A Novel Embedding Model for Relation Prediction in Recommendation Systems

Yu ZHAO†+a), Student Member, Sheng GAO†b), Patrick GALLINARI††, and Jun GUO†††, Nonmembers

SUMMARY It inevitably comes out information overload problem with the increasing available data on e-commerce websites. Most existing approaches have been proposed to recommend the users personal significant and interesting items on e-commerce websites, by estimating unknown rating which the user may rate the unrated item, i.e., rating prediction. However, the existing approaches are unable to perform user prediction and item prediction, since they just treat the ratings as real numbers and learn nothing about the ratings’ embeddings in the training process. In this paper, motivated by relation prediction in multi-relational graph, we propose a novel embedding model, namely RPEM, to solve the problem including the tasks of rating prediction, user prediction and item prediction simultaneously for recommendation systems, by learning the latent semantic representation of the users, items and ratings. In addition, we apply the proposed model to cross-domain recommendation, which is able to realize recommendation generation in multiple domains. Empirical comparison on several real datasets validates the effectiveness of the proposed model. The data is available at https://github.com/yuzhaour/da.

key words: recommendation system, latent factor model, relation prediction, cross-domain recommendation

1. Introduction

It inevitably comes out information overload problem with the increasing available data on e-commerce websites (e.g., Amazon, Taobao). Consequently, the users always are trapped by endless choices. For example, can Jack easily pick a favorite T-shirt from tens of thousands of styles of T-shirts on Amazon? Maybe not. To cope with the problem, most existing approaches [14], [15], [23], [28] have been proposed to recommend the users personal significant and interesting items on e-commerce websites, by estimating unknown rating which the user may rate the unrated item, i.e., rating prediction as \((\text{user}, \text{rating}=?, \text{item})\). Basically, the more high rating the user may rate the item, the more possible that the item would be recommended to the user. For example in Fig. 1, Amazon could proactively recommend Jack a new red T-shirt which he may rate high, taking advantage of recommendation systems. However, the existing approaches are unable to perform user prediction and item prediction, since they just treat the ratings as real numbers and learn nothing about the ratings’ embeddings (Embedding denotes the latent semantic representation.) in the training process [14], [15]. User prediction refers to the problem how to predict unknown users with given rating and item, i.e., \((\text{user}=?, \text{rating}, \text{item})\), while item prediction is similarly defined. For example in user prediction task, for investigating the popularity of the item red T-shirt, can we estimate the total number of unknown users who may rate the red T-shirt maximum rating value 5, i.e., \((\text{user}=?, \text{rating}=5, \text{item}=\text{red T-shirt})\)? User prediction and item prediction are challenge tasks due to lack of the ratings’ embeddings, and as meaningful as rating prediction. In this paper, we attempt to solve the problem including the tasks of rating prediction, user prediction and item prediction simultaneously for recommendation systems.

The problem can be reasonably considered as the relation prediction problem in multi-relational graph. Multi-relational graph involves different types of relationships between entities and consists of a vast amount of relations, each of which is formed as a triple like \((\text{subject entity}, \text{predicate relationship}, \text{object entity})\). Relation prediction refers to the generic problem how to predict unseen relations, i.e., \((?, \text{predicate relationship}, \text{object entity})\), \((\text{subject entity}, ?, \text{object entity})\) and \((\text{subject entity}, \text{predicate relationship}, ?)\). For example, in the relation prediction task in knowledge graph [3], the relationship between “Barack Obama” and “United States” is missing in the knowledge graph, can we still predict the relationship which probably should be between them? It sounds impossible at the first glance, but it is possible by utilizing existing knowledge facts for pre-
A novel model (RPEM) is proposed to learn the latent semantic representations for all users, items and ratings from the users’ rating history, which is the first study to realize rating prediction (user, rating=?, item), user prediction (user=?, rating, item) and item prediction (user, rating, item=?) simultaneously.

We present a contrastive max-margin algorithm for optimization, meanwhile, we propose a new approach for selecting the negative samples with Negative Candidate Probability Distribution.

We apply the proposed model to cross-domain recommendation, which is able to realize recommendation generation in multiple domains.

Empirical comparison validates the effectiveness of the proposed model.

In the rest of the paper, we first show related work in Sect. 2, and then present the model and learning algorithm in Sects. 3 and 4 respectively. In Sect. 5, we show the application in cross-domain recommendation. In Sect. 6, we show several experiments on real data sets. We finally conclude in Sect. 7.

2. Related Work

Recommendation systems. The recommendation algorithms can be divided into three classes: content-based [26], collaborative filtering (CF) [22], [30] and hybrid recommendation algorithms [1]. The content-based approaches analyze the content feature of the items and the attributes of the users, and then measure similarity between the users and items for recommendation, regardless of the users’ rating/selection history. However, this kind of approaches will not work well when there are not sufficient content features of items and attributes of users. In comparison, collaborative filtering (CF) approaches only use the user rating/selection history, considering the correlativity of users or items, and they do not exploit the content features of items and attributes of users. Consequently, they suffer from the sparsity problem if rating/selection history is short. CF algorithms can be further divided into two categories: memory-based CF [23], Item-KNN [28] and model-based CF [30]. The hybrid approaches combine the content-based and CF methods to avoid their limitations.

