Contents lists available at ScienceDirect

Information Sciences

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

Deep learning based recommender system using cross convolutional filters ☆

Seungyeon Lee [a,b], Dohyun Kim [b,*]

[a] Department of Industrial and Systems Engineering, Ohio State University, USA
[b] Department of Industrial and Management Engineering, Myongji University, Republic of Korea

A R T I C L E   I N F O

Article history:
Received 6 April 2021
Received in revised form 26 December 2021
Accepted 18 January 2022
Available online 25 January 2022

Keywords:
Recommender systems
Deep learning
Convolutional neural network
Cross convolutional filters

A B S T R A C T

With the recent development of online transactions, recommender systems have increasingly attracted attention in various domains. The recommender system supports the users' decision making by recommending items that are more likely to be preferred. Many studies in the field of deep learning-based recommender systems have attempted to capture the complex interactions between users' and items' features for accurate recommendation. In this paper, we propose a recommender system based on the convolutional neural network using the outer product matrix of features and cross convolutional filters. The proposed method can deal with the various types of features and capture the meaningful higher-order interactions between users and items, giving greater weight to important features. Moreover, it can alleviate the overfitting problem since the proposed method includes the global average or max pooling instead of the fully connected layers in the structure. Experiments showed that the proposed method performs better than the existing methods, by capturing important interactions and alleviating the overfitting issue.

1. Introduction

With the recent development of online transactions such as e-commerce, various types of items and services are being provided through online platforms. Users have difficulty finding, from among the vast number of items offered in the online marketplace, the product that best suits them. Recommender systems have proven to be a useful technology [2] for support of users' decision-making by reducing the number of alternatives they may prefer through analysis of their needs or behavior, etc.

Traditional recommender systems recommend items based on past user behavior or similarities between users or items. They can be classified into collaborative filtering, content-based filtering, and hybrid recommender system [17]. Collaborative filtering is a method of analyzing consumer preferences based on past consumer behavior and providing recommendations based on similar users or items [3]. Content-based filtering analyzes the contents of the user's favorite items and recommends similar items [22]. Hybrid recommender system combines two or more recommendation techniques [1]. However, traditional recommender systems have limitations when the user has purchased a very small number of items, when a new user (or item) is added, or when a very large amount of data is needed to be analyzed [13]. In addition, traditional rec-

This research was supported by Basic Science Research Program throughthe National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT Future Planning (NRF-2017R1E1A1A01077375)

* Corresponding author.
E-mail address: ftgog@mju.ac.kr (D. Kim).

https://doi.org/10.1016/j.ins.2022.01.033
0020-0255/© 2022 Elsevier Inc. All rights reserved.

ommender systems cannot effectively model certain types of content information such as texts and images provided by users or for items.

In recent years, deep learning-based models have been studied in the field of recommender systems [32]. Deep learning, one of the machine learning methods, consists of a complex artificial neural network structure. Deep learning extracts high-dimensional representations from data through a nonlinear structure and uses them to solve various problems. Also, it can handle large amounts of data and various types of high-dimensional data such as images and voices. Deep learning has achieved the best performance in various fields such as speech recognition, image recognition, natural language processing and recommender systems [14]. Many deep learning-based recommender systems have achieved higher performance than traditional recommender systems, since they can utilize content information about users and items and handle high-dimensional features efficiently. However, the existing deep learning-based methods have a disadvantage in that they do not effectively consider the interactions between user and item features, because they handle the features of both independently.

In this study, we propose a new recommender system using convolutional network (CNN) [15] with the customized filters to model user and item content information. The proposed method applies CNN to capture useful nonlinear relationships between users and items (i.e., high-dimensional interactions). The convolution layer in the proposed method preserves and models the relationships between users and items by learning the outer product matrix of the user and item feature vectors using cross convolutional filters instead of small square filters. Cross convolutional filters are two filters in which a vertical weight vector and a horizontal weight vector are combined in a cross shape. Convolution of the outer product matrix using the cross convolutional filters can give greater weight to the interactions with important features, thus capturing a variety of useful nonlinear relationships between users and items. Moreover, the proposed method does not use a multi-layer perceptron, which can incur the overfitting problem due to its complex structure [28], but directly connects the feature maps of CNN and the output node through global average pooling or global max pooling. This has the advantage of mitigating the overfitting problem and allowing nonlinear interaction to be used directly for rating prediction. In addition to the above structure, the proposed method incorporates the bidirectional encoder representations form transformer (BERT) [5] to extract semantic information from item profiles, such as movie titles. In a recommender system, cold start problems often occur when new users or items are added to the system because there is no knowledge about them. In the proposed method, the embedding vector of the item profile obtained by the pre-trained BERT model can alleviate the cold start problem because the embedding vector can reflect the semantic information of the new item's profile even without knowledge of it. In summary, the proposed method offers novelty in identifying nonlinear relationships between users and items by performing CNNs with cross convolutional filters and reducing the cold start problem by using BERT. The main contributions in this paper are summarized as follows:

- The proposed method develops a recommender system that effectively models the user-item interaction using various latest techniques, such as BERT, CNN and cross convolutional filters, based on the outer product matrix between the user and item feature vectors extracted from the content information.
- To capture useful nonlinear relationships between user and item, the proposed method applies CNN. The proposed convolutional layers use cross convolutional filters instead of general square filters to model user-item interactions represented by the outer product matrix.
- Cross convolutional filters can detect which features of the item (or user) are important from the point of view of the user (or item) by giving greater weights to interactions with important features.
- The proposed method does not use a multi-layer perceptron but directly connects the feature maps of CNN and the output node through global average pooling or global max pooling to mitigate the overfitting problem and allow nonlinear interactions to be used directly for rating prediction.
- Since the proposed method only uses content information, it does not cause the cold start problem that occurs in many recommendation systems.

The rest of the paper is organized as follows: Section 2 introduces related works. In Section 3, we explain the proposed method. Section 4 discusses experimental setup and results on various datasets. Finally, the conclusion is drawn in Section 5.

