Preprints are preliminary reports that have not undergone peer review. They should not be considered conclusive, used to inform clinical practice, or referenced by the media as validated information.

Graph Neural Networks with Rule-based Embeddings for Enhanced Recommender System

Samuel Williams (samuelwilliamswith9@hotmail.com)
Wichita State University
Benjamin Smith
Wichita State University
Daniel Carter
Wichita State University

Research Article

Keywords: Recommender, Graph, GNN, Feature Integration, Rule.

Posted Date: January 29th, 2024

DOI: https://doi.org/10.21203/rs.3.rs-3898939/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Graph Neural Networks with Rule-based Embeddings for Enhanced Recommender System

Samuel Williams[1]*, Benjamin Smith[1], Daniel Carter[1],
[1] Wichita State University; (e-mail: samuelwilliamswith9@hotmail.com, benjaminsmithcosw@gmail.com, danielcarterwichita@gmail.com)
*Correspondence: (e-mail: samuelwilliamswith9@hotmail.com)

Abstract—We are now entering the era of big data, transitioning from a previous era of information scarcity to one where we face significant information overload. In this context, the challenge of filtering out valuable information for individuals from the vast sea of internet data has become increasingly important. A graph is a crucial information organization structure consisting of nodes and edges. Content-based recommendation systems require extensive background knowledge about items and users. This knowledge can often be organized in the form of a knowledge graph. With the development of the internet, a considerable amount of knowledge has been structured into graphs, such as Wikipedia or knowledge trees and classification tree structures like e-commerce directories. This paper conducts research on recommender systems based on graph neural network (GNN) technology. Utilizing graph convolutional neural networks, a graph information extractor is constructed to merge information from the user-item bipartite graph's node neighbors and generate dense vector representations for the nodes. The model employs a multi-task learning approach by introducing the reconstruction tasks of auxiliary information graphs for users and items into the traditional collaborative filtering recommendation task through graph neural networks.

Index Terms—Recommender, Graph, GNN, Feature Integration, Rule.

1 Introduction

With the rapid development of the internet in recent years, there has been an explosive growth in network information. We are now entering the era of big data, transitioning from a previous era of information scarcity to one where we face significant information overload. In this context, the challenge of filtering out valuable information for individuals from the vast sea of internet data has become increasingly important. How to sift through massive amounts of network data to extract content that people need is a major challenge, leading to the emergence of Recommender Systems [1]. A recommender system is an algorithm designed to predict whether a user is interested in a particular item, essentially functioning as an information filtering system. For example, in video-sharing platforms like YouTube, the primary task of the recommender system is to suggest videos that users are likely to enjoy [2]. In the field of online advertising, the recommender system's primary objective is to identify advertisements that align with the user's interests, precisely delivering relevant ads to specific users. This has become a crucial revenue source for numerous commercial companies.

The advantages of collaborative filtering algorithms are quite evident; they don't require detailed information about users and items but instead rely on interaction records between users and items to complete the entire recommendation process. Therefore, this algorithm doesn't need to deal with complex information such as text or images, making it applicable to a variety of scenarios, including video recommendations, product recommendations, and music recommendations, among others. Its simplicity and universality represent the major strengths of collaborative filtering algorithms, contributing to their widespread success in various fields. However, despite the excellent performance, collaborative filtering algorithms fall short when dealing with the cold start problem. Koren et al. [3] proposed a recommendation algorithm based on matrix factorization, decomposing the rating matrix into latent feature representations for users and items, thereby exploring deeper relationships between users and items. Different from matrix factorization, which models the interaction matrix of users and items by directly multiplying their latent feature representations, Xue et al. [4] introduced a deep matrix factorization model. This model utilizes a nonlinear multilayer perceptron to replace direct multiplication, enabling a better capture of nonlinear relationships. However, most existing collaborative filtering algorithms only model representations of users and items based on their unique attributes, without considering the potential higher-order relationships between users and items in the interaction graph. He et al. [5] proposed a neural graph-based collaborative filtering algorithm. This algorithm explores high-order structural information by propagating node representations on the constructed user-item bipartite graph.

