IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 16, 2020, accepted May 1, 2020, date of publication May 6, 2020, date of current version May 19, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2992519

Deep Collaborative Filtering Based on Outer Product

YIHAN WU[1], JITONG WEI[1], JIAN YIN[1], XIN LIU[2], AND JIYONG ZHANG[2]
[1]School of Mechanical and Information Engineering, Shandong University, Weihai 264209, China
[2]School of Automation, Hangzhou Dianzi University, Hangzhou 310018, China

Corresponding author: Jian Yin (yinjian@sdu.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 61971268.

ABSTRACT Recently, deep neural networks are widely used in recommendation systems, but most of them are used to process auxiliary information of recommendation systems such as items' descriptions and images. When it comes to how to learn a better interaction function to model the relation between user latent features and item latent features, which is the most critical step in a recommendation task, most works employ matrix factorization together with the inner product. However, it is sub-optimal because of ignoring many correlations between latent factors. As deep neural networks perform well in building more complex non-linear models, we employ deep neural networks to improve the collaborative filtering algorithm, solving the problem of implicit feedback which is the most common scene in real applications. Some recent work has contributed to finding better interaction function, but these functions are not exact enough to model comprehensive correlations among latent features. In this work, we propose the Convolutional Neural Networks based Deep Collaborative Filtering model (CNN-DCF) to solve the key problem in the recommendation system. Based on the outer product and deep neural networks, we develop a correlation extraction module that can learn high-order correlations between item latent features and user latent features. Extensive experiments on the public implicit feedback dataset Yelp show that the proposed CNN-DCF model brings significant improvements over the state-of-the-art methods.

INDEX TERMS CNN, collaborative filtering, implicit feedback, neural network, outer product.

I. INTRODUCTION

In recent years, with the rapid development of the Internet and the Internet of Things (IOT) [28], data is growing rapidly at an exponential rate. However, massive data not only brings us more information and knowledge but also brings the problem of information overload. So, how to obtain the information in need of large-scale data quickly and accurately has become a major issue in both academia and industry. Nowadays, personalized recommendation technology has become an important method for handling information overload and has been widely used on the Internet. For example, most e-commerce platforms such as Amazon and Taobao, have developed personalized product recommendation engines to increase the total transaction volume (GMV). Besides, online video sites such as Netflix and YouTube use the recommendation system to increase video clicks.

The associate editor coordinating the review of this manuscript and approving it for publication was Noor Zaman.

The conventional recommendation system methods can be divided into three major categories [11], [14]: collaborative filtering methods [3], content-aware methods [2], and hybrid methods [5], [7]. At present, the most widely used method is the collaborative filtering algorithm [6], [23], [35], [38] based on matrix factorization (MF) [18], [25], [34]. MF is used to obtain user latent vector and item latent vector (also termed as user embedding and item embedding) respectively, and the correlation between user latent factors and item latent factors is modeled by an interaction function. Since MF-based approaches are highly accurate and easily scalable in addressing CF problems, some works based on MF make great progress [36], [37].

Nowadays, most matrix-based methods use inner product as the interaction function [26], [29], [33], [40]. However, applying an inner product has some drawbacks. The inner product simply combines user latent vector and item latent vector linearly, which is not exact enough to capture the complex features of the user's historical behaviors. Specifically, applying the inner product is reasonable only when the

dimensions of latent vectors are independent of each other. But because these dimensions can be considered as certain characteristics of users and items, which is not independent in the real world, using the inner product is not rational in the real recommendation tasks.

In order to find a more effective interaction function, following [31], we adopt the outer product of the user latent vector and item latent vector as an interaction function to model the relationship between latent factors in this work. Reference [31] uses the convolutional neural network (CNN) to learn the output of the outer product in the ConvNCF model. Compared with the conventional MF methods which use inner product, it gets better results because using outer product models more correlations between latent factors. Although CNN has achieved good results in the field of image processing [10], [13], [20], [24], it is irrational to use CNN in the ConvNCF model. The inspiration of CNN is that individual cortical neurons only respond to stimuli in a receptive field because features in a receptive filed have a certain correlation. Therefore, it is not necessary for each neuron to perceive the global image, while local connectivity can reduce the number of parameters. However, the interaction map obtained by the outer product obviously does not have the same local connectivity as images. So, it is unreasonable to use CNN to learn the high-order correlation between the latent factors.

In this paper, we present an effective method for recommending in an implicit feedback scenario. The contributions of our work are:

1) We develop a CNN based Deep Collaborative Filtering model (CNN-DCF) in this work. CNN-DCF combines deep learning with collaborative filtering based on the outer product, which improves the speed and accuracy when recommending for users.
2) We present a correlation extraction module to learn more comprehensive correlations between latent factors. By this special module, we can model high-order correlations among entries in interaction map and recommend for users more accurately.
3) We conduct extensive experiments on public dataset Yelp, which show that CNN-DCF performs better than state-of-the-art implicit collaborative filtering methods, proving the rationality and effectiveness of CNN-DCF.