2. Related works

2.1. Deep learning-based recommender systems

Deep learning learns high-dimensional features from data, and in this process, feature engineering is automatically performed. By designing the model structure flexibly, deep learning can effectively handle various types of data such as text and image. Moreover, it is possible to effectively reduce the training time of the model and easily extract the meaningful features inherent in the given data by incorporating a pre-trained model for complex tasks such as image recognition. In order to utilize these advantages of deep learning, deep learning-based models have been studied in the field of recommender systems. Most of the deep learning-based recommender systems outperform the traditional recommender systems [23].

NCF [9], DMF [29], ConvNCF [8], SocialCDL [31], CoDAE [20] and ECAE [30] are methods that combine deep learning with collaborative filtering in traditional recommender systems. These models learn useful features using the rating matrix and predict the rating for a user-item pair. NCF [9] is a method that implements a part of matrix factorization [12] as a neural network. NCF uses the feature vectors created from matrix factorization as the input of a multi-layer perceptron to model the high-dimensional relation(s) between them. DMF [29] directly uses the rating matrix as the input. Rating information for user and item is entered into two multi-layer perceptrons, respectively, and the rating is predicted by the cosine similarity between the feature vectors extracted from the two perceptrons. ECAE [30] is based on the collaborative denoising autoencoder (CDAE) [26], using the user or item ratings as input data. To mitigate the performance degradation and cold start issues caused by sparse rating information, ECAE uses a generative network to extract soft targets from hard targets typically marked with a rating of 0 or 1. The final prediction is obtained as the weighted sum of both outputs of the generation network and the retraining network, which is an autoencoder model using the soft targets as input. SocialCDL [31] predicts ratings with rating matrix and users' social information, and it consists of matrix factorization for ratings and sparse stacked denoising autoencoder (SSDAE) [31]. SSDAE is an autoencoder model to learn deep representations from social information. SocialCDL predicts ratings as the sum of the feature vectors of matrix factorization and hidden features obtained from SSDAE. CoDAE [20] takes correlations between users with multiple roles of rater, truster and trustee into account to learn robust representations from ratings and social networks. CoDAE uses three separate autoencoders, each taking as input a rating matrix, a truster matrix, and a trustee matrix, which is a transpose of the truster matrix. Additionally, CoDAE utilizes a shared weight matrix to learn implicit common information between input units corresponding to the same users to solve the sparseness problem in social networks and a related regularization term to learn correlations between user features obtained by the three subnetworks.

Models based on the collaborative filtering, such as DMF, ConvNCF, SocialCDL, CoDAE and ECAE, can predict ratings using an existing rating matrix. These models can work effectively when there is no available user and item content information. However, since they are collaborative filtering-based methods, it is hard to utilize content information. On the contrary, the proposed method is not based on collaborative filtering, but rather uses user and item content information to predict ratings of users, including new users.

Wide & Deep Learning [4], DeepFM [6], DNNRec [11], DLRM [19] and DCN-M [24] are methods that combine deep learning and hybrid techniques in traditional recommender systems. In these models, user and item content information are fed into the neural network to predict the rating for the corresponding user-item pair. Although it does not directly utilize the similarities between users or items, collaborative and content-based filtering are performed indirectly when predicting the rating of a specific user-item, because the predicted value is affected by the ratings of other user-item pairs with similar content information among the training data. Wide & Deep Learning [4] is composed of wide and deep models. First, the wide model is a linear regression model that handles not only the features of the data but also the interactions between features through cross-product feature transformation terms. Cross-product feature transformation represents the interactions between binary features to capture the co-occurrence of the features. On the other hand, the deep model is a multi-layer perceptron for modeling the high-dimensional representations of features. DeepFM [6] is a model that combines FM (factorization machine) [21] and a multi-layer perceptron. The FM models the two-way interactions of features, and the multi-layer perceptron models the high-dimensional interactions. DNNRec [11] consists of the embedding layer and the multi-layer perceptron. After the categorical variables are embedded, all of the embedding vectors are concatenated. They are used as the input of the multi-layer perceptron, and the rating is predicted based on the output of the multi-layer perceptron. DLRM [19] is composed of the embedding layer, the interaction layer, and the multi-layer perceptron. In addition to embedding categorical variables through the embedding layer, continuous variables are also embedded through the multi-layer perceptron. The interaction layer computes the inner product between the embedding vectors of all variables to capture the interactions between the variables. The inner product values are simply concatenated with all embedding vectors and then input into the multi-layer perceptron. The purpose of DCN-M [24] is to learn effective explicit and implicit feature interactions. DCN-M consists of a cross network to learn explicit feature interactions and a deep network to learn complementary implicit interactions.

As seen earlier, existing deep learning-based recommender system models handle features independently through a multi-layer perceptron without directly considering the interactions between user and item features. In recent years, several attempts have been made to develop recommender systems to directly reflect interactions between users and items using CNN architecture. ConvNCF [8] consists of the embedding layer and convolutional layer. Only user and item IDs, not content information, are used as input to implement collaborative filtering. After embedding user and item IDs, ConvNCF obtains an outer product matrix between user and item embedding vectors to represent the user-item interactions. The outer product matrix (i.e., interaction map) are input into the convolutional layer. In the convolutional layer, general convolutional filter is applied to learn high-order interactions. CFM [27] is composed of the embedding layer, the self-attention pooling layer and the convolutional layer. Content information is used as input. After embedding all user and item features, the attention mechanism is used to compute the importance of each feature, and then the pooling is performed. CFM calculates the outer product for all pairs of embedding vectors to represent the feature interactions. All outer product matrices are used as inputs to the convolutional layer. In convolutional layer, 3D convolutional filter is applied to learn high-order interaction signals.

