JUST PROPAGATE: UNIFYING MATRIX FACTORIZATION, NETWORK EMBEDDING, AND LIGHTGCN FOR LINK PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Link prediction is a fundamental task in graph analysis. Despite the success of various graph-based machine learning models for link prediction, there lacks a general understanding of different models. In this paper, we propose a unified framework for link prediction that covers matrix factorization and representative network embedding and graph neural network methods. Our preliminary methodological and empirical analyses further reveal several key design factors based on our unified framework. We believe our results could deepen our understanding and inspire novel designs for link prediction methods.

1 INTRODUCTION

Link prediction, i.e., predicting the existence of links between nodes, is a fundamental task in graph data mining with important applications such as recommendation systems (Ying et al., 2018), protein interaction prediction (Liu et al., 2019), knowledge discovery (Wang et al., 2017), etc. Many machine learning methods have been developed to tackle the link prediction problem (Lü & Zhou, 2011; Hasan & Zaki, 2011), ranging from early heuristic features (Barabási & Albert, 1999; Katz, 1953), statistical machine learning (Tang et al., 2015; Perozzi et al., 2014), to the recent deep learning methods (Hamilton et al., 2017; Wang et al., 2019). For example, matrix factorization (MF) (Koren et al., 2009) is a representative method for link prediction with the assumption that the graph structure has an underlying low-rank structure. Network embedding methods such as DeepWalk (Perozzi et al., 2014) and LINE (Tang et al., 2015) further consider the high-order proximity between nodes. Recently, neural network-based graph learning methods such as NGCF (Wang et al., 2019) and LightGCN (He et al., 2020) have shown superior performance, which has been attributed to the explicit modeling of higher-order neighbors.

Despite the successes of the existing approaches, the aforementioned studies are relatively independent with different assumptions and design objectives, and there lacks a general understanding of the graph-based machine learning models for link prediction. Therefore, as the first contribution of this work, we propose a unified framework for link prediction methods, covering matrix factorization, network embedding methods such as DeepWalk (Perozzi et al., 2014) and LINE (Tang et al., 2015), and graph neural network-based methods including LightGCN (He et al., 2020). Based on the unified framework, we further conduct in-depth methodological and empirical analyses. We make the following observations.

- The overall learning process of the existing methods covered by our unified framework, including the forward inference and backward gradient descent for optimization, is equivalent to a representation propagation procedure.
- Propagation kernels, which determine how the representation is propagated and whether higher-order neighbor information can be effectively utilized, greatly affect the performance of different methods for the link prediction task.
- By regarding randomly drawn negative samples as negative edges, the existing approaches essentially follow the balance theory of signed graphs in aggregating neighborhoods.

We believe our proposed unified framework and preliminary analysis results could deepen our understanding and inspire novel designs for graph-based link prediction methods.

2 PRELIMINARIES

2.1 NOTATIONS

Firstly, we introduce some notations used in the paper. Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ denotes the node set and $\mathcal{E}$ denotes the link set. The adjacency matrix is denoted as $\mathbf{A}$. The goal of link prediction is to discover potential links from the observed links $\mathbf{A}$. We assume the graph is undirected and the nodes have no features. Notice that our problem setting naturally covers the recommendation problem, i.e., $\mathcal{V} = \mathcal{U} \cup \mathcal{I}$, and links only exist between $\mathcal{U}$ and $\mathcal{I}$.

In the common setting of link prediction, there are no explicit negative samples. For example, people seldom explicitly mark whom they do not want to be friends with or products they do not want to buy. Therefore, negative sampling is usually used in link prediction to sample from pairs of nodes that do not have links as pseudo negative samples during the optimization procedure. We denote the adjacency matrix for sampled negative links as $\mathbf{B} \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{V}|}$, where $\mathbf{B}_{i,j} = 1$ represents that node i and j has a negative link. We summarize other notations in Appendix A.

