LineDi2Vec: An Edge-Based Graph Embedding on Signed Social Networks

Chen Xing[1] and Masoud Makrehchi[1]

Department of Electrical, Computer and Software Engineering,
Ontario Tech University, Oshawa, Canada
chen.xing@ontariotechu.net, masoud.makrehchi@ontariotechu.ca

Abstract. Network data plays a crucial role in various real-world applications, as connections between entities can be represented and analyzed through graphs. These include various types such as social, information, and technical networks. However, the complex topologies of these networks present challenges in converting graph data into machine-readable vector formats. Existing models like Graph Neural Network, Graph Attention Network, and node2vec have made strides in graph embeddings. Particularly for edge-related tasks, models like node2vec often resort to indirect methods like node concatenation for vector representation of edges, aiding in tasks like link sign prediction. In this paper, we introduce LineDi2vec, an innovative approach that uses a line graph to enhance the node2vec embedding method. This method effectively captures key social network theories, namely status and balance theories. LineDi2vec not only generalizes the original graphs, transforming the relationships between edges and nodes, but also maintains the topological integrity of the original graphs for effective node embedding by node2vec. We conducted an evaluation of LineDi2vec on four real-world datasets, focusing on link sign prediction. The results demonstrate its superior performance over traditional node concatenation methods and comparable efficacy to state-of-the-art GAT and GNN methods.

Keywords: Graph Embedding · Line Graph· Link Sign Prediction.

1 Introduction

The study of signed networks has gained significant attention with the proliferation of online social interactions, where relationships manifest as both positive (friendship, trust) and negative (conflict, distrust) connections [16]. Understanding these networks draws heavily from social psychology theories, particularly balance theory and status theory [18], which inform critical tasks like link sign prediction. Recent advances in network representation learning have explored two main approaches: graph neural networks (GNNs) that employ message passing mechanisms [7,9,23], and algorithmic methods like node2vec [8]. While GNNs achieve state-of-the-art performance, their requirement for careful hyperparameter tuning contrasts with the flexibility of random-walk based methods, which

nonetheless suffer from limitations in edge representation. This work bridges these approaches by proposing a novel line graph-enhanced embedding method, building upon recent developments in signed network analysis [15,5,1].

It is significant to note that a line graph-enhanced edge embedding method, called Line2vec, has already been introduced in 2019[2]. This method employs the concept of collective homophily to represent edges directly within weighted line graphs[2]. Despite its novel approach, the primary validation tasks focused on edge clustering and classification, leaving critical link-related tasks, such as link sign prediction, unaddressed. Moreover, the study[2] concentrated on undirected homogeneous graphs, thereby not covering the complexities of sign and direction prediction that are quintessential in directed social networks.

In our approach to modeling signed networks, we developed a novel edge-centric random walk algorithm that incorporates line graphs. This adaptation enables the traditionally node-focused random walk method to directly convert signed edges into vector representations. This overcomes the limitations faced by existing algorithms in embedding signed graphs into vectors. Our method also integrates relevant social theories, utilizing traditional classifiers to effectively capture and learn the distinctive features of signed edges within our embeddings. To train our model, we reconstruct the signed networks in the form of their corresponding line graphs. In this transformed structure, edges are treated as nodes, allowing for a more effective and nuanced representation of the original network's relational dynamics.

The major contributions of this paper are as follows:

- We present a novel edge-centric model that enhances existing random walk-based node embedding algorithms with the use of line graphs. This model innovatively transforms edges into nodes with associated signs, effectively addressing the limitations of node2vec in embedding signed network edges.
- We carried out link sign prediction experiments using four real-world signed social network datasets to showcase the effectiveness of our proposed model.
- Through the application of our LineDi2vec model to our case study dataset, we discovered that the model exhibits certain limitations when applied to graphs. Moreover, it appears that the model is more adept at identifying patterns within the social network beyond merely the topological information of the network.

2 Related Work

2.1 Node2vec

Node2vec is a representational learning algorithm on graphs. It is capable of learning continuous feature representations for the nodes within any given graph, which can subsequently be employed in a variety of downstream machine learning tasks.

The operation of node2vec can be divided into two main stages:

- Random Walk Generation: Initially, node2vec employs a flexible notion of second-order random walks, which are not purely random but biased based on the exploration-exploitation trade-off. This approach generates sequences or "sentences" of nodes, where each sentence represents a walk across the graph. The path chosen at each step in the walk depends not only on the structure of the graph but also on a set of parameters that control the walk's locality and fidelity to the starting node.
- Embedding Learning: The sequences of nodes collected in the first step are treated akin to sentences in natural language processing. This corpus of node sequences is then fed into a Skip-Gram model, a type of shallow neural network, to learn vector representations or embeddings for each node. These embeddings aim to place nodes that frequently co-occur in walks close to one another in the vector space, thus preserving the topological similarities between nodes within the learned embeddings.

