Unified Collaborative Filtering over Graph Embeddings

Pengfei Wang
wangpengfei@bupt.edu.cn
School of Computer Science
Beijing University of Posts and Telecommunications

Hanxiong Chen
hanxiong.chen@rutgers.edu
Department of Computer Science
Rutgers University

Yadong Zhu
edgewind11@gmail.com
Mobvista

Huawei Shen
shenhuawei@ict.ac.cn
Institute of Computing Technology
Chinese Academy of Sciences

Yongfeng Zhang*
yongfeng.zhang@rutgers.edu
Department of Computer Science
Rutgers University

ABSTRACT

Collaborative Filtering (CF) by learning from the wisdom of crowds has become one of the most important approaches to recommender systems research, and various CF models have been designed and applied to different scenarios. However, a challenging task is how to select the most appropriate CF model for a specific recommendation task. In this paper, we propose a Unified Collaborative Filtering framework based on Graph Embeddings (UGrec for short) to solve the problem. Specifically, UGrec models user and item interactions within a graph network, and *sequential recommendation path* is designed as a basic unit to capture the correlations between users and items. Mathematically, we show that many representative recommendation approaches and their variants can be mapped as a recommendation path in the graph. In addition, by applying a carefully designed attention mechanism on the recommendation paths, UGrec can determine the significance of each sequential recommendation path so as to conduct automatic model selection. Compared with state-of-the-art methods, our method shows significant improvements for recommendation quality. This work also leads to a deeper understanding of the connection between graph embeddings and recommendation algorithms.

CCS CONCEPTS

• **Information systems → Retrieval tasks and goals**; *Collaborative filtering*; *Recommender systems*;

KEYWORDS

Recommender Systems; Collaborative Filtering; Sequential Recommendation Path, Attention Mechanism, Model Selection

ACM Reference Format:
Pengfei Wang, Hanxiong Chen, Yadong Zhu, Huawei Shen, and Yongfeng Zhang. 2019. Unified Collaborative Filtering over Graph Embeddings. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3331184.3331224

*This is the corresponding author

1 INTRODUCTION

Recommender systems have become an important research area with the success of collaborative filtering methods over the decades, and a lot of work has been conducted in both industry and academia. Broadly speaking, most of the frequently used collaborative filtering algorithms can be classified as either non-sequential models [14, 19, 26] or sequential models [10, 27, 35, 41], and they have achieved important success in different recommendation scenarios.

The non-sequential approach learns a user's preference on each individual item so as to calculate the matching score between the user and an item, and then ranks the items according to the scores to generate a recommendation list. One of the most representative methods that falls into this category is Matrix Factorization (MF) [19] as well as its deep generalizations such as Neural Collaborative Filtering (NCF) [14], which learn the user and item vector representations and calculate the matching score based on vector product or a prediction network. Since user/item representations are usually learned based on all of the historical information, the related methods are good at capturing the general tastes of users by aggregating a user's complete purchase history.

Sequential approaches, on the other hand, attempt to capture the transition relationship between two or more adjacent items in user purchasing sequences so as to learn the sequential dynamics between items, and the transition relationships can be captured by, for example, factorizing the transition matrix [10, 27] or recurrent neural models [15, 31, 41]. Recently, researchers have also explored models that can capture both the user general preferences and the item dependencies for recommendation [6, 33, 35, 37, 40] so as to improve the recommendation performance.

Though these approaches are widely used in many real-world systems, each recommendation method may only be suitable for a certain type of recommendation scenario, and practitioners sometimes have to implement and compare different models so as to understand the their nature and to select the best one. The different modeling strategies of different methods raise an interesting yet important question: Given a specific recommendation scenario or a target user, which recommendation approach or specific model is more suitable to generate good recommendations? Manually testing each recommendation model so as to select the best one would require significant efforts.

In this paper, we propose a Unified Collaborative Filtering framework over Graph Embeddings (UGrec for short) to solve this problem. UGrec models user and item interactions in a graph network, and it embeds both entities and relations of the graph into a low-dimensional space. Specifically, UGrec defines the *Sequential Recommendation Path* as a basic unit to capture the correlation between users and items. In this way, their implicit interactions can be turned into a collection of explicitly paths. Mathematically, we show that many representative recommendation approaches or their variants can be mapped as different sequential recommendation paths in the graph. Based on these paths, UGrec utilizes an attention mechanism to determine the importance of the paths for each user-item pair, and more interestingly, we show that this is actually conducing automatic model selection on user-item pair level. Overall, the key contributions of our work can be summarized as follows:

- We propose a unified collaborative filtering framework over graph embeddings to learn the user and item interactions for recommendation. Based on this framework, we design sequential recommendation paths to capture the inherent relationships between users and items explicitly.
- Mathematically, we show that each type of sequential recommendation path in our model corresponds to one particular recommendation method. As a result, our model has the ability to unify several representative recommendation methods, including both sequential and non-sequential approaches.
- We propose an attention mechanism to measure the significance of each sequential recommendation path for each user-item pair. In this way, UGrec has the power to conduct automatic model selection on per user-item pair level, which provides good explainability about which model is more important to predict the relevance for a user-item pair.
- Empirically we show that our model can consistently outperform state-of-the-art baselines under different metrics for personalized recommendation, including sequential, non-sequential, joint-modeling, and graph-based baselines.

The following part of this paper is organized as follows. We firstly discuss related work in section 2, and in section 3 we introduce our framework. Experimental results and discussions are provided in section 4, and we conclude the work and highlight the future research directions in section 5.

2 RELATED WORK

In this section, we provide a brief overview of the related work from three perspectives, including the non-sequential approach, sequential approach, and the graph-based recommendation methods.

2.1 Non-sequential Recommendation

Non-sequential approach is one of the classical approaches to recommendation systems. Due to its long-time research history and the wide scope of literature, it is hardly possible to cover all non-sequential recommendation algorithms, so we review some representative methods in this subsection. Overall, non-sequential approaches try to learn the user-item matching scores to generate recommendation list. Early approaches consider the user-item rating matrix and conduct rating prediction with user-based [18] or item-based [23, 28] collaborative filtering methods. In these methods, user and item rating vector is considered as the representation vector for each user and item.

With the development of dimension reduction methods, latent factor models such as matrix factorization are later widely adopted in recommender systems, such as singular value decomposition , non-negative matrix factorization, and probabilistic matrix factorization,etc. In these approaches, each user and item is learned as a latent factor representation to calculate the matching score of each user-item pair.

Recently, the development of deep learning and neural network models has further extended collaborative filtering methods for recommendation. The relevant methods can be broadly classified into two sub-categories: similarity learning approach, and representation learning approach. The similarity learning approach adopts simple user/item representations (e.g., from one-hot) and learns a complex prediction network as the similarity function to calculate user-item matching scores [13, 14], while the representation learning approach learns rich user/item representations and adopts a simple similarity function (e.g., inner product) for matching score calculation [34, 44, 46].

Another important research direction is learning to rank for recommendation, which learns the relative ordering of items instead of the absolute preference scores. The most representative method on this direction is perhaps Bayesian personalized ranking [26], which is a pair-wise learning to rank method. It is also further generalized to take other information sources such as images [12].