2.2 LOSS FUNCTION

In this paper, we focus on a widely adopted loss function, the binary cross-entropy (BCE) loss (He et al., 2017) and leave exploring other loss functions as future works. The basic BCE loss with a L2 regularization term is formulated as

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{2}\sum\nolimits_{v\in\mathcal{V}}\left(\sum\nolimits_{i\in\mathcal{N}_A(v)}\log\sigma\left(\mathbf{X}_v^\top\mathbf{X}_i\right) + \lambda\sum\nolimits_{j\in\mathcal{N}_B(v)}\log\sigma\left(-\mathbf{X}_v^\top\mathbf{X}_j\right)\right) + \frac{\beta}{2}\|\mathbf{X}\|_2^2, \tag{1}$$

where $\mathcal{N}_A(v)$ and $\mathcal{N}_B(v)$ denote the positive and negative neighbors of node v, $\sigma(\cdot)$ is the Sigmoid activation function, λ and β are hyper-parameters to control the strength of the corresponding terms, and $\mathbf{X}$ denotes the representation of nodes. Eq. (1) can be written in an equivalent matrix form:

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{2}\mathbf{1}_{|\mathcal{V}|}\left(\left[\log\left(\sigma(\mathbf{X}\mathbf{X}^\top)\right)\odot\mathbf{A}\right] + \lambda\left[\log\left(-\sigma(\mathbf{X}\mathbf{X}^\top)\right)\odot\mathbf{B}\right]\right)\mathbf{1}_{|\mathcal{V}|}^\top + \frac{\beta}{2}\|\mathbf{X}\|_2^2, \tag{2}$$

where $\mathbf{1}_{|\mathcal{V}|}$ is a $1\times|\mathcal{V}|$ row matrix with the values being ones and $\odot$ is the Hadamard product.

2.3 A BRIEF INTRODUCTION TO CONSIDERED LINK PREDICTION MODELS

In this section, we briefly introduce link prediction models considered in our framework. We consider four representative methods divided into three categories:

- Matrix factorization, which directly optimizes the representation matrix $\mathbf{X}$ using the loss function.
- DeepWalk and LINE: these network embedding methods also optimize the representation matrix $\mathbf{X}$, but with different loss functions that consider the high-order neighborhoods between nodes.
- LightGCN: a linear GNN model that first propagates the representation as $\bar{\mathbf{X}} = \sum_{k=0}^{K}\frac{1}{K+1}\tilde{\mathbf{A}}^k\mathbf{X}$, $\tilde{\mathbf{A}}$ is a normalized adjacency matrix, e.g., $\tilde{\mathbf{A}} = \mathbf{D}_A^{-\frac{1}{2}}\mathbf{A}\mathbf{D}_A^{-\frac{1}{2}}$, $\mathbf{D}_A$ is the diagonal degree matrix, and then optimize $\mathbf{X}$ using the loss function.

Following the standard training paradigm in machine learning, we assume all these methods are optimized by the full-batch gradient descent and denote the node representations in the mth optimization step as $\mathbf{X}^{(m)}$, i.e., $\mathbf{X}^{(m+1)} = \mathbf{X}^{(m)} - \alpha\nabla_{\mathbf{X}^{(m)}}\mathcal{L}$.

3 OUR PROPOSED FRAMEWORK

3.1 THE UNIFIED FRAMEWORK

Although the aforementioned methods appear to have unrelated design objectives and loss functions based on diverse motivations, we find that they can be unified into a general framework that only

performs forward propagation, i.e., even the back propagation for gradient descends is equivalent to the forward propagation. Specifically, we have:

$$\begin{aligned}\mathbf{X}^{(m+1)} &= \mathbf{H}^{(m+1)}\mathbf{X}^{(m)} \\ &= \left(c_1\mathbf{I} + c_2\mathcal{P}_{a_1,b_1}(\tilde{\mathbf{A}})\left[\mathcal{K}_+^{(m+1)}(\mathbf{A}) - \lambda\mathcal{K}_-^{(m+1)}(\mathbf{B})\right]\mathcal{P}_{a_1,b_1}(\tilde{\mathbf{A}})\right)\mathbf{X}^{(m)},\end{aligned} \tag{3}$$

where c_1, c_2 are constants that depend on the learning rate α and regularization term β, $a_1, b_1 \in \mathbb{N}$ are model-specific constants, and $\mathbf{I}$ is the $|V| \times |V|$ identity matrix. The detailed proof and the correspondence of different methods to our framework are provided in Appendix B and Appendix C, respectively. Though Eq. (3) appears to be complicated, we elaborate the terms one by one.

