Link Sign Prediction in Signed Networks

Hao Wu
wuhao20@stanford.edu

Yiyang Li
yiyang7@stanford.com

Li Guo
liguo94@stanford.edu

***Abstract*— Interactions and relationships in social networks can be either positive or negative. Link sign prediction can be used to infer relationships that are present, but whose nature remain undetermined. Understanding the 'sign' of these links and relationships is highly relevant to a number of interesting applications, ranging from friend recommendation to fraud detection.**

Our project studied two datasets, Wiki-Vote and Slashdot, and conducted link sign prediction on them. We extracted node degree features, low order features, high order features, and defined a new node embedding algorithm: CSN2V, as features for link prediction. We fed features into two machine learning models, logistic regression and fully connected neural network, to make the prediction. We made performance comparisons among different feature combinations, classifiers, configurations of the CSN2V random walk (different q, p settings), etc. Our experimental results provide insights into which features work best on which dataset and the reason behind it. Moreover, our CSN2V algorithm confirms the utility of the Theory of Balance. Overall, this paper serves as another evidence that the sign of a link can be informed by the relationships its endpoints have with others of the surrounding social network. 4**.**

I. INTRODUCTION

Social interaction on the social network can be both positive and negative – explicit signed links can represent relationships between people: friendship or enemy, support or disagreement, approval or disapproval. Link prediction can be used to infer latent relationships that are present but not recorded by explicit links, the sign prediction problem can be used to estimate the sentiment of individuals toward each other, given information about other sentiments in the network $[11]$. The interplay of positive and negative relations is very important in many social network settings, while the vast majority of online social network research has considered only positive relationships $[10]$. Therefore, We conducted signed Link prediction of in our project.

Our project draws inspiration from 4 link prediction papers $[1-4]$. We conducted performance analysis on signed link prediction of **existing** links using different combinations of degree-type, low and high-order features with machine learning based prediction algorithm. Our evaluation metrics will account for false positive/negative rates too. Furthermore, we explored more combinations of features in effort to further improve performance on different networks and gain insights.

A. Problem Statement

Given a graph G with nodes V and edges E, where the edges could have positive or negative signs; the task is to predict the unknown sign of an existing edge given the rest of the network.

II. RELATED WORK

Liben$[1]$ provides an overview of similarity based methods for solving the "unsigned" link prediction problem. It implemented different methods for computing the "similarity" score between nodes. However, the prediction accuracy only achieves to 54.8%, which can be further improved and the methods discussed cannot infer specific characteristics of interactions.

Jure$[2]$ studied signed link prediction for online social networks, where the sign denotes positive or negative relationships between nodes. The paper extracted two types of features from the network (signed degree of nodes, sub-structure/triad counts) and utilized a machine learning model, Logistic Regression, for prediction. The paper showed a great accuracy and suggested that triad features perform better than the degree features for predicitng edges of higher embeddedness. Their prediction model provides insight into Theories of Balance and Status from social psychology $[7]$, which is broadly utilized in link prediction

[2, 8]. However, the author only considered graph features on node-degree level and on traid (loop of length 3) level, instead of considering larger sub-structures (e.g. 4-5 nodes subgraph), or other network metrics like Motifs and graphlets, or even node roles.

Kai-Yang[3] conducted a similar study to the last one in the sense that they both stem from the Social Balance theory, but his study recognized aspects that were overlooked by the previous paper. For example, it recognized false positive rate as an essential evaluation metric, and exploited higher-order features. With these higher order features added, this paper discovers that the false positive rate also drops on 3 real-world networks: Epinions, Slashdot and Wiki-Vote. This paper also abandoned using degree-type features, such as positive in-degree of a node, as they believe that nodes have their own predispositions that don't necessarily extrapolate well to the rest of the network.

