of negative
edges is very less compared to the number of positive edges and
the cost of forming a negative edge is higher than the formation
of a positive edge in signed networks. To handle this, we select
negative edges with an equal probability as the positive edges
for the discriminator to learn so that SigGAN learns to predict
the negative edge with high precision as well. Therefore, given
the graph \( G \), generator \( J \) tries to generate the positive (or nega-
tive) neighbors of \( v_j \) which are similar to its actual positive (or
negative) neighbors. We term these neighbors of \( v_j \) generated by
the generator as fake neighbors, whereas those neighbors from
the actual graph are true neighbors. The discriminator \( D \) deter-
mines if the generated neighbors of \( v_j \) are fake or true through a
two player minimax with value function \( \mathcal{V}(J, D) \), given as

\[
\min_{\theta_J} \max_{\theta_D} \mathcal{V}(J, D) = \sum_{i=1}^{V} \left( \mathbb{E}_{v_i \sim p_{\text{true}}(\cdot|v_j)} [\log D(v_i, v_j; \phi_i; \theta_D)] + \mathbb{E}_{v_i \sim J(\cdot|v_j; \theta_J)} [\log(1 - D(v_i, v_j; \phi_i; \theta_D))] \right)
\]

\( \theta_J \) represents the vector representations of all nodes \( v_i, i \neq j, 
\) (described later) which can be potential neighbors of \( v_j \). There-

fore, the optimal parameters of the generator and discrimina-
tor are learned by alternately maximizing and minimizing the
value function \( \mathcal{V}(J, D) \). At each iteration, the discriminator \( D 
\) is trained using a batch of true samples from \( p_{\text{true}}(\cdot|v_j) \) and fake
samples generated using \( J(\cdot|v_j) \). The generator \( J \) is updated
using a policy gradient. The continuous competition between \( D 
\) and \( J \) ensures generating a suitable representation of the node,
\( v_j \) that satisfies the required properties. We discuss the func-

tions of \( D \) and \( J \) next.

4.1 Discriminator

Given a list of node pairs \( (\cdot, v_j) \), the objective for the discrimina-
tor, \( D \) is to maximize the log-probability of assigning the correct
label depending on whether the edge has been sampled from the
actual network (true) or has been generated by the generator
(fake). The discriminator mainly relies on node representations
that can be compared pairwise to discriminate between true and
fake edges.

There exists a plethora of models that can be used for the discrimi-

ator; in this paper we do not contribute in investigating the
suitability of the models and leave it as a future goal. Rather,
we use a dense network to determine node representations that
can be compared to derive the likely chances of them being con-

nected with a given sign \( \phi_i \). The model uses a single hidden layer,
in which the number of units equals the dimension of the node
representations (represented by \( k \) in our case). The input to the
model is the one-hot vector of a node in the network. Each unit
in the output layer corresponds to a node in the network and is
activated by a sigmoid function. The output of the sigmoid func-
tion of the \( i \)-th output unit provides a measure of the possibil-

ity of node \( v_i \) being connected to \( v_j \) with sign \( \phi_i \). Suppose, \( h_i \in \mathbb{R}^n \)

is the one hot encoding vector of node \( v_i \) that is provided as input
to the input layer of the discriminator. Let, \( W \in \mathbb{R}^{mk \times k} \)
represent a weight matrix whose \( i^{\text{th}} \) column elements represent the



corresponding weights of the links from each of the inputs to the
\( i^{\text{th}} \) unit of the hidden layer. Thus, \( d_j = W^T h_j \) represents the hidden

t unit vector of length \( k \). Suppose, \( W' \in \mathbb{R}^{k \times n} \) represents a

matrix of the weights of hidden units to the output layer, where

https://snap.stanford.edu/data/soc-Slashdot0902.html
the $[ij]^{th}$ element of $W'$ represents the weight of the link connecting the $i^{th}$ hidden layer unit to the $j^{th}$ output unit. Thus, the output $o_i$ of each of the output units can be represented as $o_i = \sigma(W'_i \cdot d_i)$, where $W'_i$ represents the $i^{th}$ column of $W'$, and can be considered as a $k$ dimensional embedding of the node $o_i$, represented as $d_i$. Further, $\sigma$ is the sigmoid function given as $\sigma(x) = \frac{1}{1 + \exp(-x)}$. Thus, if $d_i, d_j \in \mathbb{R}^k$ represents the $k$ dimensional embedding vector for nodes $o_i$ and $o_j$, respectively, and $\phi$ represents the corresponding link sign of these nodes, then the discriminator function would be represented as

$$D(o_i, o_j, \phi) = \sigma(\phi d_i^T d_j) = \frac{1}{1 + \exp(-\phi d_i^T d_j)}$$  \hspace{1cm} (2)

As evident from equation 2, when $\phi$ is positive, $D$ returns a near to 1 value if $d_i$ and $d_j$ are similar, indicating that nodes with similar representations are most likely to be positively connected. On the other hand for a negative value of $\phi$, the discriminator would return a higher score if $d_i$ and $d_j$ are highly dissimilar. The discriminator is trained based on the loss function stated in equation 1. For each $d_j$, a stochastic gradient ascent is used to update $\theta_j$, the vector representations of $o_i$. We describe the generator model that we use in our SigGAN next.