A graph is a crucial information organization structure consisting of nodes and edges. In the context of recommendation systems, graph data can be applied from two perspectives. Content-based recommendation systems require extensive background knowledge about items and users. This knowledge can often be organized in the form of a knowledge graph. With the development of the internet, a considerable amount of knowledge has been structured into graphs, such as Wikipedia or knowledge trees and classification tree structures like e-commerce directories. Since a tree structure is itself a type of graph, effectively integrating the data from knowledge graphs into the training of recommendation systems will significantly enhance the system's breadth and depth of knowledge utilization. Velickovic et al. [6] proposed GAT, which utilizes attention mechanisms to adaptively assign weights to the neighbors of nodes during spatial convolutions. Recent works [7] have considered the heterogeneity of nodes in a heterogeneous graph, introducing heterogeneous graph attention networks that fully account for the varying importance of different types of nodes during the convolution process.

In the realm of recommendation based on graph neural networks, Rex Ying et al. [8] introduced PinSage in 2018, marking the first application of GNN in the field of business recommendations with promising results. Wang et al. [9] proposed a neural graph-based collaborative filtering algorithm, which explores high-order structural information by propagating node representations on the constructed user-item bipartite graph. Wang et al. [10] presented KGCF, applying knowledge graphs to the recommendation domain and combining graph neural networks to explore relationships between products and attributes in the knowledge graph, effectively capturing the intrinsic connections of products. Subsequently, Wang et al. [11] introduced KGAT, which integrates knowledge graphs while incorporating attention-based aggregation methods to address the inequality of high-order relationships. Fan et al. [12] proposed GraphRec, designing a graph neural network to capture social relationships and rating information in the user-item graph. Wei et al. [13] designed a MMGCN framework built upon the message-passing idea of graph neural networks, which can yield modal-specific representations of users and micro-videos to better capture user preferences.

This paper conducts research on recommender systems based on graph neural network (GNN) technology. Utilizing graph convolutional neural networks, a graph information extractor is constructed to merge information from the user-item bipartite graph's node neighbors and generate dense vector representations for the nodes. The model employs a multi-task learning approach by introducing the reconstruction tasks of auxiliary information graphs for users and items into the traditional collaborative filtering recommendation task through graph neural networks.

2 Related Work

2.1 Collaborative Filtering

Collaborative filtering-based recommendation is currently the most widely used recommendation algorithm. It discovers the latent correlations between users and items based on their historical interaction records and recommends items based on these associations. The relevant algorithms include item-based collaborative filtering algorithms [14], user-based collaborative filtering algorithms [15], and model-based collaborative filtering algorithms [16]. The major challenge of collaborative filtering algorithms is the cold start problem [17]. Collaborative filtering algorithms rely on rich interactions between users and items to be effective. However, when there are too few interactions between users and items, the performance of collaborative filtering algorithms deteriorates. For instance, in the early stages of commercial music software where user-song interactions are minimal, using collaborative filtering algorithms becomes challenging. One solution to this problem is to provide additional information about users and items to the recommendation system. Knowledge graphs, as high-quality structured data, are increasingly chosen by algorithms to address the cold start problem and enhance the recommendation capabilities of the system due to their high data quality advantages [18].

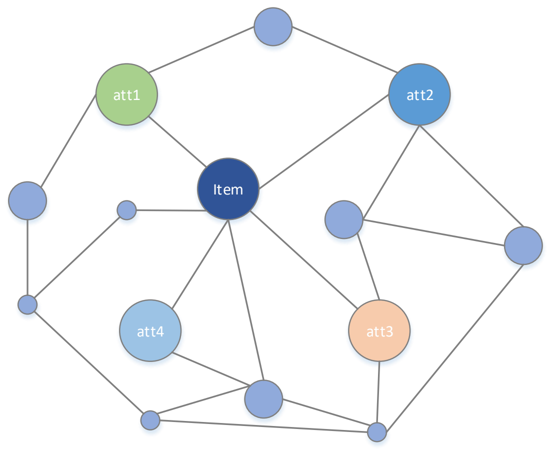

Fig. 1: Knowledge Graph based Recommendation.