Yuan[4] presented a "signed" network embedding model called SNE. The SNE adopts the log-bilinear model, assigning a pair of 'source' and 'target' feature vectors to each node. Then, the 'source' embeddings of all nodes along a given path multiplied with two signed-type vectors, corresponding to the positive or negative sign of each edge along the path, to obtain the 'target' node embedding for the destination node of any given walk [4]. A reverse pass is used to derive the 'source' embeddings, aggregated from target embeddings of nodes along a walk. This paper also presented a simpler version of their algorithm, called SNEs, where only 1 embedding is used instead of the pair of source and target embeddings. The paper conducted link prediction, on both directed and undirected signed networks and showed the effectiveness of their signed network embedding by comparing results against three state-of-the-art unsigned network embedding models.

Jerome[9] used various signed spectral similarity measures, including squared adjacency matrix, matrix exponential, and Inverted Laplacian with dimensionality reduction. The study showed that the network exhibits multiplicative transitivity, which is consistent with the Balance theory [7]. However, the prediction accuracy can be further improved, as its best model only achieves 67% accuracy with no mentioning of AUC.

III. Dataset

We firstly built three simple, signed, and directed graphs with different numbers of 4/5-order cycles, triads, etc., to test the validity of our feature extraction implementations.

Then, Wiki-Vote dataset and Slashdot dataset were utilized after successfully conducting tests on our toy graphs. The reason we chose these datasets is that they are used across most of our referenced papers, using these 2 datasets then allow us to compare our results against the existing papers.

Wiki-Vote is a network corresponding to votes cast by Wikipedia users in elections for promoting individuals to the role of admin. A signed link indicates a positive or negative vote by one user on the promotion of another (+ for a supporting vote and for an opposing vote). It has 2,794 elections with 103,747 total votes and 7,118 users participating in the elections (either casting a vote or being voted on). The resulting network contains 7,118 nodes (users) and 103,747 edges of which 78.7% are positive.

Slashdot is a network from the technology-related news website, Slashdot, where users connect to each other as friends or foes. This network contains 82,144 nodes (users), and 549,202 edges (relationships) of which 77.4% are positive. 70,284 users received at least one signed edge, and there are 32,188 users with non-zero in- and out-degree.

The following table summarizes some key characteristics of our 2 datasets which we will reference later on.