2.2 Sequential Recommendation

To capture the sequential dynamic relationship between items, sequential recommendation leverages user historical records in an ordered way for future behavior prediction and recommendation.

By integrating matrix factorization and Markov chains, factorized personalized Markov chains (FPMC) [27] embeds the transition information between adjacent behaviors into the item latent factors for recommendation, and the hierarchical representation model (HRM) [35] further extends the idea by leveraging representation learning as latent factors. These methods mostly model the local sequential patterns between every two adjacent records [41].

To model multiple-step sequential behaviors, [10] adopted Markov chains to provide recommendations with sparse sequences, and [41] proposed the dynamic recurrent basket model (DREAM) to capture global sequential patterns and to learn dynamic user interest representations based on recurrent neural network (RNN). Similarly, [15, 31] leveraged users' previous clicking and purchasing behaviors to model short-term preferences with RNN for session-based recommendation. On the other hand, [9] adopted a metric space learning approach to learn additive user-item relations for sequential recommendation, and the method is further generalized to factorization machines for sequential recommendation [25]. Recently, researchers have also explored memory networks [4] and knowledge graphs [17] for sequential recommendation.

Beyond recommendation in e-commerce, sequential recommendation has also been applied to various application scenarios such as POI recommendation [5, 7], music recommendation [3, 8, 36], and browsing recommendation [45].

2.3 Graph Embedding-based Recommendation

Because we leverage graph embeddings for unified collaborative filtering in this work, we also provide a review of the graph-based approaches to recommendation. Graph-based recommendation explores different types of relationships in heterogeneous networks to improve the recommendation performance [2, 20, 30, 44]. Yu et al. [42] took advantage of different types of entity relationships in heterogeneous information network and proposed a personalized recommendation framework for implicit feedbacks. Hu et al. [16] proposed a two-phase framework to utilize the heterogenous social networks for improving the effectiveness of offline sales. Shi et al. proposed a heterogeneous information network embedding based approach to utilize auxiliary information in networks for recommendation [29]. All of these work demonstrated that not all the connections in a heterogeneous social network are useful for recommendation. Recently translation-based embedding approach is widely applied to large-scale knowledge graphs [1, 21]. For example, TransE [1] considers the relationships by interpreting them as a translation operation over low-dimensional embeddings of the entities. Inspired by this assumption, He et al. [9] introduced a translation-based method to model the semantic relationships between entities for recommendation, and Xie et al. [38] learned graph-based POI embeddings for location-based recommendation. Vahedian et al. [32] explored a random walk sampling approach and applied it to generate extended meta-paths in weighted heterogeneous networks for recommendation.

To the best of our knowledge, the most related work with ours is the translation-based method to model third-order relationships for large-scale sequential prediction [9]. However, this approach only exploits the direct connections between user and item to predict the potential relations between entities, while in our approach, we build connections between a pair of entities by considering multiple-hop

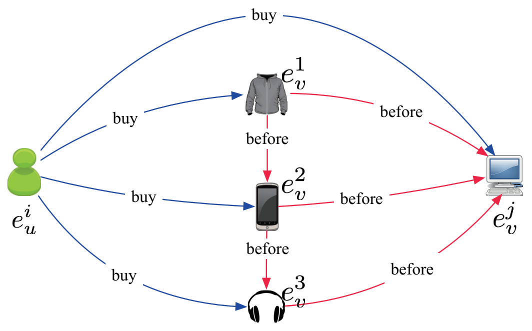

Figure 1: An example of the graph network between user e_u^i and item e_v^j. e_v^1, e_v^2, e_v^3 and e_v^j are all items that user e_u^i purchased, where $t(e_u^i, e_v^1) < t(e_u^i, e_v^2) < (e_u^i, e_v^3) < t(e_u^i, e_v^j)$. Let r_1 and r_2 denote *buy* and *before* relations, respectively, then $\{r_1\}$, $\{r_1, e_v^1, r_2\}$,$\{e_v^1, r_2\}$, $\{r_1, e_v^2, r_2\}$, $\{e_v^2, r_2\}$, $\{r_1, e_v^3, r_2\}$, $\{e_v^3, r_2\}$, $\{r_1, e_v^2, r_2, e_v^3, r_2\}$,$\{e_v^2, r_2, e_v^3, r_2\}$, $\{r_1, e_v^1, r_2, e_v^2, r_2\}$,$\{e_v^1, r_2, e_v^2, r_2\}$, $\{r_1, e_v^1, r_2, e_v^2, r_2, e_v^3, r_2\}$, and $\{e_v^1, r_2, e_v^2, r_2, e_v^3, r_2\}$ are the recommendation paths between user e_u^i and item e_v^j.

sequential recommendation paths, and each type of recommendation path represents a particular recommendation algorithms, so that the framework unifies several representative algorithms and conducts model selection automatically.

3 THE FRAMEWORK

In this section, we first introduce the problem formalization of personalized recommendation in graph network, and then describe the proposed UGrec framework in detail. In particular, we analyze how to map previous recommendation approaches to specific recommendation paths in the graph. We finally present the model learning and prediction procedure.

3.1 Notations and Definitions

For easy understanding, we present the basic notations and definitions that will be used throughout the following task. Let e_u^i and e_v^j denote the i-th user and j-th item, respectively, we construct a directed graph with e_u^i being the starting node and e_v^j being the ending node. The graph involves two types of relations, which are *buy* and *before*. The *buy* relation (denoted as r_1) connected a user and an item, which means that a user purchased an item, and the *before* relation (denoted as r_2) connects two items, which means that the user purchased one item before another item. We use $\mathcal{E}$ to denote the set of all the entities in the graph, and use $\mathcal{R} = \{r_1, r_2\}$ to denote the set of relations.

Moreover, we use $t(e_u^i, e_v^j)$ to represent the timestamp that user e_u^i purchased item e_v^j. Notations used throughout the paper are summarized in Table 1. For a triple (e_u^i, p, e_v^j), we use p to represent the *Sequential Recommendation Path* connecting e_u^i and e_v^j in the graph network, which is defined as follows:

Definition 1. ***Sequential Recommendation Path** for a triple (e_u^i, p, e_v^j). A Sequential Recommendation Path (SRP for short) $p = \{r_1, e_v^1, r_2, e_v^2, ..., r_2, e_v^k, r_2\}$ is a series of relations and entities that*

Table 1: A summary of key notations in this work.

Notation	Explanation
e_u^i, e_v^j	Entities corresponding to the i-th user and the j-th item respectively
$\mathcal{E}$	The set of entities
r_1, r_2	relations corresponding to *buy* and *before*
$\mathcal{R}$	The set of relations $\{r_1, r_2\}$
p	A Sequential Recommendation Path
$\mathcal{P}$	The set of paths for a user-item pair
$\vec{e}_u^i, \vec{e}_v^j, \vec{p}, \vec{r}_1, \vec{r}_2$	Embedding vectors of user, item, path, and relations
$\mathcal{S}$	The set of all observed user-item pairs
$\mathcal{S}'$	The set of negative sampling user-item pairs
$Pr(p \mid e_u^i, e_v^j)$	The attention function to analyze the significance of the selected path p
$D(e_u^i, e_v^j)$	The distance function for a pair (e_u^i, e_v^j)
$d(e_u^i, p, e_v^j)$	The distance function for a triple (e_u^i, p, e_v^j)

connect the user e_u^i and the item e_v^j, we use $|p|$ to represent the order of path p, which is the total number of entities and relations on the path. Given any two entities $\{e_v^m, e_v^n\} \in p$, where e_v^m and e_v^n represent the m-th and n-th entitiy in p, they keep the sequential property: $\{t(e_u^i, e_v^m) < t(e_u^i, e_v^n) | m < n\}$, which means that if the path flows through e_v^m before e_v^n, then e_v^m should be purchased earlier than e_v^n. We use this property to keep the time ordering of the entities in the path.

