Expert Systems With Applications 136 (2019) 365–375

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Contextual-boosted deep neural collaborative filtering model for interpretable recommendation

Shuai Yu[a,b], Min Yang[a,*], Qiang Qu[a], Ying Shen[c]

[a] Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China
[b] School of Computer Science, Fudan University, Shanghai, China
[c] School of Electronics and Computer Engineering, Peking University Shenzhen Graduate School, Shenzhen, China

ARTICLE INFO

Article history:
Received 6 September 2018
Revised 24 June 2019
Accepted 24 June 2019
Available online 26 June 2019

Keywords:
Recommendation systems
Interpretable recommendation
Interactive attention network
Collaborative filtering
Cold start problem

ABSTRACT

Collaborative filtering (CF) is one of the most successful recommendation techniques due to its simplicity and attractive accuracy. However, existing CF methods fail to interpret the reasons why they recommend a new item. In this paper, we propose a Contextual-boosted Deep Neural Collaborative filtering (CDNC) model for item recommendation, which simultaneously exploits both item introductions (textual features) and user ratings (collaborative features) to alleviate the cold-start problem and provide interpretable item recommendation. Specifically, we propose an interactive attention mechanism to learn the user representation, which makes use of the mutual information from both the user ratings and item introductions to supervise the representation learning of each other. With the learned attention weights, we can obtain the importance of each historical item among the historical list. Meanwhile, the attention model can assign different weights to the words in item introductions according to their importance. Therefore, CDNC can provide interpretations for the recommendations by assigning different attention weights to the historically interacted items and the words in the item introductions. On the other hand, we also learn the distributed representations of new-coming items with deep neural networks (i.e., LSTM), considering both rating and item introduction information. Finally, the user representation and the representations of new-coming item are concatenated to perform recommendation score prediction. Extensive experiments on four public benchmarks demonstrate the effectiveness of CDNC. In addition, CDNC has the advantage of interpreting the recommendations and providing user profiles for down-stream applications.

1. Introduction

With the huge volume of online information, much attention has been given to data-driven recommender systems. Those systems automatically guide users to discover products or services with respect to their personal interests from a large pool of possible options. Collaborative filtering is one of the most successful recommendation techniques in practice due to its simplicity, attractive accuracy, and scalability (Kabbur, Ning, & Karypis, 2013; Rendle, Freudenthaler, Gantner, & Schmidt-Thieme, 2009; Zhang, Lu, Chen, Liu, & Ling, 2017). It profiles a user with his/her historically interacted items and recommends items that are similar to the user's profile.

Most previous item-based CF methods apply statistical measures such as cosine similarity to estimate item similarities. The assumption of equal weights is often applied for the items in the measurement (Kabbur et al., 2013). However, treating different items in the historical list equally is not practical in real-life scenarios. On the other hand, interpretable recommender systems, which explain the underlying reasons for the potential user interest on the recommended items, are of increasing interest. Traditional methods often generate explanations from the textual data such as the content and reviews associated with the items (Chelliah & Sarkar, 2017; Chen & Wang, 2017; Tu, Cheung, Mamoulis, Yang, & Lu, 2015; Zhang et al., 2014). Yet, generating reasons of recommendation remains unsolvable when the texts are unavailable.

A common argument against conventional shallow factorization models is that the shallow models are inherently linear models which may be less effective when modeling the data with complex underlying structure. In contrast, deep neural networks, e.g., multi-layered perceptrons (MLPs), may have more expressive power to capture the complex structure of the data with the composition of multiple non-linear transformations. In addition, shallow

* Corresponding author.
E-mail addresses: yushuai2021@gmail.com (S. Yu), min.yang@siat.ac.cn (M. Yang), qiang@siat.ac.cn (Q. Qu), shenying@pkusz.edu.cn (Y. Shen).

https://doi.org/10.1016/j.eswa.2019.06.051
0957-4174/

factorization models do not explicitly consider the temporal variability of data. First, the popularity of an item may change over time. For example, movie popularity booms or fades, which can be triggered by external events such as the appearance of an actor in a new movie. Secondly, users may change their interests and baseline ratings over time (Wu, Ahmed, Beutel, Smola, & Jing, 2017). Recently, recurrent neural network (RNN) has gained significant attention by considering such temporal dynamics for both users and movies and achieved high recommendation quality (Wu, Ahmed, Beutel, & Smola, 2016; Wu et al., 2017; Zhao et al., 2019; 2018). They take the latest observations as input, update the internal states and make predictions based on the newly updated states. As shown in (Devooght & Bersini, 2016), such prediction based on short-term dependencies is likely to improve the recommendation diversity.

Furthermore, most existing recommender systems take into account only the usersâ past behaviors when making recommendation. Compared with tens of thousands of items in the corpus, the historical rating set is too sparse to learn a well-performed recommender system. It is desirable to exploit the content information (e.g., textual introduction) of items to facilitate the performance of recommender system. For example, item introduction reveals a great amount of information to understand the item, as demonstrated in Li et al. (2010). Such an introduction is usually the first contact that a user has with a item, and plays an essential role in the user's decision to consume it or not.

In this paper, we propose CDNC, a contextual-boosted deep neural collaborative filtering model to alleviate the aforementioned limitations of conventional CF methods. CDNC leverages both user ratings and item introductions for learning better user and item representations so as to deal with the cold-start problem and provide interpretable recommendations. Specifically, CDNC is performed by four modules: *initial user representation learning, interactive-learning based user representation learning, item representation learning*, and *feature fusion and prediction*. In *initial user representation learning* module, we employ two separate LSTM networks to learn the hidden representations of ratings and item introductions of items that the user consumed, which capture user characteristics and consider the temporal dynamics of user interests. In *interactive-learning based user representation learning* module, an interactive attention mechanism is designed to capture the correlation of the rating and item introduction representations to learn the final user representation. It makes use of the interactive information from rating and introduction to supervise the modeling of each other. In *item representation learning* module, we also employ two LSTM network to learn the rating and introduction representations, capturing the temporal variability of the item. Finally, in *feature fusion and prediction* module, the user representation and the representation of the new-coming item are concatenated to predict the rating score for recommendation.

We summarize our main contributions as follows:

1. We leverage auxiliary information (i.e., item introductions) to improve the performance of recommender system, alleviating the cold-start problem. A hierarchical attention network is used to capture word-level and document-level features of the item introductions.
2. We design an interactive attention network to combine the strengths of the rating and item introduction information. In addition, the attention scores are used to shed light on how the new item is recommended, providing interpretable recommendations.
3. To verify the effectiveness of our model, we conduct extensive experiments on two widely used real-life datasets. The experimental results demonstrate that our model achieves competitive results with the state-of-the-art methods for the rating prediction recommendation task.

The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 gives the problem definition. Section 4 presents the architecture of our model. Section 5 presents the proposed contextual-boosted deep neural collaborative filtering model in details. In Section 6, we introduce the experimental setup, including the experimental data, compared baselines, implementation details and evaluation metrics. The experimental results and analysis are provided in Section 7. Section 8 concludes this paper and indicates the future work.

2. Related work

Recommender system is an active research field. Bobadilla, Ortega, Hernando, and Gutierrez (2013) and Lu, Wu, Mao, Wang, and Zhang (2015) described most of the existing techniques for recommender systems. In this section, we briefly review the following major approaches that are mostly related to our work.

2.1. Item-based collaborative filtering

Item-based collaborative filtering (Sarwar, Karypis, Konstan, & Riedl, 2001) is one of the most successful techniques in the practice of recommendation due to its simplicity and attractive accuracy. The main idea behind item-based CF is that the prediction of a user u on a target item i depends on the similarity of i to all items the user u has interacted with in the past. In Ning and Karypis (2011), the authors proposed a method named SLIM (short for Sparse Linear Method), which learned item similarities by optimizing a recommendation-aware objective function. SLIM minimized the loss between the original user-item interaction matrix and the reconstructed one from the item-based CF model. FISM (Kabbur et al., 2013) was one of the most widely used collaborative filtering method, which achieved the state-of-the-art performance among the item-based methods. In its standard setting, the prediction of a user u to an item i is calculated by the inner product of the historical items and the target item. Wang et al. (2018) proposed an attention-based transaction embedding (ATEM) model. It was a shallow wide-in-wide-out neural network, which learned an attentive context embedding that is expected to be most relevant to the next choice over all the observed items in a transaction. He et al. (2018) leveraged historical items as attention source to calculate the relationship between the historical items and new-coming item. Zhang, Yao, et al. (2018) proposed an integrated network to combine non-linear transformation with latent factors.