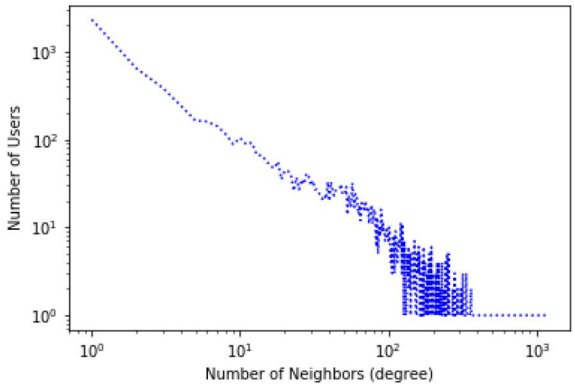

Fig. 1. Degree distribution For Wiki-Vote

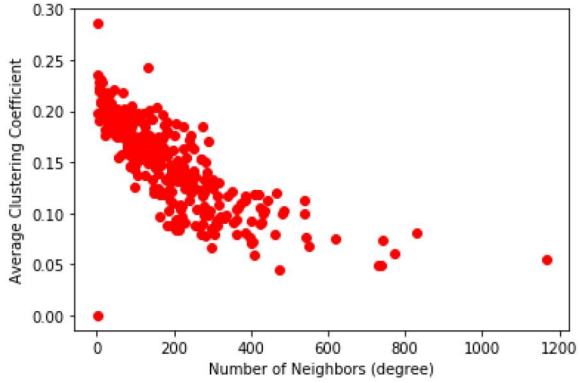

Fig. 2. Distribution of clustering coefficients vs. Node degree for Wiki-Vote

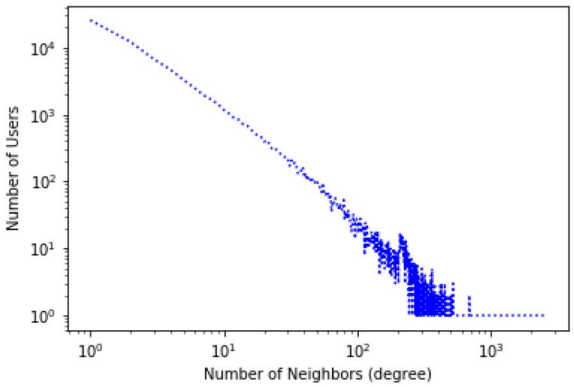

Fig. 3. Distribution of clustering coefficients vs. Node degree for Slashdot

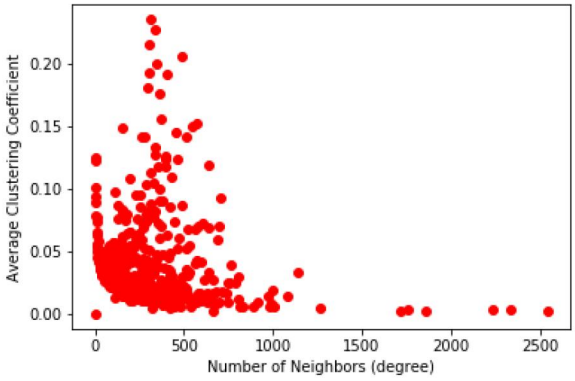

Fig. 4. Distribution of clustering coefficients vs. Node degree for Slashdot

Dataset	Wiki	Slashdot
Nodes	7,115	82,168
Edges	103,689	948,464
Avg Clustering Coeff	0.1409	0.0603
Num Triangles	608,389	602,592

TABLE I

DATASET CHARACTERISTICS

IV. METHODS

We utilized two machine learning models: logistic regression and multi-layer perceptron (MLP) with 3 hidden layers of size 64, 32, 32 respectively. We feed classifiers with combinations of different per-edge features.

Logistic regression learns a model of the form

$$P(+|x) = \frac{1}{1 + e^{-(b_0 + \sum_i^n b_i x_i)}}$$

Where x is feature vector, and $b_0, b_1, ..., b_n$ are feature weights we estimate based on training data [5].

MLP is a supervised learning algorithm that learns a non-linear function by training on a dataset. Given a feature vector f and a target t, it can learn a non-linear function approximator for either classification or regression. It is different from logistic regression, in that between the input and the output layer, there can be one or more non-linear layers, called hidden layers. As a result, an MLP can model more complex functions and could fit to the underlying pattern of datasets

better.

Inorder to be able to compare our results with existing papers, we adopt their common training scheme. We partition all edges of a network to 10% as validation set, and 90% as training set. We then train our model with 10 fold cross-validation, and average our results.

Algorithms discussed below are used for feature extraction.

Existing Algorithms

Signed Triad Motif counts We consider each triad involving the edge (u, v) with another node w. There are 16 distinct types of triads involving (u, v), by considering edge directions and edge signs of edges between u, w and v, w, which leads to $2 \cdot 2 \cdot 2 \cdot 2 = 16$ permutations. Each of these 16 triad types may provide different evidence about the sign of (u, v), adhering to Theory of Balance in Sociology. We encode this information in a 16-dimensional vector specifying the number of triads of each type that (u, v) is involved in.

Signed k-th order path counts (High Order Feature) As described in the project proposal, the counts of all different configurations of length k path with end points (i, j) can be found from the (i, j)-th entry of all permutations of:

$$(A_{+|-}^{1|T})^{k-1}$$

where 1 is identify and T is transpose (they counts for different edge directions), + means only keeping the positively weighted adjacencies and - means keeping only the negative.