Firstly, $\mathbf{H}^{(m+1)} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ is the propagation kernel that determines how the representation is propagated. It is composed of three major terms: a high-order proximity matrix $\mathcal{P}_{a_1,b_1}(\tilde{\mathbf{A}})$, a positive link kernel $\mathcal{K}_+^{(m+1)}(\mathbf{A})$, and a negative link kernel $\mathcal{K}_-^{(m+1)}(\mathbf{B})$. The high-order matrix is defined as:

$$\mathcal{P}_{a,b}(\tilde{\mathbf{A}}) = \sum\nolimits_{r=a}^{b} \frac{1}{b-a+1}\tilde{\mathbf{A}}^r, \tag{4}$$

i.e., it is a weighted sum of the ath order to the bth order normalized adjacency matrix $\tilde{\mathbf{A}}$, $a \leq b$. The positive and negative link kernels are defined as follows:

$$\begin{aligned}\mathcal{K}_+^{(m+1)}(\mathbf{A}) &= \mathbf{S}_A^{(m+1)} \odot (c_3\mathcal{P}_{a_2,b_2}(\tilde{\mathbf{A}}) + (1-c_3)\mathbf{A}) \\ \mathcal{K}_-^{(m+1)}(\mathbf{B}) &= \mathbf{S}_B^{(m+1)} \odot (c_3\tilde{\mathbf{B}} + (1-c_3)\mathbf{B}),\end{aligned} \tag{5}$$

where $c_3 \in \{0, 1\}$, $\tilde{\mathbf{B}}$ is the normalized adjacency matrix of negative links, e.g., $\tilde{\mathbf{B}} = \mathbf{D}_B^{-\frac{1}{2}}\mathbf{B}\mathbf{D}_B^{-\frac{1}{2}}$, and $\mathbf{S}_A^{(m+1)}, \mathbf{S}_B^{(m+1)}$ are score matrices that measure the difference between the similarity calculated by the representation vectors and the target link as follows

$$\mathbf{S}_A^{(m+1)} = 1 - \sigma\left(\bar{\mathbf{X}}^{(m)}\bar{\mathbf{X}}^{(m)\top}\right), \mathbf{S}_B^{(m+1)} = \sigma\left(\bar{\mathbf{X}}^{(m)}\bar{\mathbf{X}}^{(m)\top}\right), \tag{6}$$

where $\bar{\mathbf{X}}^{(m)} = \mathcal{P}_{a_1,b_1}(\tilde{\mathbf{A}})\mathbf{X}^{(m)} = \sum_{r=a_1}^{b_1} \frac{1}{b_1-a_1+1}\tilde{\mathbf{A}}^r\mathbf{X}^{(m)}$ is the propagated node representation. From Eq. (6), score matrices equal 0 when the representation perfectly reconstructs the training data.

In summary, all these four representative methods can be unified in Eq. (3), which indicates that their underlying mechanism shares strong connections. In the next section, we conduct in-depth analyses based on this framework.

3.2 Revisiting the Existing Link Prediction Methods

In this section, we revisit the existing methods based on our proposed framework.

Firstly, it is easy to see that all the four covered methods have analytical solutions only using the forward propagation, i.e., doing back propagation for gradient descends ends up being equivalent to changing the forward propagation kernel. Therefore, our framework sheds lights on novel design principles for link prediction that do no involve computationally expensive back propagations.

In view of signed graphs that sampled negative links correspond to real negative connections, the propagation step in Eq. (3) satisfies the balance theory, i.e., a sociological assumption that "the enemy of my enemy is my friend". Though this assumption is never explicitly stated for any of the covered methods, it is always behind the curtain and naturally surfaces from our unified framework, which should draw further research attention.

