Enhancing Recommender Systems Using Non-bipartite Graphs

MAYA BARUCH*, TAMAR GARBUZ*, AMNON LEVI*, and MAYA BECHLER-SPEICHER

*Equal contribution.

We propose enhancing recommender systems by augmenting the traditional bipartite user-item graph with user-to-user and item-to-item edges, forming a non-bipartite structure. This addition allows the model to explicitly leverage interior interactions in users and items, potentially improving the accuracy and relevance of the recommendation. We extend previous work done on user-item and item-item interactions by adding user-user interactions, and setting a new SoTA for the tested dataset. The source code for the experimental segment can be found in this repository.

1 INTRODUCTION

Recommender systems play a critical role in helping users navigate vast amounts of information in applications such as e-Commerce, streaming online, and social media platforms. Traditional approaches often rely on collaborative filtering, where a user-item interaction matrix is used to predict missing entries. More recently, graph neural networks (GNNs) have gained attention for modeling the bipartite relationships between users and items as graphs, where nodes represent users and items, and edges represent interactions.

However, these GNN-based recommender systems have limitations. Bipartite graph models focus solely on direct user-item interactions, often relegating important user-user and item-item relationships to implicit learning within node embeddings. For example, two users who frequently interact with the same items may have shared preferences that are not directly captured. Similarly, items that are often co-interacted by users may have inherent similarities, such as being part of the same genre or category.

To address these challenges, we propose augmenting the traditional bipartite user-item graph with user-user and item-item edges, forming a non-bipartite structure. By explicitly modeling these relationships, we enable the system to leverage collaborative signals that are otherwise overlooked. For instance, user-user edges derived from co-occurrence or social connections provide an additional layer of collaborative filtering, while item-item edges reveal shared attributes or contexts. These enhancements allow for a richer representation of relationships, potentially leading to improved recommendation accuracy and relevance.

Our approach is based on UltraGCN [Mao et al. 2021], which simplifies GNN-based recommendation by eliminating explicit message-passing layers and approximating infinite-layer convolutions through a constraint loss. Although UltraGCN achieves remarkable efficiency and scalability, it neglects the explicit incorporation of user-user relationships, which limits its ability to capture higher-order collaborative signals.

In this work, we introduce our novel non-bipartite graph-based recommendation framework and demonstrate its advantages through experiments on standard datasets. Our method explicitly incorporates non-bipartite edges into the graph structure, enabling more effective modeling of higher-order interactions. By bridging the gap between implicit and explicit modeling of diverse relationships, our work offers a scalable and effective framework for enhancing recommender systems.

Authors' address: Maya Baruch*, mayabaruch@mail.tau.ac.il; Tamar Garbuz*, garbuz@mail.tau.ac.il; Amnon Levi*, amnonlevi@mail.tau.ac.il; Maya Bechler-Speicher, mayab4@mail.tau.ac.il.

2 RELATED WORK

Recommender systems are the key to many services and applications. In such systems, users are able to interact with items and possibly assign a value to these interactions. Classic examples include watching shows and movies on a streaming service or purchasing products on an online retail service. There are a few possible goals for a recommender system: predicting interaction between a user and an item, recommending new items that a user would have positive interaction with, promoting an item to a set of users, etc. The most common recommender system problem is that of collaborative filtering [Ju et al. 2024].

The classic model of the problem is that of a matrix of user-item interactions, with the goal of predicting an empty matrix cell corresponding to a user-item pair. One famous proposed solution to this model is using matrix factorization [Koren et al. 2009].

With the rise of graph neural networks (GNNs), a useful model for collaborative filtering is the bipartite graph, using user nodes, item nodes, and interaction edges [Wang et al. 2019; Ying et al. 2018]. In such a model, the problem becomes predicting an edge instead of a matrix cell. When considering a user-item graph, GNN solutions typically model interactions using a bipartite graph [He et al. 2020; Ju et al. 2024; Wang et al. 2023].

However, this type of model ignores the topological property of user-to-user and item-to-item interactions, relegating this information to only be learned implicitly via the node embeddings.

