Knowledge-based Recommendation with Hierarchical Collaborative Embedding

Zili Zhou[1 2], Shaowu Liu[1], Guandong Xu[1 ⋆], Xing Xie[3],
Jun Yin[1], Yidong Li[4], Wu Zhang[2]

[1] Advanced Analytics Institute, University of Technology Sydney.
[2] School of Computer Engineering and Science, Shanghai University.
[3] Microsoft Research Asia.
[4] School of Computer and Information Technology, Beijing Jiaotong University.
zili.zhou@student.uts.edu.au, shaowu.liu@uts.edu.au,
guandong.xu@uts.edu.au, xingx@microsoft.com,
jun.yin-2@student.uts.edu.au, ydli@bjtu.edu.cn,
wzhang@shu.edu.cn

Abstract. Data sparsity is a common issue in recommendation systems, particularly collaborative filtering. In real recommendation scenarios, user preferences are often quantitatively sparse because of the application nature. To address the issue, we proposed a knowledge graph-based semantic information enhancement mechanism to enrich the user preferences. Specifically, the proposed **H**ierarchical **C**ollaborative **E**mbedding (**HCE**) model leverages both network structure and text info embedded in knowledge bases to supplement traditional collaborative filtering. The **HCE** model jointly learns the latent representations from user preferences, linkages between items and knowledge base, as well as the semantic representations from knowledge base. Experiment results on *GitHub* dataset demonstrated that semantic information from knowledge base has been properly captured, resulting improved recommendation performance.

1 Introduction

Recommendation has been widely used in today's business. By observing user past behaviors, recommender systems can identify items with potential to be interested by users. A popular technique in recommendations is collaborative filtering (CF) which is based on the intuition that preference history can be transferred across like-minded users. However, CF suffers from the cold-start problem in which users usually provide limited amount of preferences, i.e., preference data is quantitatively sparse, making recommendation inaccurate. Particularly, in some real recommendation scenarios, user preferences are often quantitatively sparse because of the application nature. For example, unlike watching many movies, users typically can only study a few subjects in *Coursera* [5] or contribute to a few repositories (projects) on on *GitHub* [6], both of which contain very high numbers of subjects or projects.

⋆ Corresponding author
[5] Online course platform https://www.coursera.org/
[6] Project hosting platform https://github.com/

To address the sparsity problem, researchers have proposed to extend the spare data by connecting to external knowledge graphs [19, 17, 15, 4]. This approach leverages both network structure and text info embedded in knowledge bases to supplement traditional CF. To be specific, a knowledge base is a data repository containing interlinked entities across different domains. Since knowledge base is often represented in a graph way, it is also called as knowledge graph (KG). The beauty of knowledge graph is not only the textual knowledge representations, but also the linked structure of knowledge entities. Recently, knowledge graph has emerged as a new method in recommender systems research. For example, latent features are often extracted from heterogeneous information network to represent users and items [17, 15, 4]. More recently, Zhang et al. [19] proposed the first work to build a hybrid heterogeneous information network containing both recommender system and knowledge graph.

On the other hand, although the above mentioned recommendation tasks exhibit the significant sparsity, we argue that the user choices/behaviors carry on rich semantics info which has not been fully utilized in recommendation. For example, knowing a user's interest in a subject or repository reveals lots of information about this user, such as preferences over *programming language*, *operating system*, *field of study*, *research topic*. From knowledge graph view, such information pieces are not isolated and fragmented, instead, interrelated, forming a comprehensive view of this author. This rich semantic information can play an important role in alleviating such cold-start and sparsity problem, therefore using KG-based approaches becomes an idea solution to this kind of tasks. However, previous studies on KG-based CF suffer from one or more of the following limitations: 1) rely on tedious feature engineering; 2) the high data sparsity; and 3) recommended items need to be one exact entity within knowledge base.