2.2. Matrix factorization

Modeling the long-term interests of users, the matrix factorization (MF) method and its variants have grown to become dominant in the literature (Kim, Park, Oh, Lee, & Yu, 2016; Koren, 2008; Koren, Bell, & Volinsky, 2009; Wu et al., 2018). MF based methods have been used in a broad range of applications such as recommending movies, books, web pages, relevant research and services. In MF based models, the user preference matrix is approximated as a product of two lower-rank latent feature matrices representing user profiles and movie profiles respectively. In the standard matrix factorization, the recommendation task can be formulated as inferring missing values of a partially observed user-item matrix (Koren et al., 2009). features underlying the interactions between users and items. Srebro, Rennie, and Jaakkola (2005) suggested the Maximum Margin Matrix Factorization (MMMF) model, which used low-norm instead of low-rank factorizations. Mnih and Salakhutdinov (2008) presented the Probabilistic Matrix Factorization (PMF) model that characterized the

user preference matrix as a product of two lower-rank user and item matrices. The PMF model was especially effective at making better predictions for users with few ratings. He, Zhang, Kan, and Chua (2016) proposed a new MF method which considered the implicit feedback for on-line recommendation. In He et al. (2016), the weights of the missing data were assigned based on the popularity of items. To exploit the content of items and solve the sparse issue in recommender systems, Zhao, Lu, Pan, and Yang (2016) presented the model for movie recommendation using additional visual features (e.g. posters and still frames) to better understand movies, and further improved the performance of movie recommendation.

2.3. Heterogeneous data sources for recommendation

Recently, numerous studies have been proposed to leverage heterogeneous data sources for item recommendation (Kim et al., 2016; Tu, Cheung, Mamoulis, Yang, & Lu, 2018; Zhang, Ai, Chen, & Croft, 2017b). Kim et al. (2016) leveraged item introduction information to model the item representation and then applied the item representation in a probability graph model to predict the score. Zhang et al. (2017b) used three heterogeneous data sources including rating, review, image information to jointly model the user and item representations. The representations from different sources were then integrated with an extra layer to obtain the joint representations for users and items. In the end, both the per-source and the joint representations were trained as a whole using pair-wise learning for top-N recommendation. Zheng, Noroozi, and Yu (2017) consisted of two parallel neural networks coupled in the last few layers. One of the networks focused on learning user behaviors exploiting reviews written by the user, and the other one learned item properties from the reviews written for the item.

2.4. Deep learning for recommender systems

These traditional MF methods for recommender systems are based on the assumption that the user interests and movie attributes are near static, which is however not consistent with reality. Koren (2010) discussed the effect of temporal dynamics in recommender systems and proposed a temporal extension of the SVD++ (called TimeSVD++) to explicitly model the temporal bias in data. However, the features used in TimeSVD++ were handcrafted and computationally expensive to obtain. Recently, there have been increasing interests in employing recurrent neural network to model temporal dynamic in recommender systems. For example, Hidasi, Karatzoglou, Baltrunas, and Tikk (2015) applied recurrent neural network (i.e. GRU) to session-based recommender systems. This work treated the first item a user clicked as the initial input of GRU. Each follow-up click of the user would then trigger a recommendation depending on all of the previous clicks. Wu, Wang, Liu, and Liu (2016) proposed a recurrent neural network to perform the time heterogeneous feedback recommendation. Wu et al. (2017) used LSTM autoregressive model for the user and movie dynamics and employed matrix factorization to model the stationary components that encode fixed properties. To address the cold start problem in recommendation, Cui, Wu, Liu, and Wang (2016) presented a visual and textural recurrent neural network (VT-RNN), which simultaneously learned the sequential latent vectors of userâs interest and captured the content-based representations that contributed to address the cold-start issues.

2.5. Attention network

Attention mechanisms have recently become an essential part in deep neural networks, which equip a deep neural network with the ability to focus on a subset of its inputs (or features).

Self-Attention Network (Vaswani et al., 2017) is a special case of the attention mechanism, which uses a token embedding from the source input itself as the attention source to calculate the distributed representation of the input sequence. It relates elements at different positions from a single input sequence by computing the attention between each pair of tokens. Expressive performance have been achieved by self-attention mechanism for modeling both long-range and local dependencies of the input sequence. Recently, remarkable success has been achieved by self-attention in a variety of tasks, such as reading comprehension (Hu, Peng, & Qiu, 2017) and neural machine translation (Vaswani et al., 2017).

Recently, several works have been proposed to consider the interactions cross modules when calculating the attention score of the units. For example, Li, Min, Ge, and Kadav (2017) employed an interactive mechanism to generate a supplementary question for the user when the model did not have enough information to answer a given question. User's feedback for the supplementary question was then encoded and exploited to attend over all input sentences to infer an answer. Ma, Li, Zhang, and Wang (2017) utilized the attention mechanism associated with a target to get important information from the context and computed context representation for sentiment classification. Further, IAN made use of the interactive information from context to supervise the modeling of the target which was helpful to judge sentiment. Pei et al. (2017) employed a bidirectional LSTM network to model the user temporal preference and the interactive attention network was used to model the dependency between user and item.

Our approach differs from the aforementioned related works in several aspects. First, our model simultaneously learns the user/item representations with both rating and item introduction information. Second, we design an interactive attention network to capture both the rating and item introduction information. Third, different from previous works that exploit MLP/CNN to model the item introductions, we also devise a hierarchical attention network to learn the item introductions, decomposing the time introductions into a two-level hierarchy: the word-level and sentence-level features.

3. Problem formulation

Suppose there is a sparse user-item rating matrix R that consists of N users and M items. Each entry $r_{u,i}$ denotes the rating of user u on item i. The rating is represented by numerical value from 1 to 5, where the higher value indicates the stronger preference. Meanwhile, each item i is associated with an introduction d_i describing the attributes and features of item i. Formally, the introduction of each item is a sequence of L words $d = [w_1, \ldots, w_L]$, where w_k denotes the word at index k and L is the length of the introduction. Here, we provide an example of item introduction in Table 1. Since each user has rated several items, we also define a introduction representation for user u, which is a sequence of item introductions that the user u has rated $D_u = \{d_{u,1}, \ldots, d_{u,N_u}\}$, where N_u is the number of items in historical item list of user u and $D_{u,1}$ denotes the introduction of the first item that user u rated.

In this paper, we focus on the problem of rating prediction for recommendations, where a recommendation model is learned to predict the users' ratings on unseen items given previous ratings and item introductions. The objective of rating prediction based recommendation is regarded as a supervised regression task aiming at accurately approximating ratings of users on items.

The main notations of this work are summarized in Table 2 for clarity. To prevent conceptual confusion, we use a superscript "r" to indicate the variables that are related to the rating information and a superscript "t" to indicate the variables that are related to the item introduction.

Table 1
An example of item introduction.

Item ID	Item textual Introduction
318	Bank Merchant Andy Dufresne is convicted of the murder of his wife and her lover, and sentenced to life imprisonment at Shawshank prison. Life seems to have taken a turn for the worse, but fortunately Andy befriends some of the other inmates, in particular a character known only as Red. Over time Andy finds ways to live out life with relative ease as one can in a prison, leaving a message for all that while the body may be locked away in a cell, the spirit can never be truly imprisoned.

Table 2
Notation list. We use a subscript u to annotate a user and a subscript i to annotate an item.