The cross-domain recommendation systems generally aim to exploit knowledge from source-domain to perform or improve recommendations in target-domain. Some models have been proposed to provide jointly diverse recommendations of items belonging to multiple domains [17]. According to the strategies of exploiting knowledge, the cross-domain recommendation approaches can be classified into two categories: 1) aggregating knowledge [24] and 2) linking and transferring knowledge [9], [27]. The approaches by aggregating knowledge from various source domain to perform recommendations in target domain include three cases: merging user preferences [21], mediating user modeling data [18], [24] and combining recommendations [10], and the models by linking and transferring knowledge between domains include three variants: linking domains [12], sharing latent features [13], [25] and transferring rating patterns [9], [20], [27].

Relation prediction. For relation prediction in multi-relational graph, such as Google Knowledge Graph which consist of a vast amount of knowledge facts, each of which is formed as a triple like (left-entity, relationship, right-entity), several embedding-based models [3], [22], [31] have been proposed recently. These encode the entities and relationships of the triples into latent semantic space for relation prediction. Bordes et al. [4] propose to transform the two entities into a common latent space by the corresponding left and right projection matrices of the relationship. Socher et al. [29] propose a model to encode each relationship into a tensor, where each slice of the relationship tensor is responsible for one class of entity pair.

Embedding learning. In Natural Language Processing (NLP), many language models [19] have been proposed for learning semantic knowledge from a large number of unstructured free text corpus, as encoding each word (phrase or sentence) to a semantic vector representation, namely word embedding. Word embedding can be used for various NLP tasks such as POS tagging, chunking, named entity recognition, semantic and syntactic similarity [6].

3. Approach

Problem Statement. First, we define the rating prediction
problem, which is how to predict the unknown rating with the given user and item, in the form of \( (u, r = ?, i) \), based on the users’ rating history. It can be formulated as a ranking problem, i.e., predicting a total order \( \geq u, i \in U \times I \) over users and items in terms of ratings. \( U, I \) and \( R \) to denote the set of users, items and ratings. Given the scoring function \( G : U \times R \times I \rightarrow \mathbb{R} \), so the most probable predicted rating \( (r^*) \) with the test triple \((u, r = ?, i)\) is defined as follows:

\[
r^* := \arg \min_{r \in \mathbb{R}} G(u, r, i),
\]

where \( G(\cdot) \) is the model scoring function. Similarly, for user prediction, we get a list of predicted Top-N users with the test triple \((u = ?, r, i)\) as follows:

\[
\text{Top}(u, r, i) := \arg \min_{u \in U} G(u, r, i).
\]

For item prediction, we obtain a list of predicted Top-N items \( \text{Top}(u, r, N) \) with test triple \((u, r, i = ?)\) in a similar way.

**Relation Prediction Embedding Model (RPEM).** For relation prediction in recommendation systems, we propose a new embedding model, namely RPEM, to encode the relations \((u, r, i)\), learning the latent semantic representations of the users, items and ratings. In general, entities (i.e., users and items) and relationships (i.e., ratings) are completely different objects in a multi-relational graph, it may be not capable to represent them in a common semantic space. So the basic idea of modeling the relation \((u, r, i)\) is to encode entities (i.e., users and items) and relationships (i.e., ratings) in distinct spaces, i.e., entity space and relationship spaces, and performs translation in relationship space.

More specifically, in RPEM, for each triple \((u, r, i)\), \( u \in U, r \in R \) and \( i \in I \), user and item embeddings are set as \( u, i \in \mathbb{R}^k \) and rating embedding is set as \( r \in \mathbb{R}^l \), where \( k \) and \( l \) are the dimensions of the user and item embeddings respectively. Note that \( k \) and \( l \) are not necessarily identical, i.e., \( k \neq l \). For each rating, we set two projection matrices \( Mr, Nr \in \mathbb{R}^{k \times l} \). \( Mr \) projects users from user space to rating space. \( Nr \) projects items from item space to rating space. With the mapping matrices, we define the projected vectors of users and items as

\[
u_r = uM_r, i_r = iN_r.
\]

The model scoring function is defined in rating space as

\[
G(u, r, i) := \|u_r + i_r - r\|_1.
\]

with \( \forall u, i, r, \|u\|_1 \leq 1, \|i\|_1 \leq 1, \|r\|_1 \leq 1 \). The model returns a lower score if the relation \((u, r, i)\) is true in users’ rating history and a higher one otherwise.

### 4. Learning

In this section, we present the learning algorithm for RPEM, and we propose a new approach for selecting the negative ratings with Negative Candidate Probability Distribution (NCPD) for optimization.

**Objective.** We define the following margin-based scoring function as objective for training. The main idea is that the model scoring value of true rating triple \((u, r, i)\) should be lower than the false one \((u, r’, i)\). To learn embeddings, we minimize the objective function as follows:

\[
J = \sum_{(u, r, i) \in T} \sum_{(u, r’, i) \in T'} \max\{0, \gamma + G(u, r, i) - G(u, r’, i)\},
\]

where \( \gamma \) is the dynamic margin, \( \gamma = \gamma_0 + |r - r'| \), \( \gamma_0 \) is initial margin, \( |r - r'| \) is the abstract gap between the true rating \( r \) and false one \( r’ \). The larger the difference between the true rating and the false one, the larger the \( \gamma \). \( G(\cdot) \) is the model scoring function. \( T \) is the set of true rating triples, and \( T' := \{(u, r', i) | (u, r, i) \in T, r' \in R, r' \neq r\} \).