ConvNCF and CFM calculate the outer product matrix between the embedding vectors and use it as input to the CNN model, like the proposed method. However, there is a big difference between the existing methods and the proposed method in the shape of the convolutional filter. ConvNCF and CFM model the interactions using a basic square filter which is suitable for extracting local features from the input feature map. By using the square filter, the interaction of a specific user-item feature pair in the interaction map is modeled considering some features near that pair. Therefore, it does not capture the global characteristics of that pair.

2.2. Convolutional Neural Networks (CNNs)

The convolutional neural network (CNN) [15] is the representative structure of the artificial neural networks. It is widely used in the field of image recognition. CNN generally consists of the convolutional layer, the pooling layer, and the fully connected layer. The convolutional layer is the most important component of the CNN. It uses the weight matrix, which is called the filter or kernel. The filter moves all input data in a constant stride and simultaneously performs a weighted sum of the input data and the corresponding elements of the filter. This process outputs a single feature map. In general, to detect several useful features of the input data, the convolutional layer outputs several feature maps using several filters. The output feature maps are input into the next hidden layer. The convolutional layer is characterized by weight sharing. Since one filter moves the whole of a feature map, similar features are detected regardless of location. In addition, weight sharing reduces the complexity of the model by effectively reducing the number of weights. The pooling layer serves to reduce the size of the feature map. Similarly to the convolutional layer, the pooling filter moves all input data in a constant stride, but simply takes the maximum (called max pooling) or the average value (called average pooling) from the input data within the filter area. In other words, the pooling filter has no weights to be trained. Finally, the fully connected layer is used. The flattened matrix classifies the image through the fully connected layers. In a typical CNN, the weights of the fully connected layer make up most of the parameters, thereby incurring overfitting problems [28,16]. The global pooling layer [16] can be used to alleviate this overfitting problem. Global pooling extracts one maximum (called global max pooling) or average value (called global average pooling) from each feature map using the pooling filter with the same size as the feature map.

3. Deep Learning based Recommender System using Cross Convolutional Filters

In this section, we propose a new recommender system using CNN with cross convolutional filters modified from the general square filter. The proposed method effectively models the user-item interactions, capturing useful nonlinear relationships between user and item features using cross convolutional filters. The proposed recommender system consists of: (1) variable embedding, (2) convolution with cross convolutional filters, and (3) rating prediction. First, the user and item content information (variables) are embedded through the embedding layers, and then the outer product matrix between user and item embedding vectors is obtained. The matrix represents an interactions map between user and item features. Followed by this embedding layer, the matrix is input to the convolutional layer, and nonlinear features are extracted using two cross convolutional filters. Then, in the last convolutional layer, feature maps are directly used for predicting rating through global average or max pooling. The overall schematic of the proposed method is shown in Fig. 1. In the following sections, these steps are explained in detail.

3.1. Motivation

Many factors affect users' ratings for items. Among them, there are many things that we do not know directly, such as the user's preference or the characteristics of an item that suits the user well. Therefore, finding the hidden relationships

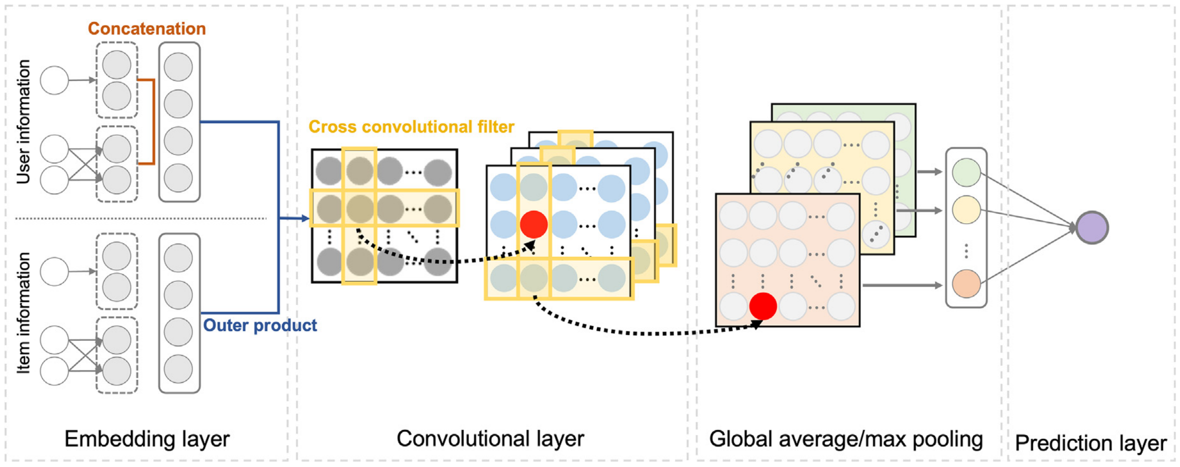

Fig. 1. Overall schematic of proposed method.

between users and items that we don't know is important for predicting ratings. To this end, we focus on hidden relationships. The purpose of the proposed method is to find hidden relationships using the content information of users and items to provide personalized recommendations. In other words, the proposed method is based on utilizing only content information without identifying users and items. For this purpose, we developed a deep learning-based model that directly models the interaction between user and item information and extracts useful non-linear relationships from them. The use of content information can also help solve the cold start problem. In [10], the content information obtained from social media sources was used to generate user profiles, which alleviated the cold start problem. Similarly, since the proposed method utilizes the content information of features the users and items have, recommendations for a new user can be easily made, even if there is no interaction between new users and items. Therefore, the proposed method can solve the cold start problem frequently encountered in many recommender systems, especially using the rating matrix, such as collaborative filtering-based models [25,18].