II. RELATED WORK

A. MPLICIT FEEDBACK IN RECOMMENDATION SYSTEM

In general, there are two typical scenarios when recommending to a user: explicit feedback which the user has a direct rating on the item and implicit feedback which does not directly represent the user's propensity. The user's interest can only be judged by whether the item has been clicked in implicit feedback scenario. Although explicit information is more reliable than implicit information, explicit information is difficult to acquire. Compared with explicit feedback, implicit feedback is more widely used in our daily lives, such as browsing websites and purchasing items. Since users do not give ratings to items clearly in most cases, users' preferences can be collected more easily and conveniently in implicit feedback. Therefore, it is necessary to study recommendation technology based on implicit feedback. However, because of its natural sparsity, it is more difficult in implicit feedback research.

Let U be the set of all users and I the set of all items. u and i are a user and an item in the system respectively. In user-item interaction matrix $\boldsymbol{Y} \in \mathbb{R}^{M \times N}$ with implicit feedback, M denotes the number of users and N denotes the number of items. Each entry y_{ui} in Y can be defined as

$$y_{ui} = \begin{cases} 1, & \textit{if user u has interacted with item i} \\ 0, & \textit{otherwise} \end{cases} \tag{1}$$

Currently, there are some methods for recommending based on implicit feedback. Rendle *et al.* propose a ranking recommendation framework called Bayesian Personalized Ranking (BPR) [29]. It transforms the ordering problem into a binary classification problem through the partial order relationship between the item pairs. Wang *et al.* propose a collaborative topic regression (CTR) model, combining collaborative filtering and probabilistic topic modeling [8]. Specifically, each item is generated by a topic model such as LDA and contains a latent variable which can capture the user's performance for the item based on historical scoring information. Depending on the topic model, this method can not only recommend existing items but also predict new items to solve common cold-start problems in recommendation systems.

B. APPLICATION OF DEEP LEARNING IN RECOMMENDATION SYSTEM

Deep Learning provides an end-to-end and non-linear modeling approach that improves the recommendation system's efficiency. Particularly, Deep Learning provides different neural network architectures for different types of data and is suitable for scenarios where multiple types of data coexist in the recommendation system. Deep Learning has become the main research direction in the field of recommendation systems. Various Deep Learning models include Multilayer Perceptron (MLP), Autoencoder, Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Attention model, and Deep Reinforcement Learning have been applied in different recommendation tasks.

Deep Learning can be used to handle large-scale data so that it is widely used in collaborative filtering recommendation issues. Salakhutdinov *et al.* first use Deep Learning in the recommendation system and propose a model that uses Restricted Boltzmann machines for collaborative filtering [27]. Wu *et al.* propose Collaborative Denoising Auto-Encoders (CDAE) which uses DAE (Denoising Autoencoder) for top-N recommendation systems [39]. Besides, because Gated Recurrent Unit (GRU) [9] can capture the dependencies between behaviors in the session,

Hidasi *et al.* [4] apply the collaborative filtering method based on the Recurrent Neural Network (RNN) [12] to use the historical behavior records in the current session to predict the probability of clicking each item in the next step. DeepFM [17] integrates the decomposition machine and MLP seamlessly. The Factorization-Machine captures linear interaction information between each pair of features through inner product and addition operation while MLP using nonlinear activation function and deep network structure to model higher-order feature interactions. Through learning from Factorization-Machine MLP paralleled, the prediction results including the combination of low-order and high-order features are obtained finally. In addition to blending with existing recommendation models such as matrix factorization and Factorization-Machine, MLP can also improve the recommendation by extract feature expression. Google's Wide & Deep model [15] consists of a wide module and a deep module, where the wide module is a single-layer perceptron, which is a generalized linear model; the deep module is a multilayer perceptron. By combining the wide module with the deep module, this model can improve memory and generalization capabilities simultaneously. The memory capability is obtained by the wide module by modeling the original features, and the generalization capability is realized by the deep module by extracting more generalized and abstract feature representations.

III. DEEP COLLABORATIVE FILTERING BASED ON OUTER PRODUCT

In this part, we compare the inner product and outer product in section A. And then, we introduce the framework of the CNN-DCF model and the objective function applied in this work in section B and section C respectively.

A. MF BASED ON OUTER PRODUCT

In the collaborative filtering algorithm based on MF, latent vectors can be obtained by MF. And the different dimensions of user latent vectors and item latent vectors can be viewed as the different features of users and items. When the inner product is employed as an interaction function to model the correlations between user latent features and item latent features, it all based on the hypothesis that dimensions of latent space are independent of each other. However, these latent factors also mean users' and items' features which are interrelated with each other in the real world. Therefore, just using an inner product to model the interaction function is illogical. Besides using the inner product, the other interaction function used in [33] is simply concatenating user latent vectors and item latent vectors. However, experiments in [31] show that just concatenating latent vectors doesn't perform well. The reason is that concatenation is too simple to learn complex information from latent features.