**Algorithm.** For learning the embeddings of the users, items and ratings, we use the stochastic gradient descent (SGD) algorithm to train the model. Algorithm 1 shows the detailed optimization algorithm. We initialize all the embeddings of users, items and ratings \((u, i, r)\), and mapping matrices \((Mr, Nr)\) with Gaussian distribution, then we perform the following procedure iteratively for a given number of iterations. First, we arbitrarily sample a small set (minibatch) of triples from the training set \( T \), and then for each positive triple in it, we select a random rating via NCPD (introduced in next) to replace the true rating. The parameters are then updated by taking a gradient step gradually. Finally, all the embeddings of users, items and ratings are normalized.

**Negative Candidate Probability Distribution.** We propose a new approach to build the corrupted fact for optimization, by replacing the true rating with incorrect one, based on Negative Candidate Probability Distribution (NCPD) instead of traditional uniform distribution. Let \( R \) be the rating set, \( r_p \) is the positive (true) rating of the given user.
and item, and corresponding negative (false) candidates $r_{c_1}, r_{c_2}, \ldots, r_{c_{|R_c|}} \in R_c$, $|R_c|$ is the size of $R_c$, where $R_c = \{r' \in R | r' \neq r_p\}$. Three factors considered in NCPD are as follows.

First, in the original users’ rating history, the ratings have quantitative information. The closer the true rating is to another one, the more likely the user will use this candidate rating to rate the item. The greater the distance between the true rating and the negative candidate, the more likely the candidate will be used as a negative fact. But when we transform the rating matrix into multi-relational graph, the relationships (ratings) become equal and parallel, losing the quantitative information. To solve the problem, we use softmax algorithm to calculate the negative candidate probability distribution $P_r$ as follows:

$$P_r(r_{c_i}) = \frac{\delta(|r_p - r_{c_i}|)}{\sum_{i=1}^{|R_c|} \delta(|r_p - r_{c_i}|)}$$

where $|r_p - r_{c_i}|$ is the abstract distance between the positive rating $r_p$ and negative one $r_{c_i}, \delta(x) = \frac{1}{1 + e^{-x}}$ is the sigmoid function. Similarly we get $P_r(r_{c_2}), P_r(r_{c_3}), \ldots$, and $P_r(r_{c_{|R_c|}})$.

Second, the user has the rating preference, indicated by the user rating frequency ratio. The greater the distance between the user true rating preference and the negative candidate preference, the more likely the candidate is a negative rating. We calculate user rating frequency statistical histogram and compute the negative candidate probability distribution $P_u$ as follows:

$$P_u(r_{c_i}) = \frac{\delta(|u_p(r_p) - u_p(r_{c_i})|)}{\sum_{i=1}^{|R_c|} \delta(|u_p(r_p) - u_p(r_{c_i})|)}$$

where $u_p(r)$ is the user preference of rating $r$,

$$u_p(r) = \frac{\text{user\_count}(r)}{\sum_{r \in R} \text{user\_count}(r)}$$

and user\_count($r$) is the rating $r$ frequency of the specific user. We get $P_u(r_{c_1}), P_u(r_{c_2}), \ldots$, and $P_u(r_{c_{|R_c|}})$ similarly.

Third, the items have the rating preference, indicated by the item rating frequency ratio. The greater the distance between the item true rating preference and the negative candidate preference, the more likely the candidate is a negative rating. We calculate item rating frequency statistical histogram and compute the negative candidate probability distribution $P_i$ as follows:

$$P_i(r_{c_i}) = \frac{\delta(|i_p(r_p) - i_p(r_{c_i})|)}{\sum_{i=1}^{|R_c|} \delta(|i_p(r_p) - i_p(r_{c_i})|)}$$

where $i_p(r)$ is the item preference of rating $r$,

$$i_p(r) = \frac{\text{item\_count}(r)}{\sum_{r \in R} \text{item\_count}(r)}$$

and item\_count($r$) is the rating $r$ frequency of the specific item. We get $P_i(r_{c_1}), P_i(r_{c_2}), \ldots$, and $P_i(r_{c_{|R_c|}})$ similarly.

Combining the three factors, we get the negative candidate probability distribution as follows,

$$\text{NCPD}(r_{c_i}) = \frac{P_r(r_{c_i}) \ast P_u(r_{c_i}) \ast P_i(r_{c_i})}{\sum_{i=1}^{|R_c|} P_r(r_{c_i}) \ast P_u(r_{c_i}) \ast P_i(r_{c_i})},$$

and we get NCPD($r_{c_1}$), NCPD($r_{c_2}$), \ldots, and NCPD($r_{c_{|R_c|}}$) similarly. Finally, we select the negative rating with Negative Candidate Probability Distribution (NCPD) to build the corrupted fact for optimization.

