Expert Systems With Applications 69 (2017) 29–39

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Collaborative filtering and deep learning based recommendation system for cold start items

Jian Wei[a], Jianhua He[a,*], Kai Chen[b], Yi Zhou[c], Zuoyin Tang[a]

[a] School of Engineering and Applied Science, Aston University, Birmingham, B4 7ET, UK
[b] Institute of Image Communication and Network Engineering, Shanghai Jiaotong University, China
[c] Department of Computer Science, Shanghai Jiaotong University, China

ARTICLE INFO

Article history:
Received 21 July 2016
Revised 17 September 2016
Accepted 28 September 2016
Available online 14 October 2016

Keywords:
Recommendation system
Data mining
Deep learning neural network
Collaborative filtering
Cold start problem

ABSTRACT

Recommender system is a specific type of intelligent systems, which exploits historical user ratings on items and/or auxiliary information to make recommendations on items to the users. It plays a critical role in a wide range of online shopping, e-commercial services and social networking applications. Collaborative filtering (CF) is the most popular approaches used for recommender systems, but it suffers from complete cold start (CCS) problem where no rating record are available and incomplete cold start (ICS) problem where only a small number of rating records are available for some new items or users in the system. In this paper, we propose two recommendation models to solve the CCS and ICS problems for new items, which are based on a framework of tightly coupled CF approach and deep learning neural network. A specific deep neural network SADE is used to extract the content features of the items. The state of the art CF model, timeSVD++, which models and utilizes temporal dynamics of user preferences and item features, is modified to take the content features into prediction of ratings for cold start items. Extensive experiments on a large Netflix rating dataset of movies are performed, which show that our proposed recommendation models largely outperform the baseline models for rating prediction of cold start items. The two proposed recommendation models are also evaluated and compared on ICS items, and a flexible scheme of model retraining and switching is proposed to deal with the transition of items from cold start to non-cold start status. The experiment results on Netflix movie recommendation show the tight coupling of CF approach and deep learning neural network is feasible and very effective for cold start item recommendation. The design is general and can be applied to many other recommender systems for online shopping and social networking applications. The solution of cold start item problem can largely improve user experience and trust of recommender systems, and effectively promote cold start items.

1. Introduction

Recommendation systems plays a central role for many online applications and e-commercial services, such as social-networking, recommendation of products such as films, music and articles (Campos, Dez, & Cantador, 2014; Linden, Smith, & York, 2003; Shi, Larson, & Hanjalic, 2014). Many big companies such as Amazon, eBay and Netflix have adopted recommendation techniques to their systems to estimate the potential preferences of customers and recommend relevant products or items to the user. Recommendation performances have huge impact on the commercial success of these companies in terms of revenue and user satisfactory.

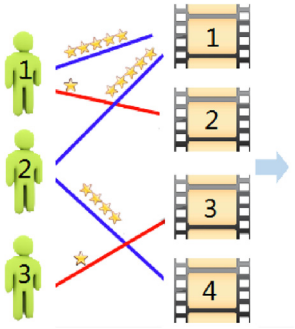
According to the type of data being collected and the ways of using them in recommendation systems, the approaches for recommendation can be classified as content-based (CB), collaborative filtering (CF) and hybrid one (Koren, Bell, & Volinsky, 2009).

CB filtering is widely used for recommendation systems design, which utilizes the content of items to create features and attributes to match user profiles. Items are compared with items previous liked by the users and the best matched items are then recommended. One major issue of CB filtering approach is that RS needs to learn user preferences for some types of items and apply these for other types of items.

CF approach is the most popular approach for recommendation systems design. It utilizes a large amount of data collected from user behavior in the past and predicts which items users will like. It does not need to analyze the content of the items. Instead, it relies on the relationship between users and items, which are typically encoded in a rating feedback matrix with each element rep-

* Corresponding author.
E-mail addresses: weij2@aston.ac.uk (J. Wei), j.he7@aston.ac.uk (J. He), kchen@sjtu.edu.cn (K. Chen), zy_21th@sjtu.edu.cn (Y. Zhou), z.tang1@aston.ac.uk (Z. Tang).

http://dx.doi.org/10.1016/j.eswa.2016.09.040
0957-4174/



	1	2	3	4
1	5	1	?	?
2	5	?	?	4
3	?	?	1	?

Fig. 1. A simplified representation for movie CF recommender systems.

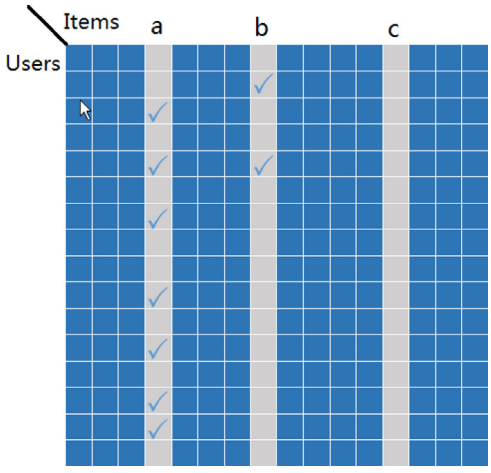


Fig. 2. Illustration of non-CS item (a), ICS item (b) and CCS item (c), where √ indicates a known rating.

resenting a specific user rating on a specific item. An illustration of the CF based recommendation is shown in Fig. 1. The left of Fig. 1 shows a relationship graph of 3 users and 4 movies, which are connected by 5 edges. Each edge is associated with a rating of 1 to 5 stars, representing the level of user preference of the connected movie. The matrix in the right of Fig. 1 is generated according to the relationship graph. The general CF recommendation task is to predict the missing ratings (such as those represented by the symbol "?" in the matrix) by given users or for given items by data mining and exploring the user-item rating matrix.

However it is widely known that CF approach suffers from sparsity and cold start (CS) problems. In the rating matrix only a small percentage of elements get values. Even the most popular items may have only a few ratings. For example, in a large Netflix rating dataset provided for Netflix Prize competition (Bennett & Lanning, 2007), there are about 100 million ratings given by over 480,000 users to about 18,000 movies. There is only around 1% of rating matrix elements receiving ratings. With a sparse rating matrix it is very challenging to estimate the relationship between items and users and make effective recommendation. Another well-known problem for CF approach is the CS problem, which can happen on new users or new items. CF approach requires a large number of ratings from a user or ratings on an item for an effective recommendation, which will not work for new users, new items or both due to few ratings available in the system. In addition, CS problem can be divided into CCS problem and ICS problem by whether number of rating records is zero or not. Generally, the sparsity of ratings for CS items is higher than 85% (Zhang et al., 2014b), and the sparsity of ratings for CCS items is 100%. Fig. 2 presents a simple illustration of the classification of CCS, ICS and non-CS items in recommendation systems.

The hybrid approach is one that combines CB filtering approach and CF approach attempting to overcome their shortcome and provide a more efficient result (Agarwal & Chen, 2009; Chen et al., 2012; Hu et al., 2013). It is noted that the majority of the works on the CS recommendation problem are trying to provide recommendation of items that may be interesting to given users. Although a lot of works have been done with the hybrid approach to solve the sparsity and cold start problems, recommendation of CS items is still an open research issue.

In this paper we investigate the CS recommendation problem of providing prediction on the popularity of given CS items to the general users, and present a solution to predict the popularity of CCS items and ICS items. There are two main motivations for this work:

- CS items need to be recommended to get ratings for improved recommendation and they should be accurately recommended to give users better experiences with the recommendation systems. Otherwise the CS items may go to an undesirable cycle of receiving no ratings.
- The estimated ratings for CS items or items that are still under planning can give a measure of popularity of such items even before they are put into market (such as books, movies, etc), therefore help make right decisions on product planning and sale strategies. The accuracy of such estimation is critically important for this type of purposes.