To learn more complicated information from data, the outer product is a better choice. Compared with methods of employing the inner product or simply concatenating latent factors, the outer product can learn more information between latent features. We show equations of the inner product (2) and outer product (3) respectively,

$$\boldsymbol{a} \odot \boldsymbol{b} = \boldsymbol{a}_{\boldsymbol{d}_1}\boldsymbol{b}_{\boldsymbol{d}_1} + \boldsymbol{a}_{\boldsymbol{d}_2}\boldsymbol{b}_{\boldsymbol{d}_2} + \ldots + \boldsymbol{a}_{\boldsymbol{d}_K}\boldsymbol{b}_{\boldsymbol{d}_K} \tag{2}$$

$$\boldsymbol{a}^{\boldsymbol{T}} \otimes \boldsymbol{b}^{\boldsymbol{T}} = \boldsymbol{a}^{\boldsymbol{T}}\boldsymbol{b} = \begin{pmatrix} \boldsymbol{a}_{\boldsymbol{d}_1}\boldsymbol{b}_{\boldsymbol{d}_1} & \cdots & \boldsymbol{a}_{\boldsymbol{d}_1}\boldsymbol{b}_{\boldsymbol{d}_K} \\ \vdots & \ddots & \vdots \\ \boldsymbol{a}_{\boldsymbol{d}_K}\boldsymbol{b}_{\boldsymbol{d}_1} & \cdots & \boldsymbol{a}_{\boldsymbol{d}_K}\boldsymbol{b}_{\boldsymbol{d}_K} \end{pmatrix} \tag{3}$$

where a, b are row vectors and denote K-dimensional latent vectors. From (2), (3), we can figure out that the output of outer product includes the output of inner product (diagonal elements). So outer product can learn more correlations between a and b than inner product.

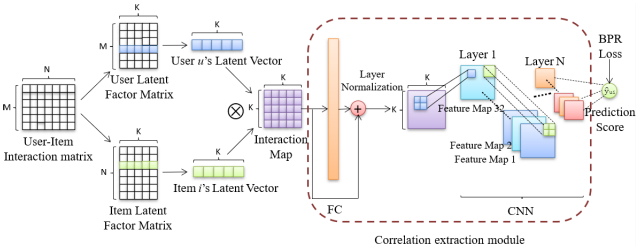


FIGURE 1. CNN-DCF model's framework.

B. CNN-DCF

To model the pairwise correlations between the dimensions of latent vectors and learn high-order correlations more comprehensively, we propose our CNN-DCF model. Fig.1 illustrates the CNN-DCF model. The aim of model is to predict a personalized score $\hat{y}_{ui}$ which can reflect the preference of user u for item i.

1) OUTER PRODUCT

We obtain user latent matrix $\boldsymbol{P} \in \mathbb{R}^{M \times K}$ and item latent matrix $\boldsymbol{Q} \in \mathbb{R}^{N \times K}$ by decomposing the User-Item interaction matrix. M is the number of users and N is the number of items. K denotes the number of latent features as well as embedding size. User latent vector $\boldsymbol{p_u}$ and item latent vector $\boldsymbol{q_i}$ are the u-th row of $\boldsymbol{P}$ and i-th row of $\boldsymbol{Q}$ respectively.

In CNN-DCF model, applying the outer product on $\boldsymbol{p_u}$ and $\boldsymbol{q_i}$, we can obtain an interaction map $\boldsymbol{O}$ via (4)

$$\boldsymbol{O} = \boldsymbol{p}_{\boldsymbol{u}}^{\boldsymbol{T}} \otimes \boldsymbol{q}_{\boldsymbol{i}}^{\boldsymbol{T}} = \boldsymbol{p}_{\boldsymbol{u}}^{\boldsymbol{T}}\boldsymbol{q}_{\boldsymbol{i}} \tag{4}$$

where $\boldsymbol{O} \in \mathbb{R}^{K \times K}$. As we have mentioned, compared with the result of the inner product, the interaction map contains more correlations between users' and items' latent features, which is more meaningful in the real world. What's more, compared with using concatenation simply [1], [32], [33], it is more useful to model latent factor's correlations explicitly when using a deep learning model on sparse data.

2) CORRELATION EXTRACTION MODULE

To extract more comprehensive information from the interaction map, we consider developing a correlation extraction module after the interaction map. This module aims to find

a better function to learn more details from the interaction map. We first flat the interaction map into a vector, then feed it into a fully connected layer so that we can learn correlations between all entries in the interaction map. In practice, around the fully connected layer, we also apply a residual connection to speed up the convergence. Besides, to reduce the number of parameters, we employ CNN. Specifically, we reshape the output of the fully connected layer into a 2-D matrix and feed it into a CNN network. The calculations are as follows.

Given the interaction map $\boldsymbol{O} \in \mathbb{R}^{K\times K}$, we first flat it to vector $\boldsymbol{V}_1$ of size K^2 and feed $\boldsymbol{V}_1$ into the fully connected layer (5).

$$f(\boldsymbol{V}_1) = a(\boldsymbol{W}_1^T \boldsymbol{V}_1 + \boldsymbol{b}_1) \tag{5}$$

We apply a residual connection around fully connected layer (6),

$$\boldsymbol{V}_2 = \boldsymbol{V}_1 + f(\boldsymbol{V}_1) \tag{6}$$

where $\boldsymbol{W}_1, \boldsymbol{b}_1$ are parameters of the fully connected layer, a is the ReLU activation function. The purpose of the residual connection is to avoid gradient vanishing and speed up the convergence.