5. Application in Cross-Domain Recommendation

Cross-domain recommendation refers to the generic problem how to offer recommendations of items in multiple domains (i.e., source domain and target domain) [7]. For example, given Jack’s rating history in the domain of clothing (source domain), in another domain of books (target domain), could Amazon recommend Jack a certain type of book which he may like? The answer is “Yes”. For example, given Jack bought a red T-shirt and rated it high before, upon the potential assumption that Jack would be a young guy, Amazon may recommend Jack the book about magic story &lt;Harry Potter;&gt; which is very popular among young people.

Given source domain data ($U_S, R, I_S$) and target domain data ($U_T, R, I_T$), there exists some latent sharing knowledge between them for most cases. Generally the rating-pattern, which represent the users’ domain-invariant preference distribution, is considered as the latent sharing knowledge. Based on it, some cross-domain recommendation algorithms [9], [27] have been proposed to offer recommendation in multiple domains, by transferring rating-pattern from source domain to target domain. So the key problem for cross-domain recommendation is how to learning robust rating-pattern in multi-domains. In this paper, based on our proposed model RPEM, the embeddings of the rating-pattern is able to be acquired directly, as the embeddings of ratings and mapping matrices (i.e., $r$, $M_r$ and $N_r$). In fact, for obtaining rating-pattern, two approaches (off-line and on-line) are available to learn RPEM in multi-domains. With the off-line approach, the embeddings of rating-pattern ($r$, $M_r$ and $N_r$) are first learnt from source domain by RPEM, and then can be transferred to target domain. They can be either used directly in target domain or fine-tuned in the follow-up training process. With the on-line approach, the embeddings of rating-pattern are simultaneously learnt from both domains. In the following experiments, we only utilize the on-line approach for cross-domain recommendation.
6. Experiments

6.1 Evaluation of the Robustness of RPEM

In the following experiments, we investigate the robustness of the proposed model on real data.

Dataset. Movielens\(^1\) contains 1,000,209 anonymous ratings of 3,952 movies made by 6,040 MovieLens users, in which ratings are in the scale 1-5 (treated as relationship R1, R2, R3, R4, R5). We split the data set into three parts as train, validation and test set. The density of the training dataset, denoted by ML1M3.38, is 3.38%. Then we extract half of the samples and a quarter of the samples from ML1M3.38 to create the sets ML1M1.7 and ML1M0.87 with densities 1.7% and 0.87% respectively. Note that ML1M3.38, ML1M1.7 and ML1M0.87 have the same validation and test set. The statistics of the datasets are showed in Table 1.

| DENSITY | USERS | ITEMS | TRAIN | VALIDATION | TEST |
|---------|-------|-------|-------|------------|------|
| 3.38%   | 6,040 | 3,952 | 806,211 | 97,028     | 96,970|
| 1.70%   | 6,040 | 3,952 | 407,108 | 97,028     | 96,970|
| 0.87%   | 6,040 | 3,952 | 208,062 | 97,028     | 96,970|

| DATASET | ML1M3.38 | ML1M1.7 | ML1M0.87 |
|---------|----------|---------|-----------|
| R1      | 45,060 (5.59%) | R1 22,812 | R1 11,918 |
| R2      | 86,630 (10.75%) | R2 43,785 | R2 22,459 |
| R3      | 210,410 (26.12%) | R3 105,814 | R3 54,185 |
| R4      | 134,372 (16.76%) | R4 72,310 | R4 25,351 |
| R5      | 182,599 (22.65%) | R5 92,375 | R5 47,190 |

| DATASET | ML1M3.38 | ML1M1.7 | ML1M0.87 |
|---------|----------|---------|-----------|
| R1      | 5,585 (5.76%) | R1 5,585 | R1 5,585 |
| R2      | 10,429 (10.75%) | R2 10,429 | R2 10,429 |
| R3      | 25,351 (22.65%) | R3 25,351 | R3 25,351 |
| R4      | 33,744 (34.78%) | R4 33,744 | R4 33,744 |
| R5      | 21,919 (22.59%) | R5 21,919 | R5 21,919 |

| DATASET | ML1M3.38 | ML1M1.7 | ML1M0.87 |
|---------|----------|---------|-----------|
| R1      | 5,585 (5.76%) | R1 5,585 | R1 5,585 |
| R2      | 10,429 (10.75%) | R2 10,429 | R2 10,429 |
| R3      | 25,351 (22.65%) | R3 25,351 | R3 25,351 |
| R4      | 33,744 (34.78%) | R4 33,744 | R4 33,744 |
| R5      | 21,919 (22.59%) | R5 21,919 | R5 21,919 |