We design two integrated recommendation models, in which item features are learned from a deep learning architecture SDAE (Vincent, Larochelle, Lajoie, Bengio, & Manzagol, 2010) using the descriptions of items retrieved online, then these features are exploited and integrated into the timeSVD++ CF model (Koren, 2010). timeSVD++ is one of the best performing CF models which tracks time changing behavior in the data and takes the temporal dynamics into account.

Our contributions are summarized as follows.

- We proposed a general framework of integrating the CF approach and machine learning algorithms to improve recommendation performance for CS items. In our proposed models, content features extracted from content descriptions (such as movie plots) by deep learning neural networks are used as the key item factor vectors in the recommendation model for CCS items and approximated by the item factor vectors in the model for ICS items. The content features are not only used loosely to determine item similarity as done in the existing hybrid approaches for cold start items, but also become key component of the recommendation models, which affects both the training of the models and prediction of the unknown ratings for CS items.
- The framework of integrating the CF approach and machine learning algorithms for CS item recommendation is general. Various CF approaches and machine learning algorithms can be used for general recommender systems. The key integration point is on the extraction of item features by machine learning algorithms and embedding the item features into the CF recommendation models.
- Based on the general framework specific system design and models are presented, in which the state of the art CF model, timeSVD++ and an advanced deep learning neural network model, SADE, are used for CS items recommendation. Application of the models to Netflix movie recommendation with nearly 100 million ratings was investigated. The experiments results showed that tight coupling of the CF approach and content based approach for recommendation is feasible and very effective. For example, the rating prediction RMSE of the proposed model IRCD-CCS for CCS item recommendation is 0.045

lower than the second best performing approach, which represents a significant performance improvement in the research field of recommendation system design.
- In addition to the design and evaluation of recommendation models for CCS items and ICS items separately, we also compared the performance of IRCD-CCS model and IRCD-ICS model on rating prediction for ICS items. In practice, recommendation systems keep introducing new items into the systems over time. If a newly introduced item is a CCS item, the CF model can not provide rating prediction for it. If the item is an ICS item, the CF models may not give good recommendation. It may be beneficial to apply a CCS recommendation model for ICS item rating prediction. We propose a scheme of switching recommendation models for ICS items and retraining the models to deal with the practical issues of transition of item status from CS to non-CS. To the best of our knowledge, this practical issue has not been studied before in the literature.

The remainder of the paper is organized as follows. Section 2 describes the related works. Section 3 presents the studied problem and the proposed recommendation models. Section 4 presents the evaluation method and experiment results. Finally Section 5 concludes the paper.

2. Related work

Technically, matrix factorization (MF) method has been applied to CF by a variety of works. MF focuses on factorizing the rating matrix into low-dimension user latent vectors and item latent vectors. Training such a model can be effectively solved by using SGD (Koren et al., 2009) or alternating least squares (ALS) (Zhou, Wilkinson, Schreiber, & Pan, 2008) to minimize the sum-squared distance. Authors Salakhutdinov and Mnih (2007) introduce the probabilistic matrix factorization (PMF) that scales linearly on large data sets and outperforms standard singular value decomposition (SVD) models. Based on PMF, several variants and generalization are proposed like Bayesian PMF (Salakhutdinov & Mnih, 2008), generalized PMF (Shan & Banerjee, 2010).

As traditional CF algorithms only rely on the relations between users and items, which are typically encoded in a U-I matrix, the recommendation performance on sparsity problem and CS problem is largely limited. A large number of approaches incorporating additional information sources beyond U-I matrix have been developed to overcome the problems. Particularly auxiliary information of users or items and interaction related information are exploited to improve recommendation accuracy (Shi et al., 2014).

Auxiliary information refers to attributes about users and items. For user attributes, trust network is incorporated into the raw ratings for prediction (Victor, Cornelis, Teredesai, & De Cock, 2008). Authors Ma, King, and Lyu (2011) propose a probabilistic factor analysis framework which takes users' social trust relations into account. Authors Zhang, Liu, Zhang, and Zhou (2010) make use of users' social tags and design a diffusion-based recommendation algorithm which is only used in social tagging systems. Authors Lika, Kolomvatsos, and Hadjiefthymiades (2014) adopt users' demographic data and apply a simple prediction rule by summing weighted ratings made by similar users to produce ratings for new users. Authos Ocepeka, Rugeljb, and Bosnica (2015) combines attribute selection and local learning into the recommendation model for CS users. Both Lika et al. (2014) and Ocepeka et al. (2015) used one of our baseline approaches (the ToU approach) in models to recommend products to CS users. Authors (Zhou, Yang, & Zha, 2011) try to learn user profiles through an additional interview process. For item attributes, collaborative topic modeling (CTR) (Wang & Blei, 2011) applies topic model and latent Dirichlet allocation (LDA) to learn item content feature. However, this model only works on implicit rating prediction problem, and latent representation is not learned effectively with highly sparse content information.

With great successes in the fields of image, video and artificial intelligence, deep learning technology attracted large interests in the recommendation system field (Georgiev & Nakov, 2013; Sainath, Kingsbury, Sindhwani, Arisoy, & Ramabhadran, 2013; Salakhutdinov, Mnih, & Hinton, 2007). Collaborative deep learning (CDL) (Wang, Wang, & Yeung, 2015) is a representative example that applies deep learning to recommendation systems by integrating stacked denoising autoencoder (SDAE) into a simple latent factor based CF model for movie and article recommendation. Nevertheless, CDL only focuses on the situation of rare users and implicit interactions between users and items, and very simple CF model is considered. The main objective of CDL is recommending top-N items, not for the explicit ratings prediction.

On the other hand, the interaction-associated information refers to the information associated with U-I interaction behavior, like timestamps and locations of the ratings being made. Recently there is a strong interest in the utilization of time information for CF, which demonstrates superior recommendation performance (Koren, 2010; Xiong, Chen, Huang, Schneider, & Carbonell, 2010; Zhang, Wang, Yu, Sun, & Lim, 2014a). TimeSVD++ (Koren, 2010) is a model that simulates the temporal dynamics of user interests by changing static biases and latent factors into time-dependent ones. Authors Xiong et al. (2010) introduce a set of additional time feature vector and use tensor factorization to learn the features. A different modeling scheme on user preferences is presented in Zhang et al. (2014a), where a latent transition matrix is used to summarize the evolving preferences for each user. Authors Xiao, Ai, Hsu, Wang, and Jiao (2015) propose a time-dependent method to compute the similarity among different users. But they are not directly applicable to CS problem.

Generally, CS problem can be classified to CS user problem and CS item problem according to the completely missing ratings for the users or the items. For CS user problem, as system information like locations and gender does not describe user interest efficiently, several recent studies attempt to enrich user profiles with information from other channels, such as social trust network (Ma et al., 2011; Victor et al., 2008), tagging system (Zhang et al., 2010), interview process (Zhou et al., 2011). But these kinds of information are hard to collect in normal conditions. In addition, it is more difficult to acquire personal information of new users because of privacy issues. By the contrast the focus of this paper is on the CS item problem with the motivations described previous. In order to alleviate the information scarcity of CS items, most research efforts so far have been devoted to profiling new items with additional information (e.g., collecting item attributes). However, there still exist a number of limitations in the existing research works. Firstly, it is hard to dig out the specific features of new items with limited rough attributes. And gathering fine-grained attributes like tags, keywords and categories are always time-consuming and costly. Secondly, most studies that combine item content information with ratings data (Schein, Popescul, Ungar, & Pennock, 2001; Wang & Blei, 2011; Wang et al., 2015) adopt generative probabilistic models and tend to overfit easily on CS item situations. The last problem with these works is that they do not take time information into account. In this paper a solution integrating deep learning and collaborative filtering approach is proposed to address these limitations and largely improve recommendation performance for CS items.