Figure 1 gives an example of all the sequential recommendation paths between e_u^i and e_v^j. As we can see, based on the definition of the *Sequential Recommendation Path*, we can remove most of the illogical paths. For example, path $\{r_1, e_v^3, r_2, e_v^2, r_2, e_v^1, r_2\}$ violates the sequential property. In this way, we can reduce the analytic space and focus on valid paths that reflect meaningful relationships between users and items.

Given the notations and definitions introduced above, our task is to select the most accurate model to predict which items that a user will most likely purchase.

3.2 UGrec

In this section, we introduce our Unified Collaborative Filtering framework over Graph Embeddings (UGrec) in detail. UGrec models the connections between users and items based on *Sequential Recommendation Path* mined from users' purchase histories. In order to analyze the significance of each sequential recommendation path, UGrec utilizes an attention mechanism to determine the significance of each path. Based on this, a meticulously designed translation learning objective function is finally used for model inference and prediction. Figure 2 shows the architecture of UGrec.

Specifically, UGrec learns over the entity set $\mathcal{E}$ and the relation set $\mathcal{R}$, and encodes both entities and relations into a low-dimensional embedding space. More formally, let $\vec{e}_u^i$ denote the user entity vector, $\vec{e}_v^j$ denote the item entity vector, and $\vec{p}$ denote the path representation. For each triple (e_u^i, p, e_v^j), we borrow the idea of TransE [1] and consider the sequential recommendation path p as a translation vector $\vec{p}$ between the two entity vectors $\vec{e}_u^i$ and $\vec{e}_v^j$. More formally, if e_u^i prefers e_v^j according the sequential recommendation path p, what we want is $\vec{e}_u^i + \vec{p} \approx \vec{e}_v^j$, which means that according to some distance metric $d(\cdot)$, $\vec{e}_v^j$ should be close to $\vec{e}_u^i + \vec{p}$. More generally, the distance will be small if the triple (e_u^i, p, e_v^j) holds; otherwise, the distance will be large. In this paper, we use the Euclidean distance to measure the distance between $\vec{e}_u^i + \vec{p}$ and $\vec{e}_v^j$, which is defined as follows:

$$d(e_u^i, p, e_v^j) = \frac{1}{2}\left\|\vec{e}_u^i + \vec{p} - \vec{e}_v^j\right\|^2 \tag{1}$$

For a pair (e_u^i, e_v^j), given all sequential recommendation paths connecting e_u^i and e_v^j, the distance between e_u^i and e_v^j is further written as:

$$D(e_u^i, e_v^j) = \sum_{p \in \mathcal{P}} Pr(p|e_u^i, e_v^j) \cdot d(e_u^i, p, e_v^j) \tag{2}$$

where $\mathcal{P}$ represents the set of sequential recommendation paths that connect e_u^i and e_v^j. Because the sequential recommendation paths are not equally reliable, in this paper, we use $Pr(p|e_u^i, e_v^j)$ to indicate the significance of the sequential recommendation path p, which is calculated as follows:

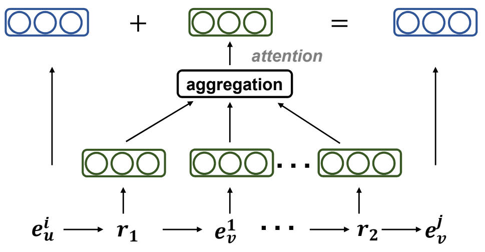

Figure 2: The conceptual architecture of UGrec. UGrec uses ADD operation to obtain the embedding of a sequential recommendation path, and then an attention mechanism is utilized to calculate the significance of the selected path.

$$Pr(p|e_u^i, e_v^j) = \frac{e^{-d(e_u^i, p, e_v^j)}}{\sum_{p' \in \mathcal{P}} e^{-d(e_u^i, p', e_v^j)}} \tag{3}$$

As we can see, $Pr(p|e_u^i, e_v^j)$ can be considered as an attention score applied on top of path p to adjust its significance, which helps our model to learn the importance of each path. Formally, given the sequential recommendation path p, we define a composition operation to obtain the path embedding, which is written as $\vec{p} = \vec{r}_1 \circ \vec{e}_v^1 \cdots \circ \vec{r}_2$. For simplicity, to obtain the representation of p, we consider the ADD operation [22] to sum up the vectors of all relations and entities, which is formalized as:

$$\vec{p} = \vec{r}_1 + \sum_{i=1}^{k} (\vec{r}_2 + \vec{e}_v^i) \tag{4}$$

In this way, given a set of pairs $\mathcal{S} = \{(e_u^i, e_v^j)\}$, the objective function of UGrec is formalized as follows:

$$\begin{aligned} L_{UGrec} &= \sum_{(e_u^i, e_v^j) \in \mathcal{S}} \Bigg\{ \sum_{(e_u^-, e_v^j) \in \mathcal{S}'} \Big[\gamma + D(e_u^i, e_v^j) - D(e_u^-, e_v^j)\Big]_+ \\ &+ \sum_{(e_u^i, e_v^-) \in \mathcal{S}'} \Big[\gamma + D(e_u^i, e_v^j) - D(e_u^i, e_v^-)\Big]_+ \Bigg\} \\ s.t. &\quad \|e\|^2 = 1, \forall e \in \mathcal{E} \end{aligned} \tag{5}$$

where $\gamma > 0$ is a margin hyper parameter, $[x]_+ = max(0; x)$ returns the maximum between 0 and x, $\mathcal{S}'$ is set of negative triples:

$$\mathcal{S}' = \{(e_u^-, e_v^j)\} \cup \{(e_u^i, e_v^-)\} \tag{6}$$

namely, for each observed pair $(e_u^i, e_v^j) \in \mathcal{S}$, we replace the user (or item) entity to some other entity e_u^- (or e_v^-) that is not connected to the original item (or user), which are sampled according the uniform distribution. Intuitively, the objective function will favor lower distance scores for valid triples compared with those invalid triples, and similar to TransE [1], the ℓ_2-norm constraint on entity embeddings help to prevent the training process to trivially minimize L by artificially increasing entity embeddings.

3.3 Connections with Recommendation Models

In this section, we discuss the connection between the UGrec framework and previous recommendation models. We show that many representative recommendation algorithms, including both sequential and non-sequential approaches, can be considered as special cases of UGrec when $\vec{r}_1 = \vec{r}_2 = \mathbf{0}$ (no translation), $Pr(p|e_u^i, e_v^j) = 1$

(equal attention), and γ=0 (no marginal loss). In this way, each recommendation algorithm corresponds to a certain type of sequential recommendation path in the UGrec model.