**Evaluation Protocol.** First, for rating prediction evaluation, we use the ranking procedure as in Formula (1). For each test relation (user, rating, item), the rating is removed and replaced by each of the ratings in turn. The scoring function of those relations are first computed by the model and then sorted by ascending order. The top-ranked rating is the final result predicted by the model. We use two metrics for evaluating the models: the mean average precision (MAP) and the mean absolute error (MAE), as follows:

\[
MAP = \frac{M}{N}, \quad MAE = \frac{\sum_{i=1}^{N} |r_i - \hat{r}_i|}{N},
\]

where \(M\) is the total times of predicting the rating correct in test data, \(N\) is the size of test data, and \(r_i\) is the true rating value and \(\hat{r}_i\) is the predicted rating. The smaller the value of MAE is, the better the model performs; whereas MAP is contrary to MAE. In addition, we calculate the MAPs of different ratings (1-5), denoted by MAP1, MAP2, MAP3, MAP4 and MAP5 respectively. Note that some compared models return a float number. We use three approaches: ROUND (e.g. 3.4→3, 3.5→4), FLOOR (e.g. 3.4→3, 3.5→3) and CEIL (e.g. 3.4→4, 3.5→4) to transform float number to integer for calculating the MAP. Experimental results indicate that the ROUND is better. For simplicity, we do not present the experimental results of FLOOR and CEIL in this paper. Second, for user (or item) prediction, we also use ranking procedure as in Formula (2). For each test relation, the user (or item) is removed and replaced by each of the users (items) in turn, and sort the scoring function by ascending order. We use the the mean of those predicted ranks and proportion of correct users ranked in the top 10 (Hits@10(%) as the evaluation metrics.

**Baseline.** For rating prediction, we compare our proposed model with the memory-based collaborative filtering methods User-KNN [23], Item-KNN [28] and two typical model-based collaborative filtering recommendation approaches: MF [2], [13], SVD++ [15], and an SVM-based multi-label classification algorithm CLR [8]. For user-KNN and item-KNN, we selected the number of neighbors \(K\) among {5,10,20}. Finally, \(K\) can be set up as 20. We use MyMediaLite\(^1\) for training MF, SVD++. We selected the learning rate among {0.00001, 0.0001, 0.001, 0.01}, clustering dimension \(k\) among {10, 50, 100}, regularization ratio among {0.01, 0.015, 0.02} on the validation sets. Finally, the parameters can be set up as {learning rate=0.001, clustering dimension \(k=50\), regularization ratio = 0.015}. For CLR, we concatenate the latent features of the user and item as its inputs, which are learnt by our proposed model in advance. For user and item prediction, we just show the results of RPEM since as far as we know traditional approaches cannot work in these two tasks\(^1\).

**Implementation.** For our model, we select the learning rate \(\lambda_u\) (for users embeddings), \(\lambda_i\) (for items embeddings) and \(\lambda_r\)

\(^1\)grouplens.org/datasets/movielens/
\(^1\)mymedialite.net/documentation/index.html
(for rating embeddings) in the stochastic gradient descent among \([0.00001, 0.0001, 0.001, 0.01]\), the initial margin \(\gamma_0\) among \([1, 2, 5, 10]\) and the embedding dimensions \(k, l\) in the range of \([20, 50, 100, 200]\) on the validation set. Finally, our model configuration can be set up as \(k = 200, l = 100, \lambda_1 = 0.0001, \lambda_2 = 0.0001, \lambda_3 = 0.00001, \gamma_0 = 2\). Note that \(\gamma_0\) is the initial margin, the final margin \(\gamma\) is adjusted to be larger than \(\gamma_0\) and the difference depends on the distance between the true rating and the false rating. The larger the rating gap, the larger the final model margin (e.g. given a true rating \(u, i, r = 4\), we construct a negative one \(u, i, r' = 1\), here we set the \(\gamma = \gamma_0 + |r - r'| = 2 + 3 = 5\). The algorithm procedure is iterated by 500 times.

**Results.** First, we evaluate our model for rating prediction in *ML1M3.38*. Observing Table 2, we can see that the MAP1 and MAP2 results of the User-KNN, Item-KNN, MF and SVD++ models are much lower than MAP3, MAP4 and MAP5, due to the fact that the R1 and R2 classes ratio (5.59%, 10.75%) of training data are much less than R3, R4 and R5 ratios (26.10%, 34.91%, 22.65%). This indicates that MAPs are sensitive to the ratings ratio of training data. The higher the ratio class in the training data, the more accurate the prediction of the rating will be in the test data. CLR algorithm performs worse on the MAP and MAE and it even breaks down on MAP2 abnormally. Since the RPEM is a classification problem, we can adjust the ratio of the five classes of training data by bootstrapping for balance (4 times R1, 2 times R2). We can see from the results that the MAPs of RPEM (balance) are closer to equilibrium approximately than others. This indicates that RPEM (balance) not only can predict the majority ratings well, but also the minority. However, the MAP and MAE for RPEM (balance) are not perfect. Finally, we train our RPEM (no balance) on original training dataset, and we see that it performs better than the other approaches on the MAP and MAE. In addition, the MAPs are also better balance compared to the other models (except RPEM (balance)). We believe that the good performance of RPEM model is due to the appropriate design of the embedding-based model. Not only learning the latent semantic representations of users and items but also the ratings, RPEM has the stronger encoding capacity compared to most other models which ignore ratings’ embedding. In addition, to the best of our knowledge, there is no existing methods which is available as baseline to realize user prediction and item prediction. Thus, we compare the RPEM with RPEM (balance) for user and item prediction. The results are shown in the Table 3. We can see that RPEM outperform RPEM (balance) on both metrics. In sum, the experiments demonstrate that our proposed model RPEM can perform rating, user and item prediction simultaneously for recommendation systems.