### 4.2 Generator

Contrary to the discriminator, the goal of the generator $J$ is to generate the links of each node $o_i$ with their corresponding sign that mimics the true adjacency distribution $p_{true}(\phi_i|v_j)$. The generator model is defined as a function $J(\phi_i|v_j)$ that generates node representations to minimize the loss function

$$\mathbb{E}_{v \sim \mathcal{D}} \left[ \log(1 - D(o_i, o_j, \phi_i; \theta_D)) \right],$$  \hspace{1cm} (3)

which is the log probability of the discriminator correctly identifying fake link samples generated by the generator. Since the generator samples through the discrete space $o_i$ to increase the probability score, to minimize the loss function we derive its gradient with respect to $\theta_D$ using policy gradient. Thus, if $N(v_j)$ denotes the neighbors of $o_j$, then

$$\nabla \theta_j \sum_{j=1}^{[V]} \left[ \mathbb{E}_{v \sim \mathcal{D}} \left[ \log(1 - D(o_i, o_j, \phi_i; \theta_D)) \right] \right] = \nabla \theta_j \sum_{j=1}^{[V]} N(v_j) \left[ J(\phi_i|v_j) \left[ \log(1 - D(o_i, o_j, \phi_i; \theta_D)) \right] \right] = \sum_{j=1}^{[V]} N(v_j) \left[ \nabla \theta_j J(\phi_i|v_j) \left[ \log(1 - D(o_i, o_j, \phi_i; \theta_D)) \right] \right] = \sum_{j=1}^{[V]} N(v_j) \left[ \sum_{i=1}^{[V]} J(\phi_i|v_j) \times \nabla \theta_j \log(J(\phi_i|v_j)) \right] \left[ \log(1 - D(o_i, o_j, \phi_i; \theta_D)) \right]$$  \hspace{1cm} (4)

Therefore, we can intuitively understand from Equation 4 that a higher value of $\log(1 - D(o_i, o_j, \phi_i; \theta_D))$ for a given $o_j$ will lower the probability of generating $o_i$ with respect to $o_j$. We use a softmax function for the generator $J(\phi_i|v_j)$ which is given as

$$J(\phi_i = s|v_j) = \frac{e^{s \cdot \phi_i^T g_j}}{\sum_{k \in \mathcal{N}(v_j)} e^{\phi_i^T g_k}},$$  \hspace{1cm} (5)

As evident from Equation 5, for a given node $o_j$, $J$ generates a neighbor $o_i$ with sign $s$ based on the representations, $g_i$ and $g_j$ of the nodes $o_j$ and $o_i$, respectively. A similar representation between the two nodes increases the probability of them being positively connected, whereas dissimilar representations increase the chances of a negative link. These representations are suitably derived based on the loss function stated in Equation 4 using gradient descent. However, a major problem with the softmax function is the high computation involved in updating the gradients. As evident from Equation 5, for an arbitrary node $o_i$, the gradients must be calculated for the entire set of vertices and hence, $J(\phi_i|v_j)$ would require updating the representations of all $[V]$ nodes. Further, the softmax function ignores the rich structural information of the graphs, where the proximity of the nodes in the network can play an important role in the link formation [54]. Existing approximation techniques that determine the softmax scores, like negative sampling and subsampling [41] do not consider the structural information. Hence, there is a need to address these issues for the practical applicability of this method. We propose a modification of the softmax approach that can mitigate these issues.

### 4.3 Modified Softmax for Signed Graphs

In this Section, we discuss the proposed variant of the softmax function that can be used for the signed graphs. Although in GraphGAN [54] a modified softmax function was proposed for unsigned networks that maintain specific properties, like, normalization, graph structure awareness, and computational efficiency, the proposed function is not applicable for signed networks. Further, an additional requirement of the signed networks is the structural balance among the nodes, which the softmax function must consider. These requirements demand the formulation of a suitable softmax function applicable for signed networks. We, hereby, summarize the four basic properties that we consider:

1. **Normalization**: The function must be normalized such that it produces a valid probability distribution, i.e., $\sum_{i \in S} J(\phi_i = t|v_j) = 1$.
2. **Graph Structure aware**: The function must take into account the structure of the graph to calculate the connectivity distribution between a pair of nodes.
3. **Computationally Efficient**: The update of the values of $\theta_D$ for deriving the gradient descent must be computationally efficient.
4. **Structurally Balanced**: The sign of the links generated must maintain Structural balance theory.