3.3.1 *Matrix Factorization (1-order).*

To show that UGrec can reduce to certain types of non-sequential approach, we only keep the 1-order sequential recommendation path $\{r_1\}$. In this way, there is only one sequential recommendation path for each (e_u^i, e_v^j) pair. We further set $\mathcal{S}'$=$\emptyset$, and refer to this reduced model as UGrec$_{mf}$, whose objective function is then written as follows:

$$\begin{aligned} L_{mf} &= \sum_{(e_u^i, e_v^j)\in\mathcal{S}} \left[\gamma + D(e_u^i, e_v^j)\right]_+ = \sum_{(e_u^i, e_v^j)\in\mathcal{S}} D(e_u^i, e_v^j) \\ &= \sum_{(e_u^i, e_v^j)\in\mathcal{S}} \frac{1}{2}\left(\|\vec{e}_u^i\|^2 + \|\vec{e}_v^j\|^2 - 2\vec{e}_u^i\cdot\vec{e}_v^j\right) \end{aligned} \tag{7}$$

In UGrec, because the ℓ_2-norm of the embedding for each entity is 1, then the objective function is further written as $L_{mf} = \sum_{(e_u^i, e_v^j)\in\mathcal{S}} \left(1 - \vec{e}_u^i\cdot\vec{e}_v^j\right)$. As we can see, in this case UGrec is actually factorizing a user-item interaction matrix to maximize the similarity between positive user-item pairs, where the user-item interaction is a binary matrix.

3.3.2 *Bayesian Personalized Ranking (1-order).*

For each pair (e_u^i, e_v^j), if we only sample one negative pair (e_u^i, e_v^-), the model is referred to as UGrec$_{bpr}$, and the objective function becomes:

$$\begin{aligned} L_{bpr} &= \sum_{(e_u^i, e_v^j)\in\mathcal{S}} \sum_{(e_u^i, e_v^-)\in\mathcal{S}'} D(e_u^i, e_v^j) - D(e_u^i, e_v^-) \\ &= \sum_{(e_u^i, e_v^j)\in\mathcal{S}} d(e_u^i, \{r_1\}, e_v^j) - d(e_u^i, \{r_1\}, e_v^-) \\ &= \sum_{(e_u^i, e_v^j)\in\mathcal{S}} \left(\vec{e}_u^i\cdot\vec{e}_v^- - \vec{e}_u^i\cdot\vec{e}_v^j\right) \end{aligned} \tag{8}$$

In this way, UGrec$_{bpr}$ is actually conducting pair-wise learning to rank similar to Bayesian Personalized Ranking (BPR).

3.3.3 *Factorized Markov Chains (2-order).*

We leave out user embeddings and only consider the item to item transitions, and take the 2-order path $p = \{e_v^k, r_2\}$ while setting $\mathcal{S}'$=$\emptyset$, then UGrec with this new architecture is denoted as UGrec$_{fmc}$, whose reduced objective function is as follows:

$$\begin{aligned} L_{fmc} &= \sum_{(e_u^i, e_v^j)\in\mathcal{S}} D(e_u^i, e_v^j) = \sum_{(e_u^i, e_v^j)\in\mathcal{S}} \frac{1}{2}\|\vec{e}_v^k - \vec{e}_v^j\|^2 \\ &= \sum_{(e_u^i, e_v^j)\in\mathcal{S}} \frac{1}{2}(\|\vec{e}_v^k\|^2 + \|\vec{e}_v^j\|^2 - 2\vec{e}_v^k\cdot\vec{e}_v^j) \\ &= \sum_{(e_u^i, e_v^j)\in\mathcal{S}} (1 - \vec{e}_v^k\cdot\vec{e}_v^j) \end{aligned} \tag{9}$$

where $e_v^k \in p$. In this case, UGrec$_{mc}$ reduces to a factorized Markov chain model.

3.3.4 *Three-order Models.*

We analyze the loss function of UGrec under 3-order paths. Specifically, we take the 3-order path $\{r_1, e_v^k, r_2\}$, and for each pair (e_u^i, e_v^j), we also sample one negative pair (e_u^i, e_v^-). The new architecture is denoted as UGrec$_3$, and the new objective function can be written as :

Table 2: Relationships between sequential recommendation path and representative recommendation models.

Sequential Recommendation Path	Reduced Model	Path Order
$(e_u^i, p = \{r_1\}, e_v^j)$	MF, BPR	$\|p\|$=1
$(p = \{e_v^k, r_2\}, e_v^j)$	FMC	$\|p\|$=2
$(e_u^i, p = \{r_1, e_v^k, r_2\}, e_v^j)$	FPMC, TransRec	$\|p\|$=3
$(e_u^i, p = \{r_1, e_v^1, r_2, \cdots, e_v^k, r_2\}, e_v^j)$	-	$\|p\| > 3$

$$\begin{aligned} L_3 &= \sum_{(e_u^i, e_v^j)\in\mathcal{S}} \sum_{(e_u^i, e_v^-)\in\mathcal{S}'} \left[D(e_u^i, e_v^j) - D(e_u^i, e_v^-)\right]_+ \\ &= \frac{1}{2}\sum_{(e_u^i, e_v^j)\in\mathcal{S}} \left(\|\vec{e}_u^i + \vec{p} - \vec{e}_v^j\|^2 - \|\vec{e}_u^i + \vec{p} - \vec{e}_v^-\|^2\right) \\ &= \sum_{(e_u^i, e_v^j)\in\mathcal{S}} \left((\vec{e}_u^i + \vec{e}_v^k)\cdot\vec{e}_v^- - (\vec{e}_u^i + \vec{e}_v^k)\cdot\vec{e}_v^j\right) \end{aligned} \tag{10}$$

The 3-order path p only contains one item entity e_v^k, where $t(e_u^i, e_v^k) < t(e_u^i, e_v^j)$ according to the sequential property. In this case, UGrec$_3$ is similar to FPMC [27] and TransRec [9]. Compared with these models, UGrec$_3$ also consists of two key components, which are the inner product of the user and item latent factors, and the inner product of the latent factors between the previous and the next item. The only difference is that FPMC and TransRec apply Bayesian learning on top of L_3 for model optimization.

3.3.5 *Selection of Distance Metrics.*

In fact, if we design different distance metrics when selecting different sequential recommendation paths, the models mentioned above can be exactly equivalent to a special case of UGrec. Take Matrix Factorization as an example, if we use the following distance metric to replace Equation (1), then according to Equation (7), Matrix Factorization can be exactly mapped into the 1-order sequential recommendation path.

$$d(e_u^i, p, e_v^j) = \frac{1}{2}\|\vec{e}_u^i + \vec{p} - \vec{e}_v^j\|_F^4 = \frac{1}{2}(1 - 2\vec{e}_u^i\cdot\vec{e}_v^j)^2 \tag{11}$$

However, considering the simplicity and generalization ability of our framework, we use Euclidean distance as the unique distance metric for optimization in this work. Based on the above analysis, we can see that UGrec is a general framework that has the ability to subsume several known recommendation methods, and it also includes a lot of new methods (higher-order recommendation paths) that we do not know before. By applying attentions on different recommendation paths, we can aggregate the power of multiple recommendation models to produce the best prediction.