To address the above issues, we propose a novel collaborative recommendation framework to integrate recommender system and knowledge graph with extensible connection between items and knowledge entities. Overall, our method constructs a multi-level network via knowledge graph to enhance sparse semantic information between users and items. Let's take example of *GitHub* recommendation task shown in Fig. 1 to show how our model works. In order to reveal the latent correlations between *GitHub* repositories,the names of which are not existent in knowledge graph, we frame the integrated system into 3-level, where users, repositories, knowledge graph entities are placed in different levels, with edges between users and repositories indicating the user has interest in the repository, edges between repositories and entities indicating that the entity is possibly related to the repository, and edges between entities meaning there is at least one specific relation between this pair of entities. A hierarchical structure heterogeneous network, which contains multiple types of nodes and multiple types of edges, is built for automatic collaborative learning. Particularly, to link recommender system and knowledge graph properly, knowledge conceptual level is proposed to indirectly map item-entities, different from previous works of direct mapping. Serving as middle level of three-level hierarchical structure model, the knowledge conceptual level can fully interconnect the whole system in a proper way, tackling the restriction that recommendation items need to be within knowledge base.

The main contributions of this paper are as follows:

– A novel KG-based recommender system with knowledge conceptual level is proposed to properly encode the correlation amongst items which are non-existent knowledge graph entities.
– A new collaborative learning algorithm is devised to deal with the proposed three-level network for sparse user preference data.
– We conducted extensive experiments on *GitHub* recommendation task, which is extremely sparse but rich semantics in user preference, to evaluate the effectiveness of our model. To the best of our knowledge, this is the first trial work of using knowledge graph embedding, to deal with semantic enhancement for entities not existent in conventional knowledge graph.

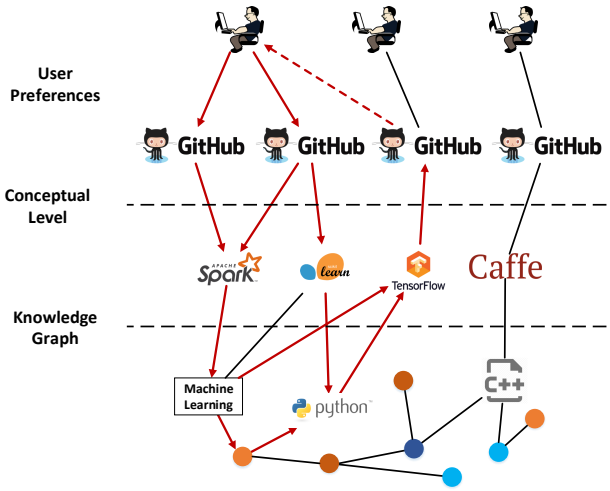


Fig. 1. Conceptual Level

The rest of the paper first introduces the basic concepts of collaborative filtering and knowledge graph, followed by a detailed discussion of the proposed hierarchical collaborative embedding model. The proposed model is compared with several baselines on *GitHub* dataset.

2 Preliminary

This section briefly summarizes necessary background of *Implicit Feedback Recommendation* and *Knowledge Graph* that form the basis of this paper.

2.1 Implicit Feedback Recommendation

This paper considers the implicit feedback recommendation problem [18], i.e., analyzing interactions among users and items instead of explicit ratings. The implicit user

feedback is encoded as a matrix $\mathbf{R} \in \mathbb{R}^{m*n}$, where $R_{ij} = 1$ if user i has interacted with item j and $R_{ij} = 0$ otherwise. The user-item interactions are defined per application scenario, e.g., a *GitHub* user "stars" (follows) a repository, or a *Coursera* user "enrolled" in a subject. Generally speaking, an interaction $R_{ij} = 1$ implies the user is interested in the item, however, the meaning of $R_{ij} = 0$ is not necessarily to be not interested. In fact, the matrix $\mathbf{R}$ is often sparse and most entries will be 0, where the 0 value indicates that the user either has no interest in the item or has interest but not interacted with the item yet. The goal of implicit feedback recommendation is to identify which 0 entries in $\mathbf{R}$ have the potential to become 1.

2.2 Knowledge Graph

The implicit feedback matrix $\mathbf{R}$ can be extreme sparse as some users may have only interacted with one or two items. Although modeling moderately sparse data has been considered by traditional CF methods, it remains a challenging problem of utilizing extreme sparse data. Fortunately, if the items contain rich semantic information, then only a few items will be able to connect the user to knowledge graph, such that more complete user profiles can be built. To be specific, knowledge graph is a semantic web consist of *entities* and *relations*, where entities represent anything in the world including people, things, events, etc., and relations connect entities that have interactions with each other. For example, in *GitHub* repository recommendation, entities can be software development concepts such as programming language *C++*, operating system *Linux*, development framework *TensorFlow*, etc. The entities are connected through relations such as "is programming language of", "have dependency on", "is operating system of", etc. Denoting entities as nodes and relations as edges, knowledge graph can be represented by a heterogeneous network with multiple types of nodes and multiple types of edges.

