Journal of Computer Science

Original Research Paper

ConvSVD++: A Hybrid Deep CF Recommender Model using Convolutional Neural Network

[1]Mohamed Grida, [2*]Lamiaa Fayed and [3]Mohamed Hassan

[1]Department of Industrial Engineering, Faculty of Engineering, Zagazig University, Egypt
[2]Faculty of Computers and Informatics, Zagazig University, Egypt
[3]Department of Information System, Faculty of Computers and Informatics, Zagazig University, Egypt

Article history
Received: 08-07-2020
Revised: 08-12-2020
Accepted: 11-12-2020

Corresponding Author:
Lamiaa Fayed
Faculty of Computers and Informatics, Zagazig University, Egypt
Email: lamiaa.fayed@yahoo.com

Abstract: Recommender systems are powerful systems that give added value to business and corporation. They are relatively recent technology and they will only keep improving in the future. The most widely used algorithms for recommender systems are categorized into the traditional recommender and deep-based recommender system. The traditional recommendation algorithm suffers from sparse data that significantly degrades recommendation accuracy. The hybrid approaches are attempts to tackle recommendation challenges. This paper addresses the integration of deep learning into traditional recommendation approaches especially, Collaborative Filtering (CF) algorithms to get a significant accurate prediction. It proposes a hybrid deep CF recommender model called ConvSVD++ that tightly integrates Convolution Neural Network (CNN) and Singular Value Decomposition (SVD++). The proposed model incorporates items' content, implicit user's feedback along with explicit item-user interaction to enhance prediction accuracy and handle sparsity problem. The proposed model is evaluated and all baseline models based on Movielens- 1M datasets. The results are evaluated using Root Mean Squared Error (RMSE) metric and it is concluded that the proposed model ConvSVD++ outperforms the baselines models. Accordingly, it is concluded that integrating CNN with SVD++ algorithm improves rating prediction accuracy.

Keywords: Recommender System, Deep Learning, Prediction, Singular Value Decomposition, Convolution Neural Network

Introduction

The Recommender System (RS) has become a significant web service in many applications. Several online companies apply RS to build up the relationship with users and enhance marketing and sales (Bobadilla *et al*., 2013). The Recommender system is introduced as a computer program that generates suggestions about new items for a particular user or customer (Lu *et al*., 2015). The most common algorithms for recommender systems are categorized into the traditional recommendation approaches, deep-based recommendation and hybrid deep recommendation. Early work considered traditional recommendation approaches that are classified into Collaborative Filtering (CF), Content-Based (CB) and hybrid approach. The CF approach makes a recommendation based on users' evaluations of items and preferences of users with similar behavior (Ricci *et al*., 2011; Xu *et al*., 2013; Herlocker *et al*., 2017; Costa-Montenegro *et al*., 2012). The CF approach is the most extensively applied recommendation algorithms (Lu *et al*., 2015). On the other hand, Matrix Factorization (MF) (Cremonesi *et al*., 2012; Vozalis and Margaritis, 2005; Kurucz *et al*., 2007) is considered as a superior CF algorithm (Bauer and Nanopoulos, 2014) due to its capability to reduce sparsity problem and improve recommendations accuracy. The content-based approach recommends items based on their content and metadata (Costa-Montenegro *et al*., 2012). CB approach compares the user's profile with items characteristics. The more the item is descriptive, the more recommendations are accurate. The hybrid approach combines collaborative and content data to improve performance (Barragáns-Martínez *et al*., 2010; Ghazanfar and Prügel-Bennett, 2014; Celdrán *et al*., 2016). Traditional recommender systems encounter a number of fundamental problems that cause a reduction in prediction accuracy, such as sparsity (Xie *et al*., 2012;

Shambour and Lu, 2015), scalability (Moreno *et al*., 2016) and cold start (Vartak *et al*., 2017; Zhang *et al*., 2010; Qiu *et al*., 2011).

Various studies have recently introduced Deep Learning (DL) to develop a recommender system. A deep learning model has a capability to learn latent features in large and complex data that are usually difficult to manipulate with standard tools (Batmaz *et al*., 2019). It also has powerful computations. A deep recommender system can be developed in accordance with collaborative or content-based models. Moreover, it can be developed basically upon the DL algorithm or integrated tightly or loosely with other traditional RS models (Zhang *et al*., 2019).