3.2. Variable embedding

Variables related to user and item content information are composed of categorical and continuous variables. Categorical variables represent exclusive categories or groups such as job information, not meaningful numerical values such as continuous variables. Categorical variables are generally converted to dummy variables. However, a dummy variable is very sparse and cannot express the meaning of categories, since it is expressed as a binary vector (i.e. a one-hot vector) that gives 1 only to the element corresponding to the category. Embedding maps a dummy variable to a dense representation in a feature space through a weight matrix (called an embedding table). In other words, a numerical feature vector for a categorical variable can be extracted. Eq. (1) expresses an embedding vector w_i^T (also called a feature vector) corresponding to the i-th category, where $W \in R^{c \times d}$ is the weight matrix and e_i is a one-hot vector of the i-th category, and c and d indicate the number of categories and the dimensions of the embedding vector, respectively. The embedding vector of the i-th category is equal to the i-th row vector of the embedding table.

$$w_i^T = e_i^T W \tag{1}$$

The continuous variable is converted into an embedding vector using a multi-layer perceptron, which serves to reduce the dimensions of the continuous variable. After embedding each variable of the user and item through different embedding layers, all of the embedding vectors for user variables are concatenated, and the same process is done for the item.

3.3. Convolution with cross convolutional filters

The convolution step is similar to the normal process of the CNN. The biggest difference between the proposed method and the typical CNN is the filter shape. Compared with the CNN's small-square-shape filter, the proposed method uses a cross-shape filter. The cross convolutional filters consist of two filters: vertical and horizontal filters. The sizes of the vertical and horizontal filters are $g \in R^{p_u \times 1}$ and $h \in R^{1 \times p_v}$, respectively, where p_u and p_v are the vertical and horizontal dimensions of the feature map. The convolution is performed using the two filters, simultaneously. Fig. 2 shows a simple operation of the cross convolutional filters. The representation for an element of input feature map is extracted through the weighted sum of the horizontal and vertical filters passing through that element. The two filters operate on all elements of the input feature map, and the output has the same size as the input feature map.

In this study, cross convolutional fiters are used to effectively model the interactions between users and items represented as a feature map in the convolution step. In order to represent interactions between users and items as a feature map, a outer product matrix between user and item embedding vectors extracted from the embedding layers is obtained.

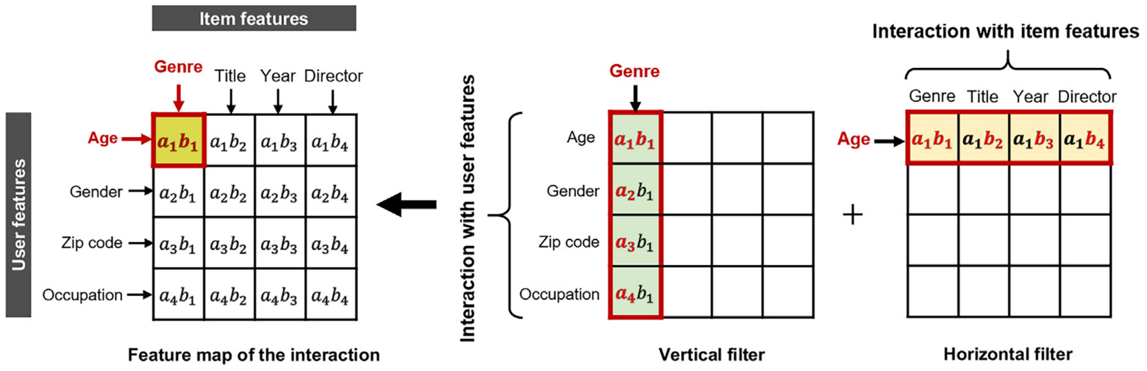
Item features
Genre
Title
Year
Director
User features
Age
Gender
Zip code
Occupation
a_1b_1 a_1b_2 a_1b_3 a_1b_4
a_2b_1 a_2b_2 a_2b_3 a_2b_4
a_3b_1 a_3b_2 a_3b_3 a_3b_4
a_4b_1 a_4b_2 a_4b_3 a_4b_4
Feature map of the interaction
Interaction with user features
Vertical filter
+
Interaction with item features
Horizontal filter

Fig. 2. Simple operation of cross convolutional filters.

By calculating the outer product, all features in the user embedding vector take all possible interactions with the features of the item embedding vector, and the same process is done for the item. In other words, all features are not handled independently, but are calculated as outer product to represent interactions. This can be seen as a feature map of the interactions. The left side of Fig. 2 shows an example of a feature map of interaction. The figure shows the outer product between the user embedding vector $\mathbf{a} = [a_1, a_2, a_3, a_4]^T$ and the item embedding vector $\mathbf{b} = [b_1, b_2, b_3, b_4]^T$. Through the filters, the interaction $a_i b_j$ between the user feature a_i and the item feature b_j is modeled considering interactions with all item features in the case of a_i and with all user features in the case of b_j.

For the convolutional layer, we define $F \in R^{p_u \times p_v}$ as the feature map of the interactions, where p_u and p_v represent the dimensions of the user and the item embedding vector, respectively. $F_{ij}^{l,k}$ is the interaction between the i-th user feature and the j-th item feature for k-th output feature map in l-th convolutional layer. $F^{0,1}$ is the input data for the first convolutional layer. Eq. (2) expresses the k-th output feature map of $(l+1)$-th convolutional layer.

$$F_{ij}^{l+1,k} = f\left(\sum_{m=1}^{M^l}\left(\sum_{u=1}^{p_u} g_u^{l+1,m,k} F_{uj}^{l,m} + \sum_{v=1}^{p_v} h_v^{l+1,m,k} F_{iv}^{l,m}\right)\right) \tag{2}$$

where M^l is the number of output feature maps for the l-th layer (i.e., the number of input feature maps for the $(l+1)$-th layer) and $f(\cdot)$ is the activation function. Also, $g \in R^{p_u \times 1}$ and $h \in R^{1 \times p_v}$ represent the vertical and horizontal filters, respectively, and $g_u^{l+1,m,k}$ indicates the u-th element of the vertical filter for the m-th input feature map in the $(l+1)$-th layer.

$$\begin{aligned} F_{ij}^{1,k} &= f\left(\sum_{u=1}^{p_u} g_u^{1,1,k} F_{uj}^{0,1} + \sum_{v=1}^{p_v} h_v^{1,1,k} F_{iv}^{0,1}\right) \\ &= f\left(\sum_{u=1}^{p_u} g_u^{1,1,k} a_u b_j + \sum_{v=1}^{p_v} h_v^{1,1,k} a_i b_v\right) \end{aligned} \tag{3}$$

Eq. (3) shows the operation for the first convolutional layer (i.e., $M^l = 1$). As shown in Eq. (3), the interaction between a_i and b_j is not simply modeled as $a_i b_j$, but rather, interactions with other features are simultaneously considered to model the global characteristics of the feature pair.