2.2 Line Graph

In graph theory, a line graph is a special kind of graph constructed to represent the relationships between the edges of another graph. The process involves creating a vertex in the line graph for each edge in the original graph. These vertices are then connected if their corresponding edges in the original graph have a common endpoint. As a result, the line graph, typically represented as L(G) for a graph G, effectively mirrors the adjacency and connectivity of the edges from the original graph.

Line graphs are notable for their unique structural properties, which include specific patterns and configurations that are not permitted, known as forbidden subgraphs[3]. These characteristics enable line graphs to be recognized and processed efficiently, often in linear time. The concept of line graphs has been expanded and explored in various contexts, leading to the study of line graphs of multigraphs[22], hypergraphs, and even weighted graphs[6], showcasing their broad applicability.

Unlike the research Line2vec[2], which also utilized line graph transformations but focused primarily on the topological characteristics and "Collective Homophily"[2] of the graph, The dynamics of directed and signed networks are often underpinned by sociological theories. These theories provide a foundational framework for understanding the complex interactions within such networks, indicating that the essence of link sign prediction transcends mere structural analysis and encompasses sociological insights, which we will explore further in the subsequent subsection.

2.3 Sociological Theory

Balance Theory and Status Theory are two pivotal sociological theories that are instrumental in the analysis and modeling of signed directed networks. In the following section, we will provide a concise overview of these two theories and

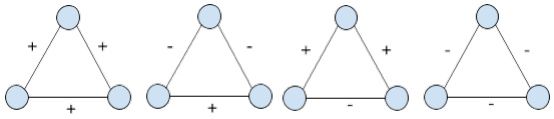

Fig. 1. Example of Balance Theory.

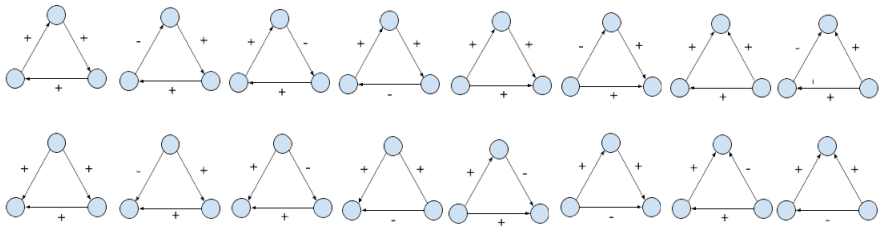

Fig. 2. Example of Status Theory.

then proceed to compare their applicability and insights across four real-world datasets.

Balance Theory: Proposed by Heider [11], Balance Theory states that signed triads tend toward stable configurations where the product of edge signs is positive (Fig. 1). A triad is balanced with either three positive edges or one positive and two negative edges, reflecting social axioms like "a friend's friend is a friend." This principle has become foundational for signed network analysis [18].

Status Theory: Status Theory [4] interprets signed directed edges as status indicators: a +A→B link signals B's higher status, while -A→B denotes lower status (Fig. 2). Though more complex than balance theory, it better explains directed social networks. Combined, these theories provide complementary perspectives for real-world signed network analysis.

Despite our proposed method for the link sign prediction task adhering to node2vec—a node embedding algorithm—and employing node concatenation to represent edges within the embedding space as our experimental baseline, it remains essential to benchmark against leading-edge sign prediction methodologies for a comprehensive evaluation. To this end, this paper selects SiGAT[12] and SDGNN[13] as comparative baselines. These state-of-the-art approaches represent alternative embedding strategies, providing a diverse context for assessing the efficacy of our proposed method against the backdrop of current advancements in the field of sign prediction.

3 Propose Method

Based on the previous discussion of sociological theory and related work on Sign Prediction, the proposed LineDi2vec algorithm is introduced in this section.

3.1 Problem Statement

For a directed sign graph $G = (V, \epsilon, s)$, where V is a set of nodes or vertices in graph G and ϵ is the set of edges between the vertices, and s is the signs assigned to the edges. The graph can be represented as adjacency matrix A where each entry A_{ij} represents the existence of an edge between vertices V_i and V_j. If $A_{ij} = 1$, then we say the edge is positive. If $A_{ij} = -1$, then we say the edge is negative. If $A_{ij} = 0$, there is not edges exists between the given nodes. We can also represent this social network by edgelist $E = (V_i, V_j, s)$ where V_i and V_j represent the start and end node IDs of given edges and s is the sign of this edge defined with 1 and -1. Given the edgelist $E = (V_i, V_j, s)$, the link sign prediction problem here becomes an edge classification problem where the model will only need to identify and cluster the s assigned to the edges.