The hybrid deep recommendation approach is one that integrates traditional and deep-based models as an attempt to overcome traditional approaches limitations and to provide an accurate recommendation. Various articles applied these models to incorporate various auxiliary information to improve accuracy and solve sparsity and cold start problems. However, it is still an open research issue.

In this study, we propose a hybrid deep Collaborative Filtering (CF) model called ConvSVD++ that incorporates items' content, implicit user's feedback along with explicit user-item interactions to enhance rating prediction accuracy and handle sparsity problem. A proposed model tightly integrates a Convolutional Neural Network (CNN) and the SVD++ approach. CNN considers contextual information that explains words neighboring, a sequence of words and position of the word that leads to a deeper understanding of the auxiliary information and learns more hidden feature representation. On the other hand, select SVD++ as it is the most common traditional approach that considers the implicit rating.

The research contributions are summarized as follow:

- To the best of our knowledge, there is no research introduce a hybrid model that combines a Convolution Neural Network (CNN) with SVD++. A proposed model is tightly integrated that learns and optimizes parameters simultaneously where two integrated approaches are mutually affected each other and allow two- way interaction between them.
- The proposed model ConvSVD++ considers the effect of the various forms of implicit feedback using the SVD++ model that leads to improving prediction accuracy
- By incorporating the CNN, it can consider the spatial structure of the input which explains words neighboring, a sequence of words and position of the word that leads to a significant improving the efficiency of extracting complex features
- The proposed model employs an efficient SVD++ algorithm that significantly enhances the computation efficiency by grouping users that shares the same implicit feedback
- An experiment demonstrates the superiority of the proposed model over the state-of-the-art models

This paper is organized as follows: Section 2 discusses the most recent related works. Section 3 presents the preliminaries for convolutional neural network and singular value decomposition approaches. Section 4 presents the proposed hybrid deep recommender model, ConvSVD++. Section 5 presents experimental result and evaluation that discusses the result of the experiment. Finally, the conclusion is presented.

Related Work

Various studies introduced hybrid deep collaborative filtering approaches to alleviate sparsity and cold start problems to enhance performance and prediction accuracy. Some studies integrate latent features from items' content and auxiliary information (like description, reviews, synopses and abstract) to the collaborative recommendation process. For instance, (Wang *et al*., 2015a) proposed a tightly integrated model for tag recommendation system. Generalized Bayesian SDAE was employed for learning feature representation (relation between items) and to be combined with relational information matrix. The model result exceeds the state of the art tag aware recommendation models. (Wang *et al*., 2015b) also introduced a hierarchical Bayesian model called Collaborative Deep Learning (CDL). CDL integrates Stacked Denoising Auto-Encoder (SDAE) with probabilistic matrix factorization. It is a tightly coupled method system that permits a recommender for two-way interaction between two components. CDL employs generalized Bayesian SDAE for learning feature representation. Similarly, (Li *et al*., 2015) applied marginalized Denoising Autoencoder (DAE) integrated to matrix factorization which is more efficient and scalable. These studies attempt to combine deep learning and collaborative filtering. Moreover, (Wang and Wang, 2014) employs probabilistic matrix factorization together with the features learned via DBNs. Zhang *et al*. (2019) introduced a hybrid approach (ConvFM) that integrates the convolution neural network with probabilistic matrix factorization. CNN is differentiated by providing high-level representation and accurate contextual information of items through word embedding and convolutional kernels. Furthermore, (Zheng *et al*., 2017) join a convolution neural network with matrix factorization. The model consists of two parallel CNN connected at the final layer with matrix factorization to extract user-item interactions for predicting rate. It handles sparsity and provides a good semantic representation of review text. Shin *et al*. (2015) utilized CNN to integrate extracted features from text

and images into their proposed boosted inductive matrix completion method.

Generally, the framework to unify this integration was investigated in (Li *et al*., 2015). This framework can interpret all previous models. It models the mappings between the latent factors used in CF and the latent layers in deep models.

Other studies incorporate items' content in addition to further sources of information about the users such as user's implicit feedback. Implicit feedback represents behavioral information about users' procurements or browsing history to learn preferences. For example, (Zhang *et al*., 2017) improve prediction accuracy by combining the item's content and the user's implicit feedback. It integrates a contractive auto-encoder with SVD++. A contractive auto-encoder captures several input variations, while SVD++ can learn the implicit feedback to improve performance.