Besides, for investigating the robustness of RPEM, we also evaluate the models on various sparse datasets: *ML1M3.38 ML1M1.7 and ML1M0.87* by rating prediction. The detailed results are presented in the comparison map (Fig. 2). From Fig. 2, we can see that along with the decrease of density of training data, the performance of all approaches is gradually degraded. The memory-based methods (User-KNN, Item-KNN) perform worse than model-based approaches (MF, SVD++) at the beginning (*ML1M3.38*). However, when the data become more and more sparse, the memory-based collaborative filtering methods degrade more slowly than model-based approaches. The performance of memory-based methods exceeds the model-based approaches when the density falls to 0.87%. Finally, we can see that even the performance of our proposed RPEM, MF and SVD++ are comparable at the beginning, but along with the density decreasing, MF and SVD++ degrade more quickly than RPEM, the gap become larger and larger. Comparing RPEM with User-KNN and Item-KNN during the whole process, the former performs better all along. This indicates that our model is more robust than the compared models on relatively sparser data.

6.2 Evaluation in Cross-Domain Recommendation (a)

We evaluate our proposed model RPEM by rating prediction for cross-domain recommendation, as in Fig. 3 (a). This case is to recommend items in IT to users in UT in target domain by exploiting DS and DF.

**Datasets.** Book-Crossing\(^{3}\) contains 278,858 users providing 1,149,780 ratings about 271,379 books. We create a

---

1. http://www2.informatik.uni-freiburg.de/%7ecziegler/BX/
dataset by randomly extracting 3374 users and 4955 books from Book-Crossing, providing 67,497 ratings, which were randomly split as shown in Table 4. The density of the dataset, denoted by BX0.35, is 0.35%. We normalize all the BX0.35 ratings scale from (1-10) to (1-5)† for consistency. The dataset is sparser than ML1M3.38, thus, it serves as the target domain dataset. ML1M3.38 serves as the source domain dataset.

Baseline. We compare our proposed model with two typical cross-domain models: CLFM [9] and PCLF [27]. CLFM model was proposed by Gao et al. [9], and it aims to learn the common rating pattern shared across domains with the flexibility in controlling the optimal level of sharing as well as the domain-specific rating patterns of users in each domain. Ren et al. [27] propose the PCLF model that can learn the common rating pattern, and capture the domain-specific rating patterns of users and clustering of items in each domain. In the experiments, we select the numbers of both user and item clusters 40 for the CLFM model, and we select user clusters \( K = 20 \), item clusters \( T = 10 \) and domain-specific item clusters \( L_{\text{source}} = 15, L_{\text{target}} = 15 \) for the PCLF model via validation set. We train all the baseline methods using the code provided by the authors.

Implementation. For RPEM, we select the learning rate \( \lambda_u, \lambda_i, \lambda_r \) in the stochastic gradient descent among \([0.00001, 0.0001, 0.001, 0.01] \), the initial margin \( \gamma_0 \) among \([1, 2, 4, 8] \) and the embedding dimension \( k, l \) in a range of \([20, 50, 100, 200] \) on the validation set. Finally, our model configuration can be set up as \([k = 200, l = 100, \lambda_u = 0.0001, \lambda_i = 0.0001, \lambda_r = 0.00001, \gamma_0 = 4] \).

Results. From Table 5, we can see that both the MAP and MAE of RPEM model improve from 52.72% and 0.5224 to 57.84% and 0.4829 respectively. This indicates that the embeddings of rating-pattern learnt by our proposed model from source domain promote better performance for target domain recommendation. We can also see that the approach of our proposed model outperforms both the CLFM and PCLF. It indicates that the embeddings of rating-pattern learnt by our proposed model perform better than the embeddings of rating-pattern learnt by compared models. We believe the reason is that the embeddings of rating-pattern learnt by RPEM (i.e., \( r, M_{\text{U}}, N_{\text{I}} \)) has more complex and rich semantic representation than others which are learnt as a single matrix.

6.3 Evaluation in Cross-Domain Recommendation (b)

We evaluate the model by rating prediction for cross-domain recommendation, as in Fig. 3 (b). This case is to recommend items \( I_T \) to users \( U_S \). Note that this case is different from cross-domain recommendation (a). This case is real cross-domain recommendation, i.e., the recommended items belong to target domain, and the recommended users don’t belong to target domain but source domain.

Dataset. The metadata is crawled from e-commerce website JD.COM. We select 4 domains for the experiments: Personal Computer (PC), Food, Appliance and Digital Product (DP). Ratings are from 1 to 5. The PC is considered as source domain. Food, Appliance and DP are target domains respectively. So we can get three domain pairs: PC+FOOD, PC+Appliance and PC+DP. The correlation in each pair increases gradually. We create several subsets by randomly extracting users and items from each domain. None of users and items in these subsets overlaps between the source domain and target domain in training data. The validation and test data consist of users who belong to source domain and items which belong to target domain. The detailed statistics of them are shown in Fig. 4.

---

1,2→1; 3,4→2; 5,6→3; 7,8→4; 9,10→5.