3.4 Learning and Prediction

In order to learn the parameters of the UGrec model, we employ stochastic gradient descent (SGD) to minimize the loss function. For each entity, we constrain the ℓ_2-norm of the embeddings as 1, which prevents the training process to trivially minimize the loss by artificially increasing entity embeddings norms [1]. We then

Table 3: Statistics of the datasets used in our experiments.

Dataset	Type	#Users	#Items	#*SRP*
Cell Phones	E-commerce	27,879	10,429	4,769,020
Cloths	E-commerce	39,387	23,033	7,232,142
Beauty	E-commerce	22,363	12,101	6,445,670
Foursquare	POI	1,083	38,333	836,635
MovieLens 1M	Movie	6,040	3,706	2,982,507

randomly select a valid triple from the training set for learning. However, in the learning procedure, we observe two key challenges.

One is that for each pair (e_u^i, e_v^j), there is a large number of paths in the graph. Though we define the *sequential recommendation path* to limit paths, the number of valid paths are still beyond computational power. Thus in our learning procedure, we limit the order of the selected sequential recommendation paths to approximate the original objective. Specifically, we consider 4 types of paths mined from the graph network, as shown in Table 2, and for the last type, the maximum order is 10.

The other challenge is that the performance of our model may not always be stable. The reason lies in the random initialization of the entities and relations, because in the first a few iterations, the entity and relation representations have not been well optimized, and directly using Eq.(3) to calculate the attention weights becomes unreliable to the model. To solve the problem, we adopt the *burn-in* training strategy: (1) In the first n_b training iterations, for each pair (e_u^i, e_v^j), we set $Pr(p|e_u^i, e_v^j) = \frac{1}{|\mathcal{P}|}$, where $|\mathcal{P}|$ is the number of sequential recommendation paths, i.e., we take all the paths equally important in the first n_b iterations. (2) After the burn-in period, we then calculate the significance of each sequential recommendation path according to Eq. (3) in the following iterations.

With the learned entities and relations, given a user e_u^i and a candidate item e_v^j, we calculate $D(e_u^i, e_v^j)$ according to Equation (2), and then construct the top-N recommendation list by ranking the items in descending order of the distance.

4 EXPERIMENT

In this section, we evaluate our proposed models by comparing with many state-of-the-art methods. We begin by introducing the experimental setup, and then analyze the experimental results.

4.1 Dataset

We evaluate different recommendation algorithms over five datasets, including three e-commerce recommendation datasets, one POI recommendation dataset, and one movie rating dataset.

- **Amazon** comprises large corpora of reviews and timestamps on various products[1]. The dataset is from Amazon.com and spans from May 1996 to July 2014. We adopt three sub-datasets of diverse size and sparsity, which are Clothing, Cell Phone, and Beauty [11, 24].
- **Foursquare** includes a large number of user check-ins at different venues from April 12, 2012 to February 16, 2013. In this work we use the check-in data in New York city[2] [39].
- **MovieLens 1M** is a popular recommendation dataset that contains 1,000,209 anonymous ratings of movies made by MovieLens users who joined MovieLens in 2000[3].

For each dataset, we select the first 70% of the interactions from each user based on timestamp to construct the training set, and adopt the remaining 30% for testing. In this way we hope to simulate the real-world scenario where we use known user behaviors to predict future behaviors. Details of the datasets are shown in Table 3.

4.2 Baselines

We take the following representative and state-of-the-art methods as baselines for performance comparison:

- **MF**: Matrix Factorization, which is a model-based collaborative filtering method[4].
- **FMC**: Factorized Markov Chain, which captures the global sequential dynamics by factorizing the item-to-item transition matrix [27].
- **PER**: Personalized Entity Recommendation [42], which treats the structural knowledge as a heterogeneous information network and extracts meta-path based latent features to represent the connectivity between users and items.
- **FPMC**: FPMC [27] uses a predictor that combines Matrix Factorization and Factorized Markov Chains so that personalized Markov behaviors can be captured.
- **HRM**: HRM [35] extends FPMC by aggregation operations (e.g., max pooling) to model complex user-item and item-item interactions.
- **TransRec**: A unified method that models the user preferences and sequential dynamics based on translations [9].
- **DREAM**: A hybrid model that leverages recurrent neural network to model the dynamic representation of users and the sequential relationship between items [41].

4.3 Evaluation Metrics

For evaluation, we adopt the widely used F1-measure, Hit, and NDCG as our evaluation metrics. We provide top-N recommendation list for each user in the testing set, where N=10. We performed significant tests using the paired t-test. Differences are considered statistically significant when the p−value is lower than 0.05.

4.4 Parameter Settings

All the embedding vectors are randomly initialized in the range of $(0, 1)$. For fair comparison, we select the best learning rate for each method in the range of {0.0001, 0.001, 0.01, 0.05, 0.1}, and vector dimension is tuned in the range of $\{100, 150, 200, 250, 300, 350\}$. We update them by conducting stochastic gradient descent (SGD). For PRE, we use the "user-item-user-item" formatted meta-path features, which is also adopted in [42, 43]. For HRM, we use max pooling strategy and set the drop rate as 0.5. For TransRec, we used the ℓ_2-normalization. For UGrec[5], we set γ as 0.1, and the number of negative samples $\kappa = 5$ in Eq.(5).

[1]http://jmcauley.ucsd.edu/data/amazon/

[2]https://www.kaggle.com/chetanism/foursquare-nyc-and-tokyo-checkin-dataset

[3]http://files.grouplens.org/datasets/movielens/

[4]https://pypi.org/project/mflib/

[5]Code is published at `anonymous site`.

Table 4: Performance on Top-10 recommendation between the baselines and our model (all the values in the table are percentage numbers with % omitted). The best performance in each row is underlined, and the starred numberes represent best baseline performance. The last column shows the absolute improvement of our results against the best baseline, which is significant at p-value≤0.05.

Dataset	Metric	Non-Sequential		Sequential Model				Graph-Based Model		Improve
		MF	PER	FMC	FPMC	HRM	DREAM	TransRec	UGrec	
Beauty	F1-measure	1.53	1.69	1.85	2.11	2.11	2.24*	2.09	3.29	1.05
	Hit	10.1	13.4	13.1	14.3	14.5	15.7*	14.1	20.2	4.50
	NDCG	4.87	5.57	6.12	7.31	7.54	8.12*	7.51	10.7	2.58
Cell Phone	F1-measure	2.93	3.03	3.13	3.33*	3.26	3.32	3.24	4.53	1.21
	Hit	7.68	10.7	20.1	21.3	22.3	23.2*	21.1	26.6	3.40
	NDCG	3.67	5.98	8.44	10.1	11.5	12.4*	10.4	14.5	2.10
Clothing	F1-measure	0.95	1.02	1.52	2.05	2.14	2.35*	2.01	3.81	1.46
	Hit	7.46	9.76	18.2	19.2	20.1	20.7*	19.3	23.4	2.70
	NDCG	1.79	2.61	9.57	10.9	11.1	12.1*	10.1	13.4	1.30
Foursquare	F1-measure	1.32	1.63	3.81	4.12	4.52	4.60*	4.12	5.73	1.13
	Hit	20.3	24.5	50.2	53.4	54.4	56.7*	52.7	60.6	3.90
	NDCG	15.2	18.4	26.7	35.6	37.1	37.6*	36.1	39.2	1.60
MovieLens 1M	F1-measure	2.63	2.77	1.53	3.44	3.82	3.92*	3.33	5.39	1.46
	Hit	36.5	38.2	22.5	42.5	43.3	45.2*	42.2	50.1	4.90
	NDCG	14.9	15.9	9.78	16.9	18.3	19.2*	17.1	22.8	3.60

4.5 Comparison against Baselines

We compare our UGrec model to the baseline methods on recommendation task. The performance results are shown in Table 4.