Other studies consider the temporal dynamic factor that reflects the dynamic and time-drifting of user-item interactions. Wei *et al*. (2017) considered the temporal dynamics of user preferences and item features. The deep learning neural network SDAE is responsible for the extraction of item content features, while the timeSVD++ model is responsible for the prediction of unknown ratings. Xiong *et al*. (2019) introduced a recommendation framework joining temporal dynamics, CNN-based text features and item correlation. It covers the item cold-start problem for diverse cold start situations.

As it is evident, Autoencoder and CNN deep learning models are the most frequently applied in recommendation design due to its power in extracting complex hidden features to handle sparsity and improve prediction accuracy. However, the auto-encoder deep learning model utilizes a bag-of-words model that fails to define the contextual information of the documents. Contextual information explains words neighboring, a sequence of words and position of the word that leads to a deeper understanding of the document. Furthermore, deep understanding assists to enhance rating prediction accuracy. Additionally, CNN can employ pre-trained word embedding models like Glove (Pennington *et al*., 2014) which provide a deep understanding of the item. SDAE can't apply the word embedding model because it applies a bag of words model.

All prior discussed studies prove that adopting both user and item features in developing RS will lead to enhance recommendation and rating prediction accuracy. A proposed model combines items' features with the user's implicit feedback. CNN is responsible for extracting items latent features, while the SVD++ model is responsible to consider the effect of the user's implicit feedback. A proposed model takes the advantages of CNN to gain significant performance improvement. To the best of our knowledge, the proposed model is the first hybrid approach that couples CNN with SVD++.

Preliminaries

Singular Value Decomposition (SVD)

Singular Value Decomposition (SVD) was utilized in two directions: Extract the latent association between users and items that help predicting user's ratings on items (Sarwar *et al*., 2000) or shrink the dimensionality of user-items space. The basic concept is to decompose user-item matrix $R_{m\times n}$ into two low-rank matrices that denote item factors and user factors, as depicted in Fig. 1. for declaration, each item i is denoted by a vector $q_i \in R^f$ and each user u is denoted by a vector $p^u \in R^f$. For each item i, the components of q_i measure the level to which the item holds those factors, positive or negative. For a given user u, the elements of p_u quantity the level of interest the user has in items that are high on the corresponding factors (again, these may be positive or negative). The rating prediction is estimated by the dot product $q_i^T p_u$, captures the overall user's interest in features of the item. The rating is predicted by Equation 1:

$$r_{ui}^{-}(t) = q_i^T p_u \tag{1}$$

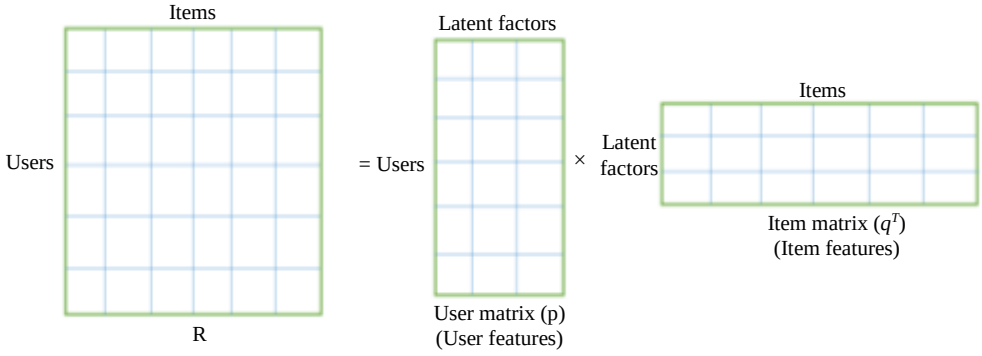

Fig. 1: SVD matrix decomposition

Biased SVD

Unlike the conventional SVD model, the biased SVD introduces user and item bias terms as in Equation 2:

$$r_{ui}^{-}(t) = \mu + b_i + b_u + q_i^T p_u \quad (2)$$

where, μ is the average rating, b_u denotes the observed deviations of user u, b_i is the bias term for item i. The objective function of the model is to minimize the regularized squared error on training process while learning the parameters:

$$\min_{q^*,p^*,b^*} \sum_{u,i,t} \left(r_{ui} - r_{ui}^{-}\right)^2 + \lambda_1 \left[b_i^2 + b_u^2 + \left\| q_i \right\|^2 + \left\| p_u \right\|^2 \right] \quad (3)$$

where, λ_1 is a constant that controls the degree of regularization and usually determined by cross-validation. Stochastic gradient descent or alternating least squares can be employed for optimization.