It is noted that cross convolutional filters can give more weight to interactions with important features through weight sharing, and can take into account features that are more important than non-essential features to capture useful nonlinear relationships between users and items. Cross convolutional filters can detect which features of the item are important from the user's point of view and which features of the user are important from the item's point of view. Fig. 3 shows an example of weight sharing for the horizontal filter with the greatest weight for the genre. Because of the weight sharing, the greatest weight is given to the interaction between the genre and all user features. Modeling the relationship between a specific feature pair considering all interactions with other features may seem to increase the complexity of the model. However, this is solved through the weight sharing property of the convolutional filter. The proposed method can capture the interactions between features from various viewpoints by outputting multiple feature maps, and can model high-dimensional relations by using multiple convolutional layers.

3.4. Rating prediction

In the prediction step of the proposed method, the global pooling layer [16] is used, instead of the fully connected layer, to alleviate the overfitting problem in CNN. The proposed method extracts one feature vector representing all feature maps by applying global average pooling or global max pooling for each feature map in the last convolutional layer. The vector is directly connected to the output node. In other words, by extracting a useful feature vector from feature maps modeled from

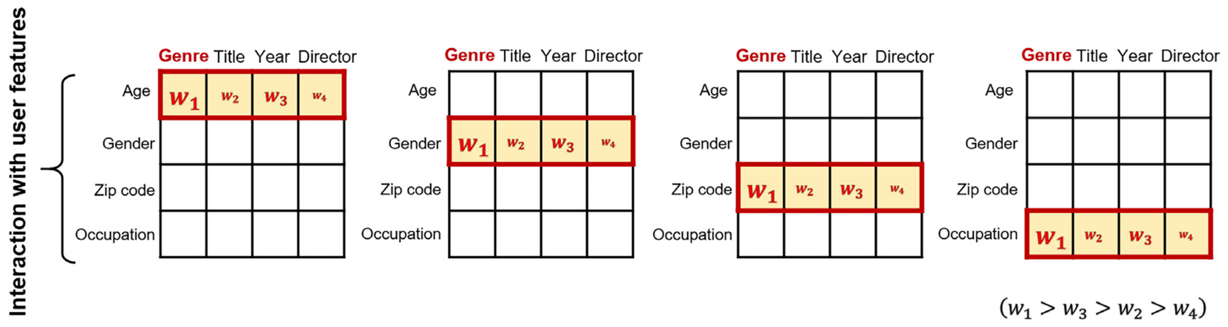
Interaction with user features
Genre Title Year Director
Age
Gender
Zip code
Occupation
w_1 w_2 w_3 w_4
$(w_1 > w_3 > w_2 > w_4)$

Fig. 3. Example of weight sharing for horizontal filter with greatest weight for genre.

various perspectives and linking it with the output node, high-dimensional interaction is directly utilized for rating prediction.

4. Experiments

4.1. Experimental setup

Computational experiments were conducted using two datasets to compare the performance of the proposed method in terms of accuracy with other deep learning-based recommender systems including DLRM [19], DNNRec [11], DCN-M [24] and ECAE [30].

Similarly to the proposed method, DNNRec, DLRM and DCN-M use content information as input and extract feature vectors using different embedding layers for each variable. However, in the case of DNNRec, the extracted feature vectors are simply combined and then used as the input of a multi-layer perceptron to predict the rating. In DLRM, after calculating the dot product among all of the embedding vectors, the dot product is concatenated with the embedding vectors and entered into the multi-layer perceptron. DCN-M embeds only the dummy variables and uses the continuous variables as it is. The embedding vectors are fed into the cross network and deep network. ECAE is a collaborative filtering-based method, and the experiments using this model were performed to compare the proposed method with the collaborative filtering-based model.

To demonstrate the effect of cross convolutional filters, we also conducted the additional experiments with the proposed model using the general square filter (3×3). Furthermore, as mentioned in Section 3, the cross convolutional filters can detect which features of the item (or user) are important from the point of view of the user (or item). To verify this, we checked the weights of the vertical and horizontal filters of the first convolutional layer in the proposed method trained with the optimal hyper-parameters. The weights of all vertical and horizontal filters were multiplied by 10^3 and the standard deviation of each feature, and then the absolute value of each weight were averaged for all filters to reflect the deviation of each feature and clearly represent the difference in weights in Fig. 6.

As mentioned above, the proposed method predicts a rating using only content information. In this way, it is possible to perform a recommendation with only content information of a new item (or user) even if there is no information on the new item (or user). Therefore, the proposed method eliminates the cold start problem. Further experiments were performed to see if the proposed method could provide useful recommendations for new users without the cold start problem. For experiments, the test dataset consists of ”new” users, not included in the training and validation datasets. New users were randomly composed of 10%, 20%, and 30% of all users in dataset. All models were newly trained using the optimal hyper-parameters found in comparative experiments.

Since all of the models are composed of different structures, we defined protocols that are commonly used in experiments for the models and optimized the hyper-parameters of each model respectively. We utilized the stochastic gradient descent (SGD) optimizer with a 0.01/0.001 learning rate and initialized parameters using Xavier initialization. All of the models used 0.33 LeakyReLU and sigmoid activation functions in the hidden and output layers, respectively. All of the datasets were processed with a batch size of 32, and all of the variables were embedded with the same size. Additionally, we experimented by adjusting the hyper-parameters of the total number of epochs, hidden layers, hidden nodes and embedding vector size.