Although using knowledge graph in recommendation is promising, it is assumed that the recommended items are entities in knowledge graph. This assumption may hold for recommending movies or tourism destinations where the items are already entities in knowledge graph, but it becomes invalid for items that are non-existent in knowledge graph, such as repositories in *GitHub*. Therefore, the link between non-existent items and knowledge graph entities must be identified together with reliability and importance measures.

For ease of reference, notations used throughout this paper are summarized as following. U_i represents vector of user i; V_j represents vector of item j; E_i represents vector of entity i; B_j represents bias vector of item j; W_k represents weigh of entity k; I_j represents possibly related entities set of item j; R represents factorized matrix of relation r; r represents vector of relation r; M_r represents subspace mapping matrix of relation r; $p(i, j, j')$ represents preference function of triple (user i, item j, item j'); $\mathcal{X}_{i,j,j'}$ represents user preference term of training function; $\mathcal{Y}_{h,r,t,t'}$ represents knowledge graph embedding term of training function; $\mathcal{Z}$ represents regularization term of training function; f_r represents knowledge graph triple score function used in training function; f_r^{TransR} represents TransR score function; f_r^{RESCAL} represents RESCAL score function.

2.3 Problem Definition

The research problem of this paper can be defined as follows: given quantitatively sparse but semantically dense user feedback data, how to leverage knowledge graph to perform semantic enhancement for items that do not exist in knowledge graph, such that the item recommendation quality can be improved.

3 Hierarchical Collaborative Embedding

In this section, we propose the **H**ierarchical **C**ollaborative **E**mbedding model (**HCE**) to bridge knowledge graph to CF, which jointly learns the embedding of elements, including users, items, entities, and relations.

3.1 Knowledge Graph Structured Embedding

With large amount of knowledge being extracted from open source, knowledge graph was proposed to store the knowledge with graph structure. The knowledge facts are represented by triples, each triple has two entities (head entity and tail entity) and one relation in between. Given all triples, the entities and relations can be considered as nodes and edges, respectively, resulting a large scale of heterogeneous knowledge graph. To capture the latent semantic information of entities and relations, several embedding based methods [3, 2, 9, 8] were proposed. These methods embed entities and relations into a continuous vector space, in which the latent semantic information can be reasoned automatically according to vector space position of entities and relations.

Two state-of-the-art knowledge graph structured embedding methods are employed in this paper: RESCAL [9] and TransR [8]. One important advantage of these two methods is the capability of modeling multi-relational data where more than one relation may exist between two entities.

RESCAL uses three-way tensor to represent triples set, each element of a triple (head entity, relation, or tail entity) is represented by one dimension, and tensor factorization is used to obtain the entity and relation representations. To be specific, each entity is represented by a vector and each relation is represented by a matrix. Y is the three-way tensor which represent all the triples, Y_k is a matrix picked up from Y, it only contains triples with relation k. E_i and E_j are the representation vectors of entity h and t, W_k is the representation matrix of relation r.

The representations of entities and relations are constructed by minimizing the following objective function:

$$\min_{E,W_k} \sum_k \|Y_k - EW_kE^T\|_F^2. \tag{1}$$

In a triple (h, r, t), each entity is represented by a vector, E_h for head entity h, E_t for tail entity t, and relation r is represented by a matrix R. The RESCAL score function of a triple (h, r, t) is defined as:

$$f_r^{\text{RESCAL}}(h,t) = \|E_hRE_t\|_2^2, \tag{2}$$

TransR uses a different score function for triples. Given a triple (h, r, t), the head and tail entities are represented by vectors E_h and E_t, respectively. Each relation is represented by a vector $\mathbf{r}$ together with a matrix M_r. TransR firstly maps entity h and t into subspace of relation r by using matrix M_r:

$$E_h^r = M_r E_h, E_t^r = M_r E_t. \tag{3}$$

The score function of TransR is defined as follows:

$$f_r^{\text{TransR}}(h,t) = \|E_h^r + \mathbf{r} - E_t^r\|_2^2. \tag{4}$$

In learning process, we pick up a true triple (h, r, t) and generate a false triple by replacing one entity of the triple by another entity: (h, r, t'). Then we make the score value of true triple larger than that of false triple: $f_r(h,t) > f_r(h,t')$.