We first analyze the performance of non-sequential models (MF and PER) on these datasets. We see that PER shows a better performance than MF on all the evaluation metrics. The reason lies in that compared with MF, PER can diffuse the user preferences based on the pre-defined meta-paths, and based on this, PRE has the ability to build complex types of connections between users and items, instead of the direct user-item interactions used in MF.

We then make comparison between sequential and non-sequential models. It can be seen that FMC performs better than MF and PER on the Beauty, Cell Phone, Clothing, and Foursquare datasets, while performs worse on MovieLens 1M dataset. We assume the potential reason may be that Makov chain based recommendation is more suitable in E-commerce and POI recommendation scenarios, where the short-term relationship between items is more important, while in the Movie recommendation scenario, users' long-term tastes on movie could be more important to provide movie recommendation, which is consistent with the previous findings such as [23].

We can also see that FPMC performs better than FMC in almost all cases. Because FPMC considers both the user-item preferences and the item-item dynamic relationships, this observation shows that jointly modeling the two perspectives helps to improve the recommendation performance, which is a good supportive evidence for the motivation of this work. This is actually not a surprising observation, and similar results have been shown in previous work [27, 35, 41].

We also found that HRM performs better than FPMC and TransRec on all datasets. The reason might be that for FPMC and TransRec, both of them assume that the sub-models as equally important, while HRM considers the interaction between two sub-models based on the max-pooling strategy. However, the max-pooling aggregation strategy adopted by HRM is applied on embedding-dimension level, and this makes it difficult to interpret which recommendation sub-model takes the most significant effect for the prediction. Besides, the max-pooling strategy is difficult to tune when considering more sub-models. By using recurrent neural network (RNN) to capture longer sequential behaviors, DREAM considers more and much longer interactions between users and items, and obtains better performance than HRM, FPMC and TransRec among the baselines.

Finally, by designing the sequential recommendation paths to model the various types of interaction between users and items, UGrec has the ability to consider a much wider scope of recommendation algorithms than previous models, and it also removes most of the illogical paths between users and items. In addition, UGrec further applies an attention mechanism on each sequential recommendation path, so that it has the ability to learn the significance of each sub-model to conduct automatic model selection on per user-item pair level. Experimental results also show that UGrec outperforms all of the baselines in terms of all the evaluation measures on the five datasets. Take the Beauty dataset as an example, when compared with the best baseline (i.e. DREAM), the performance improvement of UGrec in terms of absolute value is around 1.08%, 4.50%, and 2.58% on F1, Hit, and NDCG, and the relative percentage improves are 46.9%, 28.7%, and 31.8%, respectively.

4.6 Analysis of Model Learning

As we adopted two learning strategies to approximate our objective function so as to overcome the challenges in terms model trainingIn this section, we analyze the effect of these two learning strategies based on the experimental datasets.

Table 5: Performance of UGrec in terms of F1-score along with the increasing order of SRPs on the five datasets. All numbers in the table are percentage values with '%' omitted.

Dataset	=1	<=2	<=3	<=5	<=7	<=9
Beauty	1.63	2.52	2.89	2.13	3.22	3.29
CellPhone	3.31	4.12	4.32	4.45	4.51	4.53
Clothing	1.13	2.39	3.22	3.65	3.72	3.81
Foursquare	1.43	4.58	5.32	5.57	5.72	5.73
MovieLens 1M	2.89	4.23	4.67	4.87	5.12	5.31

4.6.1 ***Limiting the SRP Order.*** In UGrec we use Sequential Recommendation Path (SRP) to describe the relations between users and items. There are usually a large amount of SRPs for each entity pair, and it is impractical to enumerate all the possible paths. In order to guarantee the computational efficiency of our model, UGrec limits the order of the sequential recommendation paths mined from the graph network. Here we analyze the performance of UGrec when using different orders. Specifically, for each pair (e_u^i, e_v^j), we first test the performance of UGrec by only applying the first-order sequential recommendation path on the five datasets. Then we gradually increase the order of sequential recommendation paths used in UGrec. By this we would like to check how the performance changes along with the increasing order of sequential recommendation paths. In this experiment, we tried path order $|p|=\{1, 3, 5, 7, 9\}$ over five datasets, and the performance of UGrec in terms of F1-measure is reported in Table 5.

We see that as the sequential recommendation path order increases, the recommendation performance in terms of F1-score also increases, and the observation is consistent on all five datasets. The improvement is quite significant when the order increases from 1 to 2. This result is reasonable since when mining sequential recommendation paths with order less than 3, both sequential approaches and non-sequential approaches are considered by UGrec, and thus UGrec can conduct model selection to improve the performance.

When considering higher-order sequential recommendation paths (e.g. $|p| > 5$), we found that the performance gain between two consecutive trials decreases, the reason may be that the lower-order sequential recommendation paths already contain sufficient information for recommendation, while higher-order paths are less reliable in describing the user and item interactions. It indicates that if we continue to mine more high-order paths, it will increase the computational complexity but will only bring very small performance improvements.

4.6.2 ***Burn-in Training Policy.*** To learn the proposed UGrec, we utilize the burn-in procedure for optimization. One parameter in this procedure is the number of burn-in iterations, denoted as n_b. Here we study the impact of n_b on the final performance. More specifically, we tried $n_b \in \{0, 100, 200, 300, 400, 500\}$ on the Beauty dataset (other datasets have similar results), and Figure 3 shows the test performance of UGrec in term of F1 against the number of burn-in interactions.

As we can see, when setting $n_b = 0$, UGrec does not obtain satisfactory performance. It means that directly using Eq.(3) to assign the significance of each path is not a good strategy in the initial learning procedure. After a "burn-in" period, we can see

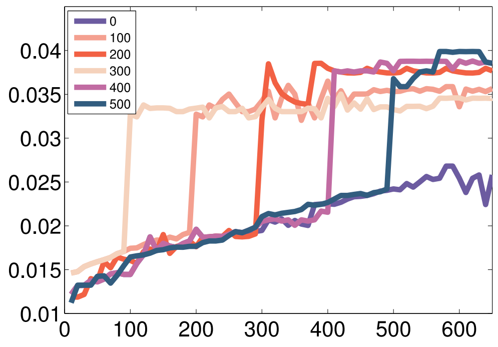

Figure 3: Performance variation in terms of F1-measure (vertical axis) against the number of burn-in iterations (horizontal axis) on the Beauty dataset.

that the performance in terms of F1 increases significantly, and the reason is that after the period, UGrec has obtained relatively stable embeddings of the user and item entities. Based on this, the learning strategy of UGrec then changes from $Pr(e_v^i, e_v^j) = 1$ to Eq.(3), UGrec can continue to adjust the attention of different paths for each user and item pair. By paying more attention to the significant sequential recommendation paths, the performance of UGrec improves significantly. This observation, on the other hand, further verifies that not all of the sequential recommendation paths are equally important for user and item pairs in a graph network [16, 29].