Some of our other features:

Similarity Score	Expression
common neighbors	$\mid \Gamma(x) \cap \Gamma(y) \mid$
Jaccards coefficient	$\frac{\mid \Gamma(x) \cap \Gamma(y) \mid}{\mid \Gamma(x) \cup \Gamma(y) \mid}$
preferential attachment	$\mid \Gamma(x) \mid * \mid \Gamma(y) \mid$

TABLE II
DIFFERENT SIMILARITY SCORE

Cumulative Signed Node2vec

After reading the SNE paper[4], we discovered a shortcoming with their algorithm. Thus, we proposed a better algorithm for obtaining node embeddings in signed networks, and we call it Cumulative Signed Node2vec (CSN).

But first, let's analyze the embedding formulation described in the paper:
Given a random walk path [h, u_1, u_2, ... , u_l], where h is our starter/target node.

$$V_h^{(t)} = \sum_{i=1}^{l} c_i \odot V_{u_i}^{(t-1)}$$

where $c_i := c_+$ if $W_{u_i,u_{i+1}} = 1$ and $c_i := c_-$ if $W_{u_i,u_{i+1}} = -1$
V is node embedding, W_{u_i,u_j} is edge weight, c_+ and c_- are 2 trainable vectors of the same size to the embedding, t is time step, and $\odot$ denotes element-wise multiplication.

For a new stop(node) in the path, this formulation does not consider signs of all previous edges when incorporating the new node's embedding to that of the target node; rather, their algorithm is only concerned with the sign of a single local step. Drawing from the Theory of Balance, this formulation is not ideal, and a simple example can illustrate why:

Say that we have 3 nodes a, b, and c. In the first configuration, when we look at edge (b, c), it makes sense that c's embedding should contribute negatively to a, since an enemy(c) of my friend(b) is also an enemy to me(a). While in the second configuration, when we look at edge (b, c), it instead makes sense that c's embedding should contribute positively to a because enemy(c) of my enemy(b) should be my friend (a).

Thus, although the weight of edge (b, c) are both negative in the above 2 configurations. The effect they imply on c's contribution to a are different. And the type of contribution of some node, positive or negative, would actually be

better captured by multiplying all the signs of edges up until that node in the walk. This is consistent with the Theory of Balance, and is the intuition behind our own algorithm.

Therefore, we propose a new formulation for node embedding derivation based on Word2Vec: we treat each random walk as a sentence, and each node visited in a walk is a word. Each node v is associated with 2 "signed" word: w_{v+} and w_{v-}, and which word to use for a node depends on the "cumulative sign" of the node in the walk from the starter. We define the cumulative sign for node v_l as follows:

$$sign(v_l) = \prod_{i=start}^{l-1} sign(v_i, v_{i+1})$$

Where $sign(v_i, v_{i+1})$ denotes the sign, 1 or -1, of edge (v_i, v_{i+1}). Then we use w_{v+} if $sign(v_l) = 1$ and w_{v-} otherwise.

For example, in the first configuration above, the "sentence" starting from a would be {a, b, -c}, while the "sentence" would be {a, -b, c} for the second configuration.

Using the power iteration approach, for each random walk [v_0, v_1, v_2 ... v_l], we update the embedding of start node v_0 at time step t with the following formula:

$$E_{v_0}(t) = \prod_{i=0}^{l-1} sign(v_i, v_{i+1}) E_{v_{i+1}}(t-1)$$

Then, we keep updating the embeddings until convergence.

In reality, we train for the embedding of all the signed words using a standard word2vec model, and the final embedding for a node v is the element-wise sum of its negative and positive word embeddings. This is from the intuition that both the positive and negative embedding of a node captures "meaning", structural and semantic, about a given node, and adding them up would aggregate these meanings. e.g $-x$ negatively contributes to y, and x contributes to z; then x should be a combination of both.

Our algorithm generates node embeddings, but the link sign prediction problem requires edge features. Thus, the edge feature should be generated from the node embeddings of its two endpoint nodes. Their are many ways to combine node features, including concatenation, hadarmard product, dot product, and l2 distance; all of which will be explored in our experiments to determine the optimal way of combining features.