R	The user-item rating matrix
$r_{u,i}$	The rating of user u on item i
N, M	The number of users and items
d_i	Introduction of item i
$\hat{r}_{ui}$	Prediction score of user u on item i
emb_u^r	Rating representation of user u
emb_u^t	Textual representation of user u
emb_i^r	Rating representation of item i
emb_i^t	Textual representation of item i
h_i^r	Hidden state of the user's historical list at index i
h_k^{t1}	Hidden state of the word-level LSTM at index k in the introduction
h_i^{t2}	Hidden state of the document-level LSTM at time step i
u_i	Context vector from word-level self-attention network of item i
N_u	The number of items in historical item list of user u

4. Architecture of our approach

The proposed CDNC model, depicted in Fig. 1, consists of four modules. We employ an interactive attention network to learn user representation, which jointly learns the representations of rating and introduction of items the user has interacted with. In particular, a long short-term memory (LSTM) network is first applied to learn the initial rating representation of items the user interacted with. In addition, we use a hierarchical LSTM network to learn the initial introduction representation of the items. Second, an interactive attention network is proposed to learn a better user representation, which makes use of the interactive attentions from the rating representation and the introduction representation to supervise the modeling of each other. Third, similar to user representation learning, we also learn the representation of new items by exploiting the rating and introduction knowledge with LSTM networks. Finally, the user representation and the item representation are concatenated to perform recommendation, via a feature fusion layer and a prediction layer. Next, we elaborate each component of the proposed CDNC model in detail.

5. Our methodology

5.1. Initial user representation learning

Since the LSTM network is good at capturing the temporal preference, we employ an LSTM network to learn the *initial rating rep-*

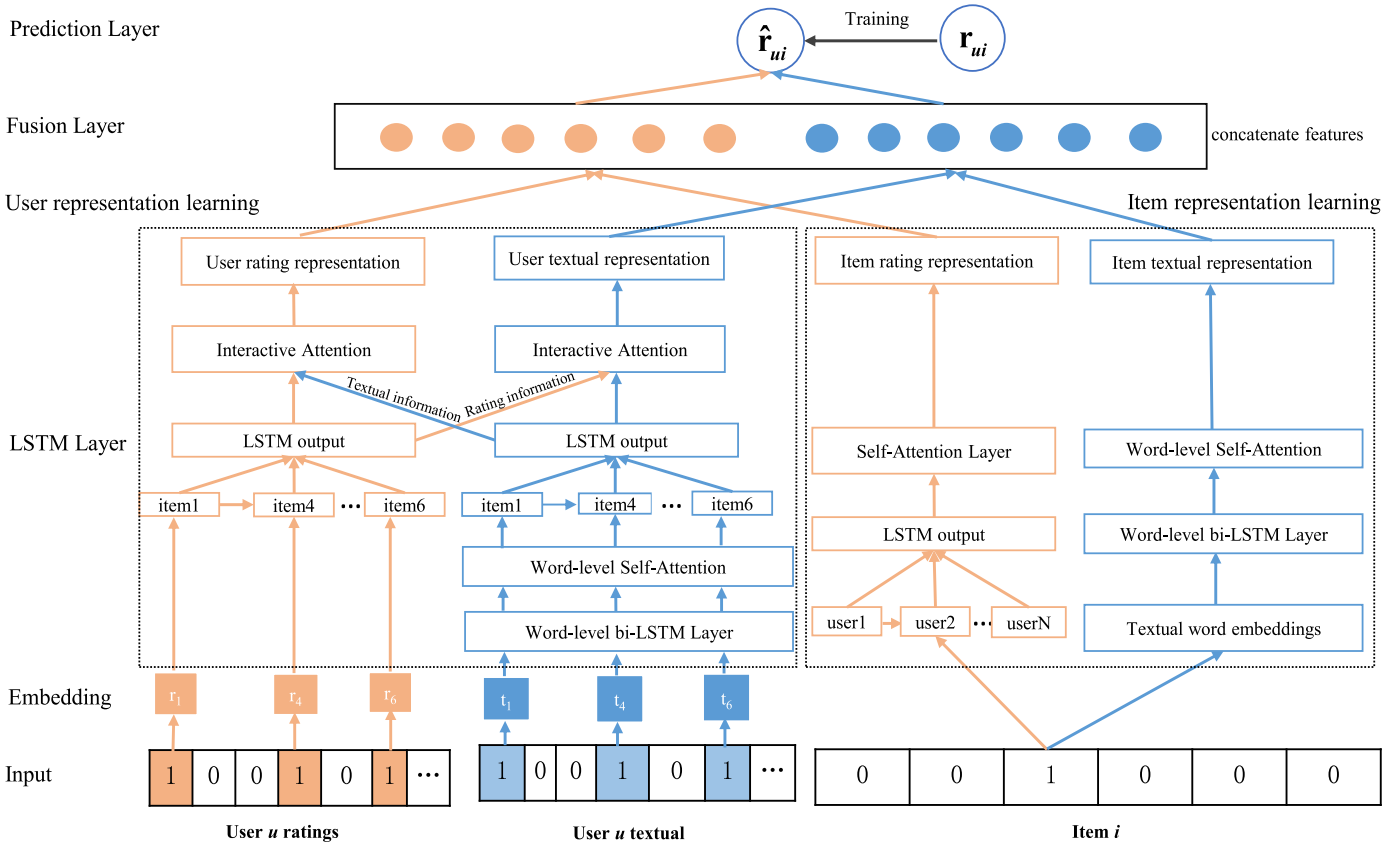

Fig. 1. Overview of the architecture of CDNC.

resentation for a given user u. Meanwhile, we also use a bidirectional LSTM network to learn the *initial introduction representation* for user u.

5.1.1. Initial rating representation

Given a user-item pair (u, i), we encode the involved user and item using the one-hot representation. Formally, we encode the rating of each item into an one-hot vector $\mathbf{y}_i \in \mathbb{R}^{|N|\times 1}$, where $|N|$ denotes the number of the items. In the one-hot vector representation, only the ith element is equal to 1 and the others are equal to 0. Each one-hot item vector is mapped to a low-dimensional rating embedding v by embedding layers. The hidden states of items are then learned by LSTM layer. Formally, given the input rating embedding v_i at index i in the user's historical list, the hidden state $h_i^r \in \mathbb{R}_r^d$ (d_r is the number of hidden states for each LSTM unit) can be update from previous hidden state h_{i-1}^r, which is computed by:

$$h_i^r = \mathbf{LSTM}(h_{i-1}^r, v_i) \tag{1}$$

After processed by the embedding and LSTM layers, the initial rating representations for user u can be represented by the hidden states $H^r = [h_1^r, h_2^r, \ldots, h_{N_u}^r]$, where N_u is the number of items in historical item list of user u.

5.1.2. Initial introduction representation

We employ a hierarchical LSTM architecture to learn the introduction representation for user u, which decomposes the item introductions of historical items into a two-level hierarchy: word-level and document-level.

First, each word w in the item introduction is mapped to a low-dimensional word embedding $e_w \in \mathbb{R}^{d_t}$ through a word embedding layer, where d_t denotes the dimension of word embedding.

Second, We encode each word embedding e into a hidden representation h^t with word-level LSTM network. The word-level LSTM is structurally similar to the item LSTM. Formally, the hidden state of the word-level LSTM at index k in the introduction is:

$$h_k^t = \mathbf{LSTM}(h_{k-1}^t, e_k) \tag{2}$$

While the uni-directional LSTM is able to compose an unbounded context history, it fails to encode the future context. Thus, we adopt a bi-directional LSTM (Graves & Schmidhuber, 2005), which runs in two chains: one forward through the input and the other backward. The forward hidden state at position k summarizes the tokens preceding position k and the backward hidden state summarizes the tokens following position k. We concatenate the hidden states of both forward and backward LSTMs as a hidden representation: $h_k^{t_1} = [\overrightarrow{h_k^{t_1}}, \overleftarrow{h_k^{t_1}}]$.

Third, we use an self-attention network to estimate the importance of each word. Our goal is to encode the introduction into a vector that represent the whole semantic information of the item introduction. Formally, we encode the introduction as follows:

$$u_i = \sum_{k=1}^{N_{d_i}} \omega_k h_k^{t_1} \tag{3}$$

$$\omega_k = softmax(z_k^w) \tag{4}$$

$$z_k^w = V_0^T \sigma(W_0 h_k^{t_1} + b_0) \tag{5}$$

where L_{d_i} is the length of introduction d_i. V_0 and W_0 are weight matrices, b_0 is the bias term. u_i is the introduction representation of ith item. ω_k is the attention weight of the kth word in the introduction d_i.