As the number of burn-in iterations n_b increases, the performance improvement is less and less significant. For example, when we increase n_b from 200 to 300, the relative performance improvement in terms of F1 is about 0.03%. It indicates that after 200 iterations, we have obtained relatively stable entity and relation representations, and if we continue to burn more iterations, it will bring less performance improvement but larger computational complexity. Therefore, we set n_b as 200 on the Beauty dataset and other datasets in our experiment.

4.7 Analysis over Different Type of SRPs

In this subsection, we study the performance of the special cases of UGrec introduced in Section 3.3. Specifically, we adopt the three types of sequential recommendation paths shown in Table 2, which are $\{r_1\}$, $\{e_v^k, r_2\}$, and $\{r_1, e_v^k, r_2\}$ that reduce to 1-order (MF/BPR), 2-order (FMC), and 3-order (FPMC/TransRec) models, respectively. For 1-order model, we take the matrix factorization setting shown in Section 3.3.1. We then run our UGrec model on each of the three path types separately, and the corresponding models are denoted as $UGrec_{mf}$, $UGrec_{fmc}$, and $UGrec_3$, respectively. Performance of the three models are shown in Table 6.

As we can see, $UGrec_{fmc}$ is better than $UGrec_{mf}$ in most cases, while $UGrec_3$ obtains the best performance compared with $UGrec_{fmc}$ and $UGrec_{mf}$. This observation is consistent with the result in Table 4. It verifies that by selecting specific order of sequential recommendation paths in modeling, the simplified version of UGrec model ($UGrec_{mf}$, $UGrec_{fmc}$, and $UGrec_3$) is able to obtain similar performance with the corresponding conventional recommendation model (MF, FMC, and FPMC/TransRec). The results also show

Table 6: Performance of UGrec when selecting different types of sequential recommendation paths on five datasets. All numbers are percentage numbers with '%' omitted.

Dataset	Metric	$UGrec_{fmc}$	$UGrec_{mf}$	$UGrec_3$
Beauty	F1-measure	1.63	1.71	2.34
	Hit	10.9	12.5	14.6
	NDCG	5.32	6.98	7.33
Cell Phone	F1-measure	3.31	3.23	3.42
	Hit	7.93	19.8	21.5
	NDCG	3.82	8.32	10.6
Clothing	F1-measure	1.12	2.03	2.42
	Hit	7.53	18.4	19.9
	NDCG	2.12	9.33	10.7
Foursquare	F1-measure	1.43	3.35	4.43
	Hit	21.1	50.3	53.4
	NDCG	16.7	28.3	36.5
MovieLens	F1-measure	2.89	1.35	3.76
	Hit	37.1	22.7	38.9
	NDCG	15.2	9.91	17.1

that although UGrec maps these conventional recommendation approaches to sequential recommendation paths in an approximate manner based on a unique distance metric, it can still achieve comparable results with the corresponding conventional model by learning over the same relation type between users and items.

4.8 Analysis of Automatic Model Selection

In this section, we analyze the significance of different types of sequential recommendation path over each recommendation dataset, so as to understand how UGrec conduct automatic model selection by learning the attention weights of different paths.

Specifically, for each (e_u^i, e_v^j) pair that UGrec recommends correctly in the testing set, we identify the order of the highest-attention sequential recommendation path. Based on this, we calculate the percentage of each order. For example, if there are 10 correctly recommended pairs, and among the 10 recommendations, 2-order path was learned as the highest attention path for 3 times, then the percentage of 2-order path will be 0.3. The percentage distributions on five datasets are shown in Figure 4.

As we can see, 2-order sequential recommendation path captures the most significant attention on Amazon and Foursquare datasets, while on the MovieLens dataset, 1-order path captures the most significant attention. This observation is interesting and is consistent with the previous experiments. It shows that sequential behavior is important for e-commerce recommendation and POI prediction tasks, because users' next purchase is usually influenced by what they recently purchased, and the next place to go usually depends on where they are currently located in, and similar observations are also shown in previous literature [7, 23, 27]. Since 2-order paths have the ability to capture the sequential dynamic relationship between items, UGrec automatically learns higher attention scores for 2-order paths.

However, users' preference on movie types are usually relatively stable, as a result, the general preference plays a more important role in the movie recommendation scenario. Since 1-order sequential recommendation path has the ability to match each item with a user's overall preference embedding, UGrec learns higher attention scores for 1-order paths in the movie recommendation task. Overall, experimental results imply that UGrec is able to learn appropriate attention weights for the sequential recommendation paths so as to conduct automatic model selection for a particular recommendation scenario.

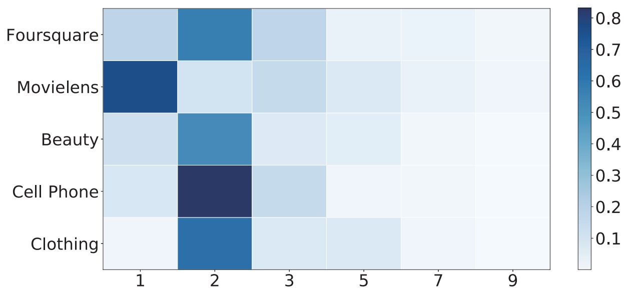

Figure 4: Order distribution on the five datasets. x-axis denotes the order of the highest-attention sequential recommendation path. Each cell indicates the frequency of the corresponding order of sequential recommendation paths.

5 CONCLUSION

In this paper, we propose a unified collaborative filtering framework over graph embeddings (UGrec) for personalized recommendation and automatic model selection. UGrec formalizes the user and item interactions in a graph network, and adopts sequential recommendation paths to represent the correlations between users and items. Mathematically, we show that many conventional recommendation algorithms can be approximately mapped as a certain type of such sequential recommendation path. In addition, UGrec leverages an attention mechanism to determine the significance of each sequential recommendation path so as to conduct model selection automatically. Compared with state-of-the-art methods, our framework offers significant improvement in terms of recommendation performance. Further analysis on the attention mechanism also shows that UGrec can learn reasonable attention weights for model selection according to a specific recommendation scenario.

UGrec is a flexible framework, except for the entity and relation types considered in this work, we can also take many other entity and relation types into consideration. Besides, though we analyzed how several representative recommendation algorithms can be covered by the UGrec framework, there are still many other interesting models that we did not explore. Interestingly, UGrec could have the ability to cover these models when more entities types are added into the graph network, and we will further analyze the relationship between these models and our framework in the future.

6 ACKNOWLEDGE

This research work was supported by the National Natural Science Foundation of China under Grant No.61802029, and the fundamental Research for the Central Universities under Grant No.500419741. We would like to thank the anonymous reviewers for their valuable comments.

REFERENCES

[1] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data *(NIPS'13)*. Curran Associates Inc., USA, 2787–2795.

[2] Robin Burke. 2000. Knowledge-Based Recommender Systems. In *ENCYCLOPEDIA OF LIBRARY AND INFORMATION SYSTEMS*. Marcel Dekker, 2000.