3.2 Knowledge Conceptual Level Connection

This work focuses on recommender systems without direct connection to knowledge graph, i.e., most recommendation items do not exist in knowledge graph. For example, a *GitHub* project with a customized name is not an entity in knowledge graph. Consequently, methods such as [19] that rely on direct mapping between items and knowledge graph entities are not applicable. However, by extracting content information from items, such as item description and user reviews, potential links between items and entities can be constructed. To bridge recommender system and knowledge graph with item-entity links, we propose a collaborative learning model with hierarchical structure of three levels: the recommender system level, the knowledge graph level, and the knowledge conceptual level (KCL). The KCL plays a key role in the model to connect the other two levels and enables collaborative learning.

Creating the knowledge conceptual level has two challenges. The first challenge is how to filter irrelevant linkages. The automated extraction of item content introduces lots of irrelevant information for the recommendation task. For example, in *GitHub* project recommendation, the project description may include vocabulary of specific areas such as biology and chemistry, which are off-topic of general purpose coding recommendation. While this information is irrelevant, it is actually linked to knowledge graph entities, thus introducing noise data.

The second challenge is how to measure the influences of knowledge graph entities on recommendation items. The entities have different influences on items, thus the links between items and entities must be weighed in order to represent an item precisely. To tackle these two challenges, the proposed Knowledge Conceptual Level implements the filtering and weighing functionalities. To be specific, a weighed link function is used to represent each item with their possibly related entities (automatically extracted from side information). The representation of an item is the weighted sum of vectors of possibly related entities plus a bias term B_j. Maximizing the weighed link function of each item is one of the targets in collaborative learning process. The weighed link function of an item is defined as follows:

$$V_j = B_j + \sum_{k \in I_j} W_k E_k \tag{5}$$

where V_j is the representation of item j, E_k is the representation of entity k, W_k is the weigh parameter of entity k, and I_j is the set containing all the entities which are possibly related to item j. If the entity is unrelated to current recommendation task, the weigh parameter should be lowered to near zero during learning process, if the influential degree of the entity is minor, the weigh parameter should be lowered accordingly. The filtering and weighing are both achieved by knowledge conceptual level.

3.3 Collaborative Learning

To integrate recommender system with knowledge graph, the proposed collaborative learning framework learns the embedding representations of both recommender system elements (users and items) and knowledge graph elements (entities and relations).

Because of user feedback is implicit, similar to some previous works [14, 19], we use pairwise ranking of items in our learning approach. Given user i, item j and item j', using $F_{i,j}$ to represent the feedback of user i for item j, if $F_{i,j} = 1$ and $F_{i,j'} = 0$, then we consider user i prefer item j over item j', we use preference function $p(i, j, j')$ to represent this pairwise preference relation, and $p(i, j, j') > 0$. More specifically, in our model, we use same-dimension vector representation for user and item, the preference function is defined as following,

$$p(i, j, j') = \ln \sigma(U_i^T V_j - U_i^T V_{j'}) \tag{6}$$

U_i is the vector representing user i, V_j is the vector representing item j, $V_{j'}$ is the vector representing item j', σ is sigmoid function.

Integrating knowledge graph embedding and knowledge conceptual level, the collaborative learning leverage the information from both user feedback and knowledge graph. by repeating following procedure. Jointly, we aim to maximize the likelihood function in Eq. 7 and the overall learning algorithm is summarized in Alg. 1.

$$\begin{aligned}
&\mathcal{L} = \sum_{(i,j,j') \in D} \mathcal{X}_{i,j,j'} + \sum_{(h,r,t,t') \in S} \mathcal{Y}_{h,r,t,t'} + \mathcal{Z} \\
&\mathcal{X}_{i,j,j'} = \ln \sigma(U_i^T V_j - U_i^T V_{j'}) \\
&\mathcal{Y}_{h,r,t,t'} = \ln \sigma(f_r(h, t) - f_r(h, t')) \\
&\mathcal{Z} = \frac{\lambda_U}{2} \|U\|_2^2 + \frac{\lambda_E}{2} \|E\|_2^2 + \frac{\lambda_R}{2} \|R\|_2^2 + \frac{\lambda_B}{2} \|B\|_2^2 + \frac{\lambda_W}{2} \|W\|_2^2 \\
&V_j = \sum_{k \in I_j} W_k E_k
\end{aligned} \tag{7}$$

4 Experiment

In this section, we introduce the dataset, the baselines and the results of comparison experiments.