The datasets were divided into training data, validation data, and test data. After randomly dividing the dataset into 90:10, 10% was used as test data, 10% of the other 90% was used as validation data, and the rest was used as training data.

As performance measures, RMSE (root mean squared error) and MAE (mean absolute error) were used, and as the loss function for training, MSE (mean squared error), typically employed with regression problems, was used.

4.2. Datasets

The datasets used to validate the proposed method were MovieLens 1 M and 100 K [7], which are the movie ratings evaluated by users. Not all users can rate all movies, so there are user-item pairs that do not have a rating value. In general, if there is no rating value, it is utilized by treating it as zero, but this study did not consider the pairs without the rating. That is, the model was trained and evaluated using the actual ratings of the user-item pairs.

Detailed dataset descriptions and statistics are as follows. MovieLens 1 M dataset consists of information on 6,040 users and 3,883 movies, along with about 1 million ratings for user-item pairs. MovieLens 100 K dataset consists of 943 users, 1,632 movies, and about 100,000 ratings. The two datasets consist of the same variables for items and users, and the rating is an integer value from 1 to 5. In addition, all users have at least 20 sets of rating information.

Since the proposed method predicts ratings only using the content information without user and item IDs, user and item IDs were not used in the experiments. User information consists of age, gender, occupation and zip code, all of which may be related to the ratings. All of the user information variables were treated as dummy variables. Item information includes the title, year, and genre. The movie title may contain some of the content of the movie and may reflect the user’s preferences. Therefore, in order to detect meaning from the movie title, the proposed method extracted the feature vector of the title using BERT [5], and used it as the variable for the title. BERT is an NLP model that achieves the best performance in various tasks, and it is possible to extract the embedding vector that expresses text data well with only a pre-trained BERT model.

Using BERT's embedding vector not only indicates the meaning of the title, but also reduces the number of variables. Both datasets need more than 1,500 variables when treating the titles as dummy variables, but only 786 numeric variables are used when using the embedding vectors of BERT.

4.3. Results

The experimental results are summarized in Table 1. They show that the proposed method is highly competitive. As can be seen from the results, the proposed method achieved lower RMSE and MAE than the other methods, including proposed method with the square filters, and improved performance when using global average pooling rather than global max pooling. It is noted that, even the proposed method with the square filters showed higher performance than other existing models. These results offer two meaningful inferences. First, directly considering interactions between items and user variables is critical for predicting user-item ratings. Second, two cross convolutional filters effectively extract useful features from the interactions.

Fig. 4 shows the behavior of the MSE loss function with respect to the change of epoch in the optimal hyper-parameters of each model. The *x*-axis represents the epoch, and the *y*-axis indicates the MSE loss for the rating. The loss of the proposed method decreases slowly relative to the other methods at the beginning of training, but it reaches the lower loss after about 50 epochs. On the other hand, in the case of DLRM and DNNRec, the loss sharply decreases at the beginning, but the loss converges quickly and eventually rises, which can be viewed as the overfitting problem. To check the occurrence of overfitting, we compared the losses of training data and validation data. Fig. 5 shows the behavior of the loss with respect to the change for each model's training data and validation data. The losses of validation data for (b) DCN-M, (c) DLRM and (d) DNNRec sharply decrease at the beginning and then hardly decrease. The difference between the training and validation data increases significantly after certain points. This overfitting phenomenon can be interpreted as a structural problem of these models. These models consist of a multi-layer perceptron, which handles features independently. Therefore, at the beginning of learning, each feature is directly used to extract useful representation, thereby rapidly increasing the prediction accuracy, but as training progresses, it creates an overfitting phenomenon that predicts only training data well. As shown in Fig. 5(a), the loss of the validation data for the proposed method gradually decreases. As with the other models, the difference between the training and validation data increases after a certain point, but the differentiation begins at the lower loss and is much smaller than in the other models. The reason is that nonlinear interactions expressed in feature maps are directly used to predict the rating through global max or average pooling, rather than through the fully connected layer, thereby mitigating the overfitting problem.

The weights of the vertical and horizontal filters represent the weights given to the interactions with the user features and the item features, respectively. Fig. 6 shows the average weights of the vertical and horizontal filters of the first convolutional layer in the proposed method as trained with the optimal hyper-parameter. As can be seen in Fig. 6, gender had the greatest weight in the vertical filter, followed by occupation. This indicates that gender and occupation are features that are highly correlated with items (movies). In addition, Fig. 6 indicates that the features highly correlated with users are genre and title, not year. The results show that the proposed method effectively considers the interactions with important features, giving higher weights to these interactions through the cross convolutional filters.

Table 2 shows the experimental results for the cold start problem. The results are RMSE with respect to the percentage of new users for MovieLens 100 K. The proposed method achieved the highest performance. Note that the proposed method provided similar results regardless of the percentage of new users. The experimental results show that the proposed method can effectively detect the relationships between users and items and accurately predict the rating with only content information. This means that the user's past ratings were not utilized, which indicates that the proposed method can be applied well to new users or items, alleviating the cold start problem.

Table 1
Experimental results.