3. Proposed recommendation model

In this section, we propose two integrated recommendation models with CF and deep learning , called IRCD-CCS and IRCD-ICS

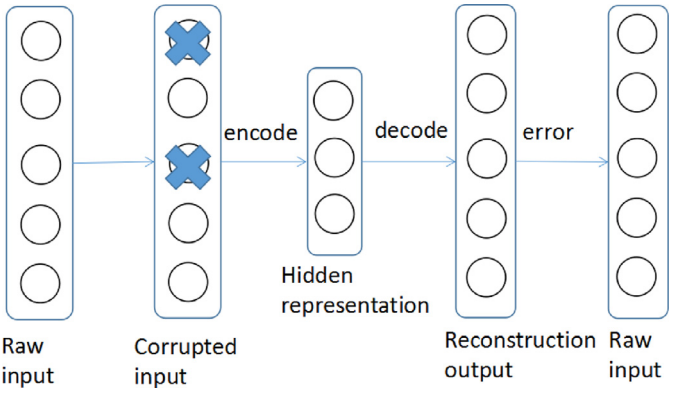


Fig. 3. A graphic structure of SDAE.

for CCS items and ICS items, respectively. A recommendation system is assumed with U users and V non-CS items. In addition it is assumed there are J CCS items, which receive no ratings from the users until the time of investigation, and I ICS items, which receive only a few ratings from the users. We let rating $r_{ui}(t)$ denote the rate by user u on item i at time t. The recommendation task considered in this paper is to estimate the unknown ratings for both CCS and ICS items based on the known ones. We let $\hat{r}_{ui}(t)$ denote the predicted values of $r_{ui}(t)$.

3.1. Deep learning of content features

As traditional CF models are not able to estimate the ratings for CS items, additional content descriptions for the items are obtained for the proposed model. Item features are extracted from the content descriptions and used with a CF model for CS item rating estimation.

Firstly the raw content information of all items are processed to generate vectors based on the bag of words approach. These item associated vectors are then learned by a SDAE to obtain item content features, which are then used in the CF models. SDAE is a deep network that is stacked by multiple denoising autoencoders (DAEs). Each layer of SDAE is trained as a DAE by minimizing the error in reconstructing its input (which is the output of the previous layer). Usually we consider the first half layers of the network as an encoding part and the last half layers as a decoding part. Encoding part tries to learn the feature representations of the noise-corrupted input, and decoding part tries to reconstruct the clean input itself in the output. An example structure of SDAE is shown in Fig. 3.

Formally, given a set C of vectors as raw content information of all items, an L-layer SDAE solves the following optimization problem:

$$\min_{W_l, b_l} \|C - C_L\|^2 + \lambda \sum_l \|W_l\|^2, \tag{1}$$

where C_L denotes the output of layer L of the network and W_l and b_l denote the weight matrix and bias vector of layer l of the network. More details on the SDAE structure and training are referred to Wang et al. (2015). Once the model is trained, the item content features could be obtained from the hidden layer $C_{L/2}$ of the network. For a given item i, the feature representation, denoted by θ_i, is a vector with low dimensions.

It is noted that apart from the goal of learning the features from the rating records, another goal of using SDAE is to reduce the dimensionality of the item content-based vectors to be same with latent factor vectors, which can then be fused into the CF process.

3.2. timeSVD++ model

The CF model used in the proposed IRCD is timeSVD++. There are several variants of timeSVD++ model. In this paper the latent factor based variant is considered. For a latent factor based model, a rating by a user u on an item i is computed by the inner product of a vector q_i (the item factor for item i) and a vector p_u (the user factor for user u) using the following formula:

$$\hat{r}_{ui}(t) = q_i^T p_u. \tag{2}$$

In order to take the biases, additional implicit feedback and temporal effects into account, timeSVD++ uses a revised prediction rule by adding some baseline predictors to (2) as follows:

$$\hat{r}_{ui}(t) = \mu + b_i(t) + b_u(t) + q_i^T \left[p_u(t) + |N(u)|^{\frac{1}{2}} \sum_{j \in N(u)} y_j \right]. \tag{3}$$

Here, μ denotes the overall mean rating, $b_i(t)$ and $b_u(t)$ indicate the time-aware biases of item i and user u respectively. Item factors do not change with time as they are more static in nature than humans. The set $N(u)$ contains the items rated by user u. The factor $|N(u)|^{\frac{1}{2}} \sum_{j \in N(u)} y_j$ indicates the perspective of implicit feedback, where y_j is a vector for item j related to implicit feedback and is to be learned from training process.

The biases $b_i(t)$ and $b_u(t)$ for items and users, respectively, are computed by the following formulae:

$$b_i(t) = b_i + b_{i,Bin(t)}, \tag{4}$$

$$b_u(t) = b_u + \alpha_u \cdot dev_u(t) + b_{u,t}. \tag{5}$$

It is noted that the time-aware item bias $b_i(t)$ is composed of a stationary part b_i and a time changing part $b_{i,\,Bin(t)}$, where the whole timeline is split into time-based bins $Bin(t)$. For user bias $b_u(t)$, b_u represents the stationary part, $\alpha_u \cdot dev_u(t)$ captures a possible gradual drift, in which the time deviation $dev_u(t)$ is defined as:

$$dev_u(t) = sign(t - t_u) \cdot |t - t_u|^{\beta}, \tag{6}$$

and $b_{u,\,t}$ denotes the day-specific sudden drift. Similar to user biases, user factors also become time-aware as $p_u(t)$.

$$p_u(t) = (p_{u1}(t), p_{u2}(t), \ldots, p_{ud}(t)). \tag{7}$$

$$p_{uk}(t) = p_{uk} + \alpha_{uk} \cdot dev_u(t) + p_{uk,t} \quad k = 1, \ldots, d. \tag{8}$$

Here d is the dimensionality of user factors.

In order to learn the model parameters, the system minimizes the regularized squared error on the training ratings:

$$\min_{q*, p*, b*} \sum_{u,i,t} (r_{ui} - \hat{r}_{ui})^2 + \lambda \Big[\|q_i\|^2 + \|p_u\|^2 + \|\alpha_u\|^2 + \|p_{ut}\|^2 + \sum_{j \in N(u)} \|y_j\|^2 + b_i^2 + b_{i,Bin(t)}^2 + b_u^2 + \alpha_u^2 + b_{u,t}^2 \Big]. \tag{9}$$

According to the above regularized squared error function, a SGD optimization method is used to iteratively learn the model parameters. The algorithm loops through all ratings in the training set iteratively and updates each parameter according to the associated gradient until system converges.

3.3. Rating prediction model for CCS items

Next we present the IRCD-CCS model for rating prediction of CCS items. We first present the computation of content similarity. Two baseline rating prediction approaches based on content similarity are then presented. Finally the IRCD-CCS model for CCS items is presented, which integrates content similarities based approach and timeSVD++ model.