Before CNN layers, $\boldsymbol{V}_2$ is reshaped into a 2-D matrix $\boldsymbol{E} \in R^{K\times K}$. Then, a set of CNN layers followed by a fully connected layer are applied to get the prediction score $\hat{y}_{ui}$. The computation is as follows:

$$\hat{y}_{ui} = \boldsymbol{W}\,CNN\,(\boldsymbol{E}) + \boldsymbol{b} \tag{7}$$

3) RATIONALITY OF CNN-DCF

Here are the reasons why CNN-DCF is more reasonable to learn complete information from the interaction map which is the output of the outer product.

In the interaction map, each element o_{ij} is the product of k_i-th user latent factor and k_j-th item latent factor, reflecting the correlation between them. The fully connected layer captures the correlation among every element in interaction map $\boldsymbol{O}$. Of course, an MLP can be simply employed after the fully connected layer, which is termed as ONCF-mlp in [31]. However, applying CNN above the fully connected layer is a better choice. Here are two reasons:

- **Comparison with ConvNCF.** We have learned correlations between all elements in the interaction map via the fully connected layer. As a result, despite the local connectivity of CNN, the correlations among different elements of the interaction map will not be lost when employing CNN-DCF.
- **Comparison with ONCF-mlp.** In CNN, each filter is replicated across the entire visual field and these replicated units share the same parameters. Thus, applying correlation extraction module instead of MLP can reduce the number of parameters significantly.

C. OBJECTIVE FUNCTION

Since recommendation can be considered as a personalized ranking task, we optimize the model with a ranking-aware objective function in CNN-DCF. In implicit feedback, observed interactions should be ranked higher than unobserved interactions. So, we use the Bayesian Personalized Ranking (BPR) objective function for optimization [29]. Being different from usual approach which items are ranked by sorting them according to prediction data, BPR optimizes for correctly ranking item pairs instead of scoring single them.

In the BPR optimization criterion, there is a partial order $>_u$. If user u has viewed item i but not item j, we can assume that u prefers i over j, i.e., $i >_u j$. Besides, there are two assumptions when applying BPR:

1) All users act independently with each other.
2) The ordering of each pair of items for a user is independent of the ordering of every other pair.

In order to find the best personalized ranking, the following posterior probability should be maximized.

$$p(\Theta|>_u) \propto p(>_u|\Theta)\,p(\Theta)$$

To simply the formula, we apply Θ instead of parameters in the model. Based on the above two assumptions, $p(>_u|\Theta)$ can be rewritten as follows:

$$\prod_{u\in U} p(>_u|\Theta) = \prod_{(u,i,j)\in U\times I\times I} p(i>_u j|\Theta)^{\delta((u,i,j)\in\mathcal{D})} \cdot(1-p(i>_u j|\Theta))^{\delta((u,i,j)\notin\mathcal{D})}$$

where

$$\delta(b) := \begin{cases} 1, & if\ b\ is\ true \\ 0, & else \end{cases},$$

and $\mathcal{D} := \{(u,i,j)|\, i\in y_u^+ \wedge j\in y_u^-\}$. y_u^+ denotes items that have interacted with user u and y_u^- denotes items that have not interacted with user u. Hence, this formula can be simplified to:

$$\prod_{u\in U} p(>_u|\Theta) = \prod_{(u,i,j)\in \mathrm{D}} p(i>_u j|\Theta)$$

More specifically, in this work, we define the individual probability that user u prefers item i to item j as $p(i>_u j|\Theta) := \sigma(\hat{y}_{ui}(\Theta) - \hat{y}_{uj}(\Theta))$, where $\sigma(x) := \frac{1}{1+e^{-x}}$ is the sigmoid function.

We have discussed the likelihood function. To complete the Bayesian modeling approach, we assume prior density $p(\Theta)$ is a normal distribution with zero mean and variance-covariance matrix Σ_Θ, i.e., $p(\Theta) \sim N(0, \Sigma_\Theta)$. In the following, we set $\Sigma_\Theta = \lambda_\Theta I$ to reduce the number of unknown hyperparameters. Now we can obtain the personalized ranking objective function BPR objective function by calculating the maximum posterior estimator.

$$\begin{aligned} BPR :&= \ln p(\Theta|>_u) \\ &= \ln p(>_u|\Theta)\,p(\Theta) \\ &= \ln \prod_{(u,i,j)\in D} \sigma\left(\hat{y}_{ui}(\Theta) - \hat{y}_{uj}(\Theta)\right) p(\Theta) \end{aligned}$$

$$= \sum_{(u,i,j)\in D} \ln \sigma\left(\hat{y}_{ui}(\Theta) - \hat{y}_{uj}(\Theta)\right) + \ln p(\Theta)$$
$$= \sum_{(u,i,j)\in D} \ln \sigma\left(\hat{y}_{ui}(\Theta) - \hat{y}_{uj}(\Theta)\right) - \lambda_\Theta \|\Theta\|^2$$

where λ_Θ are model specific regulation parameters.

IV. EXPERIMENTS

A. EXPERIMENTAL SETTINGS

To evaluate our model comprehensively, we conduct experiments in order to answer the following questions:

Q1 Do our proposed model performs better than the state-of-the-art recommendation models?

Q2 Do applying outer product instead of inner product improve the performance of models?

Q3 Is learning more comprehensive correlations from interaction map helpful?