2.2 Knowledge Graph-Based Recommender Model

These types of algorithms directly utilize knowledge graph embedding models like TransE [19], TransR [20], etc., to generate embeddings for the knowledge graph. The embeddings are then incorporated as input features to provide additional knowledge about items to the recommender system, enhancing the prediction accuracy [21]. For example, MKR [22] designs a separate cross & compress unit, sharing feature parameters between the recommendation and knowledge graph embedding tasks, enabling the recommendation task to benefit from the knowledge graph embedding task for improved performance. CKE8 simultaneously incorporates visually related knowledge, text knowledge, and structured knowledge from the knowledge graph as features into collaborative filtering algorithms to enhance predictive capabilities. CKE uses TransR to generate embeddings for structured knowledge in the knowledge graph. DKN [23] utilizes TransD [24] to generate embeddings for entities in news and constructs embeddings for news using embeddings generated from sentences, characters, and entities in the news. User embeddings in DKN are composed of embeddings from news articles clicked by the user. KSR [25] employs TransE to generate entity embeddings and designs a Key-Value Memory Network to utilize knowledge from the knowledge graph. KTUP [26] utilizes TransH [27] to simultaneously learn the recommender model and the Knowledge Graph Completion model. PRE [28] uses path-based similarity [29] to calculate the recommendation probability for items. FMG [30] constructs a metagraph by combining multiple metapaths and further utilizes metapath-based similarity for recommendation. HERec [31] and Metapath2vec [32] employ metapaths for random walks, generating embeddings for entities based on the points sampled from random walks. Metapaths are often constructed by experts, making them semantically clear and structurally reasonable, which is a significant advantage of using metapaths in these methods.

3 METHODOLOGY

3.1 Graph Feature Extraction

By embedding the items involved in the user's historical interactions and taking the average, the visited item vector is obtained. The item information embedding layer is represented by a matrix named $N_i \times D$, capable of converting the input item ID into a dense vector of D dimensions. The average pooling result can be obtained using the following formula:

$$h_v = \frac{1}{|N|} \sum_{i \in N_m} \alpha_i \tag{1}$$

Average pooling takes the average of all activation values within the pooling domain. High positive activation values may offset low negative activation values, leading to the loss of discriminative information. Additionally, using the average method to utilize information in the embedded vectors for pooling does not assign appropriate weights. Items closer to the prediction time are more likely to represent the user's short-term preferences, having a greater impact on the user's next clicked item. On the other hand, historical items further away from the prediction time reflect the user's long-term stable preferences. Therefore, a more focused approach to aggregating historical interaction information is to treat it as a time series feature, introducing a sequence feature extractor.

Let f be the transformation function of the recurrent network's hidden layer. The transformation for the t-th time step of the hidden layer vector is as follows:

$$h_t = f(x, h_{t-1}) \tag{2}$$

The encoder transforms the hidden state at each time step into a context vector as follows:

$$c = q(h_1, \ldots, h_T) \tag{3}$$

where q is a mapping function, and in this context where dealing with fixed-length sequences, the choice of q is a concatenation and a single-layer fully connected operation:

$$q = \theta(W \cdot concat(h_1, \ldots, h_T) + b) \tag{4}$$

3.2 Rule filtering

All connected paths between users and items, established through the "interact" relation, where the direction from users to items is ignored, can serve as candidate inference rules for the interact predicate. Constructing the knowledge graph as an undirected graph, the input being the triplets $(u, interact, i)$, this paper utilizes a bidirectional breadth-first search to obtain all connected paths with a length of at most I between u and i. These paths serve as candidate rules for the interact predicate.