To predict the ratings for CCS items, we first use similarity measure to relate CCS items to the non-CS items, and predict the ratings for the CCS items from their most related non-CS items. Based on the item features obtained from the SDAE deep learning process, we use Pearson's correlation coefficient formula to compute the similarity between CCS items and non-CS items. For any two feature vectors θ_i and θ_j of items i and j, the similarity is computed as below:

$$s_{ij} = \frac{\sum_{k=1}^{d} \left(\theta_{ik} - \bar{\theta}_i\right) \cdot \left(\theta_{jk} - \bar{\theta}_j\right)}{\sqrt{\sum_{k=1}^{d} \left(\theta_{ik} - \bar{\theta}_i\right)^2 \cdot \sum_{k=1}^{d} \left(\theta_{jk} - \bar{\theta}_j\right)^2}}. \tag{10}$$

where $\bar{\theta}_i$ and $\bar{\theta}_j$ are the mean values of vectors θ_i and θ_j.

We consider two baseline approaches to predict ratings for CCS items. The first approach predicts the ratings for CCS items from their M most similar non-CS items within the whole non-CS item set after the missing ratings for the non-CS items are predicted, which is called Top-of-All (ToA) approach. Let $\mathcal{S}^M(j)$ denote the set of the M most similar non-CS items to a CCS item j, $j \in [1, J]$. For the ToA approach, the following formula is used to predict rating by user u on CCS item j:

$$\hat{r}_{uj} = \frac{\sum_{i \in \mathcal{S}^M(j)} \hat{r}_{ui} s_{ij}}{\sum_{i \in \mathcal{S}^M(j)} s_{ij}}. \tag{11}$$

It is noted that the ratings $\hat{r}_{ui}$ can be both real and predicted ones.

Alternatively, we can use the second approach to predict the ratings by a user for CCS items from their M most similar non-CS items within the set of non-CS items rated by the user, which is called Top-of-User (ToU) approach. Let $\mathcal{S}^M(u, j)$ denote the set of the M most similar non-CS items among the non-CS items rated by u to a CCS item j, $j \in [1, J]$. For the ToU approach, the following formula is used to predict rating by user u on CCS item j:

$$\hat{r}_{uj} = \frac{\sum_{i \in \mathcal{S}^M(u,j)} r_{ui} s_{ij}}{\sum_{i \in \mathcal{S}^M(u,j)} s_{ij}}. \tag{12}$$

Note that in this case, the ratings r_{ui} are real ratings in the training set.

The above two simple and straightforward approaches make rating prediction entirely based on similar non-CS items and ignore the other information in the rating matrix. According to our experiments, the ToU approach has higher accuracy than the ToA approach. Next we propose an integrated model by combining the ToU approach and timeSVD++ together.

As conventional CF methods are not directly applicable to the CCS problems, we set the item factor q_i in timeSVD++ model to item content feature θ_i and replace the overall average rating μ and item biases $b_i(t)$ with predicted ratings, which is generated by the ToU approach. We extend the ToU approach to generate predicted ratings for all the real ratings in the training set.

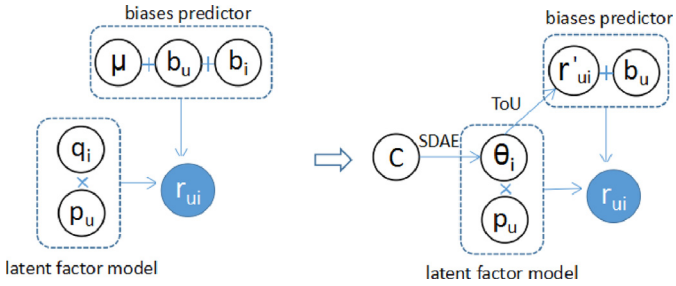


Fig. 4. The graphical modification of IRCD-CCS framework.

The following prediction rule is used in the proposed IRCD-CCS model:

$$\hat{r}_{ui}(t) = b_u(t) + \theta_i^T \left[p_u(t) + |N(u)|^{\frac{1}{2}} \sum_{j \in R(u)} y_j \right] + \frac{\sum_{j \in \mathcal{S}^M(u,i)} r_{uj} s_{ij}}{\sum_{j \in \mathcal{S}^M(u,i)} s_{ij}}. \tag{13}$$

and the corresponding regularized squared error function is:

$$\min_{p*,b*} \sum_{u,i,t} (r_{ui} - \hat{r}_{ui})^2 + \lambda \Big[\|p_u\|^2 + \|\alpha_u\|^2 + \|p_{ut}\|^2 + \sum_{j \in N(u)} \|y_j\|^2 + b_u{}^2 + \alpha_u{}^2 + b_{u,t}{}^2 \Big]. \tag{14}$$

Because all the parameters are user-associated, it can be used to predict ratings for CCS items after training.

It is noted that in the IRCD-CCS model for CCS items, CCS items did not receive any rating from users. Therefore the CCS items do not participate in the timeSVD++ model training. However, to enable prediction with the rule (13), we train the IRCD-CCS model with the rating matrix by setting the item factors q_i of non-CCS items to their content feature $theta_i$. It means that the content features learned from the SDAE is utilized instead of the item features hidden in the rating matrix in the IRCD-CCS model for the CCS items.

Fig. 4 shows a graphical framework of traditional MF model (left part) and IRCD-CCS model (right part). For each user u and item i, traditional MF model computes the predicted rating r_{ui} by adding the latent factors with biases predictor. Latent factor is the inner product of item factor q_i and user factor p_u. Biases predictor include the overall mean rating μ, item bias b_i and user bias b_u. Compared with traditional MF, the IRCD-CCS model first applies SDAE to learn the item content feature θ_i from the raw item content information C. Then the ToU approach is used to obtain a preliminary predicted rating r'_{ui} based on the similarity measure of item content feature. In the model training item factor q_i is set to item content feature θ_i. Finally ratings can be prediction by adding the two parts together with the rule shown in (13).

It is noted that the IRCD-CCS model needs pre-processing to generate the rating prediction for every real ratings in the training set offline, but the online prediction can be computed immediately with the prediction rule.

3.4. Rating prediction model for ICS items

The IRCD-ICS model modifies the timeSVD++ by applying content features learned from SDAE into item latent factor training process. In our proposed model IRCD-ICS for ICS items, the prediction rule (3) is reused, but the model parameters are learned by minimizing a different regularized squared error function. The

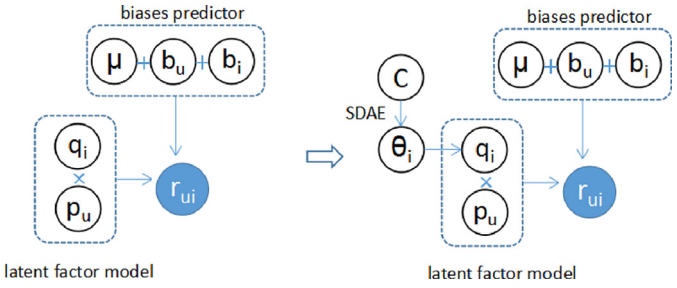


Fig. 5. The graphical modification of IRCD-ICS framework.

modified regularized squared error function is shown below:

$$\min_{q*,p*,b*} \sum_{u,i,t} (r_{ui} - \hat{r}_{ui})^2 + \lambda \Bigg[\|q_i - \theta_i\|^2 + \|p_u\|^2 + \|\alpha_u\|^2 + \|p_{ut}\|^2 + \sum_{j \in N(u)} \|y_j\|^2 + b_i^2 + b_{i,Bin(t)}^2 + b_u^2 + \alpha_u^2 + b_{u,t}^2 \Bigg]. \quad (15)$$

Note that the dimensionality of both q_i and θ_i is d.