V. Results and Discussion

It's mentioned that Wiki Election dataset only contains 21.6% negative edges, which means classes are imbalanced. Therefore, accuracy will not be a good enough metric in this case, which is why we also use AUC, a more robust metric.

The experiments we carried out aim to verify 4 things. First, whether a more sophisticated machine learning model will help improve our prediction performance. Second, exactly which configuration of second-order random walk (q, p value) will give us the most effective CSN2V node embeddings for link prediction. Third, which way of combining our CSN2V node embeddings make up the best edge feature for the best prediction performance. Lastly, which combination of features (each optimally tuned) arrive at the best prediction AUC and accuracy.

For our first goal, we adopted a simple Logistic Regression machine learning model, and later a 3 layered Neural Network with ReLu activation to compare their performances.

Feature Combination	Logistic Regression	Neural Network
Low	0.675	0.676
High	0.752	0.781
Low + High	0.828	0.828

TABLE III
Logistic Regression vs. Neural Network - AUC, Wiki-Vote

As we can see from the above results, using the 3 layered neural network achieves slightly

Feature Combination	Logistic Regression	Neural Network
Low	0.609	0.636
High	0.792	0.892
Low + high	0.829	0.917

TABLE IV

LOGISTIC REGRESSION VS. NEURAL NETWORK - AUC, SLASHDOT

Configuration	AUC	Accuracy
q = 1, p = 1	0.564	0.7811
q = 100, p = 1	0.662	0.8103
q = 100, p = 0.01	0.692	0.8253
q = 0.01, p = 1	0.5508	0.7781

TABLE VI

DIFFERENT CONFIGURATION OF CSN2V ON SLASHDOT

better results in general, and the intuition is that a deeper model can model more sophisticated mathematical functions and thus fitting to the underlying pattern of our data better.

For our second goal, we considered 4 configurations of q and p of our second-order random walk[12]. To give some background, p is the unnormalized return probability, which is the probability to return back to the previous node, and q is unnormalized walkaway probability, which is the probability to move outwards. We then derived CSN2V node embeddings for these 4 configurations, and used them for the link sign prediction task individually. Finally, we compared results.

The 4 configurations of p and q pairs are: (q = 1 and p = 1), (q = 0.01 and p = 1), (q = 100 and p = 1), (q = 100 and p = 0.01). We always walk 10 times from each node, and the walk length is always 80. We also selected to use the concatenation of node embeddings and their hadamard product for every edge as its features, as this combination in general produce better accuracy than others (more details later on).

Configuration	AUC	Accuracy
q = 1, p = 1	0.547	0.7964
q = 100, p = 1	0.6024	0.808
q = 100, p = 0.01	0.6148	0.81
q = 0.01, p = 1	0.5384	0.7971

TABLE V

DIFFERENT CONFIGURATION OF CSN2V ON WIKI-VOTE

As shown in the above table, the configuration of (q = 100, and p = 0.01) achieves the best AUC and prediction accuracy. The intuition is that second-order random-walk with a high q and low p prioritizes BFS styled exploration, which gives the structural role of each node. In our case, the BFS styled random walk explored the close neighborhood of each node, understanding possible triads within its egonet. This is equivalent to deriving the low-order triad features, and unsurprisingly, its performance is very similar to that of the low order features. One could argue that a deeper BFS might account for the high order features as well, but that would require longer and more walks and careful tuning of p and q, which we have not explored yet.