Different methods have diverse strategies in using the high-order information from neighborhoods. To be more specific, from Eq. (3), it is easy to see that both a_1, b_1 in the high-order proximity matrix $\mathcal{P}_{a_1,b_1}(\tilde{\mathbf{A}})$ and a_2, b_2 in the positive link kernel $\mathcal{K}_+^{(m+1)}(\mathbf{A})$ affects the high-order proximity (please see Appendix C for the exact values of different methods). For MF and LINE, they can aggregate one extra order proximity during each optimization step. Therefore, MF and LINE can learn from high-order neighborhoods by accumulating steps during optimization. Such a result is in sharp contrast with the current belief that MF and LINE can only preserve the first-order proximity between nodes. In addition, DeepWalk and LightGCN can aggregate multiple orders of neighborhoods during each

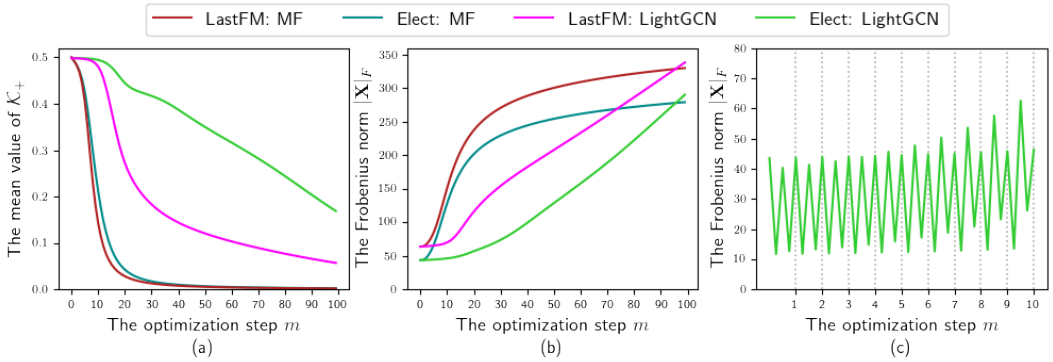

Figure 1: The analysis results of MF and LightGCN. (a) The mean value of $\mathcal{K}_+$ as the optimization step increases; (b) The Frobenius norm of $\mathbf{X}$ as the optimization step increases; (c) The Frobenius norm $\|\mathbf{X}\|_F$ for LightGCN as the optimization substeps increase.

optimization step through two separate mechanisms: DeepWalk constructs a high-order proximity in the objective function using random walks, while LightGCN contains an internal propagation step. In short, how different methods utilize high-order proximities is vital to the model design.

To empirically analyze different methods based on our framework, we conduct experiments on two recommendation datasets: Elect and LastFM. We mainly compare the results of MF and LightGCN with some analyses. More details about the experimental settings can be found in Appendix D.

The results are shown in Appendix E. Not surprisingly, LightGCN achieves better results than MF, which is consistent with the literature. Based on the results, we try to investigate a more fundamental question: "What factors drive the success of link prediction models?" Using our framework, we preliminary examine two targets: the positive link kernel $\mathcal{K}_+$ and the representation matrix $\mathbf{X}$.

For the positive link kernel, recall that it measures how well the node representation can reconstruct the positive links. Therefore, its average value is a good indicator of the model learning process. To analyze its influence, we plot the mean value of $\mathcal{K}_+$ as the optimization progresses in Figure 1(a). The figure clearly shows that the mean values for MF drop much faster than LightGCN, indicating that MF quickly fits the training positive data. However, the final results of LightGCN are much better (both methods have already adopted early stopping to prevent overfitting). Therefore, the empirical results show that we need to properly control the fitting curve to realize effective link prediction models.

Next, we analyze the representation matrix $\mathbf{X}$. Specifically, since inner product is used to measure the similarity between node representations, both the length and direction of representation vectors affect the similarity. Here, we focus on the average length of representation vectors and plot the Frobenius norm of $\mathbf{X}$ vs. the optimization steps in Figure 1(b). The results indicate that the representation of LightGCN has smoother lengths than MF. To further analyze the underlying reason, we decompose one optimization step of LightGCN, i.e., one step of Eq. (3), into four substeps (please check Appendix C for details): (1) Propagate K layers using $\tilde{\mathbf{A}}$; (2) Propagate one layer using $\mathcal{K}_+^{(m+1)}(\mathbf{A})$; (3) Propagate another K layers using $\tilde{\mathbf{A}}$; (4) Update the representations.