It is noted that the trained IRCD-ICS model is used for both ICS items and the general non-CS (NCS) items (with relatively larger number of ratings). We do not create separate recommendation models for the ICS items and the general NCS items. For the ICS items, it is desirable to learn the item factor from the content description as much as possible, as there is very little useful information to be learned for the ICS items from the rating matrix. Therefore the content features play a key role in the recommendation for ICS items. For the NCS items, both rating matrix and content description (content features) can be utilized and should be used to improve the rating prediction performance. The introduction of factor $\|q_i - \theta_i\|^2$ can help achieve the goal of learning item factors for both ICS and NCS items.

Compared with timeSVD++, the following main changes are made on the update rule of item factor vectors. For each given rating $r_{ui}(t)$, the prediction error is computed by $e_{ui}(t) = r_{ui}(t) - \hat{r}_{ui}(t)$. The original update equation of item factor q_i used in timeSVD++ is:

$$q_i \leftarrow q_i + \gamma \left\{ e_{ui}(t) \left[p_u(t) + |N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} y_j \right] - \lambda \cdot q_i \right\}. \quad (16)$$

In the proposed model the following update rule for the item factor q_i is used:

$$q_i \leftarrow q_i + \gamma \left\{ e_{ui}(t) \left[p_u(t) + |N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} y_j \right] - \lambda \cdot (q_i - \theta_i) \right\}, \quad (17)$$

where γ denotes the learning rate. The interested readers are referred to Koren (2010) for more details on the timeSVD++ model learning process. Fig. 5 shows a graphical framework for traditional MF model (left part) and IRCD-ICS model (right part). For the IRCD-ICS model the item content feature θ_i is utilized to learn item factor q_i in the training process according to rule (17).

4. Performance evaluation

In this section, the recommendation performance of the proposed models IRCD-CCS for CCS items and IRCD-ICS for ICS items is evaluated. The experiment data preparation and results are presented and discussed.

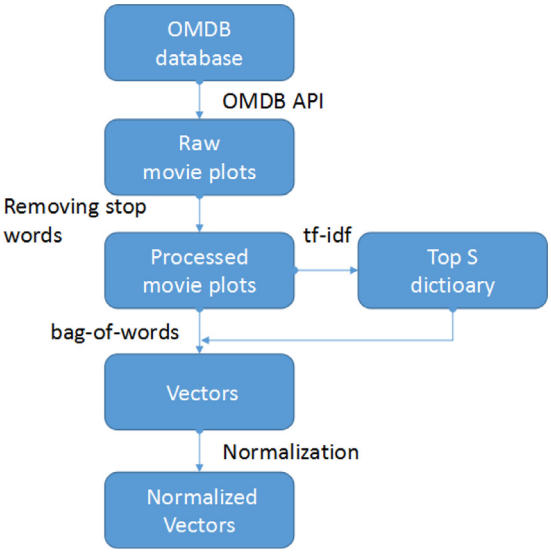


Fig. 6. Workflow of data preprocessing on movie plots.

4.1. Experiment dataset and settings

A large real-world dataset created by the Netflix Prize is used to evaluate the proposed models. The Netflix dataset contains more than 100 million explicit ratings on a scale of 1 to 5 stars for 17,770 movies defined by 480,189 anonymous users. The corresponding timespan ranges from Dec 12, 1999 to Dec 12, 2005.

In order to predict ratings for CS items we also collect the plots of the movies from IMDB to extract item content information. We first collected the corresponding movie plots by OMDb API[1], which is a free web service to obtain movie information. A Python-based program was written, which traverses the movies in the Netflix dataset and automatically sends search requests of plots according to the movie titles to the OMDb database. Then the collected movie plots were filtered by removing stop words, which refer to the most common words in a language offering little useful information. In the experiments the list of stop words was obtained with a built in Python package. Then we computed the term frequency-inverse document frequency (tf-idf) of each word in the corpus. Tf-idf is the product of term frequency and inverse document frequency. Term frequency refers to the number of occurrence of words in a document, and inverse document frequency is calculated by dividing the total number of documents by the number of documents containing the word. Generally tf-idf is used as a weighting factor to reflect how important a word is to a document. Based on tf-idf, the most important S words with the highest tf-idf values are chosen to form a dictionary. S is set to 20,000 in our experiments. Each movie plot is then represented by a S-dimensional bag-of-words vector, in which each entry of the vector indicates the count of corresponding word occurs in the plot. The last step is the normalization for all the vectors. After the movies with missing plots were removed, the final dataset has 476,691 users, 14,657 movies, and 95,975,845 ratings.

A workflow of obtaining and processing the movie plots is presented in Fig. 6.

As we are interested in the performance of rating prediction for both CCS items and ICS items, the original dataset is partitioned into one training set and one test set for CCS items and ICS items, respectively. The movies are ordered by the timestamp of their first received rating. The histogram of movies on the date of

[1] http://www.omdbapi.com

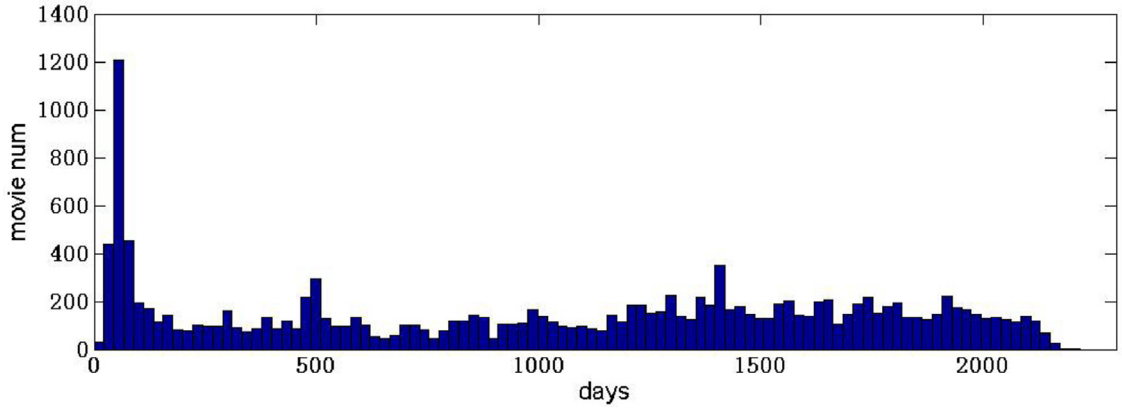


Fig. 7. Histogram of movies on the date of their first ratings.

Table 1
Statistics of the training and test datasets for CCS movie experiment.

	$L = 100$		$L = 300$	
	Training set	Test set	Training set	Test set
Number of users	476,691	11,764	476,691	51,171
Number of movies	14,557	100	14,357	300
Number of ratings	95,959,733	16,112	95,874,146	101,699
Mean rates	3.6042	3.4975	3.6043	3.5652

their first rating is shown in Fig. 7. We divide the entire timespan of the dataset (2240 days) into 100 intervals, and count how many movies fall into each interval. Generally, new movies appeared late, and thus located at far right in the figure. For the preparation of training and test set for CCS experiments, we choose the most recent L movies for test set and the other movies for training set. The ratings of the L CCS movies are used for testing and it is guaranteed that none of these CCS movies could be seen by any user in the training set. To prepare the ICS training and test sets, we choose the most recent K movies for test set and the other movies for training set. Meanwhile the earliest N ratings of each tested movie are added into training set and all the remaining ratings of these K ICS movies are used for test purpose.

For the experiment setting of some parameters, we chose some typical values for performance evaluation. The proposed recommendation models can work with general configurations of the parameters. As there is a large space of values for the parameters, it is very time consuming to test all the combinations of the settings for the parameters. After quick check on the recommendation performance with a number of parameters settings, we determined the settings for model parameters (e.g., setting dimensionality d of feature vectors to 50) with those giving good performance.