We next discuss the proposed softmax approach. Given a node, we introduce sign-specific relevance probability of its neighbor. For a node $o_j$, the positive relevance probability of its neighbor $o_i$ is given as

$$p(\phi_i = s|v_j) = \frac{e^{s \cdot \phi_i^T g_j}}{\sum_{k \in \mathcal{N}(v_j)} e^{\phi_i^T g_k}} \cdot \sum_{t \in \mathcal{S}} e^{t \cdot g_j}.$$  \hspace{1cm} (6)

whereas the negative relevance probability is obtained by replacing $s$ by $s'$. The sign-specific relevance probability indicates how likely a neighbor $o_i$ of node $o_j$ would be connected by a link with a given sign, s or $s'$. Similar to the GraphGAN approach, for each node $o_j$, we initially make a Breadth First Search (BFS) traversal with $o_j$ as the root node. Let $o_{r_0} \rightarrow o_{r_1} \rightarrow \cdots \rightarrow o_{r_n}$ be a path in the BFS tree with root at $o_{r_0}$. Let $\pi_{r_{n-1}} = p(\phi_i = s|v_{r_{n-1}})$ be the positive relevance probability of node $o_{r_{n-1}}$ with respect to its uplink parent $o_{r_{n-1-1}}$. Therefore, $\pi_{r_{n-1}}$ is the relevance probability...
of the node at hop \( m \) (written as sub-script of \( \pi \)) from the node at hop 0 (written as super-script of \( \pi \)) connected by an edge with positive sign, \( s \). From structural balance theory, we can derive the sign of the edge connecting node at level 0 and \( m \) through the level, \( m = 1 \). For example, on the basis of structural balance theory for triads, the sign of the edge between node at level 0 and \( m \) would be positive, \( s \), if the sign of the edge node at level 0 and \( m = 1 \) is \( s \) and the sign of the edge node at level \( m = 1 \) and \( m \) is \( s \). Additionally, the sign of the edge between node at level 0 and \( m \) would also be positive if the sign of the edge node at level 0 and \( m = 1 \) is \( s' \) and the sign of the edge node at level \( m = 1 \) and \( m \) is \( s' \). Similarly, we can find the the sign of the edge between node at level 0 and \( m \) would be negative if the sign of the edge between node at level 0 and \( m = 1 \) and \( m \) is different. Therefore, we can extend the positive and negative relevance probability, respectively, for nodes that are more than one hop away by recursive formulations given as

\[
\pi^0_0 = \left( \pi^0_{m-1,s} \pi^m_{m-1,s} + \pi^0_{m-1,s'} \pi^m_{m-1,s'} \right)
\]
\[
\pi^0_{m,s'} = \left( \pi^0_{1,s} \pi^1_{m-1,s'} + \pi^0_{1,s'} \pi^1_{m-1,s} \right)
\]

We are interested in determining the modified softmax function \( f(\phi|\alpha) \) for any arbitrary node \( \alpha \) in the tree with respect to the root node \( \alpha_r \). Assuming that the graph is strongly connected, a BFS tree with respect to a root will traverse all the nodes through a single unique path. This assumption holds true for any undirected graph; several real-world directed networks are also strongly connected with a large core and hence can be applied to these. Thus, we define \( f(\phi|\alpha) \) as

\[
J(\phi = s|\alpha) = \pi^0_{s,\alpha} + \pi^0_{s',\alpha} \pi^1_{m-1,s'}
\]

We prove using appropriate theorems that the generator function \( f(\phi|\alpha) \) satisfies the first three requirements that are mentioned above, whereas, we intuitively show that the modified softmax can also learn the structural balance property if the same exists in the network.

**Theorem 4.1.** For a given node \( \alpha \), using the modified softmax we get \( \sum_{i \in j} \sum_{\phi} J(\phi_i = t|\phi_j) = 1 \)

Proof. We start by showing that for any sub-tree \( ST_m \) (rooted at \( \alpha_m \)) of the BFS tree with \( \alpha_m \) as the root, the total sum of the softmax scores of all the nodes in the sub-tree, with respect to \( \alpha_r \), is the softmax score of \( \alpha_m \) calculated with respect to \( \alpha_r \). We use this concept for the sub-trees rooted at the child nodes of the BFS tree of \( \alpha_m \). From the expression of sign-specific relevance probability stated in Equation 6, we can directly conclude that for a given node \( \phi_i \), if \( N(\phi_i) \) be its neighbors then

\[
\sum_{\phi_j \in N(\phi_i)} \sum_{\phi} J(\phi_i = t|\phi_j) = \frac{1}{|N(\phi_i)|}
\]

Initially, we consider the case where the sub-tree is rooted at the node \( \alpha_m \), which has only leaf nodes as its children. Let the children be denoted as \( \alpha_m, \alpha_m, \ldots, \alpha_m \). So,

\[
\sum_{i \in ST_l \alpha_m} \sum_{\phi} J(\phi_i = t|\alpha_m)
\]

can be written as

\[
\sum_{i \in ST_l \alpha_m} J(\phi_i = t|\alpha_m) + \sum_{i \in \alpha_m} \sum_{\phi} J(\phi_i = t|\alpha_m),
\]

where \( \alpha_m \) denotes the child nodes of \( \alpha_m \). From Equation 7, we have

\[
\sum_{i \in ST_l \alpha_m} J(\phi_i = t|\alpha_m) = \left( \pi^0_{m-1,s} \pi^m_{m-1,s} + \pi^0_{m-1,s'} \pi^m_{m-1,s'} \right)
\]