We further plot the Frobenius norm of LightGCN with respect to these four substeps in Figures 1(c) and (d). The results show that substeps (1) and (3) can reduce the length while steps (2) and (34 increase the length. The reason is that the propagations in (1) and (3) use the normalized adjacency matrix $\tilde{\mathbf{A}}$, which can control the length of the representation. The results show that properly choosing the normalization of adjacency matrix is crucial for the success of link prediction methods.

4 Conclusion

In this paper, we introduce a unified framework for four representative link prediction methods including MF, DeepWalk, LINE, and LightGCN. We reveal several key design principles of these methods based on our proposed framework. We believe our proposed framework and in-depth analyses can open up new research opportunities for developing link prediction methods.

REFERENCES

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286 (5439):509–512, 1999.

William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 1025–1035, 2017.

Mohammad Al Hasan and Mohammed J Zaki. A survey of link prediction in social networks. In *Social network data analytics*, pp. 243–275. Springer, 2011.

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pp. 173–182, 2017.

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pp. 639–648, 2020.

Leo Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

Leilei Liu, Yi Ma, Xianglei Zhu, Yaodong Yang, Xiaotian Hao, Li Wang, and Jiajie Peng. Integrating sequence and network information to enhance protein-protein interaction prediction using graph convolutional networks. In *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 1762–1768. IEEE, 2019.

Linyuan Lü and Tao Zhou. Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications*, 390(6):1150–1170, 2011.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 701–710, 2014.

Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *Proceedings of the eleventh ACM international conference on web search and data mining*, pp. 459–467, 2018.

Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pp. 1067–1077, 2015.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12): 2724–2743, 2017.

Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*, pp. 165–174, 2019.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 974–983, 2018.

Wenhui Yu and Zheng Qin. Sampler design for implicit feedback data by noisy-label robust learning. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 861–870, 2020.

A MAIN NOTATIONS

Table 1: Notations used in the paper

Notations	Descriptions
$\mathbf{A} \in \{0,1\}^{\lvert\mathcal{V}\rvert\times\lvert\mathcal{V}\rvert}$	The adjacency matrix of positive links
$\mathbf{B} \in \{0,1\}^{\lvert\mathcal{V}\rvert\times\lvert\mathcal{V}\rvert}$	The adjacency matrix of negative links by negative sampling
$\mathbf{D}_A/\mathbf{D}_B$	The diagonal degree matrix of $\mathbf{A}/\mathbf{B}$
$\mathcal{N}_A(u)$	The positive neighborhood of node u
$\mathcal{N}_B(u)$	The sampled negative neighborhood of node u
$\mathbf{X}^{(m)}$	The node representation matrix at the mth epoch
α	The learning rate
β	The strength for the L_2 regularization
$\mathbf{H}^{(m)}$	The propagation kernel at the mth epoch
$\tilde{\mathbf{A}}(\tilde{\mathbf{B}})$	The normalized adjacency matrix of positive and negative links
$\mathbf{S}_A^{(m)}/\mathbf{S}_B^{(m)}$	The scoring matrix for positive and negative links at the mth epoch
$\sigma(\cdot)$	The Sigmoid activation function
$\odot$	The Hadamard product

B PROOF

In this section, we first give the modified loss function of the corresponding methods in the matrix format. Then, we take LightGCN as an example to show the derivation of our unified framework, while other methods can be proved in similar ways.