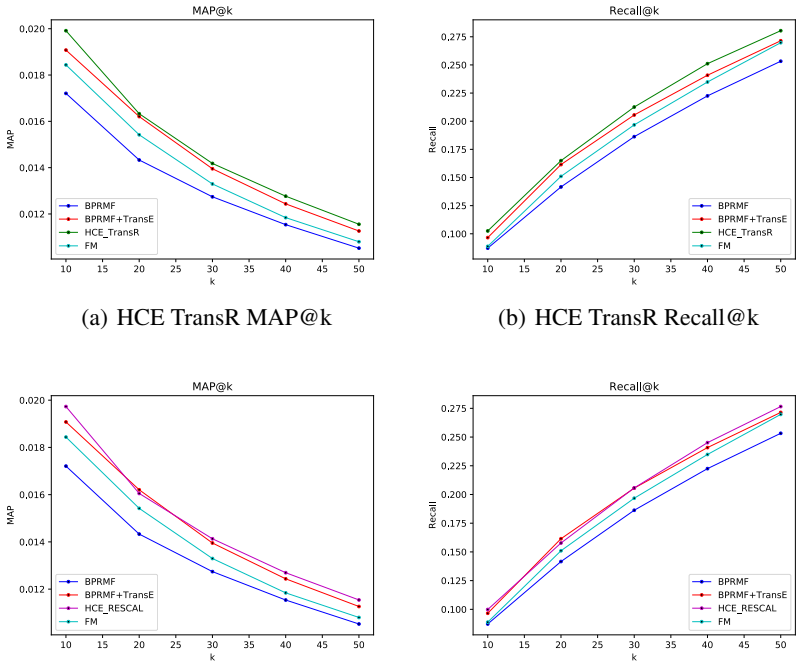
Algorithm 1 *HCE* Algorithm

Input: User preferences, Knowledge graph, Item&entity links.
Training:
Step 1: Draw pairwise user-entity triple set $\mathcal{D}$.
Step 2: Repeat
for each $(u_i, v_j, v_{j'}) \in \mathcal{D}$ **do**
  Draw pairwise entity-relation quadruple set $\mathcal{S}_{j,j'}$
  Draw possibly related entities set I_j and $I_{j'}$
  for each $(h, r, t, t') \in \mathcal{S}_{j,j'}$ **do**
    Represent item by embedding of entities:
    $V_j = B_j + \sum_{k \in I_j} W_k * E_k$
    $V_{j'} = B_{j'} + \sum_{k \in I_{j'}} W_k * E_k$
    Compute interaction of user and item:
    $\mathcal{X}_{i,j,j'} = \ln \sigma(U_i^T V_j - U_i^T V_{j'})$
    Compute score of entity-relation quadruple:
    $\mathcal{Y}_{h,r,t,t'} = \ln \sigma(f_r(h,t) - f_r(h,t'))$
    Compute regularization:
    $\mathcal{Z} = \|U_i\|, \|E_{\{h,t,t'\}}\|, \|R_r\|, \|B_{\{j,j'\}}\|, \|W_{\{j,j'\}}\|$
    maximize $\mathcal{X}_{i,j,j'} + \mathcal{Y}_{h,r,t,t'} + \mathcal{Z}$
Predictions:
for each $u_i \in U$ **do**
  Recommend items for user i in order
  $j_1 > j_2 > ... > j_n$ ($U_i^T V_{j1} > U_i^T V_{j2} > ... > U_i^T V_{jn}$).