For the experiment datasets, the rating statistics for the training and test sets are presented in Tables 1 and 2 for CCS experiment and ICS experiment, respectively.

We first conduct experiments for CCS movies and compare our model IRCD-CCS against three recommendation models: ToA model, ToU model, and simple average (SA) model. ToA model and ToU model are presented in Section 3.3. As the existing models for CCS item prediction use different content information and features, they are not directly comparable to the proposed model. Therefore the simple prediction model SA for CCS items is used: every CCS item rating by a user is set to the average of ratings of the user.

On the other hand, the baseline models for ICS movie experiment includes ALS, SGD, timeSVD++ and CDL. ALS and SGD are two major algorithms for learning parameters. In our experiment, we use them to minimize the error function of (2). As described in the previous section, timeSVD++ is a time-aware CF model, taking the temporal effects into consideration. CDL is a model jointly performing deep representation learning and CF, which applies an ALS-style algorithm for learning. ALS and SGD provide recommendation based on plain ratings, while timeSVD++ and CDL utilize additional time information and item content information respectively.

The overall experiment procedure has four major steps, which are described below:

- Configure the system parameters: the parameters to be configured include learning rate, regularization and factor dimension for each model as described in Section 4.1. These settings remain the same throughout the experiments.
- Prepare training and test sets: as described in Section 4.1, training and test sets are generated respectively according to the configurations with different test sizes (L for CCS experiment and K for ICS experiment). Moreover, the number of training samples for tested ICS items N is also configured to different values in ICS experiments.
- Train the model by fitting the data in training set: specifically, ALS and CDL apply the ALS algorithm to update model parameters, while IRCD models, SGD and timeSVD++ learn model parameters by SGD algorithm. ToA and ToU approaches compute the prediction result directly using Eqs. (11) and (12).
- Predict the ratings in the test set: this is done by using the trained models. The prediction performance is then computed and recorded.

Table 2
Statistics of the training and test datasets for ICS movie experiment ($N = 5$).

	$K = 100$		$K = 200$		$K = 300$	
	Training set	Test set	Training set	Test set	Training set	Test set
Number of users	476,691	11,603	476,691	31,725	476,691	50,918
Number of movies	14,657	100	14,657	200	14,657	300
Number of ratings	95,960,233	15,612	95,924,771	51,074	95,875,646	100,199
Mean rates	3.6042	3.5135	3.6043	3.4288	3.6043	3.5718

Table 3
Performance comparison of prediction models for CCS movies with Netflix dataset.

Approaches	ToA		ToU		IRCD-CCS		SA
	$M = 20$	$M = 100$	$M = 20$	$M = 100$	$M = 20$	$M = 100$	
RMSE ($L = 100$)	1.155	1.224	1.133	1.113	1.075	**1.053**	1.146
RMSE ($L = 300$)	1.134	1.218	1.140	1.127	1.096	**1.082**	1.157

Table 4
Performance comparison of prediction models for ICS movies with Netflix dataset.

Algorithms	RMSE	RMSE	RMSE
	($K = 100$)	($K = 200$)	($K = 300$)
ALS	1.124	1.112	1.097
SGD	1.070	1.076	1.058
timeSVD++	1.053	1.074	1.053
CDL	1.179	1.151	1.148
IRCD-ICS	**1.049**	**1.070**	**1.048**

4.2. Results and analysis

In this subsection, the proposed models are evaluated in terms of RMSE and compared with other baseline recommendation models. RMSE is an objective metric widely used for performance evaluation of recommendation system models, which is defined as:

$$RMSE = \sqrt{\frac{1}{N_p}\sum_{u,i}(\hat{r}_{ui} - r_{ui})^2}, \tag{18}$$

where N_p denotes the total number of predictions.

After tuning the parameters on the validation set, we compare our proposed IRCD models with other baseline models. Dimensionality d of the feature vectors is set to 50 for all the models.

4.2.1. Performance evaluation of IRCD-CCS model on CCS movies

Firstly we evaluate the prediction performance of IRCD-CCS model for CCS items. Table 3 presents the prediction result RMSE, against the number of most related items M, the size of the test dataset for CCS movies, L for CCS new movies. The SA model, ToA model, ToU model and IRCD-CCS model are compared. The number M of most related items is configured to 20 and 100, and the size $L$of the test dataset for CCS movies is configured to 100 and 300, respectively.

It is noted that the IRCD-CCS model performs the best for all the investigated scenarios. Its performance is significantly better than the baseline models. The result shows an improvement of about 0.05 on RMSE compared to the second best model ToU. The performance of both IRCD-CCS model and ToU model improves largely as M increases. On the contrary, the ToA model works well with small M (e.g., with only 20 most related ICS items for rating prediction), but with a large M such as 100 it has a poor performance, which is even worse than the SA model. This can be explained by that the influence of prediction error is accumulated as M increases.

4.2.2. Performance evaluation of IRCD-ICS model on ICS movies

Next we compare the prediction performance RMSE of the IRCD-ICS model with the existing models for ICS movies. Table 4 presents the experiment results. In the experiments the number of associated training ratings N is set to 5. The size K of the tested ICS movies is set to 100, 200 and 300. It is observed that the proposed method IRCD-ICS achieves the best accuracy in all cases. CDL performs the worst even though it applies SDAE for the content information learning. The main reason is that CDL is proposed on the use of implicit rating data instead of explicit data. The large gap (around 0.05) between SGD and ALS demonstrates that SGD is effective on making better predictions for ICS items than ALS. By modeling the temporal dynamics, timeSVD++ outperforms SGD by a significant margin of 0.002 to 0.017. Compared to timeSVD++, IRCD-ICS model shows further consistent improvement of more than 0.004 with inclusion of content information and deep learning process (Table 4).

To have a deep comparison of timeSVD++ and the proposed model IRCD-ICS, we investigate how the prediction performance RMSE changes with the training iterations. Representative training curves are presented in Fig. 8. For both models the RMSE decreases monotonically without overfitting problem. The IRCD-ICS model converges faster than timeSVD++ in the training process. For the case of K being 100, the IRCD-ICS model takes only 25 iterations to end while timeSVD++ needs more than 35 iterations, which means more computation time.

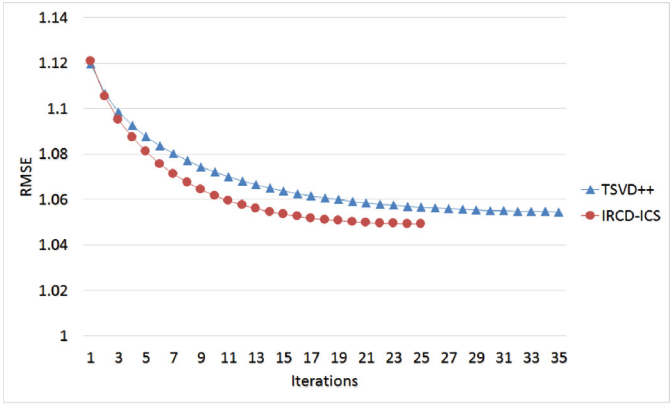
4.2.3. Performance evaluation of both models on ICS movies

In general if an item has a sufficient number of ratings the IRCD-ICS model for this item will certainly outperform the IRCD-CCS model. But when the number of ratings for an ICS items is close to zero, it is not clear whether IRCD-ICS model or IRCD-CCS model performs better. Therefore in this subsection we compare the two models IRCD-CCS and IRCD-ICS on recommending ICS items. To evaluate and compare the models under different degrees of rating matrix sparsity, the number of associated training ratings N is set to 1, 3, 5 and 7 for ICS movies. Fig. 9 shows the RMSE results of timeSVD++ and the IRCD-ICS model with different N for the ICS movies. The green dashed line indicates the RMSE of IRCD-CCS model, which is irrelevant to N. As we can see from Fig. 9 , the RMSE value of both timeSVD++ and IRCD-ICS decreases as N increases. However, when N is less than 5, the ICS item based models including timeSVD++ and IRCD-ICS even perform worse than IRCD-CCS model. It is shown that ICS item based models do not make good prediction of ratings for items with only a small number of ratings, in which case the CCS item based model is preferred.