UltraGCN [Mao et al. 2021] presents a highly efficient graph convolutional network model for recommendation by eliminating explicit message passing layers, instead of approximating infinite-layer convolutions through a constraint loss. This simplification significantly reduces training time without compromising on accuracy. Despite its successes, UltraGCN does not incorporate user-user relationships, which are often crucial for capturing shared user interests. UltraGCN's authors noted that user-user relationships derived from co-occurrence graphs add little benefit, possibly due to the broad nature of users' interests compared to item attributes. They suggest using a social network graph for better user-user modeling, leaving it as a future direction.

3 METHOD

Due to its high efficiency and leading results, we follow in the footsteps of [Mao et al. 2021] in constructing the user-item binary cross-entropy losses for constraint and optimization:

$$\mathcal{L}_C = -\sum_{(u,i)\in N^+} \beta_{u,i} log(\sigma(e_u^\top e_i)) - \sum_{(u,j)\in N^-} \beta_{u,j} log(\sigma(e_u^\top e_j)), \quad \beta_{u,i} = \sqrt{\frac{d_u + 1}{d_u^2(d_i + 1)}}$$

$$\mathcal{L}_O = -\sum_{(u,i)\in N^+} log(\sigma(e_u^\top e_i)) - \sum_{(u,j)\in N^-} log(\sigma(e_u^\top e_j))$$

Where N^+, N^- are positive and negative samples of edges in the user-item graph, and e_u, e_i are the model embeddings for the user u and item i respectively.

Due to their similarity, we combine $\mathcal{L}_C, \mathcal{L}_O$ into a single cross entropy loss:

$$\mathcal{L}_{CE} = \sum_{(u,i)\in N^+} (w_1 + w_2\beta_{u,i}) log(\sigma(e_u^\top e_i)) - \sum_{(u,j)\in N^-} (w_3 + w_4\beta_{u,j}) log(\sigma(e_u^\top e_j))$$

Where w_1, w_2, w_3, w_4 are tuneable hyper-parameters.

We also utilize a weight normalization loss:

$$\mathcal{L}_W = \sum_{w \in M} |w|_2^2$$

We propose a user-item graph enriched with user-user and item-item edges, transforming it into a non-bipartite graph. These edges can be predefined or learned. Direct connections could represent social ties between users or common properties among items. This structure allows the model to learn from interactions typically left implicit.

The UlgraGCN architecture models item-item relationships explicitly, in addition to user-item relationships. Our work aims to extend UltraGCN by explicitly modeling user-user relationships using methods such as:

3.1 Matrix Factorization

By factorizing the user-item interaction matrix, we represent each user with a latent vector, enabling us to capture user-user similarity through vector comparison. The matrix factorization objective is the square error loss between the user embeddings and the positive embeddings. Denote the user embeddings by E^u, and the positive embeddings by E^p. Then the loss is

$$\mathcal{L}_{MF} = \sum_{i=1}^{|I|} \sum_{j=1}^{|U|} (E^u{}_{i,j} - E^p{}_{i,j})^2,$$

where $|U|$, $|I|$ are the numbers of users and items respectively. This loss is incorporated into the overall objective with a weighting coefficient λ_{mf}, which is tuned using grid search.

3.2 Co-occurrence-based Similarity

We compute the overlap between user interaction sets, providing a similarity score between users based on their co-interacted items.

We calculate the weighted co-occurrence user-user matrix by first calculating the weighted adjacent matrix $G^U \in \mathbb{R}^{|U| \times |U|}$:

$$G^U = AA^\top$$

In a similar manner to [Mao et al. 2021] we can then obtain an approximation of the infinite-layer convolution on the co-occurrence graph for a user-user similarity weight $\omega^U_{u,v}$:

$$\omega^U_{u,v} = \frac{G^U_{u,v}}{g_u - G^U_{u,u}} \sqrt{\frac{g_u}{g_v}}, \quad g_w = \sum_k G^U_{w,k}$$

We can than obtain $\mathcal{L}_U$, a complementary loss to Mao et al. [2021]'s $\mathcal{L}_I$, which employed a similar build for item-item co-occurence:

$$\mathcal{L}_U = - \sum_{(u,i) \in N^+} \sum_{v \in S(u)} \omega^U_{u,v} log(\sigma(e_v^\top e_i))$$

Where $S(u)$ is a sample of size K^U, typically set to 10 in our experiments. This loss constraints the model to better represent co-occurring users with similar embeddings.