3.2 Line DiGraph Tranformation

As stated previously, the link sign prediction task here has become an edge sign classification problem with sign s being assigned to each edges. For the original graph $G = (V, \epsilon, s)$, we already have a similar expression as the edgelist form: $E = (V_i, V_j, s)$. Both of these two expressions have the same adjacency matrix A. Therefore, for a line graph:

$$L(G) = (V, \epsilon) \tag{1}$$

$$V = G(\epsilon) \tag{2}$$

and

$$L(\epsilon) = L(V_i, V_j) \tag{3}$$

if $L(V_i, V_j)$ share a common endpoint in G. Figure 4 shows an example of line graph transformation on a small sample graph.

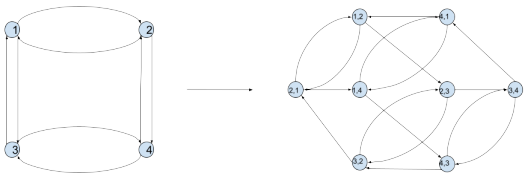

Fig. 3. An example of Line Digraph Transformation: the edge are transferred into nodes and the edges of the line graph also have directions based on common nodes in original graph.

Here, we can simply assign the s in edgelist $E = (V_i, V_j, s)$ to $L(G)$ by the Whitney isomorphism theorem[24]. In this scenario, the edges of the original/root graph G are now vertices of the line graph $L(G)$, with the original edge sign assigned to them. This transforms the link prediction problem into a node classification problem, a classic challenge for node-centric embedding methods like node2vec[8] to address.

3.3 Link Sign Prediction

The line graph $L(G) = (V, \epsilon, s)$ we created can then be processed using the node2vec algorithm. As previously stated, node2vec is a random walk-based algorithm that treats random walks as sentences in a language corpus, aiming to generate a one-to-one node-based vector representation for further analysis. Subsequently, signs can be assigned to the nodes in the embedding vector space generated by node2vec, which reduces the problem from node classification to binary classification. The vector space representation, along with the assigned signs to its nodes, are well-organized for processing by any downstream classifier.

3.4 Time Complexity of LineDi2vec

The analysis of LineDi2vec's time complexity constitutes a significant aspect of our study, particularly due to the implications of line graph transformation. As discussed in related works section, this transformation can substantially increase the quantity of nodes processed by the node2vec algorithm, which exhibits a time complexity of $O(n \log n)$ as highlighted by [20]. Theoretically, for a graph comprising n nodes, its corresponding line graph could encompass up to n^2 edges, leading to a quadratic time complexity of $O(n^2)$. Given that the output from the line graph transformation serves as the input for the node2vec algorithm, the overall time complexity is derived as:

$$O(n^2) + O(n \log n) = O(n^2 + n \log n) \approx O(n^2) \tag{4}$$

However, in practical scenarios, particularly in social networks and their line graphs, a linear time relationship is often observed, where the number of edges maintains a linear correlation with the number of nodes. Under these circumstances, encountered throughout this paper, the time complexity of LineDi2vec adjusts to:

$$O(n) + O(n \log n) = O(n + n \log n) \approx O(n \log n) \tag{5}$$

Thus, the time complexity of LineDi2vec aligns with that of node2vec, as $O(n)$ grows significantly slower than $O(n \log n)$. It is crucial to note, however, that the identical time complexities of LineDi2vec and node2vec do not necessarily translate to equivalent processing durations. Indeed, when comparing the operational times of LineDi2vec against node2vec and node concatenation approaches, a linear correlation with the time complexity $O(n)$ is observed.

4 Experiment

In this section, we focus on conducting link sign prediction to evaluate the efficacy of our model in enhancing signed network embeddings. Link sign prediction involves forecasting the unobserved signs of existing edges in a test dataset, utilizing information from a training dataset. This task is a key metric for assessing the performance of network embedding methods.[18][5] We have adopted their experimental framework for our analysis.