Q4 Is the CNN-DCF a better choice in application compared with other models based on the outer product?

1) DATASET

We implement a series of experiments on the public dataset Yelp. Yelp open dataset is a user rating dataset published by the review website Yelp. It is an explicit feedback data, so we transform it into implicit data by marking each entry as 0/1(0/1 indicate whether the user reviewed the item). Because the original Yelp dataset is too sparse to evaluate recommendation models, we filter out users and items with less than 10 interactions. As a result, there are 25,815 users, 25,677 items, and 730,791 times interactions in the final dataset.

2) EVALUATION PROTOCOLS

We adopt the leave-one-out evaluation to measure the performance of our models. For each user, we select the last item the user interacted with and other 999 items the user has never interacted with to form the test dataset. We employ Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) as the evaluation standard to measure the performance. HR@k is a recall-based metric which measures if the testing item is in the top-k position among all items in the test set. NDCG@k assigns higher scores to items at top-k ranks. For all models below, we calculate the average of every user's evaluation score.

3) BASELINES

We compare CNN-DCF with these methods:

ItemPop This model recommends the most popular item to users, which is not a personalized model. This model is always taken as a baseline for recommender methods.

MF-BPR [29] This is an MF model which adopts BPR function as the loss function.

MLP [33] This model concatenates user latent vector and item latent vector, then uses MLP to learn interaction function. It is a Neural network-based Collaborative Filtering (NCF) model.

TABLE 1. HR@K performance.

Models	HR@5	HR@10	HR@20
ItemPop	0.0710	0.1147	0.1732
MF-BPR	0.1752	0.2817	0.4203
MLP	0.1766	0.2831	0.4203
JRL	0.1858	0.2922	0.4343
NeuCF	0.1881	0.2958	0.4385
ConvNCF	0.1901	0.3010	0.4403
CNN-DCF	**0.1936**	**0.3042**	**0.4438**

JRL [38] This model is an NCF model too. It uses inner product to model correlations between user latent vector and item latent vector. Different from Generalized Matrix Factorization (GMF), it uses multiple hidden layers above the inner product.

NeuCF [33] This model combines the advantage of GMF and MLP to learn interaction function between user latent factors and item latent factors. It unifies the advantages of linearity of MF and non-linearity of MLP.

ConvNCF [31] This model employs multiple convolution layers above the interaction map to learn high-order information from the user's past behaviors.

4) PARAMETER SETTINGS

For MLP, JRL and NeuCF, we pre-train their embedding layers by MF-BPR. As for models based on outer product, namely ConvNCF and CNN-DCF, we pre-train their embedding layers via OMF (Outer product-based MF), which places a fully connected layer after the interaction map.

As for the correlation extraction module in CNN-DCF, we employ 5-layers CNN and 6-layers CNN when $K = 32$ and $K = 64$ respectively. Besides, we set each hidden layer with 32 feature maps and we use a stride of 2 for both $K = 32$ and 64. Especially, for our proposed model, we optimize the embedding layers and the rest part of the model by using mini-batch Adagrad with different learning rates respectively. Furthermore, for a fair comparison, we employ L_2 regularization to avoid overfitting for all compared methods. The regularization coefficients are tuned in the range of [0.001, 0.01, 1, 10, 100].

B. PERFORMANCE COMPARISON (Q1)

Setting k=5, 10, 20 respectively, we compare the top-k recommendation performance on the Yelp dataset by employing different models (Table 1 and 2). Table 3 shows the average improvement of CNN-DCF over the baseline respectively. From these tables, we find the following conclusions.

- ConvNCF and CNN-DCF, which both apply the outer product as interaction function, perform better than other models which use the inner product. It proves that applying the outer product improves the performance of models.

TABLE 2. NDCG@K performance.

Models	NDCG@5	NDCG@10	NDCG@20
ItemPop	0.0365	0.0505	0.0652
MF-BPR	0.1104	0.1447	0.1796
MLP	0.1103	0.1446	0.1792
JRL	0.1177	0.1519	0.1877
NeuCF	0.1189	0.1536	0.1895
ConvNCF	0.1243	0.1543	0.1919
CNN-DCF	**0.1268**	**0.1589**	**0.1940**

TABLE 3. Average improvement over baseline.

Baselines	RI
ItemPop	+122.90%
MF-BPR	+7.05%
MLP	+6.99%
JRL	+4.03%
NeuCF	+3.27%
ConvNCF	+2.16%

RI indicates the average improvement of CNN-DCF over the baseline when applied in Yelp

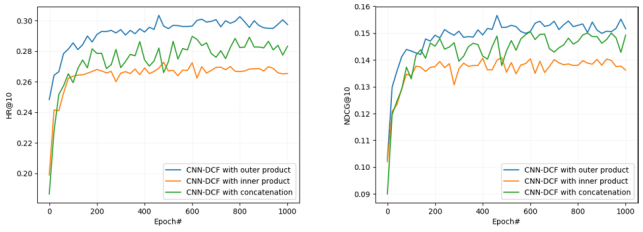


FIGURE 2. HR@10, NDCG@10 of applying different operations above the embedding layer in each epoch.

- CNN-DCF achieves better performance and gets high improvement over all other models. It proves that learning more comprehensive correlations among interaction map's entries is more effective.