Dataset	Model	RMSE	MAE
MovieLens 100 K	Proposed method (Global average pooling)	**0.9379**	**0.7343**
	Proposed method (Global max pooling)	0.9384	0.7347
	Proposed method with square filters	0.9443	0.7467
	DCN-M	0.9552	0.7573
	DLRM	0.9634	0.7599
	DNNRec	0.9546	0.7532
	ECAE	0.9513	0.7566
MovieLens 1 M	Proposed method (Global average pooling)	**0.8989**	**0.7048**
	Proposed method (Global max pooling)	0.9003	0.7074
	Proposed method with square filters	0.9007	0.7095
	DCN-M	0.9220	0.7379
	DLRM	0.9019	0.7101
	DNNRec	0.9076	0.7122
	ECAE	0.9057	0.7110

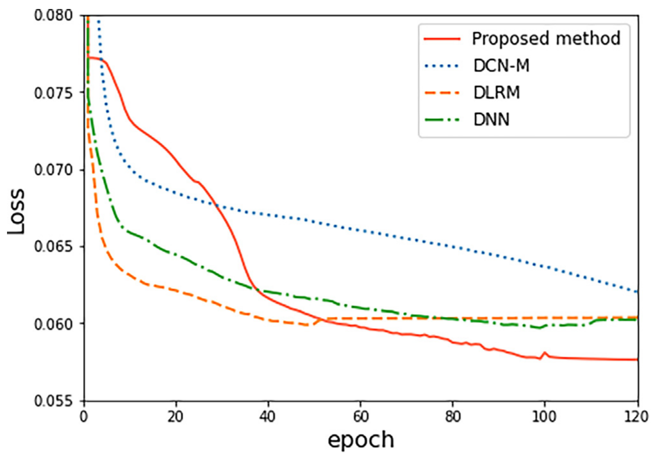

Fig. 4. Behavior of MSE loss with respect to change of epoch for each model's validation data. The red line represents the losses of the proposed method using global average pooling. The green, orange and blue lines represent the losses of the DCN-M, DLRM and DNNRec, respectively.

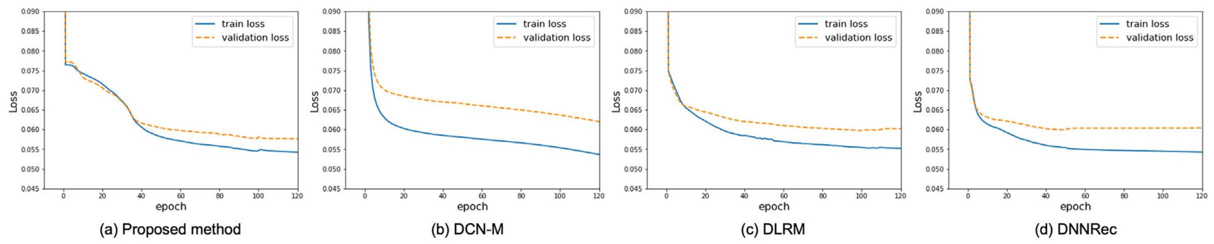

Fig. 5. Behavior of loss with respect to change for each model's training data and validation data: (a) Proposed method, (b) DCN-M, (c) DLRM, and (d) DNNRec.

Age					Gender					Occupation					Zip code				
2.06	1.22	1.88	1.67	1.85	**13.7**	2.67	**5.58**	**8.85**	**12.30**	2.22	**2.74**	2.61	2.52	1.72	2.06	1.74	0.85	0.80	0.60

(a) Vertical filter

Year					Genre					Title				
2.632	5.49	2.68	2.79	2.10	4.38	**11.09**	**13.39**	**8.84**	7.10	2.45	**9.84**	1.93	4.87	**9.95**

(b) Horizontal filter

Fig. 6. Weights of vertical and horizontal filters. The vertical and horizontal filters represent the weights given to the interactions with the user features and the item features, respectively.

Table 2
Prediction results of new users on MovieLens 100 K dataset (% indicates the rate of new users).

Model	10%	20%	30%
Proposed method (GAP)	**1.0248**	**1.0377**	**1.0383**
DCN-M	1.0455	1.0576	1.0588
DLRM	1.0977	1.0982	1.1883
DNNRec	1.1077	1.1153	1.1413

Table 3
Comparisons of time cost for each model.

Model	Training (s)	Inference (s)
Proposed method (GAP)	24.25	1.59
DCN-M	32.30	1.82
DLRM	12.95	1.62
DNNRec	10.66	1.54

To evaluate computational efficiency, we performed experiments comparing execution time of all models. Table 3 shows the training time per epoch and the inference time (in seconds) for each model. Both training and inference times are averages of 10 trials on the training data. As can be seen in Table 3, the proposed method and DCN-M are somewhat less computational efficient compared to other models during training. However, the inference time of the proposed method is similar to or even shorter than those of other models. Although the computation efficiency analysis indicates that the proposed approach has low computational efficiency in training step, the training time may not be considered practically significant. What is important, however, is that the inference (prediction) time is close to the other methods, and the prediction accuracy of the proposed method is superior to other models without the cold start problem.

5. Conclusion

In this paper, we proposed a deep learning-based recommender system that can extract useful features from user and item content information and directly model the user-item interactions. By constructing a feature map for the interaction through the outer product of user and item features, the proposed model does not handle each feature independently, but instead, directly utilizes the feature interactions between users and items. Furthermore, the proposed method effectively models the interaction through cross convolutional filters. The cross convolutional filters give greater weights to important features, thereby taking into account more important features than non-critical features, and capturing useful nonlinear relationships between users and items (i.e., high-dimensional interactions). The experiments showed that the proposed method yielded better performance than the other methods without a multi-layer perceptron structure, implying that it may be considered as a useful alternative in a variety of recommendation tasks. Moreover, the experiments were conducted using only content information without ID information. This suggests that the proposed model can effectively identify the nonlinear interaction between user-item content information. In this study, we evaluated the performance of the proposed model on movie rating data only. Meaningful future research will entail validating the proposed algorithm with various datasets. Additionally, the model can be further simplified by weighting the cross convolutional filter for non-critical features to zero, and interactions between users and items can be modeled more efficiently by considering only interactions with important features. Therefore, efforts to efficiently consider the interactions between features can be considered to be a promising avenue of future research.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT Future Planning (NRF-2017R1E1A1A01077375)

References

[1] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, IEEE Transactions on Knowledge and Data Engineering 17 (2005) 734–749.
[2] B.T. Betru, C.A. Onana, B. Batchakui, Deep learning methods on recommender system: A survey of state-of-the-art, International Journal of Computer Applications 162 (2017) 17–22.
[3] J.S. Breese, D. Heckerman, C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, in: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, 1998, pp. 43–52.
[4] H.T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, et al, Wide & deep learning for recommender systems, in: Proceedings of the 1st workshop on deep learning for recommender systems, 2016, pp. 7–10.
[5] J. Devlin, M.W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2018, pp. 4171–4186.
[6] Guo, H., Tang, R., Ye, Y., Li, Z., He, X., 2017. Deepfm: A factorization-machine based neural network for ctr prediction, pp. 1725–1731..
[7] F.M. Harper, J.A. Konstan, The movielens datasets: History and context, Acm transactions on interactive intelligent systems (tiis) 5 (2015) 1–19.