B.1 LOSS

For matrix factorization, a commonly used loss function is the original BCE loss, i.e., $\mathcal{L}_{\text{MF}} = \mathcal{L}_{\text{BCE}}$. For LINE, its loss function can be written as follows

$$\mathcal{L}_{\text{LINE}} = -\frac{1}{2}\mathbf{1}_{|\mathcal{V}|}\left(\left[\log\left(\sigma(\mathbf{X}\mathbf{X}^\top)\right)\odot\tilde{\mathbf{A}}\right] + \lambda\left[\log\left(-\sigma(\mathbf{X}\mathbf{X}^\top)\right)\odot\mathbf{B}\right]\right)\mathbf{1}_{|\mathcal{V}|}^\top + \frac{\beta}{2}\|\mathbf{X}\|_2^2, \tag{7}$$

where $\tilde{\mathbf{A}} = \mathbf{D}_A^{-1}\mathbf{A}$ and $\mathbf{B}$ is the expectation of negative sampled links as follows

$$\mathbf{B} \propto \mathbf{D}_A^{\frac{3}{4}}. \tag{8}$$

For DeepWalk, based on the analysis by Qiu et al. (2018), when the walk length $L \to +\infty$, the loss function of DeepWalk in the matrix form with the window size $w \in \mathbb{N}^+$ can be written as

$$\mathcal{L}_{\text{DW}} = -\frac{1}{2}\mathbf{1}_{|\mathcal{V}|}\left(\left[\log\left(\sigma(\mathbf{X}\mathbf{X}^\top)\right)\odot\mathcal{P}_{1,w}(\tilde{\mathbf{A}})\right] + \lambda\left[\log\left(-\sigma(\mathbf{X}\mathbf{X}^\top)\right)\odot\tilde{\mathbf{B}}\right]\right)\mathbf{1}_{|\mathcal{V}|}^\top + \frac{\beta}{2}\|\mathbf{X}\|_2^2, \tag{9}$$

where $\tilde{\mathbf{A}} = \mathbf{D}_A^{-1}\mathbf{A}$ and $\tilde{\mathbf{B}} = \mathbf{D}_B^{-1}\mathbf{B}$.

As for LightGCN, the loss function can be written in a matrix form as

$$\mathcal{L}_{\text{LGC}} = -\frac{1}{2}\mathbf{1}_{|\mathcal{V}|}\left(\left[\log\left(\sigma(\bar{\mathbf{X}}\bar{\mathbf{X}}^\top)\right)\odot\mathbf{A}\right] + \lambda\left[\log\left(-\sigma(\bar{\mathbf{X}}\bar{\mathbf{X}}^\top)\right)\odot\mathbf{B}\right]\right)\mathbf{1}_{|\mathcal{V}|}^\top + \frac{\beta}{2}\|\mathbf{X}\|_2^2 \tag{10}$$

where $\bar{\mathbf{X}} = \sum_{k=0}^{K}\frac{1}{K+1}\tilde{\mathbf{A}}^k\mathbf{X}$, $\tilde{\mathbf{A}} = \mathbf{D}_A^{-\frac{1}{2}}\mathbf{A}\mathbf{D}_A^{-\frac{1}{2}}$.

B.2 LightGCN Derivation

In this subsection, we take LightGCN as an example to show how to derive our unified framework. The derivative of Eq. (10) is calculated as

$$\begin{aligned}\frac{\partial \mathcal{L}_{\text{LGC}}}{\partial \mathbf{X}^{(m)}} &= \left(\beta\mathbf{I} - \sum_{k=0}^{K} \frac{\tilde{\mathbf{A}}^k}{K+1}\left[\left(1-\sigma(\bar{\mathbf{X}}^{(m)}\bar{\mathbf{X}}^{(m)\top})\right)\odot\mathbf{A} - \lambda\sigma(\bar{\mathbf{X}}^{(m)}\bar{\mathbf{X}}^{(m)\top})\odot\mathbf{B}\right]\sum_{k'=0}^{K}\frac{\tilde{\mathbf{A}}^{k'}}{K+1}\right)\mathbf{X}^{(m)} \\ &= \left(\beta\mathbf{I} - \mathcal{P}_{1,K}(\tilde{\mathbf{A}})\left[\mathbf{S}_A^{(m+1)}\odot\mathbf{A} - \lambda\mathbf{S}_B^{(m+1)}\odot\mathbf{B}\right]\mathcal{P}_{1,K}(\tilde{\mathbf{A}})\right)\mathbf{X}^{(m)}.\end{aligned} \tag{11}$$