Using $G^I = A^\top A \in \mathbb{R}^{|I| \times |I|}$ yields a similar term for item-item interactions:

$$\mathcal{L}_I = - \sum_{(u,i) \in N^+} \sum_{j \in S(i)} \omega^I_{i,j} log(\sigma(e_u^\top e_j))$$

We combine the two into the co-occurrence similarity loss $\mathcal{L}_{CS}$:

$$\mathcal{L}_{CS}(\lambda_I, \lambda_U) = \lambda_I \mathcal{L}_I + \lambda_U \mathcal{L}_U$$

We combine all previous losses into a unified loss term $\mathcal{L}$:

$$\mathcal{L} = \mathcal{L}_{CE}(w_1, w_2, w_3, w_4) + \gamma \mathcal{L}_W + \mathcal{L}_{CS}(\lambda_I, \lambda_U) + \lambda_{MF} \mathcal{L}_{MF}$$

4 EXPERIMENTS AND RESULTS

Datasets. Due to computational constraints, we chose to focus our experiments on the Amazon-Book dataset. We were motivated to choose it over the other popular datasets (Gowalla, Yelp2018) for two reasons: The first reason is simply that it is the largest by a big margin. The second reason is that as discussed, we used UltraGCN as our baseline. Amazon-Book is the dataset for which UltraGCN obtained the largest improvement, so further improving it could prove the most meaningful.

Evaluation Protocol. We follow previous works in using the standard train-test splits provided in the dataset in order to allow a one-to-one comparison between model performances. To avoid misattributing any improvement, when comparing to the base model, UltraGCN, all reported results use the exact same hyper-parameters with the exception of the new loss's coefficient. To further allow for a fair comparison, we report Recall@20 and NDCG@20 as they are the metrics contested in the prior works.

Evaluated Models. We compare our approach with existing models.

Code. We provide our code to allow the reproduction of the results in github.com/amnonlevitau/RecZoo.

Resources. Our experiments utilize a Slurm-based university cluster and a personal GPU with 24GB memory. We train and evaluate our model against state-of-the-art baselines. Due to the minimalistic nature of the added terms, the computational complexity is nearly identical to that of UltraGCN, and thus an efficiency analysis is omitted since it would only serve to reinstate the results seen in [Mao et al. 2021].

4.1 Results

We appended the UltraGCN architecture by adding 2 new losses that constrain the model to better represent the user-user relationships: co-occurrence and matrix factorization.

As co-occurrence was already utilized by Mao et al. [2021] in the item-item relationship constraint, we implemented it for user-user relationships as a default, and indeed as can be seen in Table 1, it deteriorated the results. Due to computational and time constraints, only a small number of λ_U values were tested. The reported result uses $\lambda_U = \lambda_I$ for symmetry.

When setting $\lambda_U = 0$, we evaluated different values for λ_{MF} and found that $\lambda_{MF} = 35$ yields optimal results when freezing all UltraGCN hyper-parameters in their original reported optimal values for both Recall (Fig. 1) and NDCG (Fig. 2). Using the discovered optimal value, we evaluated the model performance with the new loss term, resulting in a statistically significant improvement over the previous state-of-the-art result for Recall@20, as well as a potential

Table 1. Performance comparison on Amazon-Books

Model	Recall@20	NDCG@20
NGCF	0.0344	0.0263
NIA-GCN	0.0369	0.0287
LR-GCCF	0.0335	0.0265
LightGCN	0.0411	0.0315
DGCF	0.0422	0.0324
UltraGCN	0.0681	0.0556
Ours (CS)	0.0623	0.0502
Ours (MF)	**0.0683**	0.0564
Improvement	0.26%	1.47%
p-value	0.036	0.18