[8] He, X., Du, X., Wang, X., Tian, F., Tang, J., Chua, T.S., 2018. Outer product-based neural collaborative filtering, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18, International Joint Conferences on Artificial Intelligence Organization. pp. 2227–2233. 10.24963/ijcai.2018/308..
[9] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, T.S. Chua, Neural collaborative filtering, in: Proceedings of the 26th international conference on world wide web, 2017, pp. 173–182.
[10] J. Herce-Zelaya, C. Porcel, J. Bernabé-Moreno, A. Tejeda-Lorente, E. Herrera-Viedma, New technique to alleviate the cold start problem in recommender systems using information from social media and random decision forests, Information Sciences 536 (2020) 156–170.
[11] R. Kiran, P. Kumar, B. Bhasker, Dnnrec: A novel deep learning based hybrid recommender system, Expert Systems with Applications 144 (2020) 113054.
[12] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (2009) 30–37.
[13] S.S. Lakshmi, T.A. Lakshmi, Recommendation systems: Issues and challenges, IJCSIT) International Journal of Computer Science and Information Technologies 5 (2014) 5771–5772.
[14] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (2015) 436–444.
[15] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (1998) 2278–2324.
[16] Lin, M., Chen, Q., Yan, S., 2013. Network in network. arXiv preprint arXiv:1312.4400..
[17] J. Lu, D. Wu, M. Mao, W. Wang, G. Zhang, Recommender system application developments: a survey, Decision Support Systems 74 (2015) 12–32.
[18] M.K. Najafabadi, A. Mohamed, C.W. Onn, An impact of time and item influencer in collaborative filtering recommendations using graph-based model, Information Processing & Management 56 (2019) 526–540, https://doi.org/10.1016/j.ipm.2018.12.007.
[19] Naumov, M., Mudigere, D., Shi, H.J.M., Huang, J., Sundaraman, N., Park, J., Wang, X., Gupta, U., Wu, C.J., Azzolini, A.G., et al., 2019. Deep learning recommendation model for personalization and recommendation systems. arXiv preprint arXiv:1906.00091..
[20] Y. Pan, F. He, H. Yu, A correlative denoising autoencoder to model social influence for top-n recommender system, Frontiers of Computer Science 14 (2019), https://doi.org/10.1007/s11704-019-8123-3.
[21] Rendle, S., 2010. Factorization machines, in: 2010 IEEE International Conference on Data Mining, pp. 995–1000..
[22] F. Ricci, L. Rokach, B. Shapira, Recommender Systems Handbook, 2nd ed., Springer Publishing Company, 2015, Incorporated.
[23] A. Singhal, P. Sinha, R. Pant, Use of deep learning in modern recommendation system: A summary of recent works, International Journal of Computer Applications 180 (2017) 17–22.
[24] Wang, R., Shivanna, R., Cheng, D.Z., Jain, S., Lin, D., Hong, L., Chi, E.H., 2020. DCN-M: improved deep & cross network for feature cross learning in web-scale learning to rank systems. CoRR abs/2008.13535. arXiv:2008.13535..
[25] J. Wei, J. He, K. Chen, Y. Zhou, Z. Tang, Collaborative filtering and deep learning based recommendation system for cold start items, Expert Systems with Applications 69 (2017) 29–39, https://doi.org/10.1016/j.eswa.2016.09.040.
[26] Y. Wu, C. DuBois, A.X. Zheng, M. Ester, Collaborative denoising auto-encoders for top-n recommender systems, in: Proceedings of the Ninth ACM International Conference on Web Search and Data Mining, Association for Computing Machinery, New York, NY, USA, 2016, pp. 153–162, https://doi.org/10.1145/2835776.2835837.
[27] Xin, X., Chen, B., He, X., Wang, D., Ding, Y., Jose, J., 2019. Cfm: Convolutional factorization machines for context-aware recommendation., in: IJCAI, pp. 3926–3932..
[28] Q. Xu, M. Zhang, Z. Gu, G. Pan, Overfitting remedy by sparsifying regularization on fully-connected layers of cnns, Neurocomputing 328 (2019) 69–74.
[29] Xue, H.J., Dai, X., Zhang, J., Huang, S., Chen, J., 2017. Deep matrix factorization models for recommender systems., in: IJCAI, pp. 3203–3209..
[30] P. Yiteng, F. He, H. Yu, A novel enhanced collaborative autoencoder with knowledge distillation for top-n recommender systems, Neurocomputing 332 (2018), https://doi.org/10.1016/j.neucom.2018.12.025.
[31] Yiteng, P., He, F., Yu, H., 2020. Learning social representations with deep autoencoder for recommender system. World Wide Web 23..
[32] S. Zhang, L. Yao, A. Sun, Y. Tay, Deep learning based recommender system: A survey and new perspectives, ACM Computing Surveys (CSUR) 52 (2019) 1–38.

Seungyeon Lee received the B.S. and M.S. degrees in industrial and management engineering from Myongji University, Korea in 2018 and 2020, respectively. Her research interests include statistical data mining, deep learning, recommender system and graph data analysis.

Dohyun Kim received the M.S. and Ph.D. degrees in industrial engineering from KAIST, Korea in 2002 and 2007, respectively. Currently, he is an associate professor with the Department of Industrial and Management Engineering, Myongji University. His research interests include statistical data mining, deep learning and graph data analysis.