Then, we have

$$\begin{aligned}\mathbf{X}^{(m+1)} &= \mathbf{X}^{(m)} - \alpha\frac{\partial \mathcal{L}_{\text{LGC}}}{\partial \mathbf{X}^{(m)}} \\ &= \left((1-\alpha\beta)\mathbf{I} + \alpha\mathcal{P}_{1,K}(\tilde{\mathbf{A}})\left[\mathbf{S}_A^{(m+1)}\odot\mathbf{A} - \lambda\mathbf{S}_B^{(m+1)}\odot\mathbf{B}\right]\mathcal{P}_{1,K}(\tilde{\mathbf{A}})\right)\mathbf{X}^{(m)},\end{aligned} \tag{12}$$

which is a special case of Eq. (3) by setting $c_1 = 1-\alpha\beta$, $c_2 = \alpha$, $a_1 = a_2 = b_2 = 0$, $b_1 = K$ and $\tilde{\mathbf{A}} = \mathbf{D}_A^{-\frac{1}{2}}\mathbf{A}\mathbf{D}_A^{-\frac{1}{2}}$.

C Detailed Correspondence of Link Prediction Models

We summarize the detailed correspondence of different models in our unified framework in Table 2.

Table 2: The detailed correspondence of different models in our unified framework.

Model	c_1	c_2	c_3	a_1	b_1	a_2	b_2	$\tilde{\mathbf{A}}$	$\tilde{\mathbf{B}}$
MF	$1-\alpha\beta$	α	0	0	0	0	0	—	—
LINE	$1-\alpha\beta$	α	1	0	0	1	1	$\mathbf{D}_A^{-1}\mathbf{A}$	$\propto \mathbf{D}_A^{\frac{3}{4}}\mathbf{B}$
DeepWalk	$1-\alpha\beta$	α	1	0	0	1	w	$\mathbf{D}_A^{-1}\mathbf{A}$	$\mathbf{D}_B^{-1}\mathbf{A}$
LightGCN	$1-\alpha\beta$	α	0	0	K	0	0	$\mathbf{D}_A^{-\frac{1}{2}}\mathbf{A}\mathbf{D}_A^{-\frac{1}{2}}$	—

D Experimental Settings

D.1 Datasets

We adopt the following two public datasets in our experiments:

- Elect is a dataset from the Amazon-review collection about electronics, which is widely used for item recommendation tasks. We use the dataset processed by Yu & Qin (2020).
- LastFM is collected from the music community site. It records the download relationships between users and artists. We use the dataset processed by He et al. (2020).

The statistics of the datasets are summarized in Table 3

Table 3: The statistics of datasets

Dataset	# Nodes	# Links	Density
Elect	2,957	35,931	1.645%
LastFM	6,381	52,668	0.620%

D.2 Evaluation Metrics

Following previous works (Wang et al., 2019; He et al., 2020), we treat all non-interacted products as candidate links and output the link prediction scores and perform the top-K recommendation task, where K is set as 20. As for the evaluation metrics, we choose Precision@K, Recall@K, and NDCG@K. For all the metrics, higher values indicate better performance. We repeat all experiments five times with different random seeds and report the average results.

D.3 Hyper-parameters

We tune the learning rate α in $\{1e-5, 1e-4, 1e-3, 1e-2, 1e-1\}$ and the propagation layers for LightGCN in $\{1, 3, 5\}$. For each observed user-item interaction, we randomly sample one negative pair from non-interacted items before training. Besides, we set $\lambda = 1$ for all models. For all methods, we adopt an early stopping strategy with a patience of 10, i.e., stop the optimization if the validation accuracy does not improve for 10 steps.

E Experimental Results

Table 4: The result of link prediction for different methods.

Dataset	Metric	MF	LightGCN
Electronics	Precision@20	0.0079	0.0157
	Recall@20	0.0547	0.1125
	NDCG@20	0.0321	0.0652
LastFM	Precision@20	0.0302	0.0557
	Recall@20	0.1112	0.1981
	NDCG@20	0.0767	0.1468