Finally, we use a document-level LSTM network to capture the temporal dynamics of user interests based on the textual introductions of items in the historical item list of user u. Document-level LSTM keeps track of the past items by iteratively processing each item introduction step by step. Formally, the hidden states of the document-level LSTM at time step i is:

$$h_i^{t_2} = \mathbf{LSTM}(h_{i-1}^{t_2}, u_i) \tag{6}$$

The initial introduction representations of the items in user u's historical list can be represented by the hidden states $H^{t_2} = [h_1^{t_2}, h_2^{t_2}, \ldots, h_{N_u}^{t_2}]$, where N_u is the number of items in historical item list of user u.

5.2. Interactive attention for user representation learning

In order to learn better user representation, we combine the strength from both rating and item introduction information. The user representation consists of two parts: user rating representation and user textual representation.

With the item rating representation $H^r = [h_1^r, h_2^r, \ldots, h_{N_u}^r]$ and the initial introduction representation $h_{N_u}^{t_2}$, the interactive attention mechanism calculates the attention vector $\alpha \in \mathbb{R}^{d_\alpha}$ for item ratings of user u as:

$$emb_u^r = \sum_{i=1}^{N_u} \alpha_i h_i^r \tag{7}$$

$$\alpha_i = softmax(z_i^r) \tag{8}$$

$$z_i^r = V^T \sigma(W_1 h_i^r + W_2 h_{N_u}^{t_2} + b) \tag{9}$$

where z_i^r is an alignment model which scores the contribution of item i to the rating representation of user u. V, W_1, W_2 are the weight matrices, b is the bias. To form a probability distribution over the items, we normalize the scores across the items using *softmax* function and get attention score α_i. $\sigma(\cdot)$ is the activation function. In this paper, we choose *sigmoid* as activation function because *sigmoid* has more stable performance than *relu* and *tanh* in all the experiments.

Only using the attention vector α cannot capture the interactive information of the rating and introduction information of items, and lacks the ability of discriminating the importance of the items in user's historical list. To make use of the interactive information between the rating and introduction information of items, we also use the rating representation as attention source to attend to the introduction representations of items. Similarly, we can calculate the attention vector β for the introductions as:

$$emb_u^t = \sum_{i=1}^{N_u} \beta_i h_i^{t_2} \tag{10}$$

$$\beta_i = softmax(z_i^t) \tag{11}$$

$$z_i^t = V^T \sigma(W_1 h_i^{t_2} + W_2 h_{N_u}^r + b) \tag{12}$$

To reduce the amounts of parameters of our model, we share the weight matrices V, W_1, W_2 and bias b when calculating the attention score. Finally, we obtain the final user representation as the concatenation of both rating and introduction representations of items in the user's historical list: $emb_u = [emb_u^r, emb_u^t]$.

5.3. Item representation learning

Similar to the learning of user representation, we also learn the item representation of the new-coming item i.

For the item rating representation, we use the historical user list who consumed this item as the input of a embedding layer followed by a LSTM layer to obtain the dynamic information of the item. We also add an additional self-attention network to estimate

the importance of the historical users for the item representation learning. We refer the readers to Section 5.1 for the implementation details of self-attention network.

For the item introduction representation, we first use an bidirectional LSTM network to capture the semantic information of text. After that, we leverage a word-level self-attention network to estimate the importance of each word in the item introduction. The implementation details of self-attention network can be found in Section 5.2. Thus, we can also obtain the final item representation $emb_i = [emb_i^r, emb_i^t]$.

5.4. Softmax smoothing

Based on our empirical observation, the standard softmax fails to learn from users' historical data and perform accurate recommendation. By analyzing the attention weights learned by our model, we reveal that the performance of the model is largely hindered by the *softmax* function, due to the large variances of user historical items. The attention weights of the items from long history list are distinguishable. Similar to He et al. (2018) and Yu et al. (2019), we introduce a symbol μ to smooth the denominator of the original attention formula. μ can be set in a range of [0, 1]. If $\mu = 1$, then Eq.(13) degenerates into the original *softmax*. One typically choice of the value of β is between zero and one. This smooth setting leads to much better performance than standard *softmax* function.

$$\alpha_i = \frac{exp(z_i^r)}{[\sum_{k=1}^{N_u} exp(z_k^r)]^{\mu}} \tag{13}$$

5.5. Fusion layer and prediction layer

So far, we have introduced how to obtain the representation of user u and item i. A natural way to combine the user and item representations is to calculate the rating score with the inner product of the user and item representations. To provide more flexibility to the model, we add a fusion layer to assign different weights to the two parts. The formulation of the information fusion layer is given as follows:

$$\hat{r}_{ui} = c \times \text{sigmoid}(\mathbf{W}_p^T[emb_u^r \otimes emb_i^r, emb_u^t \otimes emb_i^t]) \tag{14}$$

$$\mathbf{W}_p = \begin{bmatrix} \omega \mathbf{W}_p^r \\ (1-\omega)\mathbf{W}_p^t \end{bmatrix} \tag{15}$$

where $\mathbf{W}_p^r$ and $\mathbf{W}_p^t$ are hidden parameters of the fusion layer. $\otimes$ denotes element-wise multiplication. ω is a constant scalar, which is a trade-off parameter of the two parts. To obtain stable performance, we set $\omega = 0.7$. c is a scalar that is used to scale up the corresponding formula.

5.6. Model optimization

The parameters of our rating prediction model is learned in a supervised manner. In particular, given a labeled training dataset D, we minimize directly the mean squared error (MSE) between the predicted scores and the ground-truth scores as the objective function:

$$L = \frac{1}{|D|} \sum_{r_{ui} \in D} (r_{ui} - \hat{r}_{ui})^2 + \lambda ||\theta||^2 \tag{16}$$

where r_{ui} and $\hat{r}_{ui}$ are ground-truth score and the predicted score for user u on item i. $|D|$ represents the number of instances in the training corpora. λ is regularization parameter, and we set $\lambda = 0.01$ in the experiment. θ is the parameter set of our model.

We adopt the Adaptive Moment Estimation (Adam) (Kingma & Ba, 2014) as the optimization algorithm. Adam adapts the learning rate for each parameter by performing smaller updates for frequent parameters and larger updates for infrequent parameters. It yields faster convergence for our model than the vanilla SGD and relieves the pain of tuning the learning rate.

Table 3
Data statistic on three real-world datasets.

Dataset	#users	#items	#ratings	density
Movielens-1m	6040	3544	993,482	4.641%
Movielens-10m	69,878	10,073	9,945,875	1.413%
Netflix	480,189	17,770	100,480,507	1.177%
Yelp2013	70,816	15,584	622,873	0.04%

6. Experimental setup

6.1. Dataset description

To evaluate the effectiveness of our model in terms of rating prediction, we conduct extensive experiments on three real-world datasets collected from MovieLens,[1] Netflix[2] and Yelp2013.[3] These datasets consist of users' explicit ratings on items over a scale of 1 to 5. MovieLens and Netflix contain reviews for movies, while Yelp2013 consists of reviews for various businesses, products and services. Since MovieLens and Netflix do not include item introductions, we additionally collect the introduction (i.e., plot summary) of each item from IMDB[4]. In summary, each user is associated with the UserID, Gender, Age, Occupation and Zip-code. Each item contains MovieID, Title, Genres, and Introduction. Each rating record is a four-tuple (UserID, MovieID, Rating, Timestamp).

Similar to Wang, Wang, and Yeung (2015) and Wang and Blei (2011), we pre-process the item introductions in all datasets as follows: (i) set maximum length of raw documents to 300; (ii) removed stop words; (iii) calculated tf-idf score for each word; (iv) removed corpus-specific words that have the document frequency higher than 0.5; (v) selected top 8000 distinct words as the vocabulary; (vi) removed all non-vocabulary words from raw documents. As a result, the average numbers of words per document are 97.09 on MovieLens-1m, 92.05 on MovieLens- 10m and 91.50 on Netflix, respectively. We remove the items that do not have introductions from each dataset. To alleviate the severe sparsity problem in Netflix dataset, the users that have less than 3 ratings are also removed from the Netflix dataset. The detailed statistics are presented in Table 3. we randomly split each dataset into a training set (80%), a validation set (10%), and a test set (10%).