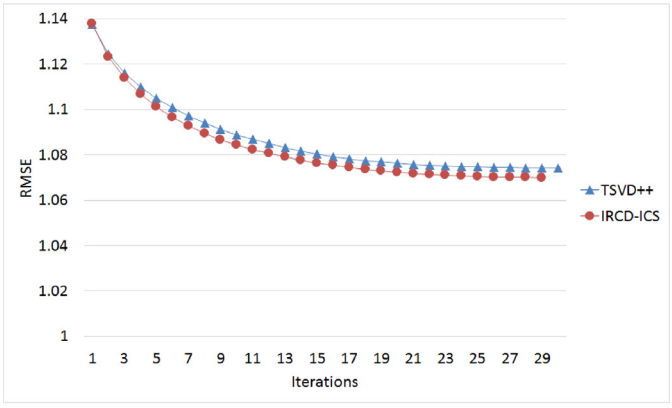
4.3. Discussions

In the above experiments, different recommendation models are used for CCS items and non-CCS (ICS and NCS) items. However, in practical operation of recommendation systems, if ratings for a CCS item are received from users, the item becomes an ICS item. In this case, for this new ICS item rating prediction with the model trained and used for CCS items may be worse than with a model used for ICS items. However, the existing recommendation model for non-CCS items does not use any information (ratings and content description) related to this ICS item, which just changed its status from CCS item. Therefore there is a need to retrain the recommendation model for non-CCS items with extra rating and content description information from the new non-CCS items.

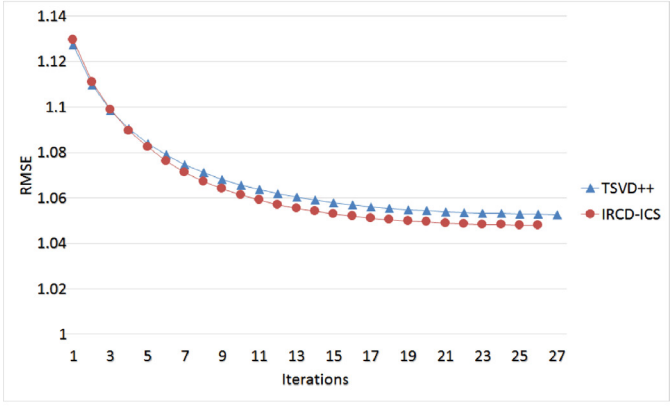
There are two key issues to consider with regards to the retraining of the model for non-CCS items: 1) how frequently the



(a) K=100.



(b) K=200.



(c) K=300.

Fig. 8. Training curves of timeSVD++ and IRCD-ICS model.

model should be retrained? 2) which part of the model should be retrained?

For the issue of retraining frequency, we do not need immediately retrain the ICS model whenever some items change their status from CCS to ICS. As the recommendation of items is not made continuously, there is no need to retrain the ICS model too frequently, which will incur a very high computation cost. Even if the IRCD-ICS model is not retrained when a CCS item changes to ICS item, the CCS model can be used temporally for recommendation of that item. According to the evaluation results, there is not large performance degradation when the CCS model is applied to ICS items. Therefore we can design a recommendation system,

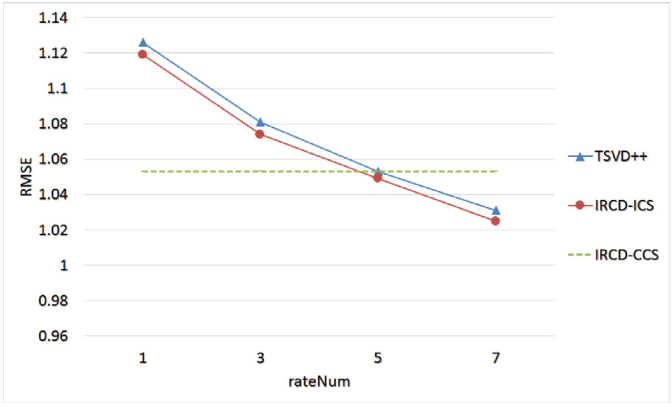


Fig. 9. Performance comparison of IRCD-CCS and IRCD-ICS models for rating prediction of ICS models, $K = 100$.

which collects and stores the new ratings made by the users, and regularly retrains the ICS and CCS recommendation models, for example, in the scale of days or weeks according to the rating activities.

For the issues of which parts of the models to be trained, there is very little impact in the trained model with inclusion of new ratings to the training set, as they take very small proportion of the total ratings in the training set. To reduce computation loads, in the model retraining, we can keep the values for trained model parameters (such as q_i, p_u and b_i) which are well trained in the previous round of training process and should be stable in short term, and only learn the parameter values for the new items.

5. Conclusions

Recommendation of cold start items is challenging and still an open research issue for recommendation systems. Cold start items can be classified to complete cold start (CCS) items which receive no ratings and incomplete cold start (ICS) items which receive more than zero but very few ratings. In this paper we proposed two recommendation models to address the recommendation problems for CCS and ICS items, respectively. The models combine a time-aware collaborative filtering (CF) model timeSVD++ with a deep learning architecture SDAE. The deep learning neural network SDAE is responsible for the extraction of item content features, while the timeSVD++ model is responsible for prediction of unknown ratings. It considered temporal dynamics of user preferences and item features. A large number of experiments were run to evaluate the proposed models in terms of recommendation prediction error RMSE on Netflix dataset. The results showed that our models outperformed existing baseline approaches for cold start item recommendation. From our analysis and experiments, the impact of including the time and item content information is very large. Especially for CCS problem, our model can successfully takes the advantage of CF latent factor models to gain significant performance improvement. In addition, we also compared our proposed models on the ICS new items recommendation with different degrees of rating matrix sparsity. It was found out that the ICS item-based model does not make good recommendations for items that received very few ratings (e.g. 3 ratings). In that case the CCS item based model should be used instead of the ICS item based model.

In the future we plan to extend our recommendation models for cold start items and work on the following research directions. First, we are interested in the investigation of the recommendation performance for CCS and ICS items with more system configurations and parameters setting, in order to reveal more insights to their impact on recommendation performance and system opti-

mization. Second, we create and maintain two separate recommendation models for CCS items and ICS items, respectively. This approach requires extra storage and computation resources. We plan to design a recommendation model, which is applicable to recommendation of both CCS and ICS items. Third, recommendation models are evaluated by the RSME of rating predictions, which may not effectively reflect the performance of real recommendation systems. We are interested in the design of an additional performance evaluation approach, which can take item recommendation decisions into account and quantify the impact of the decisions on user acceptance of recommended items. Finally, in this paper we run experiments of cold start item recommendation on Netflix movies. We are interested in the application of the models to the recommendation of other products such as online music.

Acknowledgment

The work is partially supported by National Natural Science Foundation of China (Grant No. 61221001), and 111 Program (B07022), and Shanghai Key Lab of Digital Media Processing and Transmissions STCSM (14XD1402100).

References

Agarwal, D., & Chen, B. C. (2009). Regression-based latent factor models. In *Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 19–28).