---

Table 4 The statistics of ML1M3.38 (source domain) and BX0.35 (target domain) which are used for cross-domain recommendation (a).

| DATASET | ML1M3.38 | BX0.35 |
|---------|----------|--------|
| DENSITY | 3.38%    | 0.35%  |
| USERS   | 6,040    | 3,374  |
| ITEMS   | 3,952    | 4,955  |
| TRAIN   | 806,211  | 57,749 |
| VALIDATION | -        | 4,878  |
| TEST    | -        | 4,870  |
Table 6  Experimental results of different approaches for real cross-domain recommendation. Best results are in bold.

| MOD   | PC+FOOD | PC+APPLIANCE | PC+DIGIT |
|-------|---------|--------------|----------|
|       | MAP(%)  | MAE          | MAP(%)   | MAE      | MAP(%) | MAE      |
| MF    | 79.11   | 0.3716       | 83.34    | 0.3193   | 84.77  | 0.3102  |
| SVD++ | 81.91   | 0.2899       | 86.28    | 0.2384   | 87.09  | 0.2200  |
| RPEM  | 81.97   | 0.2200       | 85.24    | 0.1744   | 87.30  | 0.1529  |

Baseline. None of users and items overlaps between source domain and target domain, so the CLFM and the PCLF cannot work in this case. We compare our proposed model with two approaches: Matrix Factorization (MF) [14] and SVD++ [15]. As far as we know, they are the only two approaches which can fit this case. For using MF and SVD++, we combine the DS and DT datasets (row matrices) to generate a big hybrid matrix for factorization.

Implementation. Like the experiment above, we select parameters on validation set. Finally, the parameters of MF and SVD++ can be set up as [learning rate= 0.001, clustering dimension $k = 50$, regularization ratio $= 0.015$]. Our model configuration can be set up as $\{k = 200, l = 100, \lambda_r = 0.0001, \lambda_i = 0.0001, \lambda_e = 0.00001, \gamma_0 = 2\}$ on the three pair of cross-domain datasets, the number of iterations has been set as 500.

Results. Table 6 shows the results of the compared models in the PC+FOOD, PC+APPLIANCE and PC+DIGIT for real cross-domain recommendation. We can observe that the SVD++ model performs slightly better than the MF model. Our model performs the best. It implies that the rating-pattern embeddings can help cross-domain recommendation even if none of the users and items overlaps. In addition, we also discover that the performance depends on the correlation between the source domain and target domain. The more related the domain are, the better the performance. Our proposed model can capture the correlation to improve recommendation generation.

7. Conclusion

Motivated by relation prediction in multi-relational graph, in this paper we propose a novel embedding model, namely RPEM, which is available for rating, user and item prediction in recommendation systems, by learning the latent semantic representation of the users, items and ratings. We also apply the proposed model to cross-domain recommendation. Empirical comparison on several real datasets validates the effectiveness of the proposed model.

Acknowledgments

Thanks to Spyridoula G. for proofreading my manuscripts. This work was supported by the Natural Science Foundation of China under Grant No. 61300080, No. 61273217, No. 61671078, the 111 Project under Grant No. B08004 and FP7 MobileCloud Project under Grant No. 612212. The authors are partially supported by 360 Innovation, BUPT Excellent Ph.D. Students Foundation.

References

[1] G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions,” Proc. IEEE Trans. Knowl. Data Eng., vol.17, no.6, pp.734–749, 2005.
[2] S. Bhagat, U. Weinsberg, S. Ioannidis, and N. Taft, “Recommending with an agenda: Active learning of private attributes using matrix factorization,” Proc. Recsys, pp.65–72, 2014.
[3] A. Bordes, N. Usunier, and A. Garcia-Duran, “Translating embeddings for modeling multi-relational data,” Proc. NIPS, pp.2787–2795, 2013.
[4] A. Bordes, J. Weston, R. Collobert, and Y. Bengio, “Learning structured embeddings of knowledge bases,” Proc. 25th Annual Conference on Artificial Intelligence, 2011.
[5] I. Cantador, I. Fernandez-Tobias, and S.B.P. Cremonesi, “Cross-domain recommender systems.”
[6] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, “Natural language processing (almost) from scratch,” Proc. The Journal of Machine Learning Research, vol.12, pp.2493–2537, 2011.
[7] I. Fernandez-Tobias1, I. Cantador, M. Kaminskas, and F. Ricci, “Cross-domain recommender systems: A survey of the State of the Art,” Spanish Conference on Information Retrieval, 2012.
[8] J. F¨urnkranz, E. H¨ullermeier, E.L. Mencia, and K. Brinker, “Multilabel classification via calibrated label ranking,” Proc. Machine Learning, vol.73, no.2, pp.133–153, 2008.
[9] S. Gao, H. Luo, D. Chen, S. Li, P. Gallinari, and J. Guo, “Cross-domain recommendation via cluster-level latent factor model,” Proc. In Machine Learning and Knowledge Discovery in Databases, vol.8189, pp.161–176, 2013.
[10] S. Givon and V. Lavrenko, “Predicting social-tags for cold start book recommendations,” Proc. 3rd ACM Conference on Recommender Systems, pp.333–336, 2009.
[11] Z. Ganter, S. Rendle, C. Freudenthaler, L. Schmidt-Thieme, “MyMediaLite: A Free Recommender System Library,” Proc. In the fifth ACM conference on Recommender systems, pp.305–308, 2011.
[12] M. Kaminskas, I. Fernandez-Tobias, and I. Cantador, “Ontology-based identification of music for places,” Proc. 13th International Conference on Information and Communication Technologies in Tourism, pp.436–447, 2013.
[13] Y. Koren, “Factorization meets the neighborhood: A multifaceted collaborative filtering model,” Proc. 14th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp.426–434, 2008.
[14] Y. Koren, R.M. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” Proc. IEEE Computer, vol.42, no.8, pp.30–37, 2009.
[15] R. Kumar, B.K. Verma, and S.S. Rastogi, “Social popularity based svid++ recommender system,” Proc. International Journal of Computer Applications, vol.87, no.14, pp.33–37, 2014.
[16] B. Li and X. Xue, “Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction,” Proc. IJCAI, vol.9, pp.2052–2057, 2009.
[17] B. Li, Q. Yang, and X. Xue, “Transfer learning for collaborative filtering via a rating-matrix generative model,” Proc. 26th Annual International Conference on Machine Learning, pp.617–624, 2009.
[18] Y. Low, D. Agarwal, and A. Smola, “Multiple domain user personalization,” Proc. 17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp.123–131, 2011.
[19] T. Mikolov, W. tau Yih, and G. Zweig, “Linguistic regularities in continuous space word representations,” Proc. HLT-NAACL, pp.746–751, 2013.
[20] O. Moreno, B. Shapira, L. Rokach, and G. Shani, “TALMUD: Transfer learning for multiple domains,” Proc. 21st ACM Conference on Information and Knowledge Management, pp.425–434, 2012.
[21] M. Nakatsuji, Y. Fujiwara, A. Tanaka, T. Uchiyama, and T. Ishida, “Recommendations over domain specific user graphs,” Proc. 19th European Conference on Artificial Intelligence, pp.607–612, 2010.