6.2. Baseline methods

We evaluate and compare our proposed model with the following state-of-the-art baselines:

- **Probabilistic Matrix Factorization (PMF)** (Mnih & Salakhutdinov, 2008) is a standard rating prediction model that only uses ratings for collaborative filtering.
- **Singular Value Decomposition (SVD)** (Sarwar, Karypis, Konstan, & Riedl, 2002) finds the matrix $\hat{R} = U^T V$ of the given rank which minimizes the sum-squared distance to the target matrix R.
- **SVD++** (Koren, 2008) integrates explicit feedback with implicit feedback into one model and achieves state-of-the-art performance.

[1] https://grouplens.org/datasets/movielens.
[2] https://www.netflix.com.
[3] https://www.yelp.com/dataset.
[4] https://www.imdb.com.

- **Collaborative Topic Regression (CTR)** (Wang & Blei, 2011) Collaborative Topic Regression is a state-of-the-art recommendation model, which combines collaborative filtering (PMF) and topic modeling (LDA) to use both ratings and documents.
- **Collaborative Deep Learning (CDL)** (Wang et al., 2015) Collaborative Deep Learning is another state-of-the-art recommendation model, which enhances rating prediction accuracy by analyzing documents using Stacked Denoising AutoEncoder (SDAE).
- **Multi-layered Perceptron (MLP)** (He et al., 2017) is the standard deep neural network model with ReLU activations.
- **Joint Representation Learning (JRL)** (Zhang et al., 2017b) exploits mulit-view neural network to learn user-item representations.
- **Neural Matrix Factorization (NeuMF)** (He et al., 2017) is a strong deep neural baseline that combines MF with a side MLP network.
- **Convolutional MF (ConvMF)** (Kim et al., 2016) is a strong baseline that integrates convolutional neural network (CNN) into probabilistic matrix factorization.

For all the baseline methods, we select the best hyper-parameters of the models on the validation set.

6.3. Evaluation metrics

In order to facilitate comparison with other models quantitatively, we adopt root mean squared error (RMSE) which is a standard evaluation metric for supervised regression model (rating prediction). RMSE measures the differences between values predicted by the model and the values observed (Gunawardana & Shani, 2009). For example, by using RMSE, predicting a 2 score when the user actually gave a 1 score to a item is less of a mistake than predicting 5 score to the same item. Formally, the RMSE score is computed by:

$$RMSE = \sqrt{\frac{\sum (\hat{r}_{ui} - r_{ui})^2}{\#\ of\ ratings}}$$

We also used mean absolute error (MAE) to evaluate our model, which is widely used in previous studies (Ma, Yang, Lyu, & King, 2008; Singh & Gordon, 2008; Srebro et al., 2005). MAE measures the absolute value of the difference between the forecasted value and the actual value, which tells us how big of an error we can expect from the forecast on average. Note that lower MAE values represent higher recommendation accuracy (Jin, Zhou, & Mobasher, 2005).

$$MAE = \frac{\sum |r_{ui} - \hat{r}_{ui}|}{\#\ of\ ratings}$$

We use the validation set to choose the best parameters of the proposed model by using the early-stopping strategy. For reliability of our results, we repeat this procedure for 5 times with different data partitioning and get the average test errors over the 5 validation runs.

7. Experimental results

7.1. Quantitative results

Table 4 shows the RMSE and MAE scores of CDNC and the baseline methods on Movielens-1m dataset. Note that "Improve" indicates the relative improvements of CDNC over the best competitor. Compared to the baseline methods, our model substantially and consistently outperforms the baseline methods by a noticeable margin on all the experimental datasets. For example, CDNC improves 0.597% on RMSE, 1.037% on MAE over the baseline methods. The significant performance gap between CDNC and PMF indicates that deeper understanding of documents helps learn better latent models that are more accurately even when enough ratings are given.

Table 4
Experiment Results on Movielens-1m Dataset. Numbers with * mean that improvement from our model is statistically significant over the baseline methods (t-test, p-value < 0.05).

Model	RMSE	MAE
PMF	0.8971	0.706
SVD	0.8730	0.686
SVD++	0.8620	0.673
CTR	0.8969	0.706
CDL	0.8879	0.691
ConvMF	0.8549	0.676
MLP	0.8773	0.687
NeuMF	0.8631	0.674
JRL	0.8531	0.675
Ours	**0.8480**	**0.668***
Improve	0.597%	1.037%

Table 5
Experiment Results on Movielens-10m Dataset. Numbers with * mean that improvement from our model is statistically significant over the baseline methods (t-test, p-value < 0.05).

Model	RMSE	MAE
PMF	0.8331	0.652
SVD	0.8135	0.631
SVD++	0.8108	0.629
CTR	0.8275	0.646
CDL	0.8186	0.637
ConvMF	0.7930	0.606
MLP	0.8143	0.632
NeuMF	0.8107	0.674
JRL	0.7913	0.605
Ours	**0.7882**	**0.601***
Improve	0.392%	0.662%

Table 6
Experiment Results on Netflix Dataset. Numbers with * mean that improvement from our model is statistically significant over the baseline methods (t-test, p-value < 0.05).

Model	RMSE	MAE
PMF	0.9237	0.721
SVD	0.9112	0.717
SVD++	0.9053	0.714
CTR	0.9216	0.726
CDL	0.9103	0.719
ConvMF	0.8968	0.706
MLP	0.9043	0.713
NeuMF	0.8970	0.706
JRL	0.8914	0.701
Ours	**0.8823***	**0.692***
Improve	1.021%	1.284%

For Movielens-10m and Netflix datasets, our model also achieves state-of-the-art performance compared to the strong baselines. Table 5 shows the overall rating prediction error of CDNC and other competitors on Movielens-10m. The improvements of CDNC over the best competitor (i.e., JRL) are 0.392% on RMSE and 0.662% on MAE. Table 6 shows the performances on Netflix dataset. CDNC also outperforms other competitors, the improvement of RMSE is 1.021% and MAE is 1.284%. We can observe that for sparse datasets, Movielens-10m and Netflix, JRL outperforms NeuMF and other methods which only use rating information. Our model further outperforms JRL. It implies that our model can better utilize the item introductions and rating information with attention mechanism to boost the recommendation performance.

We also verify the performance of our model on another application domain (Yelp reviews) except movie recommendation.

Table 7
Experiments results Yelp2013 dataset. Numbers with * mean that improvement from our model is statistically significant over the baseline methods (t-test, p-value < 0.05).

Model	RMSE	MAE
PMF	1.853	1.427
SVD	1.591	1.130
SVD++	1.575	1.117
CTR	1.169	0.915
CDL	1.562	1.119
ConvMF	1.182	0.912
MLP	1.586	1.135
NeuMF	1.591	1.130
JRL	1.072	0.844
Ours	**1.045***	**0.822***
Improve	2.519%	2.607%

Table 8
Ablation test (RMSE scores) on the three datasets.

Dataset	CDNC	w/o text	Improve
Movielens-1m	**0.8480**	0.8508	0.32%
Movielens-10m	**0.7882**	0.7916	0.43%
Netflix	**0.8823**	0.8867	0.50%

Table 7 shows the overall rating prediction error of CDNC and other baseline methods. For Yelp2013 dataset, our model also outperforms the baselines by a noticeable margin. The improvements of CDNC over the best competitor (i.e., JRL) are 2.519% on RMSE and 2.607% on MAE.

7.2. Ablation study

In order to analyze the effectiveness of introduction (document description) information, we also report the results of CDNC when discarding the textual data (denoted as w/o text). We set the hyper-parameter $\omega = 1$, which indicates that the textual analysis part is disabled. The RMSE results are reported in Table 8. For Movielens-1m and Movielens-10m datasets, the relative improvement is 0.32% and 0.43% respectively, when leveraging both rating and introduction information. For Netflix dataset, the relative improvement is 0.50%. This verifies that the textual data helps to comprehend the deep semantics of the items thus boosts the final performance of the recommender system.