C. PERFORMANCE OF OUTER PRODUCT (Q2)

To show the efficacy of the outer product more intuitively, we replace it with inner product and concatenation respectively. With HR@K and NDCG@K as evaluation criterion respectively, we compare their performance in each epoch (Fig.2). From Fig.2, we observe that CNN-DCF with outer product outperforms other methods on both HR@10 and NDCG@10. The improvement demonstrates the efficacy of outer product which models the correlation between different embedding dimensions. It proves the rationality of learning more comprehensive correlations by applying the outer product.

D. PERFORMANCE OF CORRELATION EXTRACTION MODULE (Q3)

To verify the efficacy of the correlation extraction module, for embedding size is 32 and 64 respectively, the HR@10 and NDCG@10 in each epoch of CNN-DCF and ConvNCF are shown in Fig.3 and Fig.4. From Fig.3 and Fig.4, we observe that no matter the embedding size is 32 or 64, CNN-DCF always performs better than ConvNCF, which proves our conclusion that the correlation extraction module is more reasonable than CNN because correlation extraction module learns more comprehensive information from interaction map.

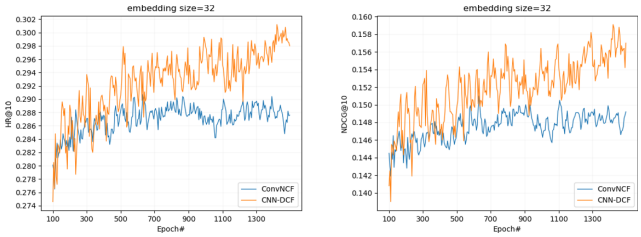


FIGURE 3. HR@10, NDCG@10 of applying CNN-DCF and ConvNCF in each epoch for embedding size = 32.

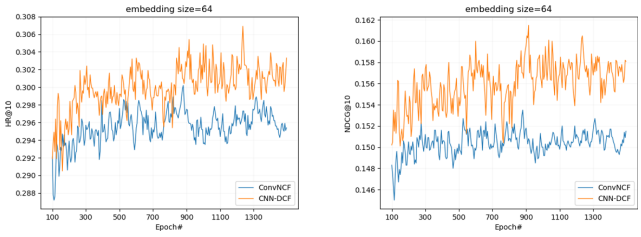


FIGURE 4. HR@10, NDCG@10 of applying CNN-DCF and ConvNCF in each epoch for embedding size = 64.

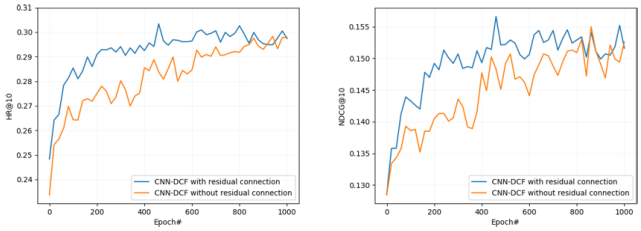


FIGURE 5. HR@10, NDCG@10 of applying CNN-DCF with and without residual connection in each epoch.

Besides, to verify the validity of the residual connection in correlation extraction module, we compare the performance of CNN-DCF with and without residual connection (Fig.5). Fig. 5 shows that CNN-DCF with the residual connection converges faster than CNN-DCF without residual connection. It proves that applying residual connection can speed up the convergence and improves efficiency and productivity in real world applications.

E. ADVANTAGES OF CNN-DCF IN APPLICATION (Q4)

As the recommendation system is using more and more in our daily life, the practicability of our proposed model should be under consideration. To measure the practicability of CNN-DCF clearly, we compare CNN-DCF with ConvNCF and ONCF-mlp which both perform well in implicit

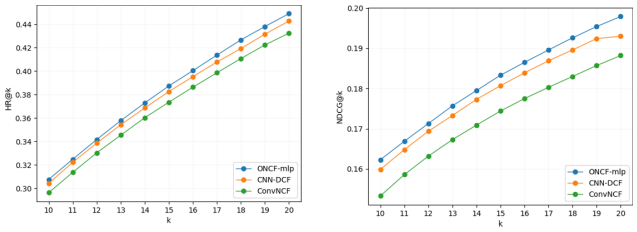


FIGURE 6. HR@k, NDCG@k of applying CNN-DCF, ConvNCF and ONCF-mlp where K ranges from 10 to 20.

recommendation tasks. For embedding size is 64, we set top-K item recommendation as the evaluation criterion where K ranges from 10 to 20. We employ MLP with three hidden layers in ONCF-mlp. Furthermore, we use the same CNN network in CNN-DCF and ConvNCF to ensure the fairness of the experiment. Fig. 6 shows the best performance of each model in the experiments.

Comparing ONCF-mlp and CNN-DCF separately, we find that their best performances are very close. Although ONCF-mlp performs a bit better than CNN-DCF because of its huge number of parameters, ONCF-mlp has some disadvantages:

1) There are a large number of parameters in ONCF-mlp. More parameters require large memories to store the model and lots of training data to train the model well. So, the model will be hard to train because it requires a lot of data to avoid overfitting.
2) The performance of the model is unstable. Because of the training difficulty, ONCF-mlp performs much worse than ConvNCF sometimes [31].
3) In order to make sure good generalization performance, it needs a lot of manual labor to adjust the regularization coefficient carefully.