4.1 Experiment Setup

Dataset: For our experiments, we utilized four real-world signed social network datasets: Bitcoin-Alpha[17][16], Bitcoin-OTC[17][16], Slashdot[19], and Epinions[21]. The Bitcoin-Alpha and Bitcoin-OTC datasets are networks from platforms where individuals trade using Bitcoin. In these networks, members rate each other on a scale from -10 (total distrust) to +10 (total trust) in increments of 1. Ratings above 0 are considered positive, while others are negative. The Slashdot dataset originates from a technology-related news website with a vibrant user community. The website's Slashdot Zoo feature allows users to label each other as friends or foes, creating a network with clear positive and negative relationships. Lastly, the Epinions dataset is derived from Epinions.com, a consumer review site. Here, members can express trust or distrust in the reviews of others, forming a network that mirrors their opinions and trust relationships.

Baseline: To evaluate the effectiveness of LineDi2vec, we have set the baseline for our model as the traditional node concatenation method. The sole distinction between our proposed method and the baseline lies in the implementation of the line graph. All other parameters, including data preprocessing and the downstream binary classifier, remain identical.

Link Sign Prediction: For the link sign prediction task, The dataset already have the sign*s*. Therefore, in our experiment, we only need to seperate dataset into training and testing set for our model to process. The train-test split ratio we utilized is 70%-30%. For the node2vec parameters, we opt for 8 walkers, 30 walk lengths, 64 dimensions, and 200 walks. The hyperparameters for node2vec, p/q, are selected from a range of 0.1 to 10, with increments of 0.1 in the 0.1 to 1 range and increments of 1 in the 1 to 10 range. The optimal p and q parameter settings for each node2vec entry are omitted here for simplicity.

4.2 Result compared with Baseline

We report the experimental results in Table 1, with the highest value for each metric highlighted. The results demonstrate that::

- Link sign prediction in directed networks can be effectively formulated as binary classification by preserving edge signs during line graph transformation, enabling compatibility with various downstream classifiers.
- Our method's strong performance demonstrates that line graph transformations preserve both structural properties and balance/status theory relationships from the original network.
- While node concatenation (node2vec) works reasonably for small datasets (Bitcoin Alpha/OTC), its effectiveness degrades with larger networks despite remaining a viable baseline approach.
- Line graph enhancement consistently outperforms node concatenation across all metrics, with particularly significant AUC improvements (average about 13% boost), demonstrating better class separation and scale/threshold invariance. (Best results shown in Table

Table 1. Experiment result with 4 dataset

Dataset	metrics	node2vec	LineDi2vec
Bitcoin Alpha	Accuracy	0.9051	**0.9224**
	F1 Score	0.9335	**0.9583**
	AUC	0.7576	**0.8813**
Bitcoin OTC	Accuracy	0.9011	**0.9294**
	F1 Score	0.9423	**0.9624**
	AUC	0.7643	**0.8784**
SlashDot	Accuracy	0.8756	**0.9135**
	F1 Score	0.7526	**0.9533**
	AUC	0.6709	**0.8674**
Epinion	Accuracy	0.8985	**0.9287**
	F1 Score	0.8214	**0.9620**
	AUC	0.8081	**0.8952**

Table 2. Result Compared with SiGAT and SDGNN

Dataset	metrics	SiGAT[12]	SDGNN[13]	LineDi2vec
Bitcoin Alpha	F1 Score	0.9714	0.9729	0.9583
	AUC	0.8872	0.8988	0.8813
Bitcoin OTC	F1 Score	0.9602	0.9647	0.9624
	AUC	0.9055	0.9184	0.8784
SlashDot	F1 Score	0.9055	0.9128	0.9533
	AUC	0.8874	0.8977	0.8674
Epinion	F1 Score	0.9593	0.9628	0.9620
	AUC	0.9333	0.9411	0.8952

4.3 Comparision with SiGAT and SDGNN

In this section, we compared our LineDi2vec result on all four dataset with the current state-of-art model SiGAT[14] and SDGNN[13]. Although these two model are based on graph attention network and graph neural network, which is not necessary the same condition with our baseline, we still want to show the great potential of our line graph enhancement since both these two use node concatenations on the representaion of edges in there embeddings. From Table 2, we can find that:

- SiGAT and SDGNN intentionally omit accuracy/precision/recall metrics, as AUC better captures class discrimination in imbalanced signed networks where positive-negative ratios vary significantly.
- LineDi2vec achieves comparable performance to SiGAT/SDGNN across all metrics. While GNN-based methods typically outperform random-walk approaches like node2vec [12,13], our results demonstrate that line graph enhancement can match their performance. This suggests line graph methods could both replace node concatenation and potentially augment existing GNN architectures.

4.4 Limitations and Draw Backs

The application of forbidden subgraph theory[3] reveals a key limitation: not every graph or network lends itself to transformation into a line graph. This constraint inherently limits the applicability of our model, confining its utility to those types of homogeneous graphs amenable to such conversion.