7.3. Cold-start problem analysis

To investigate the effectiveness of our model in dealing with the cold-start problem, we also conduct experiments on the Netflix dataset for the cold-start users. In this paper, the users that have less than 3 ratings are categorized as cold-start users and there are totally 49,394 cold-start users in Netflix dataset. The experimental results are reported in Table 9. From the results we can observe that the cold-start users benefit significantly from the movie introductions. In particular, our model achieves much better performance than other models that do not consider the context information (e.g., PMF and MLP) on cold-start users. This verifies the effectiveness of our model, which incorporates the auxiliary information (textual introduction) for alleviating the cold-start problem.

7.4. Execution time analysis

We investigate the computational cost of baseline methods and the proposed CDNC model. All these methods are run on a single NVIDIA Tesla P100 GPU. At the training phase, CDNC takes about 1 second per iteration on the three datasets. Based on our empirical observation, most compared baselines take 1–2 s per iteration on a average. All models typically converge within less than 15 epochs by using the early stopping criterion. The training time until convergence therefore varies between 1–52 h depending on the models and the datasets. At the testing phase, predicting the recommendation results is reasonably fast with a throughput of about 256 instances per second, using a batch size of 64.

Table 9
Experiments results for cold-start users on Netflix dataset. Numbers with * mean that improvement from our model is statistically significant over the baseline methods (t-test, p-value < 0.05).

Model	RMSE	MAE
PMF	1.673	1.446
SVD	1.561	1.331
SVD++	1.545	1.329
MLP	1.482	1.337
NeuMF	1.427	1.319
JRL	1.406	1.305
Ours	**1.285***	**1.181***
Improve	8.606%	9.502%

As shown in Fig. 2, the matrix factorization based methods (e.g., PMF and SVD) take less execution time than the deep learning based methods. During the training phase, our model takes more time than PMF (or SVD) and less time than JRL. While during the inference (testing) phase, our model takes comparable time with other methods. Therefore, the proposed CDNC model is practical for movie recommendation in real world.

7.5. Sensitive of the hyper-parameters

Our model introduces three additional hyper-parameters (i) ω to control the weight of the textual analysis part, (ii) μ to control the interactive attention network, and (iii) λ to control the strength of the regularizer. Here we show how the three hyperparameters impact the performance of our model and also shed lights on how to set the values.

First, we fix $\mu = 0.93$ and $\lambda = 0.1$ and vary the value of ω. As can be seen from Fig. 3, the optimal value of ω is around 0.7. When ω is too small (i.e. less than 0.2), the model benefits only from textual part thus the performance is limited. When ω is too large (i.e. large than 0.9) the model benefits only from rating part and the performance of the model drops dramatically. This further verifies the positive effect of the item introductions.

Second, we fix $\omega = 0.7$ and $\lambda = 0.1$ and vary the value of μ. Fig. 4 shows the results. We can observe that when μ is smaller than 0.6, increasing μ leads to the performance improvements of CDNC gradually. When μ is larger than 0.93, the performance of the model drops dramatically since it degenerates to the standard softmax function.

Third, we fix $\omega = 0.7$ and $\mu = 0.93$, and vary the value of λ. Fig. 5 shows the results. We can see that when λ is too small (i.e. less than 0.001) the model suffers from over fitting. When λ is too large (i.e. large than 10) the performance of the model is limited, since the values of parameters are tend to **0**.

7.6. Case study

We investigate the interpretable power of CDNC for explaining the reasons of the recommendations. Different attention scores are assigned to different items according to their intent importance. Due to the limited space, we randomly choose an user from Movielens test set for analysis and demonstration. For better illustration, we set a window size of 8. The visualization of the attention

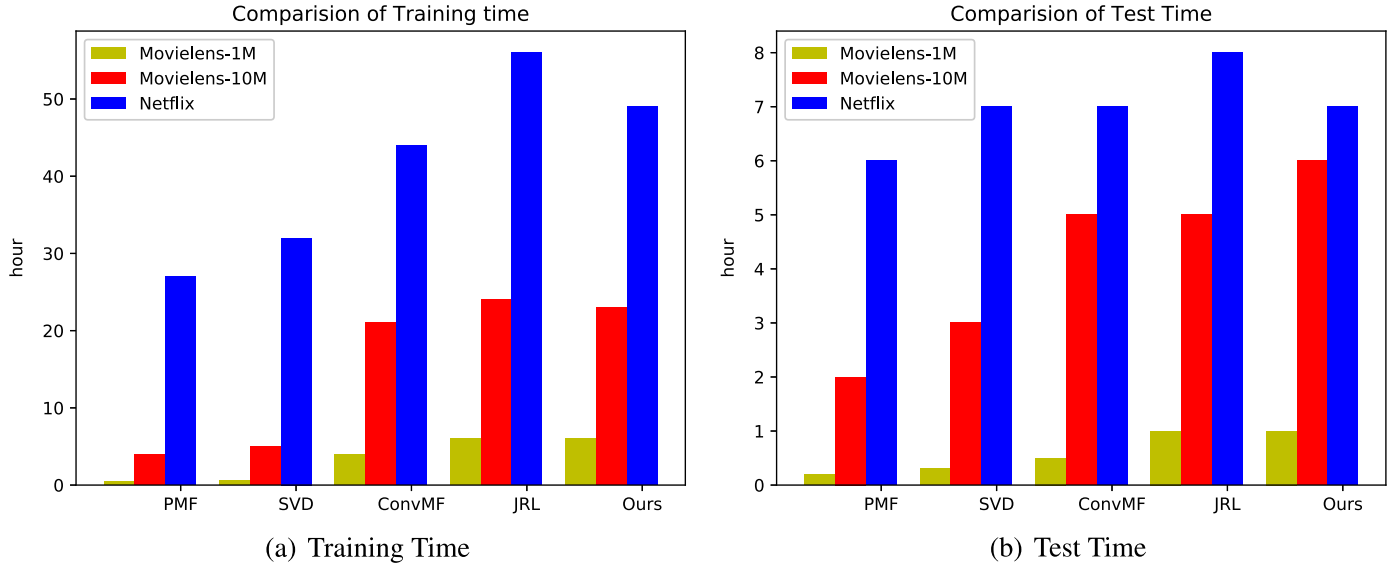

Fig. 2. Comparison of execution time.

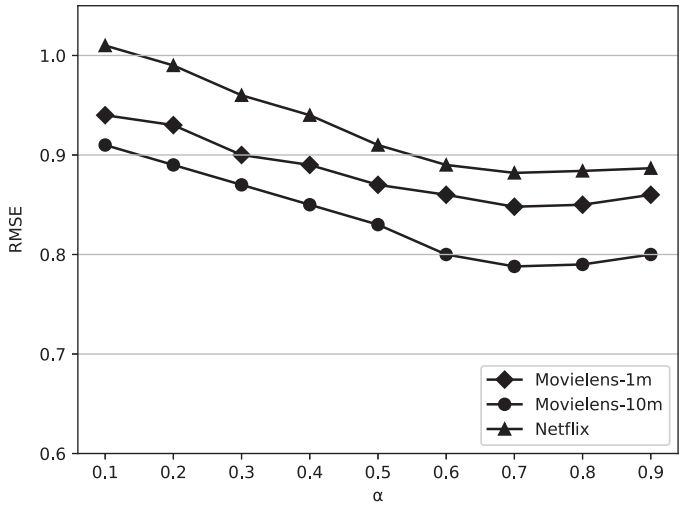

Fig. 3. Testing performance of CDNC methods by varying the parameter ω.

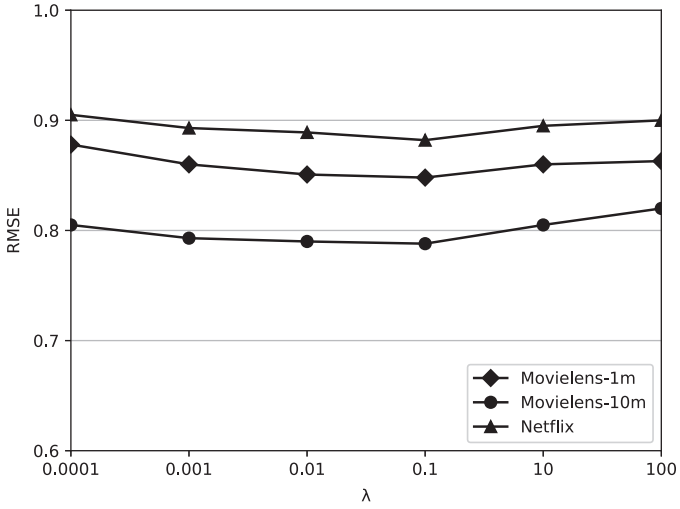

Fig. 5. Testing performance of CDNC methods by varying the L2 regularizer λ.