The difficulty of training ONCF-mlp reduces efficiency and increases costs in industrial applications. These drawbacks make it hard for ONCF-mlp to be applied in reality.

From Fig. 6, we also observe that CNN-DCF performs much better than ConvNCF. It proves the effectiveness of learning more comprehensive information from interaction map. In addition to accuracy, computational time is also to be considered in applications. It takes 4.48ms and 4.82ms respectively for CNN-DCF and ConvNCF to predict a user's favorite item among 1000 items on average. It demonstrates that CNN-DCF is more efficient when being applied in industrial applications.

n conclusion, considering both recommendation performance and training difficulty, CNN-DCF is the best choice when being applied in practice.

V. CONCLUSION AND FUTURE WORK

In this work, we explore deep neural networks for collaborative filtering based on the outer product. Specifically, we propose CNN-DCF which can learn more comprehensive correlations between user's latent features and item's latent features. Experiments on Yelp dataset show that CNN-DCF perform better than other models in recommendation task with implicit feedback.

In the future, we will combine our model with content-aware method and employ more auxiliary information like item's description, user's geographic location and time information to improve the efficacy of CNN-DCF. Besides, in order to explore CNN-DCF to use in real application scenarios like e-commerce, we will optimize our model and reduce its time consumption. We consider employing DenseNet or ResNet to explore the potential of CNN-DCF. What's more, as implicit feedback is the most common scenario, we will be committed to finding better sampling methods to improve recommendation ability in implicit feedback.

REFERENCES

[1] A. Beutel, P. Covington, S. Jain, C. Xu, J. Li, V. Gatto, and E. H. Chi, "Latent cross: Making use of context in recurrent recommender systems," in *Proc. 11th ACM Int. Conf. Web Search Data Mining WSDM*, 2018, pp. 46–54.

[2] A. Chen, "Context-aware collaborative filtering system: Predicting the user's preference in the ubiquitous computing environment," in *Location and Context-Awareness* (Lecture Notes in Computer Science), vol. 3479, Berlin, Germany: Springer, 2005.

[3] *An Integrated Approach to TV & VOD Recommendations Archived 6 June 2012 at the Wayback Machine*, Red Bee Media, London, U.K., 2012.

[4] B. Hidasi, A. Karatzoglou, and L. Baltrunas, "Session-based Recommendations with Recurrent Neural Networks," 2015, *arXiv:1511.06939*. [Online]. Available: https://arxiv.org/abs/1511.06939

[5] B. Liu, Y. Fu, Z. Yao, and H. Xiong, "Learning geographical preferences for point-of-interest recommendation," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining KDD*, 2013, pp. 1043–1051.

[6] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, "Item-based collaborative filtering recommendation algorithms," in *Proc. 10th Int. Conf. World Wide Web WWW*, Aug. 2001, pp. 285–295.

[7] C. Cheng, H. Yang, I. King, and M. R. Lyu, "Fused matrix factorization with geographical and social influence in location-based social networks," in *Proc. 26th AAAI*, Jul. 2012, pp. 17–23.

[8] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining KDD*, Aug. 2011, pp. 448–456.

[9] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN Encoder–Decoder for statistical machine translation," 2014, *arXiv:1406.1078*. [Online]. Available: https://arxiv.org/abs/1406.1078

[10] D. H. Hubel and T. N. Wiesel, "Receptive fields and functional architecture of monkey striate cortex," *J. Physiol.*, vol. 195, no. 1, pp. 215–243, Mar. 1968.

[11] D. Jannach, M. Zanker, A. Felfernig, *Recommender Systems: An Introduction*. Cambridge, U.K.: Cambridge Univ. Press, 2010.

[12] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.

[13] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biol. Cybern.*, vol. 36, no. 4, pp. 193–202, Apr. 1980.

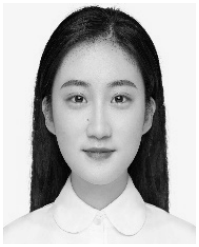
[14] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.

[15] H. Cheng, "Wide & deep learning for recommender systems," in *Proc. 10th ACM Conf. Recommender Syst.*, Sep. 2016, pp. 7–10.

[16] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," 2016, *arXiv:1608.06993*. [Online]. Available: http://arxiv.org/abs/1608.06993

[17] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "DeepFM: A factorization-machine based neural network for CTR prediction," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 1725–1731.