Bennett, J., & Lanning, S. (2007). The netflix prize. In *Proceedings of KDD cup and workshop, 2007* (p. 35).

Campos, P. G., Dez, F., & Cantador, I. (2014). Time-aware recommender systems: A comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction, 24*(1–2), 67–119.

Chen, T., Zhang, W., Lu, Q., Chen, K., Zheng, Z., & Yu, Y. (2012). SVDFeature: A toolkit for feature-based collaborative filtering. *The Journal of Machine Learning Research, 13*(1), 3619–3622.

Georgiev, K., & Nakov, P. (2013). A non-iid framework for collaborative filtering with restricted boltzmann machines. In *Proceedings of the 30th international conference on machine learning* (pp. 1148–1156).

Hu, L., Cao, J., Xu, G., Cao, L., Gu, Z., & Zhu, C. (2013). Personalized recommendation via cross-domain triadic factorization. In *Proceedings of the 22nd international conference on world wide web* (pp. 595–606).

Koren, Y. (2010). Collaborative filtering with temporal dynamics. *Communications of the ACM, 53*(4), 89–97.

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, (8), 30–37.

Lika, B., Kolomvatsos, K., & Hadjiefthymiades, S. (2014). Facing the cold start problem in recommender systems. *Expert Systems with Applications, 41*(4), 2065–2073.

Linden, G., Smith, B., & York, J. (2003). Amazon. com recommendations: Item–to-item collaborative filtering. *IEEE Internet Computing, 7*(1), 76–80.

Ma, H., King, I., & Lyu, M. R. (2011). Learning to recommend with explicit and implicit social relations. *ACM Transactions on Intelligent Systems and Technology (TIST), 2*(3), 29.

Ocepeka, U., Rugeljb, J., & Bosnica, Z. (2015). Improving matrix factorization recommendations for examples in cold start. *Experts Systems with Applications, 42*(19), 6784–6794.

Sainath, T. N., Kingsbury, B., Sindhwani, V., Arisoy, E., & Ramabhadran, B. (2013). Low-rank matrix factorization for deep neural network training with high-dimensional output targets. In *Acoustics, speech and signal processing (ICASSP)* (pp. 6655–6659).

Salakhutdinov, R., & Mnih, A. (2007). Probabilistic matrix factorization. In *NIPS: vol. 20* (pp. 1–8).

Salakhutdinov, R., & Mnih, A. (2008). Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of the 25th international conference on machine learning* (pp. 880–887).

Salakhutdinov, R., Mnih, A., & Hinton, G. (2007). Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on machine learning* (pp. 791–798).

Schein, A. I., Popescul, A., Ungar, L. H., & Pennock, D. M. (2001). Generative models for cold-start recommendations. In *Proceedings of the 2001 SIGIR workshop on recommender systems* (p. 6).

Shan, H., & Banerjee, A. (2010). Generalized probabilistic matrix factorizations for collaborative filtering. In *Data mining (ICDM), 2010 IEEE 10th international conference* (pp. 1025–1030).

Shi, Y., Larson, M., & Hanjalic, A. (2014). Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges. *ACM Computing Surveys (CSUR), 47*(1), 3.

Victor, P., Cornelis, C., Teredesai, A. M., & De Cock, M. (2008). Whom should I trust? The impact of key figures on cold start recommendations. In *Proceedings of the 2008 ACM symposium on applied computing* (pp. 2014–2018).

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., & Manzagol, P. A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research, 11*, 3371–3408.

Wang, C., & Blei, D. M. (2011). Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 448–456).

Wang, H., Wang, N., & Yeung, D. Y. (2015). Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1235–1244).

Xiao, Y., Ai, P., Hsu, C. H., Wang, H., & Jiao, X. (2015). Time-ordered collaborative filtering for news recommendation. *China Communications, 12*(12), 53–62.

Xiong, L., Chen, X., Huang, T. K., Schneider, J. G., & Carbonell, J. G. (2010). Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *SDM: vol. 10* (pp. 211–222).

Zhang, C., Wang, K., Yu, H., Sun, J., & Lim, E. P. (2014a). Latent factor transition for dynamic collaborative filtering. In *SDM* (pp. 452–460).

Zhang, D., Hsu, C. H., Chen, M., Chen, Q., Xiong, N., & Lloret, J. (2014b). Cold-start recommendation using bi-clustering and fusion for large-scale social recommender systems. *IEEE Transactions on Emerging Topics in Computing, 2*(2), 239–250.

Zhang, Z. K., Liu, C., Zhang, Y. C., & Zhou, T. (2010). Solving the cold-start problem in recommender systems with social tags. *EPL (Europhysics Letters), 92*(2), 28002.

Zhou, K., Yang, S. H., & Zha, H. (2011). Functional matrix factorizations for cold-start recommendation. In *Proceedings of the 34th international ACM SIGIR conference on research and development in information retrieval* (pp. 315–324).

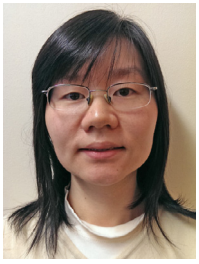
Zhou, Y., Wilkinson, D., Schreiber, R., & Pan, R. (2008). Large-scale parallel collaborative filtering for the netflix prize. In *Algorithmic aspects in information and management* (pp. 337–348).

Jian Wei Jian Wei received the BEng degree in electronic engineering from Shanghai Jiaotong University, China, in 2013. He is currently a PhD student at the School of Engineering and Applied Science in Aston University, UK. His research interests include machine learning and data mining, modeling and scheduling in distributed computing systems and fog computing for IoT big data analytics.

Dr Jianhua Heis a lecturer at Aston University, UK. He received his BSc and MSc degrees from Huazhong University of Science and Technology (HUST), China, and PhD degree from Nanyang Technological University, Singapore, in 1995, 1998 and 2002, respectively. Dr He was an associate professor of Department of Electronics and Information Engineering since 2001. He was with University of Bristol from 2004 to 2006 and with University of Essex in 2007. He is a steering committee member and associate editor of *KSII Transactions on Internet and Information Systems*, was editor of *Wireless Communications and Mobile Computing*, *International Journal of Communication Systems*, leading guest editor for four special issues for *International Journal of Distributed Sensor Networks*. He serves as chair positions for a number of conferences and TPC member for many international conference including IEEE GLOBECOM and ICC. His main research interests include big data analytics, fog computing/mobile edge computing, edge analytics, machine to machine communications, Internet of things system and technologies (e.g. for small cities and intelligent transport) and 4G/5G technologies. He has authored or co-authored over 150 technical papers in major international journals and conferences. He is a senior member of IEEE.

Dr Kai Chenreceived the Ph.D. degree from Shanghai Jiaotong University in 2003 in China. He is an academic staff member at the Institute of Image Communication and Network Engineering, Shanghai Jiaotong University, China. His major research includes information retrieving, object recognition and big data mining. He is the key member of the institute on network engineering research. He is the principal investigator of several key national projects and many IAR (Industry-Academia-Research) projects.

Dr Yi Zhou Yi Zhou received the Ph.D. degree from Shanghai Jiaotong University in 2010 in China. She is an staff member at the Computer Science Department of Shanghai Jiaotong University, China. Her major research includes object recognition and big data mining. She is work on a project of Chinese Characters Reconginition, which is supported by the National Science Foundation.



Dr Zuoyin Tangis currently a lecturer in the School of Engineering and Applied Science, Aston University, UK. She obtained her PhD degree from University of Bath, UK, in 2008. She has authored and co-authored over 40 technical papers in major international journals and conferences. Dr Tang's main research interests include resource management for big data analytics, cellular networks, Internet of things and wireless sensor networks.