Moreover, the demands on computational resources and the time necessary for converting a graph into a line graph represent a significant drawback. As highlighted by [10], this conversion process can dramatically increase the complexity of the original graphs with time complexity $O(n)$. This complexity surge subsequently results in an exponential increase in both the processing time and the computational power needed. Our empirical evidence, particularly from handling large-scale datasets like Epinions and Slashdot, illustrates this point.

5 Conclusion

This paper introduces LineDi2Vec, a novel line graph-enhanced graph embedding approach for link sign prediction, designed to overcome the limitations of indirect edge representation in conventional node embedding methods. By explicitly modeling edges through line graphs, our method captures richer relational patterns compared to traditional techniques like node2vec. Experiments on four real-world datasets demonstrate that our approach consistently outperforms baseline methods, highlighting the advantages of line graph-based embeddings for edge-centric prediction tasks. We further discuss the theoretical and practical challenges arising from line graph properties[3], providing insights for future research in graph representation learning. Our work advances the state-of-the-art in signed network analysis while opening new directions for edge-aware graph embeddings.

References

1. Mahboubeh Ahmadalinezhad and Masoud Makrehchi. Edge-centric multi-view network representation for link mining in signed social networks. *Expert Syst. Appl.*, 170:114552, 2021.
2. Sambaran Bandyopadhyay, Anirban Biswas, M Narasimha Murty, and Ramasuri Narayanam. Beyond node embedding: a direct unsupervised edge representation framework for homogeneous networks. *arXiv preprint arXiv:1912.05140*, 2019.
3. Lowell W Beineke. Characterizations of derived graphs. *Journal of Combinatorial theory*, 9(2):129–135, 1970.
4. Joseph Berger, Bernard P. Cohen, and Morris Zelditch. Status characteristics and social interaction. *American Sociological Review*, 37(3):241–255, 1972.
5. 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, 2018.
6. Tim S Evans and Renaud Lambiotte. Line graphs, link partitions, and overlapping communities. *Physical review E*, 80(1):016105, 2009.

7. Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry, 2017.
8. 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, 2016.
9. William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs, 2018.
10. Frank Harary and Robert Z Norman. Some properties of line digraphs. *Rendiconti del circolo matematico di palermo*, 9:161–168, 1960.
11. Fritz Heider. *The psychology of interpersonal relations*. Psychology Press, 2013.
12. Junjie Huang, Huawei Shen, Liang Hou, and Xueqi Cheng. Signed graph attention networks. In *Artificial Neural Networks and Machine Learning–ICANN 2019: Workshop and Special Sessions: 28th International Conference on Artificial Neural Networks, Munich, Germany, September 17–19, 2019, Proceedings 28*, pages 566–577. Springer, 2019.
13. Junjie Huang, Huawei Shen, Liang Hou, and Xueqi Cheng. Sdgnn: Learning node representation for signed directed networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 196–203, 2021.
14. Mohammad Raihanul Islam, B Aditya Prakash, and Naren Ramakrishnan. Signet: Scalable embeddings for signed networks. In *Advances in Knowledge Discovery and Data Mining: 22nd Pacific-Asia Conference, PAKDD 2018, Melbourne, VIC, Australia, June 3-6, 2018, Proceedings, Part II 22*, pages 157–169. Springer, 2018.
15. 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*, pages 509–518, 2018.
16. Srijan Kumar, Bryan Hooi, Disha Makhija, Mohit Kumar, Christos Faloutsos, and VS Subrahmanian. Rev2: Fraudulent user prediction in rating platforms. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 333–341, 2018.
17. Srijan Kumar, Francesca Spezzano, VS Subrahmanian, and Christos Faloutsos. Edge weight prediction in weighted signed networks. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 221–230. IEEE, 2016.
18. Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. Predicting positive and negative links in online social networks, 2010.
19. 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.
20. Tiago Pimentel, Adriano Veloso, and Nivio Ziviani. Fast node embeddings: Learning ego-centric representations. 2018.
21. Matthew Richardson, Rakesh Agrawal, and Pedro Domingos. Trust management for the semantic web. In *International semantic Web conference*, pages 351–368. Springer, 2003.
22. Zdeněk Ryjáček and Petr Vrána. Line graphs of multigraphs and hamilton-connectedness of claw-free graphs. *Journal of Graph Theory*, 66(2):152–173, 2011.
23. Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018.
24. Hassler Whitney. Congruent graphs and the connectivity of graphs. *American Journal of Mathematics*, 54:61–79, 1932.