A similar set of derivations for \( \sum_{i \in \alpha_m} \sum_{\phi} J(\phi_i = t|\alpha_m) \)

\[
\sum_{i \in \alpha_m} \sum_{\phi} J(\phi_i = t|\alpha_m) = \left( \pi^0_{m,s} \pi^m_{m,s} + \pi^0_{m,s'} \pi^m_{m,s'} \right)
\]

Since for a leaf node \( \pi^m_{m,s} = \pi^m_{m,s'} = 1 \), Equation 10 can be expressed as

\[
\sum_{i \in \alpha_m} \sum_{\phi} J(\phi_i = t|\alpha_m) = \pi^0_{m,s} + \pi^0_{m,s'}
\]

We collapse the sub-tree rooted at \( \alpha_m \) so as to form a single leaf node and recursively work towards the sub-tree rooted at the child nodes of the root node \( \alpha_r \). Then, for the child node \( \alpha_m \), we have

\[
\sum_{i \in ST_l \alpha_m} \sum_{\phi} J(\phi_i = t|\alpha_m) = \pi^0_{m,s} + \pi^0_{m,s'}
\]

Taking the sum over all subtrees rooted at the child nodes of \( \alpha_m \), we have

\[
\sum_{i \in \alpha_m} \sum_{\phi} J(\phi_i = t|\alpha_m) = 1,
\]

which completes our proof that the normalization property holds for the modified softmax function.

In order to show that the modified softmax considers the connectivity pattern of a node, we prove next how the modified softmax score of the root node changes with respect to the shortest path length to any node as followed by Wang et al. [54] in unsigned networks.

**Theorem 4.2.** In the modified softmax \( f(\phi|\alpha, \theta) \) decreases with increasing shortest path distance between \( \alpha \) and \( \theta \).

Proof. Since BFS tree preserves the shortest path from the root node to the other nodes, so for a root node \( \alpha_r \) the probability \( f(\phi_r|\alpha, \theta) \) for an arbitrary node \( \alpha_r \), that is \( n \) hops away
We analyze the performance of SigGAN on five datasets, Slashdot, Epinions, Bitcoin, Reddit and Wiki-RFA. A brief overview of the datasets is as follows:

- **Slashdot**: This dataset is created from the relationships among users who tag each other as either friends or foes from the news website, Slashdot [36].
- **Epinions**: This dataset is created from the consumer reviewer site, Epinions, where users share trust or distrust relationships among themselves [36].
- **Reddit**: We obtained this dataset from the SNAP database. This is a directed network of hyperlinks created from subreddit pairs created from posts that create hyperlink from one post to another [32]. The sign of the link is based on the sentiment of the source subreddit towards the hyperlinked subreddit.
- **Bitcoin**: This dataset represents the trust or distrust relationship between users who trade on the Bitcoin OTC [34] platform.
- **Wiki-RFA**: This dataset represents the voting relationship among the users, i.e., nodes represent users and the edges represent the votes among the users [57]. Therefore, the sign of the edges represent supporting, neutral, or opposing vote.

For our experiments, we considered the dataset for Wiki-RFA, Reddit and Bitcoin and follow Shen et al. [46] to create graphs for Slashdot and Epinions datasets. We outline the details of each of these datasets in the Table 1. Our observations indicate that the number of negative edges is quite low compared to the number of positive edges and the sparsity of the graphs differ across datasets.

### 5.2 Baselines

We compare SigGAN with 3 state-of-the-art deep learning-based techniques for signed link prediction. Each of these techniques generates node representations using different strategies that are used to determine the sign of the links. We briefly discuss these baselines next.

- **DNE-SBP**: Shen et al. [46] proposed a deep neural embedding with structural balance preservation (DNE-SBP) model to learn graph representations for signed networks. The method uses a semi-supervised stacked auto-encoder to explore the connectivity patterns of a node. Further, the SigGAN exploited this connectivity relationship of a node and extended structural balance theory to generate node embeddings. They explicitly showed that more importance is given in reconstructing the negative edges than the positive edges.
- **SIDE**: Kim et al. [31] proposed a method for representation learning in signed directed networks (SIDE) that incorporates sign, direction, and proximity relationships of nodes.

### Table 1: Dataset Details

| Dataset | Nodes | Positive Edges | Negative Edges |
|---------|-------|----------------|----------------|
| Slashdot | 7000  | 632103         | 110068         |
| Epinions | 7000  | 431098         | 92011          |
| Reddit   | 6999  | 734408         | 277050         |
| Bitcoin  | 5877  | 277050         | 21436          |
| Wiki-RFA | 7118  | 249580         | 18282          |

https://snap.stanford.edu/data/soc-sign-Slashdot090211.html
https://snap.stanford.edu/data/soc-sign-epinions.html
https://snap.stanford.edu/data/soc-RedditHyperlinks.html
https://snap.stanford.edu/data/soc-sign-bitcoin-otc.html
https://snap.stanford.edu/data/wiki-RFA.html
the nodes to generate low dimensional vector representations of these nodes. This model generates random walks across the graph that captures the proximity between nodes in the random walks, and subsequently generates the likelihood of co-occurrence of a node pair based on this information. Further, in the next step, using this information, SigGAN generates embedding of the nodes following a skip-gram technique with negative sampling.