For a triplet (s, p, o), RotatE maps s, p, and o to the complex vector space, defining the predicate p as the rotation from s to o. The distance between p and the predicate can be considered as the confidence evaluation criterion for the rule r. The confidence $conf(r)$ is defined as follows:

$$r : p \Leftarrow p_1 \wedge p_2 \wedge \cdots \wedge p_h \tag{5}$$

$$conf(r) = -||p - f(p)||_2 \tag{6}$$

where $|| \cdot ||_2$ represents the L_2 norm of the complex vector.

Expanding along the rules for users. Then, aggregating the expanded entity nodes back to the user nodes in reverse. The k-hop extended entity set for user u on rule r is obtained by the following formula:

$$D_u^k(r) = \{o|(s, p_k, o), s \in D_u^{k-1}(r)\} \tag{7}$$

where $k \in [1, h]$, h is the length of the rule, and $D_u^0(r) = \{u\}$. The set of entities J_i subjected to the aggregation operation during the aggregation process is represented by the following formula:

$$J_i = \{D_u^0(r) \cup \cdots \cup D_u^{h-i}(r)\} \tag{8}$$

where $i \in [1, h]$. The variable h represents the length of rule r, indicating the total number of iterations to be performed on rule r. Aggregate each point along the extension direction in sequence. The user is the point where the aggregation is initially performed, and the nodes expanded from the user are points where the aggregation is subsequently performed. The aggregation process is repeated h times, performed successively on J_1 to J_h. Construct the feature representation of the central node by aggregating the feature representations of neighboring nodes onto it and applying a non-linear activation function.

3.3 Feature Integration

It is assumed that under the guidance of L rules $\{r_1, r_2, \ldots, r_L\}$, this paper obtains L feature representations L, where h_j is the length of rule r_j. The final representation U of user u is calculated by the following formula:

$$U = LW = [U_{r_1}^{h_1}, \ldots, U_{r_L}^{h_L}]W \tag{9}$$

Where W is the weight vector with dimensions $(L \times 1)$. L is the representation matrix obtained under the guidance of L rules, with dimensions $d_r \times L$.

The final loss function is defined as follows:

$$Loss = \frac{1}{N} \sum_{i}^{N} (l_i - q(U_i^T M_i))^2 + \mu ||W||_2 \tag{10}$$

where the size of M_i is $(d_r \times 1)$. q is a non-linear function, such as sigmoid. $q(U_i^T M_i)$ calculates the probability that user u_i is interested in item m_i.