For our third goal, we experimented with 4 ways of utilizing the 2 endpoint-nodes' embeddings for edge features: dot product, hadamard product, concatenation, and l2 distance. We contrast their performances below:

Configuration	AUC	Accuracy
Hada	0.5310	0.7878
Hada + Concat	0.6148	0.8134
Hada + Concat + L2	0.6208	0.8116
Hada + Concat + dot	0.6193	0.8101

TABLE VII

DIFFERENT EMBEDDINGS IN WIKI-VOTE

Configuration	AUC	Accuracy
Hada	0.564	0.782
Hada + Concat	0.692	0.825
Hada + Concat + L2	0.696	0.822
Hada + Concat + dot	0.688	0.824

TABLE VIII

DIFFERENT EMBEDDINGS IN SLASHDOT

Based on AUC, we determined that the best way to combine node embeddings of 2 endpoints is to concatenate their node vectors with their hadamard product and l2 distance, despite that

using l2 distance seems to lower the accuracy. Thus, all the experimental results tabulated below use this specific node embedding combination.

For our last goal, we combine four types of features: node degree features (Deg), Low order features (Low), high order features (High), and Cumulative Signed Node2vec features (CSN2V).

Node degree features corresponds to the following 7 properties for a directed edge (u, v): $d^+_{out}(u)$, $d^-_{out}(u)$, $d^+_{in}(v)$, $d^-_{in}(v)$, $C(u, v)$, $d^+_{out}(u) + d^-_{out}(u)$, $d^+_{in}(v) + d^-_{in}(v)$, where $C(u, v)$ is the total number of common neighbors of u and v in an undirected sense. Low order feature corresponds to number of different triad motifs the given edge is involved in, where there are 16 types of triads in total considering 2 different edge directions and edge signs. High order feature corresponds to number of different types of length 4 or 5 cycles that the given edge is involved in.

Metrics	AUC	Accuracy
Low	0.675	0.818
Low + Deg	0.572	0.565
Low + High	0.828	0.893
Low + High + Deg	0.786	0.896
Low + CSN2V(q, p=1)	0.673	0.783
Low + CSN2V(q, p=1) + Deg	0.531	0.638

TABLE IX

PERFORMANCE FEATURE COMBINATIONS WITH DEG AND WITHOUT DEG ON WIKI-VOTE

One observation is that Degree features only lower the performance when combined with any other features. This confirms with the intuition of chiang11 [3] (section 2).

Now, given the information we have from the previous experiments, we use Neural Network as our final prediction model. For our CSN2V, we select the node embeddings derived from a random walk of (q = 100, p = 0.01), and we combine the node embeddings by concatenating their node embeddings, hadamard product, and l2 distance. Then the different combinations of features are up for comparison below:

Metrics	AUC	Accuracy
Low	0.6355	0.8103
High	0.7715	0.8923
Low + high	0.8292	0.917
CSN2V (q = 100, p = 0.01)	0.696	0.822

TABLE X

DIFFERENT FEATURE COMBINATIONS ON SLASHDOT

Metrics	AUC	Accuracy
Low	0.676	0.834
High	0.7812	0.8837
Low + High	0.828	0.886
CSN2V(q=100, p=0.01)	0.6148	0.81

TABLE XI

DIFFERENT FEATURE COMBINATIONS ON WIKI-VOTE

From above results, we have the following observations and discussion:

- High order features outperform low order features.
  First of all, both networks had small clustering coefficients. The Wiki-Vote network has an average clustering coefficient of 0.14, and the Slashdot has only 0.06, this confirms with the reasoning in Chiang[3] that there isn't enough low order triads to sufficiently inform link sign prediction. Indeed, the high order features helped achieve greater prediction accuracy and AUC on its own, and when combined with lower order features, we obtain our best results across the 2 datasets.

- High order features outperforms low order features by a larger margin on Slashdot.
  Given the properties of the Wiki-Vote and Slashdot networks[secion III] mentioned in the above point, this observation can also be justified. It is worth noting that although the Slashdot network had more than 10 times the nodes of Wiki-Vote, it had less triangles than Wiki-Vote, this fact could explain why Low order features performed worse on Slashdot than on Wiki-Vote. By the same token, the high order feature outperforms the low order feature by a larger margin on the Slashdot network.