[3] Shuo Chen, Josh L Moore, Douglas Turnbull, and Thorsten Joachims. 2012. Playlist prediction via metric embedding. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 714–722.

[4] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 108–116.

[5] Chen Cheng, Haiqin Yang, Michael R Lyu, and Irwin King. 2013. Where You Like to Go Next: Successive Point-of-Interest Recommendation.. In *IJCAI*, Vol. 13. 2605–2611.

[6] Robin Devooght and Hugues Bersini. 2017. Long and short-term recommendations with recurrent neural networks. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*. ACM, 13–21.

[7] Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, Yeow Meng Chee, and Quan Yuan. 2015. Personalized Ranking Metric Embedding for Next New POI Recommendation. In *Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI'15)*. AAAI Press, 2069–2075.

[8] Negar Hariri, Bamshad Mobasher, and Robin Burke. 2012. Context-aware music recommendation based on latenttopic sequential patterns. In *Proceedings of the sixth ACM conference on Recommender systems*. ACM, 131–138.

[9] Ruining He, Wang-Cheng Kang, and Julian McAuley. 2017. Translation-based Recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems (RecSys '17)*. ACM, New York, NY, USA, 161–169. https://doi.org/10.1145/3109859.3109882

[10] Ruining He and Julian McAuley. 2016. Fusing similarity models with markov chains for sparse sequential recommendation. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE, 191–200.

[11] Ruining He and Julian McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. *CoRR* abs/1602.01585 (2016). arXiv:1602.01585

[12] Ruining He and Julian McAuley. 2016. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback.. In *AAAI*. 144–150.

[13] Xiangnan He, Xiaoyu Du, Xiang Wang, Feng Tian, Jinhui Tang, and Tat-Seng Chua. 2018. Outer Product-based Neural Collaborative Filtering. *IJCAI* (2018).

[14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[15] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. *ICLR* (2016).

[16] Qingbo Hu, Sihong Xie, Jiawei Zhang, Qiang Zhu, Songtao Guo, and Philip S. Yu. 2016. HeteroSales: Utilizing Heterogeneous Social Networks to Identify the Next Enterprise Customer. In *Proceedings of the 25th International Conference on World Wide Web (WWW '16)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 41–50.

[17] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 505–514.

[18] Joseph A Konstan, Bradley N Miller, David Maltz, Jonathan L Herlocker, Lee R Gordon, and John Riedl. 1997. GroupLens: applying collaborative filtering to Usenet news. *Commun. ACM* 40, 3 (1997), 77–87.

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

[20] Pigi Kouki, Shobeir Fakhraei, James Foulds, Magdalini Eirinaki, and Lise Getoor. 2015. HyPER: A Flexible and Extensible Probabilistic Framework for Hybrid Recommender Systems. In *Proceedings of the 9th ACM Conference on Recommender Systems (RecSys '15)*. ACM, New York, NY, USA, 99–106. https://doi.org/10.1145/2792838.2800175

[21] Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2015. Modeling Relation Paths for Representation Learning of Knowledge Bases. *CoRR* abs/1506.00379 (2015). arXiv:1506.00379

[22] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI'15)*. AAAI Press, 2181–2187.

[23] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon.Com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Computing* 7, 1 (Jan. 2003), 76–80.

[24] Julian J. McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-based Recommendations on Styles and Substitutes. *CoRR* abs/1506.04757 (2015). arXiv:1506.04757

[25] Rajiv Pasricha and Julian McAuley. 2018. Translation-based factorization machines for sequential recommendation. In *Proceedings of the 12th ACM Conference on Recommender Systems*. ACM, 63–71.

[26] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI '09)*. AUAI Press, Arlington, Virginia, United States, 452–461.

[27] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing Personalized Markov Chains for Next-basket Recommendation. In *Proceedings of the 19th International Conference on World Wide Web (WWW '10)*. ACM, New York, NY, USA, 811–820.

[28] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. ACM, 285–295.

[29] Chuan Shi, Binbin Hu, Wayne Xin Zhao, and Philip S. Yu. 2017. Heterogeneous Information Network Embedding for Recommendation. *CoRR* abs/1711.10730 (2017). arXiv:1711.10730

[30] Y. Sun and J. Han. 2013. Meta-path-based search and mining in heterogeneous information networks. *Tsinghua Science and Technology* 18, 4 (August 2013), 329–338. https://doi.org/10.1109/TST.2013.6574671

[31] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. ACM, 17–22.

[32] Fatemeh Vahedian, Robin Burke, and Bamshad Mobasher. 2017. Weighted Random Walk Sampling for Multi-Relational Recommendation. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization (UMAP '17)*. ACM, New York, NY, USA, 230–237. https://doi.org/10.1145/3079628.3079685

[33] Kiewan Villatel, Elena Smirnova, Jérémie Mary, and Philippe Preux. 2018. Recurrent Neural Networks for Long and Short-Term Sequential Recommendation. *arXiv preprint arXiv:1807.09142* (2018).

[34] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1235–1244.

[35] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning Hierarchical Representation Model for NextBasket Recommendation. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '15)*. ACM, New York, NY, USA, 403–412.

[36] Xiang Wu, Qi Liu, Enhong Chen, Liang He, Jingsong Lv, Can Cao, and Guoping Hu. 2013. Personalized next-song recommendation in online karaokes. In *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 137–140.

[37] Liang Xiang, Quan Yuan, Shiwan Zhao, Li Chen, Xiatian Zhang, Qing Yang, and Jimeng Sun. 2010. Temporal recommendation on graphs via long-and short-term preference fusion. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 723–732.

[38] Min Xie, Hongzhi Yin, Hao Wang, Fanjiang Xu, Weitong Chen, and Sen Wang. 2016. Learning graph-based poi embedding for location-based recommendation. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 15–24.

[39] Dingqi Yang, Daqing Zhang, Vincent. W. Zheng, and Zhiyong Yu. 2015. Modeling User Activity Preference by Leveraging User Spatial Temporal Characteristics in LBSNs. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45, 1 (2015), 129–142.

[40] Koren Yehuda. 2010. Collaborative Filtering with Temporal Dynamics. *Commun. ACM* 53, 4 (April 2010), 89–97.

[41] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. A Dynamic Recurrent Model for Next Basket Recommendation. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '16)*. ACM, New York, NY, USA, 729–732.

[42] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. 2014. Personalized Entity Recommendation: A Heterogeneous Information Network Approach. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining (WSDM '14)*. ACM, New York, NY, USA, 283–292.

[43] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative Knowledge Base Embedding for Recommender Systems *(KDD '16)*. ACM, New York, NY, USA, 353–362.

[44] Yongfeng Zhang, Qingyao Ai, Xu Chen, and W. Bruce Croft. 2017. Joint Representation Learning for Top-N Recommendation with Heterogeneous Information Sources. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM '17)*. ACM, New York, NY, USA, 1449–1458.

[45] Yongfeng Zhang, Min Zhang, Yiqun Liu, Chua Tat-Seng, Yi Zhang, and Shaoping Ma. 2015. Task-based recommendation on a web-scale. In *Big Data (Big Data), 2015 IEEE International Conference on*. IEEE, 827–836.

[46] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 425–434.