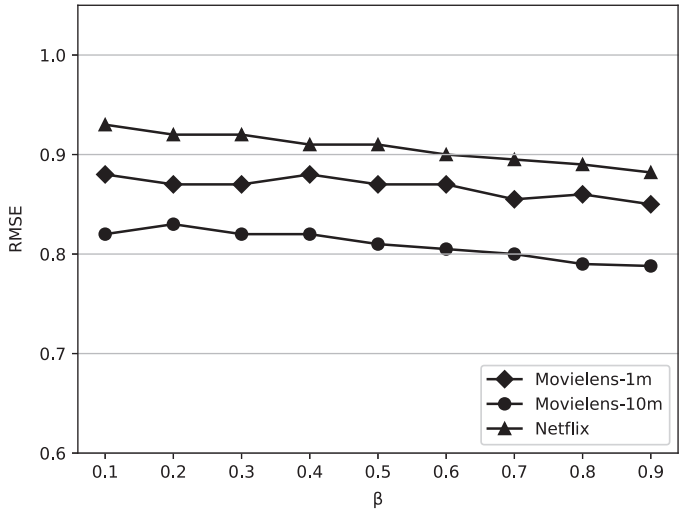

Fig. 4. Testing performance of CDNC methods by varying the attention factor μ.

weights of the items and how the model makes predictions are shown in Fig. 5. The horizontal axis denotes the watched movies by the chosen user, and the vertical axis denotes the movies to be predicted. Each entry of the heatmap denotes how much the watched movie contributes in the process of predicting new items. For example, from Fig. 6, we can observe that the movie *Nikita* contributes most in the prediction of the movie *Men in black*, since these two movies are both action movies. On the other hand, the movie *Escape from New York* contributes little to the prediction of *In the Army Now*, since they belong to different categories. The recommender system focuses selectively on parts of the items to acquire information for recommendation.

We also investigate the power of word-level attention for interpretable recommendation. Different attention scores are assigned to different words according to their importance. We choose the movie *X-Men* to visualize the attention scores on the plot summary words. The color depth indicates the importance degree of the words. The darker the color, the more important the word. As shown in Fig. 7, some words such as *evil plan, fear, war, storm* characterize the essence of the story of the movie.

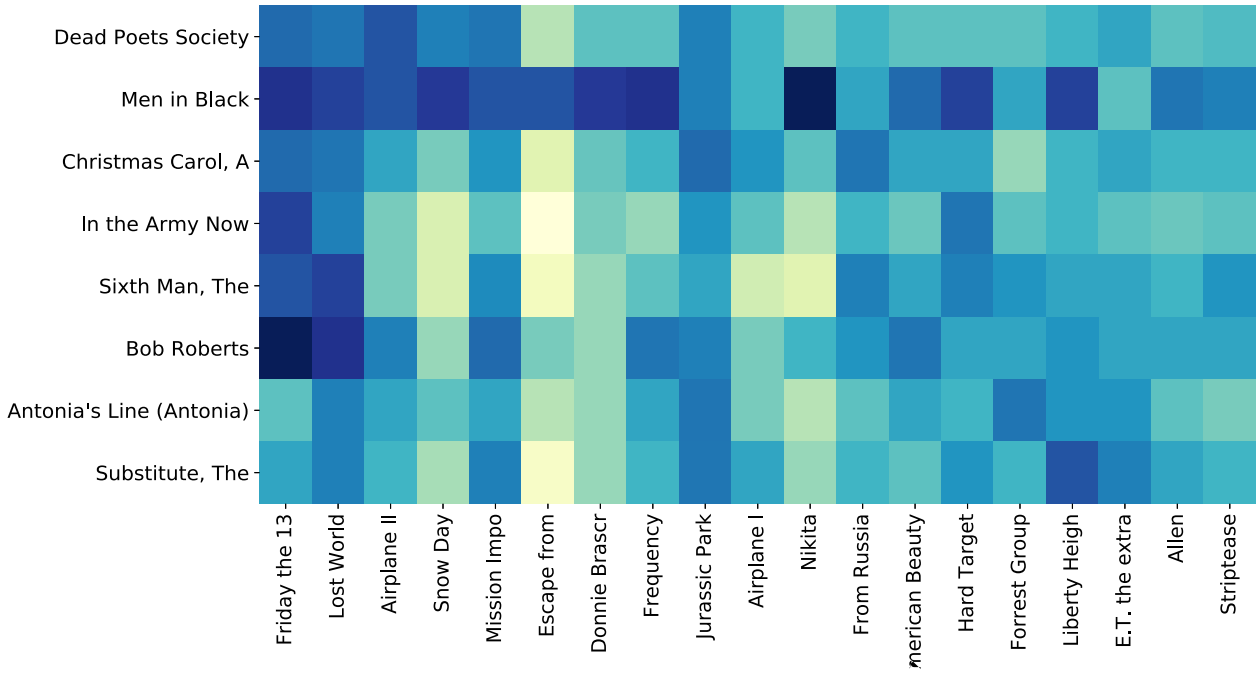

Fig. 6. The visualization of the prediction of CDNC. Each line shows how the historical movies contribute to the movie being predicted.

Fig. 7. The visualization of the plot summary of the movie *X-Men*.

8. Conclusion and future work

In this paper, we proposed a novel contextual-boosted deep neural collaborative filtering model, called CDNC, for rating prediction recommendation task. Our model simultaneously exploited the historical ratings and item introductions that the user interacted with via interactive attention mechanism to learn comprehensive user representation and alleviate the cold start problem. Meanwhile, we also used a hierarchical attention network to capture the textual features of the new-coming item, and learned the item representation composing of historical rating and item introduction information. A feature fusion layer and a prediction layer were finally employed to obtain the prediction results. To verify the effectiveness of our model, we conducted extensive experiments on three widely used real-world datasets. Experimental results showed that the proposed CDNC model achieved the state-of-the-art performance. In addition, our model has strong compatibility. When the ratings or item introductions are missing, our model can still obtain acceptable prediction results. This complements the existing deep textual recommendation models for rating predictions.

In the future, we will extend our CDNC model to pair-wise learning and top-n recommendation. Another emerging direction is to explore more auxiliary information such as knowledge base to further improve the recommendation performance.

Credit Author Statement

1. Shuai Yu is responsible for implementing the work and collecting data.
2. Min Yang is responsible for writing this paper and helps implement the work.

3. Qiang Qu helps to implement the work and rewrite the paper.
4. Ying Shen helps to review and edit the paper.

Conflict of interest

This piece of the submission is being sent via mail.

Credit authorship contribution statement

Shuai Yu: Conceptualization, Data curation, Methodology, Formal analysis. **Min Yang:** Writing - original draft. **Qiang Qu:** Project administration, Writing - original draft. **Ying Shen:** Validation, Writing - review & editing.

Acknowledgments

This work was also partially supported by the SIAT Innovation Program for Excellent Young Researchers (Grant No. Y8G027), the CAS Pioneer Hundred Talents Program, Shenzhen Fundamental Research Project (No. JCYJ20180302145633177), and Natural Science Foundation of Guangdong Province (No. 2018A030313943). Min Yang was sponsored by CCF-Tencent Open Research Fund.