4.1 Dataset

To demonstrate the effectiveness of proposed method, we collected *GitHub* dataset and conduct experiments on it. The *GitHub* dataset is chosen for several reasons. Firstly, the user feedback is implicit which is more realistic in real-world recommendation. Besides, the *GitHub* dataset is quantitatively sparse but semantically dense, the dataset consists of $3,798$ users, $2,477$ items and $22,096$ interactions. Defining the density ratio as $iteration\_num/(user\_num * item\_num)$, the ratio of *GitHub* dataset is 0.0026. In contrast, the popular MovieLens-1M dataset has a density ratio of 0.0119 even if only 5-star ratings are considered. Though the *GitHub* dataset is quantitatively sparse, it is semantically dense, the repositories are highly related with each other based on their semantic information including some simple entity-based interactions, such as some repositories use programming language "C++" or some repositories use toolkit "TensorFlow". We also leverage some complex interactions from knowledge graph, for example, a repository uses toolkit "TensorFlow" which is implemented by "C++" which is the programming language of another repository. We do recommendation on *GitHub* dataset not only based on historical cooccurrence of items but also based on semantic information enhanced by knowledge graph. The other reason we use *GitHub* dataset is that the items (repositories) can't be directly mapped to entities of knowledge graph

because of its highly customized item name. Although directly mapping is used in some previous works, it fails in recommendation tasks where item names are customized.



(a) HCE TransR MAP@k (b) HCE TransR Recall@k

(c) HCE RESCAL MAP@k (d) HCE RESCAL Recall@k

Fig. 2. MAP@k and Recall@k result.

4.2 Baselines

We choose following methods as baselines of our experiment, **BPRMF** (Bayesian Personalized Ranking based Matrix Factorization), **BPRMF+TransE** and **FM** (Factorization Machines).

BPRMF ignores the knowledge graph information, it only focuses on historical user feedback, the results are learnt by using pairwise item ranking based matrix factorization.

BPRMF+TransE uses almost the same setting as our proposed models (RESCAL-based HCE and TransR-based HCE), while it only considers part of knowledge graph information. By using TransE knowledge graph embedding method, it ignores the multi-relational data.

FM [12, 13] is another popular solution for integrating side information into recommendation tasks. While it is limited by only considering the entities as items' features and ignoring the semantic structural relation between entities.

4.3 Comparison

To measure both the precision and recall of recommendation results, we use MAP@k (mean average precision) [6] and Recall@k [5] in our experiments. Due to utilizing two knowledge graph embedding methods, RESCAL and TransR, in proposed Hierarchical Collaborative Embedding (HCE) model, we compare RESCAL-based HCE and and TransR-based HCE with baselines (BPRMF, FM, BPRMF+TransE) respectively.

Each experiment is repeated five times with different random seeds and we report the MAP and Recall values by varying the position k in Fig. 2. The results can be summarized as follows: 1) Results of FM model are better than BPRMF, because BPRMF totally ignores knowledge graph information, knowledge graph information is useful to improve the recommendation results. 2) The improvement of FM is limited, less than BPRMF+TransE model, because FM model doesn't consider relation structure of knowledge graph, integrating knowledge graph structured embedding in our proposed model by using knowledge conceptual level effectively elevates MAP@k and Recall@k scores. 3) Although BPRMF+TransE model is effective, it is still outperformed by both RESCAL-based HCE model and TransE-based HCE model, because the latter two models consider the multi-relational data of knowledge graph.

The effectiveness of proposed Hierarchical Collaborative Embedding (HCE) framework is presented. Knowledge Conceptual Level serves as the core component of HCE framework appropriately.

5 Related work

In this section, we introduce two related works, Knowledge graph structured embedding and Implicit Collaborative Filtering. Knowledge graph structured embedding leverages relational learning methods [10] to extract the latent semantic information of knowledge graph elements including entities and relations. Collaborative filtering learns users' interests from their feedback, either explicit or implicit.

5.1 Knowledge Graph Structured Embedding

Based on different assumptions, each structured embedding method proposes a model to represent knowledge graph triple which consists of head entity, relation and tail entity. There are three categories of models, direct vector space translating, vector space translating with relation subspace or hyperplane mapping, and tensor factorization. Considering of knowledge graph is a multi-relational heterogeneous network, Bordes et al. [2] then proposed another model using direct vector space translating model, which ignore multi-relation problem but make the model much more efficient in training speed. Nickel et al. [9] proposed a new type of relational learning methods based on tensor factorization, which is efficient in both speed and accuracy. Lin et al. [8] use relation subspace mapping instead of hyperplane mapping. Except the models mentioned above, there are some other structured embedding models [3, 20]. In this work, we integrate [2] into our framework as one baseline, and we use [9] and [8] as important components of our proposed model.