- Low + High performs the best.
  This confirms what's proposed in the chiang11 paper, that for many nodes with low clustering coefficient (not in any triads), high order features serve as a great supplement and improve overall link prediction performance. Moreover, high order features brings more information from larger parts of the graph, which aids the other more local features.

We also compare our CSN2V performance with SNEs. The reason we are not comparing with SNEst is that our algorithm does not assign 2 embeddings to each node, aka a source embedding and a target embedding. Our algorithm uses the same embedding regardless of whether a node is pointed to or from in a random walk, thus making CSN2V most comparable to SNEs where each node is assigned 1 embedding only. It is also worth mentioning that the SNE paper did not use AUC as a metric, and accuracy is not a good metric due to the class imbalance of the dataset. Since the only common dataset we used with the SNE paper is Slashdot, we tabulate the results below:

Method	Accuracy
CSN2V	0.822
SNEs	0.6080
SNEst	0.9328

TABLE XII
CSN2V VS SNES ON SLASHDOT

This shows that our algorithm outperforms SNEs on the Slashdot dataset, which validates our hypothesis that introducing the Theory of Balance will improve signed-node2vec's applicability to the specific task of link sign prediction. Furthermore, this result may entail that our CSN2V algorithm, if incorporate the 2-embedding approach, could potentially achieve better results too, and this is an exciting future direction that we would love to explore. Moreover, the AUC of the SNE algorithms was not calculated in the paper, so the comparison of our results are not as legitimate.

To find our code and result:
https://github.com/Matt-F-Wu/CS224W_Project

VI. CONCLUSION

Link sign prediction is a well studied problem with many proposed solutions. Given only the structure of the network, we can achieve more than 82% AUC and over 90% accuracy on datasets like Wiki-Vote and Slashdot with low-order and high-order features combined. We also saw the potential of a Cumulative Signed Node2vec algorithm in the task of link sign prediction, which draws inspiration from the Theory of Balance. It is worth noting that each of these algorithms corresponds to and may even stem from theories in sociology and the deep understanding of different types of human interaction. By understanding the type of the networks and the nature of interactions within them, we may be able to develop better-performing algorithms that are customized for the data and the problem we have.

Contributions:

Hao Wu: creating the CSN2V algorithm and implementing the high order feature extraction program, training models with teammates. implement the training framework.

Yiyang Li: implemented node2vec feature, node2vec extractor, helped develop data pipeline, and trained the models with teammates

Li Guo: Extracted degree feature and Low order feature, training models with teamates. Conducted data visualization.

REFERENCES

[1] Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. Journal of the American society for information science and technology, 58(7):10191031, 2007.

[2] J. Leskovec, D. Huttenlocher, J. Kleinberg. Predicting Positive and Negative Links in Online Social Networks. In Proc. WWW, 2010.

[3] K. Chiang, I. S. Dhillon, N. Natarajan, and A. Tewari. Exploiting Longer Walks for Link Prediction in Signed Network. In Proc. CIKM, 2011.

[4] Yuan, S., Wu, X., Xiang, Y.: SNE: signed network embedding. In: Advances in Knowledge Discovery and Data Mining, pp. 183195. Springer, Cham (2017).

[5] 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. ACM, 2016.

[6] Neural Network https://scikit-learn.org/stable/modules/neural networkssupervised.html
[7] D. Cartwright, F. Harary. Structural balance: A generalization of Heider's theory. Psychological review, 1956.
[8] S. Marvel, S. Strogatz, J. Kleinberg. Energy landscape of social balance. Physical Review Letters, 103, 2009.
[9] J. Kunegis, A. Lommatzsch, C. Bauckhage. The Slashdot Zoo: Mining a social network with negative edges. In Proc. WWW, 2009.
[10] M. E. J. Newman. The structure and function of complex networks. SIAM Review, 45:167256, 2003.
[11] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. J. Amer. Soc. Inf. Sci. and Tech., 58(7):10191031, 2007.
[12] Lecture 9: Graph Representation Learning