References

Bobadilla, J., Ortega, F., Hernando, A., & Gutierrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems, 46*, 109–132.
Chelliah, M., & Sarkar, S. (2017). Product recommendations enhanced with reviews. In *Proceedings of RecSys*.
Chen, L., & Wang, F. (2017). Explaining recommendations based on feature sentiments in product reviews. In *Proceedings of the 22nd international conference on intelligent user interfaces* (pp. 17–28). ACM.
Cui, Q., Wu, S., Liu, Q., & Wang, L. (2016). A visual and textual recurrent neural network for sequential prediction. arXiv:1611.06668.
Devooght, R., & Bersini, H. (2016). Collaborative filtering with recurrent neural networks. arXiv:1608.07400.
Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks, 18*(5–6), 602–610.
Gunawardana, A., & Shani, G. (2009). A survey of accuracy evaluation metrics of recommendation tasks. *Journal of Machine Learning Research, 10*(Dec), 2935–2962.
He, X., He, Z., Song, J., Liu, Z., Jiang, Y.-G., & Chua, T.-S. (2018). Nais: Neural attentive item similarity model for recommendation. *IEEE Transactions on Knowledge and Data Engineering, 30*(12), 2354–2366.
He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T.-S. (2017). Neural collaborative filtering. In *WWW* (pp. 173–182).
He, X., Zhang, H., Kan, M.-Y., & Chua, T.-S. (2016). Fast matrix factorization for online recommendation with implicit feedback. In *SIGIR* (pp. 549–558). ACM.
Hidasi, B., Karatzoglou, A., Baltrunas, L., & Tikk, D. (2015). Session-based recommendations with recurrent neural networks. *ICLR*.
Hu, M., Peng, Y., & Qiu, X. (2017). Reinforced mnemonic reader for machine comprehension. *CoRR*. arXiv: 1705.02798
Jin, X., Zhou, Y., & Mobasher, B. (2005). A maximum entropy web recommendation system: combining collaborative and content features. In *SIGKDD* (pp. 612–617). ACM.
Kabbur, S., Ning, X., & Karypis, G. (2013). Fism: Factored item similarity models for top-n recommender systems. In *SIGKDD* (pp. 659–667). ACM.
Kim, D., Park, C., Oh, J., Lee, S., & Yu, H. (2016). Convolutional matrix factorization for document context-aware recommendation. In *RecSys* (pp. 233–240). ACM.
Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization arXiv:1412.6980.
Koren, Y. (2008). Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDDinternational conference on knowledge discovery and data mining* (pp. 426–434).
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.
Li, H., Min, M. R., Ge, Y., & Kadav, A. (2017). A context-aware attention network for interactive question answering. In *SIGKDD* (pp. 927–935). doi:10.1145/3097983.3098115.
Li, Y., Nie, J., Zhang, Y., Wang, B., Yan, B., & Weng, F. (2010). Contextual recommendation based on text mining. In *Proceedings of the 23rd international conference on computational linguistics: Posters* (pp. 692–700). Association for Computational Linguistics.
Lu, J., Wu, D., Mao, M., Wang, W., & Zhang, G. (2015). Recommender system application developments: A survey. *Decision Support Systems, 74*, 12–32.
Ma, D., Li, S., Zhang, X., & Wang, H. (2017). Interactive attention networks for aspect-level sentiment classification. In *IJCAI* (pp. 4068–4074).
Ma, H., Yang, H., Lyu, M. R., & King, I. (2008). Sorec: Social recommendation using probabilistic matrix factorization. In *CIKM* (pp. 931–940). ACM.
Mnih, A., & Salakhutdinov, R. R. (2008). Probabilistic matrix factorization. In *NIPS* (pp. 1257–1264).
Ning, X., & Karypis, G. (2011). Slim: Sparse linear methods for top-n recommender systems. In *ICDM* (pp. 497–506). IEEE.
Pei, W., Yang, J., Sun, Z., Zhang, J., Bozzon, A., & Tax, D. M. (2017). Interacting attention-gated recurrent networks for recommendation. In *CIKM* (pp. 1459–1468). ACM.
Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2009). Bpr: Bayesian personalized ranking from implicit feedback. In *UAI* (pp. 452–461). AUAI Press.
Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *WWW* (pp. 285–295). ACM.
Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2002). Incremental singular value decomposition algorithms for highly scalable recommender systems. In *ICCIS* (pp. 27–28). Citeseer.
Singh, A. P., & Gordon, G. J. (2008). Relational learning via collective matrix factorization. In *SIGKDD* (pp. 650–658). ACM.
Srebro, N., Rennie, J., & Jaakkola, T. S. (2005). Maximum-margin matrix factorization. In *Advances in neural information processing systems* (pp. 1329–1336).
Tu, W., Cheung, D. W., Mamoulis, N., Yang, M., & Lu, Z. (2015). Activity-partner recommendation. In *Pacific-asia conference on knowledge discovery and data mining* (pp. 591–604). Springer.
Tu, W., Cheung, D. W., Mamoulis, N., Yang, M., & Lu, Z. (2018). Activity recommendation with partners. *ACM Transactions on the Web (TWEB), 12*(1), 4.
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., … Polosukhin, I. (2017). Attention is all you need. In *NIPS* (pp. 6000–6010).
Wang, C., & Blei, D. M. (2011). Collaborative topic modeling for recommending scientific articles. In *SIGKDD* (pp. 448–456). ACM.
Wang, H., Wang, N., & Yeung, D.-Y. (2015). Collaborative deep learning for recommender systems. In *SIGKDD* (pp. 1235–1244). ACM.
Wang, S., Hu, L., Cao, L., Huang, X., Lian, D., & Liu, W. (2018). *Attention-based transactional context embedding for next-item recommendation*. AAAI.
Wu, C., Wang, J., Liu, J., & Liu, W. (2016). Recurrent neural network based recommendation for time heterogeneous feedback. *Knowledge-Based Systems, 109*, 90–103.
Wu, C.-Y., Ahmed, A., Beutel, A., & Smola, A. J. (2016). Joint training of ratings and reviews with recurrent recommender networks. *ICLR*.
Wu, C.-Y., Ahmed, A., Beutel, A., Smola, A. J., & Jing, H. (2017). Recurrent recommender networks. In *WSDM* (pp. 495–503). ACM.
Wu, H., Zhang, Z., Yue, K., Zhang, B., He, J., & Sun, L. (2018). Dual-regularized matrix factorization with deep neural networks for recommender systems. *Knowledge-Based Systems, 145*, 46–58.
Yu, S., Wang, Y., Yang, M., Li, B., Qu, Q., & Shen, J. (2019). Nairs: A neural attentive interpretable recommendation system. In *Proceedings of the twelfth ACM international conference on web search and data mining* (pp. 790–793). ACM.
Zhang, F., Lu, Y., Chen, J., Liu, S., & Ling, Z. (2017). Robust collaborative filtering based on non-negative matrix factorization and r_1-norm. *Knowledge-Based System, 118*, 177–190. doi:10.1016/j.knosys.2016.11.021.
Zhang, S., Yao, L., Sun, A., Wang, S., Long, G., & Dong, M. (2018). Neurec: On nonlinear transformation for personalized ranking. In *IJCAI* (pp. 3669–3675). doi:10.24963/ijcai.2018/510.
Zhang, Y., Ai, Q., Chen, X., & Croft, W. B. (2017b). Joint representation learning for top-n recommendation with heterogeneous information sources. In *CIKM* (pp. 1449–1458). ACM.
Zhang, Y., Lai, G., Zhang, M., Zhang, Y., Liu, Y., & Ma, S. (2014). Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th international ACM SIGIR conference on research & development in information retrieval* (pp. 83–92). ACM.
Zhao, L., Lu, Z., Pan, S. J., & Yang, Q. (2016). Matrix factorization+ for movie recommendation.. In *IJCAI* (pp. 3945–3951).
Zhao, W., Wang, B., Yang, M., Ye, J., Zhao, Z., Chen, X., & Shen, Y. (2019). Leveraging long and short-term information in content-aware movie recommendation via adversarial training. *IEEE Transactions on Cybernetics*.
Zhao, W., Wang, B., Ye, J., Gao, Y., Yang, M., & Chen, X. (2018). Plastic: Prioritize long and short-term information in top-n recommendation using adversarial training.. In *Ijcai* (pp. 3676–3682).
Zheng, L., Noroozi, V., & Yu, P. S. (2017). Joint deep modeling of users and items using reviews for recommendation. In *WSDM* (pp. 425–434). ACM.