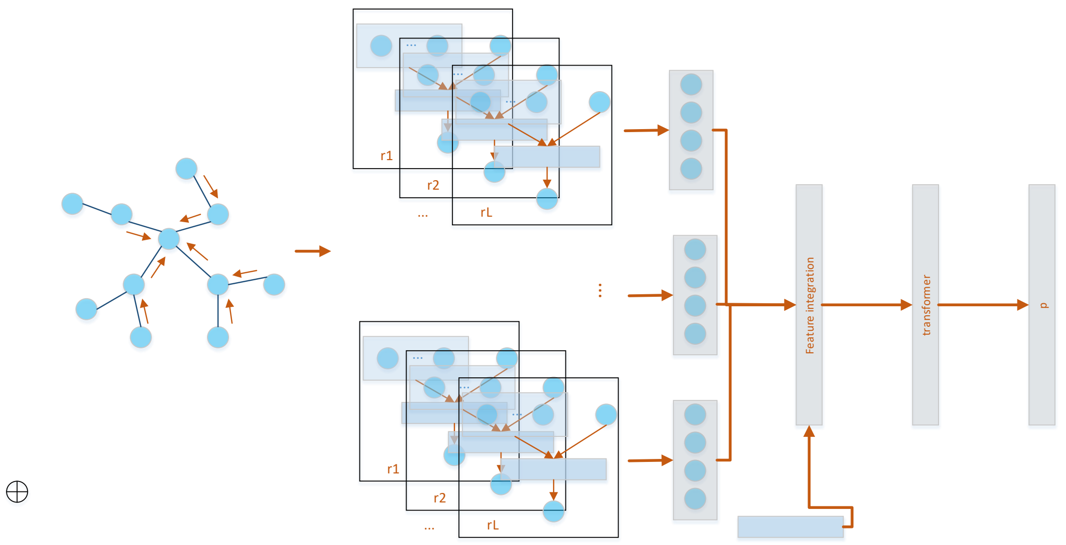

Fig. 2: Framework of Our proposed Model.

TABLE 1: The statistical data of Last.FM and MovieLens-1M.

Data	MovieLens-1M	Last.FM
users	6036	1872
items	2445	3864
interactions	753772	42346
entities	182011	9366
KG Triples	1241995	15518

4 EXPERIMENTS

4.1 Datasets

We used the publicly available music dataset Last.FM and the movie dataset MovieLens-1M as recommendation datasets. Last.FM has over 20 million users every month using the website. Leveraging this large user base, the site provides real, reliable, and high-quality music recommendation data. This article used the publicly available dataset from the website, which includes 1871 users, 3864 songs, and 42,346 user-music interactions. MovieLens-1M is a widely used recommendation dataset in the movie domain, released by the GroupLens research project at the University of Minnesota. The dataset comprises 6036 users, 2445 items (movies), and 753,772 user-movie interactions. The complete statistical data for the three datasets is shown in Table 1, where ”Users” represents the number of users, ”Items” represents the number of items, ”Interactions” represents the number of interactions between users and items, ”Entities” represents the number of entities in the knowledge graph, and ”KG Triples” represents the number of triples in the knowledge graph.

4.2 Baselines

PER [28] is a representative recommendation algorithm based on meta-paths. This algorithm treats the entire knowledge graph as a Heterogeneous Information Network (HIN) and calculates path-based similarity using meta-paths for recommendation predictions.

CKE [33] is a representative recommendation algorithm designed to use only entity embeddings. In the original CKE paper, it combines three types of knowledge—structured knowledge, text knowledge, and visual knowledge—into a collaborative filtering algorithm for recommendation.

DeepFM [34] is also a general deep recommendation model that combines a factorization machine component with a deep neural network component. It models interactions between low-level and high-level features by sharing input. In this paper, we use a similar input setup as DeepWide.

RKGE [35] is a representative algorithm that automatically mines paths and models them for designing a recommendation system. The paths mined by this algorithm refer to connected entities between users and items, excluding the predicates between entities.

4.3 Metrics

Accuracy (ACC) is the ratio of all correctly classified samples to the total number of samples in a classification task. Due to the significant impact of sample distribution on accuracy results, it can lead to bias in model evaluation. Therefore, it is generally not used as a standalone evaluation metric in tasks. However, it can to some extent reflect the user perception when the model is applied in practical scenarios.

AUC simplifies the vast multi-class classification task by transforming it into a binary classification task through negative sampling, making it an efficient training strategy. When evaluating binary classification algorithms, AUC is a widely recognized metric, representing the area under the Receiver Operating Characteristic (ROC) curve. The ROC curve plots the false positive rate against the true positive rate.

$$AUC = \frac{\sum_{i \in p} rank_i - \frac{M(1+M)}{2}}{M \times N} \tag{11}$$

where M is the number of positive samples, N is the number of negative samples, and the formula is equivalent

TABLE 2: The experimental results on the MovieLens-1M.

Model	AUC	ACC
PER	0.677	0.645
CKE	0.713	0.668
DeepFM	0.735	0.679
RKFE	0.755	0.701
OURS	0.783	0.723

TABLE 3: The experimental results on Last.FM.