(3) SNE: Yuan et al. [61] proposed a technique for signed network embedding (SNE) that uses a log-bilinear model for generating the node embeddings. The model considers the node representations of all the nodes along different paths of a node to generate its embedding. The model computes the embedding of the target node given the information of the predecessors along the path and the signed relationship among these predecessors by maximizing the log-likelihood.

For our experiments, we perform stochastic gradient descent to update parameters in SigGAN with learning rate 0.001. For each iteration, we set the number of positive samples as 20 and repeat this for 10 epochs for both generator and discriminator respectively with batch size as 32. Subsequently, we set the number of epochs for SigGAN as 10 and consider the dimension of node embedding as 50. We set the values of each hyper-parameter and its effect on the performance of SigGAN.

| Dataset  | Slashdot | Epinions | Reddit | Bitcoin | Wiki – RFA |
|----------|----------|----------|--------|---------|------------|
| L1       | 0.74     | 0.883    | 0.93   | 0.86    | 0.76       |
| L2       | 0.76     | 0.878    | 0.94   | 0.87    | 0.76       |
| Had      | 0.79     | 0.89     | 0.96   | 0.86    | 0.75       |
| Avg      | 0.78     | 0.883    | 0.93   | 0.868   | 0.77       |
| Concat   | 0.74     | 0.87     | 0.92   | 0.87    | 0.74       |

Table 2: Mean micro F1-score of the SigGAN for different edge vector representations

6.1 Link Prediction

Link prediction in signed networks is to identify the sign of an edge. We compare the performance of SigGAN with the baselines, DNE-SBP, SIDE and SNE in identifying the sign of the edge. Based on the experiments followed in the existing research works, we use a logistic regression classifier [16, 46] that is trained on the vector representation of the edges as features. Since SigGAN and the baselines generate the node vector representations, we initially derive the corresponding edge vector representation, $e_{ij}$. We follow 5 different methods, L1-norm, L2-norm, Hadamard product, average and concatenation to determine $e_{ij}$ as followed in existing research works [31, 46]. We, then, train the logistic regression model on the training set for the edge representation.

We show the mean micro F1-score for different types of edge representations for SigGAN in Table 2. We observe that the edge representation by Hadamard product performs the best for Slashdot, Epinions and Reddit, edge representation by average for Wiki-RFA and edge representation by L2-norm for Bitcoin respectively. Furthermore, the edge representation by Hadamard product for Wiki-RFA and Bitcoin is very near to the best performance with a difference of 0.02 and 0.01 respectively. Based on these observations, we choose edge representation by Hadamard product for SigGAN.

In [31, 46], it has been shown that the baselines have better performance when the logistic regression model is trained by the edge vector representation by Hadamard product than the other representations. Therefore, in order to maintain uniformity, we consider the edge vector representation by Hadamard product for our comparison results. We show the mean micro F1-score in Table 3. Our observations from Table 3 indicates the SigGAN provides better performance than all the existing research works in all the datasets. The mean micro F1-score of the SigGAN is around 3 – 11% more than the existing research works.

6.1.1 Hyperparameter Analysis: We discuss next how we set the values of the different parameter or hyperparameter for SigGAN. For each parameter or hyperparameter, we vary its values and calculate the mean micro F1-score for 5 independent runs for each value. Therefore, we select that value for a parameter or
| Dataset            | Slashdot | Epinions | Reddit | Bitcoin | Wiki – RFA |
|--------------------|----------|----------|--------|---------|------------|
| SigGAN             | 0.79     | 0.89     | 0.96   | 0.86    | 0.75       |
| DNE – SBP          | 0.68     | 0.83     | 0.86   | 0.83    | 0.70       |
| SIDE               | 0.73     | 0.87     | 0.93   | 0.84    | 0.70       |
| SNE                | 0.73     | 0.86     | 0.93   | 0.84    | 0.69       |

Table 3: Mean micro F1-score of the SigGAN with the existing research works

| Appr   | 80% | 60% | 40% | 20% | 80% | 60% | 40% | 20% | 80% | 60% | 40% | 20% |
|--------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| SigGAN | 0.72| 0.76| 0.78| 0.89| 0.86| 0.87| 0.88| 0.76| 0.73| 0.76| 0.74|    |
| DNE – SBP | 0.71| 0.76| 0.78| 0.89| 0.84| 0.86| 0.84| 0.72| 0.73| 0.73| 0.71|    |
| SIDE   | 0.71| 0.76| 0.75| 0.87| 0.84| 0.86| 0.87| 0.74| 0.75| 0.73| 0.71|    |
| SNE    | 0.75| 0.75| 0.77| 0.75| 0.76| 0.76| 0.84| 0.74| 0.73| 0.71| 0.71|    |