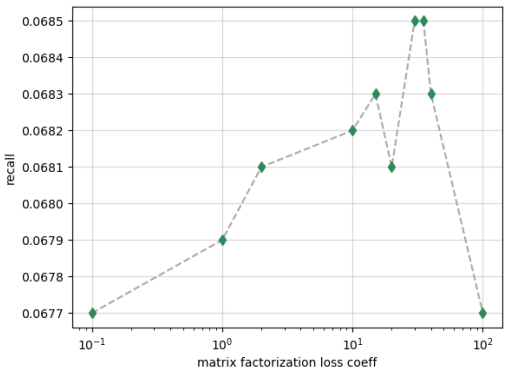

Fig. 1. Recall@20 values for different $\mathcal{L}_{MF}$ values

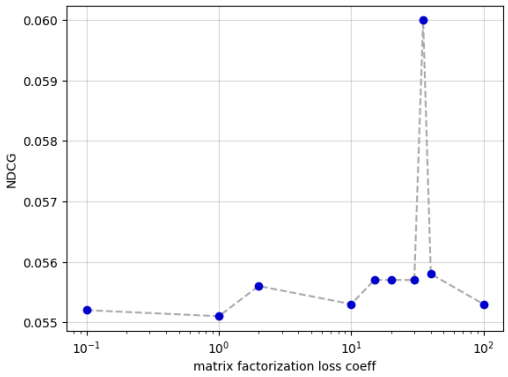

Fig. 2. NDCG@20 values for different $\mathcal{L}_{MF}$ values

improvement in NDCG (not reported an improvement due to an abnormal deviation resulting in a low p-value). See Table 1.

By using the exact dataset, evaluation protocol, and evaluation code, we were able to import the performance results from [Mao et al. 2021]. Despite that, we reevaluated the UltraGCN performance on our own and were able to reproduce their results up to a margin of error.

5 FUTURE WORK

Future work may explore dynamic edge addition based on real-time user interactions (enabling adaptive graphs for evolving systems) or utilize more complex node features for edge determination (leveraging external data sources could refine edge determination and enhance recommendation quality). Additionally, fine-tuning edge thresholds may improve model performance.

We also propose two other methods that can be explored for improving user-user relations:

Personalized PageRank: A bipartite user-item graph allows us to perform random walks from a user node, propagating through item nodes and reaching other users. The resulting distribution reflects the strength of user-user connections.

Projection of Bipartite Graph: Create user-user connections by linking users who have interacted with the same items. The edge weight reflects the number of shared items, providing an implicit collaborative filtering signal.

6 CONCLUSION

Unsatisfied with the negligence of user-user interactions in previous GNN-based recommender systems, we proposed two methods that incorporate them into a model. The first method employs a technique previously seen in item-item interaction modeling, while the second was not previously used in a non-bipartite setting. We observed that the new method offered a Pareto improvement of an existing state-of-the-art model, achieving a new SoTA.

REFERENCES

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. 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*. 639–648.

Wei Ju, Zheng Fang, Yiyang Gu, Zequn Liu, Qingqing Long, Ziyue Qiao, Yifang Qin, Jianhao Shen, Fang Sun, Zhiping Xiao, et al. 2024. A comprehensive survey on deep graph representation learning. *Neural Networks* (2024), 106207.

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

Kelong Mao, Jieming Zhu, Xi Xiao, Biao Lu, Zhaowei Wang, and Xiuqiang He. 2021. UltraGCN: ultra simplification of graph convolutional networks for recommendation. In *Proceedings of the 30th ACM international conference on information & knowledge management*. 1253–1262.

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

Yu Wang, Yuying Zhao, Yi Zhang, and Tyler Derr. 2023. Collaboration-aware graph convolutional network for recommender systems. In *Proceedings of the ACM Web Conference 2023*. 91–101.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 974–983.