Model	AUC	ACC
PER	0.627	0.605
CKE	0.646	0.614
DeepFM	0.661	0.637
RKFE	0.705	0.652
OURS	0.725	0.683

to evaluating the number of positive-negative sample pairs for which the score of the positive sample is greater than the score of the negative sample.

4.4 Experimental Setting

When obtaining test results, this paper divided the dataset into training, validation, and test sets in a 6:2:2 ratio. The early stopping strategy was employed to determine the appropriate number of training epochs. Specifically, if the algorithm's evaluation metrics on the validation set did not improve for three consecutive training epochs, the training process was halted. The reported results are averaged over 5 runs for each experiment.

4.5 Result Analysis

According to the experimental results, it can be observed that the proposed structure performs the best in both datasets. On the other hand, the performance of PER is not satisfactory on both datasets. This is because PER requires manual selection of metapaths, and in scenarios with a large number of entity types in the knowledge graph, manual parameter setting can introduce interference, making it difficult for the model to learn effectively. Additionally, NeuMF, which cannot effectively utilize auxiliary features, performs relatively worse. It is noteworthy that NeuMF performs better on the sparser Last.FM dataset, demonstrating the generalization ability of neural networks. The experimental results also demonstrate that our model's multitask design effectively leverages graph structure information to assist in the recommendation task.

5 Conclusion and Future Work

A recommender system is an algorithm designed to predict whether a user is interested in a particular item, essentially functioning as an information filtering system. For example, in video-sharing platforms like YouTube, the primary task of the recommender system is to suggest videos that users are likely to enjoy. The advantages of collaborative filtering algorithms are quite evident; they don't require detailed information about users and items but instead rely on interaction records between users and items to complete the entire recommendation process. Therefore, this algorithm doesn't need to deal with complex information such as text or images, making it applicable to a variety of scenarios, including video recommendations, product recommendations, and music recommendations, among others. A graph is a crucial information organization structure consisting of nodes and edges. In the context of recommendation systems, graph data can be applied from two perspectives. Content-based recommendation systems require extensive background knowledge about items and users. This knowledge can often be organized in the form of a knowledge graph. With the development of the internet, a considerable amount of knowledge has been structured into graphs, such as Wikipedia or knowledge trees, and classification tree structures like e-commerce directories. Since a tree structure is itself a type of graph, effectively integrating the data from knowledge graphs into the training of recommendation systems will significantly enhance the system's breadth and depth of knowledge utilization. This paper conducts research on recommender systems based on graph neural network (GNN) technology. Utilizing graph convolutional neural networks, a graph information extractor is constructed to merge information from the user-item bipartite graph's node neighbors and generate dense vector representations for the nodes. The model employs a multi-task learning approach by introducing the reconstruction tasks of auxiliary information graphs for users and items into the traditional collaborative filtering recommendation task through graph neural networks.

References

[1] M. B. Dias, D. Locher, M. Li, W. El-Deredy, and P. J. Lisboa, "The value of personalised recommender systems to e-business: a case study," in *Proceedings of the 2008 ACM conference on Recommender systems*, 2008, pp. 291–294.

[2] B. Marlin and R. S. Zemel, "The multiple multiplicative factor model for collaborative filtering," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 73.

[3] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[4] H.-J. Xue, X. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems." in *IJCAI*, vol. 17. Melbourne, Australia, 2017, pp. 3203–3209.

[5] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th international conference on world wide web*, 2017, pp. 173–182.

[6] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[7] H. Linmei, T. Yang, C. Shi, H. Ji, and X. Li, "Heterogeneous graph attention networks for semi-supervised short text classification," in *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, 2019, pp. 4821–4830.

[8] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018, pp. 974–983.

[9] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*, 2019, pp. 165–174.

[10] H. Wang, M. Zhao, X. Xie, W. Li, and M. Guo, "Knowledge graph convolutional networks for recommender systems," in *The world wide web conference*, 2019, pp. 3307–3313.