Table 4: Comparison results of mean micro F1 score of SigGAN with the baselines for different levels of sparsity

| Appr   | 80% | 60% | 40% | 20% | 80% | 60% | 40% | 20% | 80% | 60% | 40% | 20% |
|--------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| SigGAN | 0.81| 0.83| 0.84| 0.86| 0.90| 0.92| 0.93| 0.94|    |    |    |    |
| DNE – SBP | 0.79| 0.81| 0.81| 0.82| 0.86| 0.85| 0.85| 0.85|    |    |    |    |
| SIDE   | 0.80| 0.81| 0.81| 0.83| 0.91| 0.89| 0.90| 0.92|    |    |    |    |
| SNE    | 0.81| 0.81| 0.82| 0.82| 0.90| 0.90| 0.90| 0.91|    |    |    |    |

Table 5: Comparison results of mean micro F1 score of the SigGAN with the baselines for different levels of sparsity

Figure 1: Comparing the effect of change in batch size, learning rate and embedding dimension on Mean Micro F1-score for Epinions in Figure 1a, Figure 1b and Figure 1c respectively.

Figure 2: Comparing the effect of change of number of epochs for Generator and Discriminator and number of epochs of SigGAN on Mean Micro F1-score for Epinions dataset in Figure 2a and Figure 2b respectively.
We investigate the performance of SigGAN in handling sparse data. We consider edge representations by L1-norm, L2-norm, Hadamard product, average and concatenation. We show our observations for Epinions in Figure 1 and Figure 2 respectively. Based on our observations from Figure 1a, we find the value of batch-size of 32 provides the maximum mean micro F1-score irrespective of the function for edge representation except for L2-norm and Hadamard product when batch size is 100. However, our observations indicate that the increase in results for L2-norm and Hadamard product when batch size is 100 is very small and further, it might lead to overfitting if we increase the batch size to 100. Therefore, we consider 32 as batch size for SigGAN. Similarly, we observe for learning rate as 0.001, SigGAN has the maximum mean micro F1-score. Additionally, to decide the embedding dimension, we vary the dimension from 25 – 100 and calculate the result of SigGAN. Our observations as shown in Figure 1c indicate that increasing the dimension from 50 to 100 provides almost similar mean micro F1-score and dimension of size 25 has the least mean micro F1-score. As we want to have as low dimension as possible, we choose 50 as the node embedding dimension. We specify the number of epochs by which we train SigGAN as s and t as the number of epochs for which discriminator and generator runs for each value of s. Our observations from Figure 2a and Figure 2b indicate that SigGAN has the maximum mean micro F1-score irrespective of the function for edge representation when both s and t are 10 and further, the increase in the number of epochs has very less impact in the mean micro F1-score. Therefore, the modified Softmax specifically for signed networks and simultaneous training of SigGAN by equal number of positive and negative edges leads to quicker convergence as observed in Figure 2a and Figure 2b. Therefore, based on our training mean micro F1-score, we set these values for all the parameter and hyperparameter values.

### 6.2 Handling Sparsity

We investigate the performance of SigGAN in handling sparse graphs. For our experiments, we induce sparsity in the original graph by randomly removing 20%, 40%, 60% and 80% of the edges from the original graph. For each of these sparse graphs, we generate the embedding of each node using SigGAN. Further, to create a graph with specific sparsity, say 20%, we repeat the procedure 5 times so that we have 5 different graphs with 20% sparsity. For each of these graphs, we then evaluate the performance of SigGAN in predicting the sign of the link by following the procedure discussed in Section 6.1. Additionally, since we follow edge representation by Hadamard product in Section 6.1, we report only the results for edge representation by Hadamard product here. We repeat the same experiment for Slashdot, Epinions, Reddit, Bitcoin and Wiki-RFA. We calculate the mean micro F1-score as the average of micro F1-score over 5 different graphs with same sparsity and micro F1-score over a graph is calculated by 5-fold cross validation. We show our observations in Table 4 and Table 5 respectively. We discover that SigGAN ensures better performance than the baselines in all the datasets when the sparsity is increased except for when 80% of the edges were removed for Slashdot, Reddit and 40% for Wiki-RFA. Subsequently, we observe that SigGAN ensures almost consistent performance irrespective of the sparsity in the signed network, i.e., 0.72 – 0.78 for Slashdot, 0.86 – 0.89 for Epinions, 0.73 – 0.76 for Wiki-RFA, 0.81 – 0.86 for Epinions and 0.90 – 0.94 for Reddit.

### 6.3 Case Study: Analysis of Embedding by SigGAN

We now study whether the network representations learned by SigGAN follows the extended structural balance property, i.e., positively connected nodes are closer in the representation space than the negatively connected nodes, as discussed previously in [27]. For our experiments, we follow the procedure adopted in existing research works [27, 31]. In order to measure this, we calculate the average positive edge distance, APED as the average distance between the vector representation of the positively connected nodes and ANED for negatively connected nodes, i.e.,

\[
APED = \frac{(\sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij}^+ d_{ij})}{(\sum_{i=1}^{n} A_{ij}^+)} \tag{15}
\]

\[
ANED = \frac{(\sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij}^- d_{ij})}{(\sum_{i=1}^{n} A_{ij}^-)} \tag{16}
\]