5.2 Collaborative Filtering using Implicit Feedback

Popularized by the *Netflix prize*[7], traditional methods focus on explicit feedback such as *ratings*. However, the last decade has seen a growing trend towards exploiting implicit feedback such as *clicks* and *purchases*. Implicit feedback has a major advantage of eliminating the needs of asking users explicitly. Instead, user feedback is collected silently, resulting more user-friendly recommender systems. Hu et al. [7] and Pan et al. [11] investigated item recommendation from implicit feedback and propose to impute all missing values with zeros. More recently, Shi et al. [16] and Bayer et al. [1] extended Bayesian Personalized Ranking (BPR) [14] for optimizing parameters from implicit feedback. In this paper, we employ an optimization strategy similar to BPR, but with semantic information modeling. Standard BPR methods are also used as baselines for our experiments.

6 Conclusions

In this paper, we proposed Hierarchical Collaborative Embedding framework, which integrates recommender system with knowledge graph into a three-level model. The information of knowledge graph is leveraged to improve the results of quantitatively sparse but semantically dense recommendation scenarios. Experiment was conducted on real-world *GitHub* dataset showing that semantic information from knowledge graph has been properly captured, resulting improved recommendation performance. To the best of our knowledge, this is the first attempt of using knowledge graph embedding to perform semantic enhancement for items that do not exist in knowledge graph, by using the proposed Knowledge Conceptual Level. LS as well. For future work, we would like to add additional layers to the network to capture higher order interactions of items and entities.

7 Acknowledgement

The authors thank the reviewers for their helpful comments. This work was partially supported by the Major Research Plan of National Science Fundation of China [No. 91630206].

References

1. Immanuel Bayer, Xiangnan He, Bhargav Kanagal, and Steffen Rendle. A generic coordinate descent framework for learning from implicit feedback. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1341–1350. International World Wide Web Conferences Steering Committee, 2017.
2. Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795, 2013.

[7] `http://www.netflixprize.com/`

3. Antoine Bordes, Jason Weston, Ronan Collobert, Yoshua Bengio, et al. Learning structured embeddings of knowledge bases. In *AAAI*, volume 6, page 6, 2011.
4. Robin Burke, Fatemeh Vahedian, and Bamshad Mobasher. Hybrid recommendation in heterogeneous networks. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 49–60. Springer, 2014.
5. Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *International Conference on Machine Learning*, pages 233–240, 2006.
6. Sander Dieleman and Benjamin Schrauwen. Deep content-based music recommendation. In *International Conference on Neural Information Processing Systems*, pages 2643–2651, 2013.
7. Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 263–272. IEEE, 2008.
8. Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, pages 2181–2187, 2015.
9. Maximilian Nickel. *Tensor factorization for relational learning*. PhD thesis, lmu, 2013.
10. Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2015.
11. Rong Pan, Yunhong Zhou, Bin Cao, Nathan N Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. One-class collaborative filtering. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 502–511. IEEE, 2008.
12. Steffen Rendle. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 995–1000. IEEE, 2010.
13. Steffen Rendle. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(3):57, 2012.
14. Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pages 452–461. AUAI Press, 2009.
15. Chuan Shi, Zhiqiang Zhang, Ping Luo, Philip S Yu, Yading Yue, and Bin Wu. Semantic path based personalized recommendation on weighted heterogeneous information networks. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 453–462. ACM, 2015.
16. Yue Shi, Alexandros Karatzoglou, Linas Baltrunas, Martha Larson, Nuria Oliver, and Alan Hanjalic. Climf: learning to maximize reciprocal rank with collaborative less-is-more filtering. In *Proceedings of the sixth ACM conference on Recommender systems*, pages 139–146. ACM, 2012.
17. Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM international conference on Web search and data mining*, pages 283–292. ACM, 2014.
18. Xiao Yu, Xiang Ren, Yizhou Sun, Bradley Sturt, Urvashi Khandelwal, Quanquan Gu, Brandon Norick, and Jiawei Han. Recommendation in heterogeneous information networks with implicit user feedback. In *ACM Conference on Recommender Systems*, pages 347–350, 2013.
19. Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 353–362. ACM, 2016.
20. Zili Zhou, Guandong Xu, Wenhao Zhu, Jinyan Li, and Wu Zhang. Structure embedding for knowledge base completion and analytics. In *International Joint Conference on Neural Networks*, pages 737–743, 2017.