[11] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, "Kgat: Knowledge graph attention network for recommendation," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 950–958.

[12] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, "Graph neural networks for social recommendation," in *The world wide web conference*, 2019, pp. 417–426.

[13] Y. Wei, X. Wang, L. Nie, X. He, R. Hong, and T.-S. Chua, "Mmgcn: Multi-modal graph convolution network for personalized recommendation of micro-video," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 1437–1445.

[14] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*, 2001, pp. 285–295.

[15] A. Gunawardana and G. Shani, "A survey of accuracy evaluation metrics of recommendation tasks." *Journal of Machine Learning Research*, vol. 10, no. 12, 2009.

[16] S. Jiang, X. Qian, J. Shen, Y. Fu, and T. Mei, "Author topic model-based collaborative filtering for personalized poi recommendations," *IEEE transactions on multimedia*, vol. 17, no. 6, pp. 907–918, 2015.

[17] Y. Wei, X. Wang, Q. Li, L. Nie, Y. Li, X. Li, and T.-S. Chua, "Contrastive learning for cold-start recommendation," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021, pp. 5382–5390.

[18] Y. Wei, X. Wang, X. He, L. Nie, Y. Rui, and T.-S. Chua, "Hierarchical user intent graph network for multimedia recommendation," *IEEE Transactions on Multimedia*, vol. 24, pp. 2701–2712, 2021.

[19] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," *Advances in neural information processing systems*, vol. 26, 2013.

[20] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 29, no. 1, 2015.

[21] Y. Wei, W. Liu, F. Liu, X. Wang, L. Nie, and T.-S. Chua, "Lightgt: A light graph transformer for multimedia recommendation," in *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2023, pp. 1508–1517.

[22] H. Wang, F. Zhang, M. Zhao, W. Li, X. Xie, and M. Guo, "Multi-task feature learning for knowledge graph enhanced recommendation," in *The world wide web conference*, 2019, pp. 2000–2010.

[23] H. Wang, F. Zhang, X. Xie, and M. Guo, "Dkn: Deep knowledge-aware network for news recommendation," in *Proceedings of the 2018 world wide web conference*, 2018, pp. 1835–1844.

[24] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, "Knowledge graph embedding via dynamic mapping matrix," in *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)*, 2015, pp. 687–696.

[25] J. Huang, W. X. Zhao, H. Dou, J.-R. Wen, and E. Y. Chang, "Improving sequential recommendation with knowledge-enhanced memory networks," in *The 41st international ACM SIGIR conference on research & development in information retrieval*, 2018, pp. 505–514.

[26] Y. Cao, X. Wang, X. He, Z. Hu, and T.-S. Chua, "Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences," in *The world wide web conference*, 2019, pp. 151–161.

[27] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 28, no. 1, 2014.

[28] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han, "Personalized entity recommendation: A heterogeneous information network approach," in *Proceedings of the 7th ACM international conference on Web search and data mining*, 2014, pp. 283–292.

[29] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," *Proceedings of the VLDB Endowment*, vol. 4, no. 11, pp. 992–1003, 2011.

[30] H. Zhao, Q. Yao, J. Li, Y. Song, and D. L. Lee, "Meta-graph based recommendation fusion over heterogeneous information networks," in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 2017, pp. 635–644.

[31] C. Shi, B. Hu, W. X. Zhao, and S. Y. Philip, "Heterogeneous information network embedding for recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 2, pp. 357–370, 2018.

[32] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 2017, pp. 135–144.

[33] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 353–362.

[34] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, "Learning deep structured semantic models for web search using clickthrough data," in *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, 2013, pp. 2333–2338.

[35] Z. Sun, J. Yang, J. Zhang, A. Bozzon, L.-K. Huang, and C. Xu, "Recurrent knowledge graph embedding for effective recommendation," in *Proceedings of the 12th ACM conference on recommender systems*, 2018, pp. 297–305.