where, \(A_{ij}^+\) is 1 if node \(i\) and node \(j\) are positively connected else \(A_{ij}^-\) is 1 if node \(i\) and node \(j\) are negatively connected else 0 and \(d_{ij}\) is the Euclidean distance between the vector representations of \(i\) and \(j\). Therefore, for a network to follow extended structural balance theory, ANED should be greater than APED. We randomly select equal number of positively and negatively connected edges, i.e., 40% of the total negatively connected edges. We repeat the same experiment for all the datasets, i.e., Slashdot, Epinions, Reddit, Bitcoin and Wiki-RFA. The results are shown in Table 6 which indicates that APED, is around 1.5 – 4.0 less than ANED, irrespective of the dataset. We discover SigGAN has the best performance for Epinions and Bitcoin where the average distance between positively and negatively connected nodes are the highest. Our observations confirm that SigGAN can ensure that the positively connected nodes have a smaller distance than the negatively connected nodes. Thus, these results indicate that for signed networks, the SigGAN ensures positively connected node pairs are placed closer than the negatively connected node pairs and can, therefore, ensure extended structural balance theory property.

### 6.4 Summary of Insights

We discuss SigGAN in comparison to GraphGAN [54] and its applicability to signed networks followed by an analysis of the performance of the SigGAN with respect to the baselines followed by the shortcomings of SigGAN next. We develop SigGAN based on GraphGAN [54] which utilizes the inherent characteristics of the graph structure in a GAN based framework to generate node embedding. Although GraphGAN [54] is very effective for link prediction in unsigned networks, it cannot be directly applied to signed networks. Therefore, in this paper, we propose SigGAN which is not a direct extension of GraphGAN but explicitly considers the specific characteristics of signed networks, like structural balance theory and high imbalance in number of positive and negative edges. We propose a modified softmax for signed networks which integrates structural balance theory with the existing properties from unsigned networks, like normalization and graph structure awareness. Therefore, by integration of these properties in the proposed modified softmax, we ensure that the generator is highly effective in generating negative samples for the discriminator. We show in Subsection 4.3 how we formulate modified softmax such that it considers the properties. We prove appropriate theorems in Subsection 4.3 to show
that normalization, graph structure awareness and computational efficiency, are satisfied in the generator. Furthermore, to show that structural balance theory is satisfied by SigGAN, we prove through intuition in Subsection 4.3 and through a case study in Subsection 6.3. In order to handle high imbalance in number of positive and negative edges and high information content in the negative edges, we train the discriminator with equal number of positive and negative edges. This further ensures that SigGAN can ensure prediction of negative edge effectively. Our observations in Subsection 6.1 and Subsection 6.2 shows that SigGAN has a higher mean micro F1-score than the baselines. As the micro F1-score considers the efficiency in predicting both the positive and negative edges for original and sparse graphs respectively. Therefore, we believe the design of the modified softmax specific for signed networks for generator and simultaneous training by both positive and negative edges for discriminator are the main reasons behind success of SigGAN. While the generator is effective in generating negative samples irrespective of the sign of the edge considering the signed network characteristics, the discriminator learns to handle the imbalance in data.

6.4.1 Limitations of SigGAN: Although SigGAN ensures high effectiveness in comparison with the existing baselines irrespective of the dataset and presence of sparsity, we observe few limitations in SigGAN which we discuss next. A critical analysis of SigGAN would be that we did not explore the applicability of different discriminator models which can improve the performance. We evaluate the execution time for SigGAN and the other baselines on a machine with an Intel Core i7 4.2GHz CPU and 32 GB RAM. We observe during sparsity analysis that SigGAN is linearly scalable when the fraction of edges is increased gradually from 20% of the total edges to all the edges irrespective of the dataset. The time taken are 32, 40, 51, 60 and 72 minutes respectively for 20%, 40%, 60%, 80% and 100% of the edges for Epinions. Furthermore, on comparing the execution time of SigGAN with the baselines, we observe that the execution time of DNE-SBP is the best followed by SigGAN and SIDE. For example, the execution time on Epinions was 50, 72, 86, 106 minutes for DNE-SBP, SigGAN, SIDE and SNE respectively. We show our observations on Epinions as it is the dataset with the most number of edges (around 0.8 million edges). Therefore, although SigGAN has better performance than DNE-SBP, it requires more time to converge. We intuitively believe by exploring different discriminator models and by including structural role based information of the nodes in modified softmax, we can lower the execution time of SigGAN. We consider these as the future directions of the paper.

7 CONCLUSIONS

We propose a Generative Adversarial Network (GAN) based approach, SigGAN, for generating a representation of the nodes in signed networks. In signed networks, nodes with positively connected edges have similar representations whereas the representations of the nodes with negative edges have dissimilar representations. SigGAN is computationally efficient, handles imbalance in negative and positive edges and considers structural balance theory. Furthermore, it can be applied for predicting the sign of a link with high effectiveness. Validation on 5 datasets, like Slashdot, Epinions, Reddit, Bitcoin and Wiki-RFA indicate the SigGAN can predict links with a high mean micro F1-score of 0.75 – 0.96 which is higher than the existing state-of-the-art search works. As a future goal, we plan to extend SigGAN by exploring different discriminator functions, integration of role based information for node embedding and its applicability for directed signed networks. We, further, intend to make SigGAN applicable for temporal signed networks as most of the signed relationships formed in real life often undergo change in polarity with respect to time.