[18] H. Shan and A. Banerjee, "Generalized probabilistic matrix factorizations for collaborative filtering," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2010, pp. 1025–1030.
[19] H. Wang, N. Wang, and D. Yeung, "Collaborative deep learning for recommender systems," in *Proc. 21st ACM KDD*, 2015, pp. 1235–1244.
[20] K. Fukushima, "Neocognitron," *Scholarpedia*, vol. 2, no. 1, p. 1717, 2007.
[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[22] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, no. 5, pp. 359–366, Jan. 1989.
[23] G. Linden, B. Smith, and J. York, "Amazon.Com recommendations: Item-to-item collaborative filtering," *IEEE Internet Comput.*, vol. 7, no. 1, pp. 76–80, Jan. 2003.
[24] M. Matsugu, K. Mori, Y. Mitari, and Y. Kaneda, "Subject independent facial expression recognition with robust face detection using a convolutional neural network," *Neural Netw.*, vol. 16, nos. 5–6, pp. 555–559, Jun. 2003.
[25] M. Wu, "Collaborative filtering via ensembles of matrix factorizations," in *Proc. KDD Cup Workshop*, Aug. 2007, pp. 43–47.
[26] P. K. Jain and K. Ahmad, "5.1 definitions and basic properties of inner product spaces and Hilbert spaces," in *Functional Analysis*, 2nd ed. New Delhi, India: New Age International, 1995, ch. 5, sec. 1, p. 203.
[27] R. Salakhutdinov, A. Mnih, and G. Hinton, "Restricted Boltzmann machines for collaborative filtering," in *Proc. 24th Int. Conf. Mach. Learn. ICML*, 2007, pp. 791–798.
[28] *Rouse, Margaret, Internet of Things (IoT)*, IOT Agenda, Aug. 2019. [Online]. Available: https://internetofthingsagenda.techtarget.com/definition/Internet-of-Things-IoT
[29] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proc. UAI*, 2009, pp. 452–461.
[30] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," 2015, *arXiv:1505.00387*. [Online]. Available: http://arxiv.org/abs/1505.00387
[31] X. He, X. Du, X. Wang, F. Tian, J. Tang, and T.-S. Chua, "Outer product-based neural collaborative filtering," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 2227–2233.
[32] X. He and T.-S. Chua, "Neural factorization machines for sparse predictive analytics," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Aug. 2017, pp. 355–364.
[33] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. Chua, "Neural collaborative filtering," in *Proc. 26th WWW*, Apr. 2017, pp. 173–182.
[34] X. Luo, M. Zhou, H. Leung, Y. Xia, Q. Zhu, Z. You, and S. Li, "An Incremental-and-Static-Combined scheme for Matrix-Factorization-Based collaborative filtering," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 1, pp. 333–343, Jan. 2016.
[35] X. Luo, Z. You, M. Zhou, S. Li, H. Leung, Y. Xia, and Q. Zhu, "A highly efficient approach to protein interactome mapping based on collaborative filtering framework," *Sci. Rep.*, vol. 5, no. 1, p. 7702, Jul. 2015, doi: 10.1038/srep07702.
[36] X. Luo, M. Zhou, S. Li, Y. Xia, Z. You, Q. Zhu, and H. Leung, "An efficient second-order approach to factorize sparse matrices in recommender systems," *IEEE Trans. Ind. Informat.*, vol. 11, no. 4, pp. 946–956, Aug. 2015.
[37] X. Luo, M. Zhou, Y. Xia, and Q. Zhu, "An efficient non-negative Matrix-Factorization-Based approach to collaborative filtering for recommender systems," *IEEE Trans. Ind. Informat.*, vol. 10, no. 2, pp. 1273–1284, May 2014.
[38] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Adv. Artif. Intell.*, vol. 2009, pp. 1–19, Oct. 2009.
[39] Y. Wu, C. DuBois, A. X. Zheng, and M. Ester, "Collaborative denoising auto-encoders for Top-N recommender systems," in *Proc. 9th ACM Int. Conf. Web Search Data Mining WSDM*, 2016, pp. 153–162.
[40] Y. Zhang, Q. Ai, X. Chen, and W. B. Croft, "Joint representation learning for Top-N recommendation with heterogeneous information sources," in *Proc. ACM Conf. Inf. Knowl. Manage.*, Nov. 2017, pp. 1449–1458.



YIHAN WU was born in Jinan, Shandong, China, in 2000. She is currently pursuing the bachelor's degree in computer science and technology with Shandong University. She is also a Volunteer Service Advanced Individual with Shandong University. Since 2019, she has been working on Recommender System supported by the Natural Science Foundation of China. She received the second-class scholarship twice and more than ten provincial and university level awards.

JITONG WEI was born in Jinan, Shandong, China, in 1999. She is currently pursuing the bachelor's degree in computer science and technology with Shandong University. Since 2019, she has been working on Recommender System. She received the second-class scholarship twice and more than ten provincial and university level awards.

JIAN YIN received the Ph.D. degree from Shandong University, China, in 2015. He is currently an Associate Professor with the School of Mechanical, Electrical and Information Engineering, Shandong University. His research interests include recommender systems and intelligent information processing.

XIN LIU received the Ph.D. degree from Nanyang Technological University, Singapore, in 2012. From 2012 to 2014, he was a Postdoctoral Researcher with LSIR Laboratory, EPFL, Switzerland. He is currently an Associate Professor with the School of Automation, Hangzhou Dianzi University. His research interests include recommender systems, trust modeling, natural language processing, and computer vision.

JIYONG ZHANG received the B.S. and M.S. degrees in computer science from Tsinghua University, in 1999 and 2001, respectively, and the Ph.D. degree in computer science from the Swiss Federal Institute of Technology at Lausanne (EPFL), in 2008. He is currently a Distinguished Professor with Hangzhou Dianzi University. His research interests include intelligent information processing, machine learning, data sciences, and recommender systems.