[22] M. Nakatsuji, “Semantic Sensitive Simultaneous Tensor Factorization.” International Semantic Web Conference. Springer International Publishing, pp.411–427, 2016.

[23] E. Neuhold, “Personalization and user profiling & recommender systems,” Proc. WIIM Information Management Proseminar, 2003.

[24] W. Pan, N. Liu, E. Xiang, and Q. Yang, “Transfer learning to predict missing ratings via heterogeneous user feedbacks,” Proc. 22nd International Joint Conference on Artificial Intelligence, pp.2318–2323, 2011.

[25] W. Pan, E. Xiang, and Q. Yang, “Transfer learning in collaborative filtering with uncertain ratings,” Proc. 26th AAAI Conference on Artificial Intelligence, pp.662–668, 2012.

[26] M. Pazzani and D. Billsus, “Learning and revising user profiles: The identification of interesting web sites,” Proc. Machine Learning, vol.27, pp.313–331, 1997.

[27] S. Ren and S. Gao, “Improving cross-domain recommendation through probabilistic cluster-level latent factor model,” Proc. 29th AAAI Conference on Artificial Intelligence. 2015.

[28] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-based collaborative filtering recommendation algorithms,” Proc. WWW, pp.285–295, May 2001.

[29] R. Socher, D. Chen, C.D. Manning, and A.Y. Ng, “Reasoning with neural tensor networks for knowledge base completion,” Proc. NIPS, pp.926–934, 2013.

[30] X. Su and T.M. Khoshgoftaar, “A survey of collaborative filtering techniques,” Proc. Artificial Intelligence, 2009.

[31] Y. Zhao, S. Gao, P. Gallinari, and J. Guo, “Knowledge base completion by learning pairwise-interaction differentiated embeddings,” Proc. Data Mining and Knowledge Discovery, vol.29, no.5, pp.1486–1504, 2015.

Yu Zhao received the B.S. degree from the Southwest Jiaotong University (SWJTU) in 2006 and the M.S. degree from Beijing University of Posts and Telecommunications in 2011. He is currently pursuing the Ph.D. degree at BUPT. He has visited at Universite Pierre et Marie Curie (UPMC), France and is presently visiting at University of Rochester, USA. His research interests include natural language processing, recommendation system, etc.

Sheng Gao has been an Assistant Professor with the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, since 2012. He received the bachelor’s and master’s degrees from BUPT in 2003 and 2006, respectively, and the Ph.D. degree from Universite Pierre et Marie CURIE (Paris 6), Paris, France, in 2011. His current research interests include machine learning, data mining, information recommendation, and social network analysis. He has published over 20 academic papers on world-wide famous journals or conferences, including ISI, WWW, CIKM, and ECML.

Patrick Gallinari has been a Professor with Pierre et Marie Curie, Paris 6 University, Paris, France, since 1992, and the Director of the Computer Science Laboratory, LIP6, since 2005. He received the Ph.D. degree in computer science from the University of Compiegne, Compiegne, France, in 1985. His current research interests include machine learning applications and information retrieval.

Jun Guo is vice-president of BUPT, Beijing, China. He received the bachelor’s and master’s degrees from BUPT in 1982 and 1985, respectively, and the Ph.D. degree from Tohoku Gakuin University, Tohoku, Japan, in 1993. His current research interests include pattern recognition, etc. He has published more than 100 papers on international journal and conference, including SCIENCE, Nature Online Magazine of Scientific Reports, and IEEE TPAMI.