REFERENCES

[1] Agrawal, P., Garg, V. K., and Narayanam, R. Link label prediction in signed social networks. In IJCAI (2013), pp. 2591–2597.
[2] Biagi, G., Tang, J., and Liu, H. Signed link analysis in social media networks. In IJCSM (2016), pp. 539–542.
[3] Bhownick, A., K., Meneni, K., and Mitra, B. On the network embedding in sparse signed networks. In Pacific-Asia Conference on Knowledge Discovery and Data Mining (2019), Springer, pp. 94–106.
[4] Cartwright, D., and Harary, F. Structural balance: a generalization of heider’s theory. Psychological review 63, 5 (1956), 277.
[5] Cha, D.-K., Kang, J.-S., Kim, S.-W., and Choh, J. Rating augmentation with generative adversarial networks towards accurate collaborative filtering. In The World Wide Web Conference (2019), pp. 2616–2622.
[6] Cha, D.-K., Kang, J.-S., Kim, S.-W., and Lee, J.-T. Cgan: A generic collaborative filtering framework based on generative adversarial networks. In Proceedings of the 25th ACM international conference on information and knowledge management (2018), pp. 137–146.
[7] Chakraborty, R., and Chakraborty, N. Are we friends or enemies? let’s ask thy neighbour! in Proceedings of the 25th International Conference on Intelligent User Interfaces Companion (2020), pp. 61–62.
[8] Chen, X., Guo, J.-F., Pan, X., and Zhang, C. Link prediction in signed networks based on connection degree. Journal of Ambient Intelligence and Humanized Computing (2017), 1–11.
[9] Chen, X., Guo, J.-F., Pan, X., and Zhang, C. Link prediction in signed networks based on connection degree. Journal of Ambient Intelligence and Humanized Computing 10, 5 (2019), 1747–1757.
[10] Dass, D. Positive and negative link prediction algorithm based on sentiment analysis in large social networks. Wireless Personal Communications 102, 3 (2018), 2183–2198.
[11] Denton, E., Chintalapalli, S., Szlam, A., and Fergus, R. Deep generative image models using a laplacian pyramid of adversarial networks. arXiv preprint arXiv:1506.05751 (2015).
[12] Derr, T., Ma, Y., and Tang, J. Signed graph convolution networks. In 2018 IEEE International Conference on Data Mining (ICDM) (2018), IEEE, pp. 929–934.
[13] Derr, T., Wang, C., Wang, S., and Tang, J. Relevance measurements in online signed social networks. In Proceedings of the 14th International Workshop on Mining and Learning with Graphs (MLG) (2018).

| Dataset   | Slashdot | Epinions | Reddit | Bitcoin | Wiki – RFA |
|-----------|----------|----------|--------|---------|------------|
| APED      | 73       | 54.5     | 10.90  | 45      | 10         |
| ANED      | 74.50    | 58.56    | 12     | 48      | 11         |

Table 6: We show APED and ANED for Slashdot, Epinions, Reddit, Bitcoin and Wiki-RFA.
knowledge discovery and data mining

Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining (2016), pp. 1273–1282.

Proceedings of the 25th ACM Conference on Hypertext and Social Media (New York, NY, USA, 2014). HT ’14, ACM, pp. 144–157.

Proceedings of the 24th international conference on world wide web (2015), pp. 1067–1077.

Proceedings of the Eighth ACM International Conference on Web Search and Data Mining (2015). ACM, pp. 87–96.

Proceedings of the 26th ACM Conference on Hypertext and Social Media (New York, NY, USA, 2015). HT ’15, ACM, pp. 527–536.

Proceedings of the 27th ACM Conference on Hypertext and Social Media (New York, NY, USA, 2016). HT ’16, ACM, pp. 144–153.

Proceedings of the Ninth ACM International Conference on Web Search and Data Mining (2016). ACM, pp. 1375–1378.

Proceedings of the 28th ACM Conference on Hypertext and Social Media (New York, NY, USA, 2017). HT ’17, ACM, pp. 148–157.

Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (2018). ACM, pp. 592–600.

Proceedings of the 26th ACM SIGKDD international conference on Knowledge discovery and data mining (2016), pp. 1225–1234.

Proceedings of the 27th ACM Conference on Hypertext and Social Media (New York, NY, USA, 2016). HT ’16, ACM, pp. 144–153.

Proceedings of the 28th ACM Conference on Hypertext and Social Media (New York, NY, USA, 2017). HT ’17, ACM, pp. 148–157.

Proceedings of the 29th ACM Conference on Hypertext and Social Media (New York, NY, USA, 2018). HT ’18, ACM, pp. 148–157.

Proceedings of the 30th ACM Conference on Hypertext and Social Media (New York, NY, USA, 2019). HT ’19, ACM, pp. 148–157.

Proceedings of the 31st ACM Conference on Hypertext and Social Media (New York, NY, USA, 2020). HT ’20, ACM, pp. 148–157.