SVD++

SVD++ considers the effect of implicit feedback. SVD++ method provides accuracy superior to SVD and also supportive for users that providing much more implicit feedback than explicit one new item factors are used to signify users based on the set of items that they implicitly rated as in the following Equation 4:

$$r_{ui}^{-} = \mu + b_i + b_u + q_i^T \left[p_u + \left|R(u)\right|^{-\frac{1}{2}} \sum_{j \in R(u)} y_j \right] \quad (4)$$

where, $R(u)$ is a set of items implicitly rated by user u. A user is modeled by $p_u + \left|R(u)\right|^{-\frac{1}{2}} \sum_{j \in R(u)} y_j$. Where p_u is a user vector that is learned from the given explicit ratings. This vector is complemented by the sum $\left|R(u)\right|^{-\frac{1}{2}} \sum_{j \in R(u)} y_j$ which represents the aspect of implicit feedback. Since the y_j's are centered around zero (by the regularization), the sum is normalized by $\left|R(u)\right|^{-\frac{1}{2}}$, to stabilize its variance across the range of observed values of $|R(u)|$.

The objective function is to minimize the associated regularized squared error. Stochastic gradient descent is the learning algorithm that iterates all known ratings in K, estimating:

$$b_u \leftarrow b_u + \gamma_1 \cdot \left(e_{ui} - \lambda_1 \cdot b_u\right) \quad (5)$$

$$b_i \leftarrow b_i + \gamma_1 \cdot \left(e_{ui} - \lambda_1 \cdot b_i\right) \quad (6)$$

$$q_1 \leftarrow q_1 + \gamma_2 \cdot \left(e_{ui} \left(p_u + \left|R(u)\right|^{-\frac{1}{2}} \sum_{j \in R(u)} y_j \right) - \lambda_2 \cdot q_1 \right) \quad (7)$$

$$p_u \leftarrow p_u + \gamma_2 \cdot \left(e_{u1} \cdot q_1 - \lambda_2 \cdot p_u\right) \quad (8)$$

$$\forall j \in R(u) : y_j + \gamma_2 \cdot \left(e_{ui} \cdot \left|R(u)\right|^{-\frac{1}{2}} \cdot q_i - \lambda_2 \cdot y_i \right) \quad (9)$$

SVD++ can model various forms of implicit feedback. For example, if a user u rent a set of items in $R^1(u)$ and browse a set of items in $R^2(u)$, the model modified as:

$$r_{ui}^{-} = \mu + b_i + b_u + q_i^T \begin{bmatrix} p_u + \left|R^1(u)\right|^{-\frac{1}{2}} \sum_{j \in R^1(u)} y_j^{(1)} \\ + \left|R^2(u)\right|^{-\frac{1}{2}} \sum_{j \in R^2(u)} y_j^{(2)} \end{bmatrix} \quad (10)$$

Convolutional Neural Network

Convolution Neural Network (CNN) is a type of feed-forward neural network that achieves a significant performance in computer vision and Natural Language Processing (NLP). CNN is similar to a neural network which is made up of numerous neurons associated with weights and biases. Each neuron takes various inputs, sum them up, then applies the activation function to get the output. CNN is different from neural networks where the input can be a multi-channeled (3 channel in the case of image). Unlike non-convolutional neural networks, CNN considers the spatial structure of the input that leads to a significant performance in computer vision and document analysis. A CNN composed of an input layer, several hidden layers and an output layer. The hidden layer consists of convolutional, pooling operations and connected layers which significantly improving the efficiency of extracting complex features (Zhang *et al*., 2019):

I. Convolution layer generates local features of input data. It consists of a set of independent filters that are initialized randomly to be learned by the network afterward. The size of the filter is smaller than the input. Each filter is rolled overall spatial locations of the input and compute the dot product between the input and weights defined in the filter at every spatial position. The results are summed up into one number that represents all the pixels the filter observed. Convolution layer output is called an activation map (*feature map)*. Its input can be the original input or a prior layer output that is an alternative activation map. The convolutional layer output is achieved by stacking the activation maps

of all filters and its volume will be smaller than the input. Every neuron in the activation map is only linked to a small local area of the input

II. The pooling layer extracts a high-level representation of data by making the down sampling of the input. Pooling processes each activation map independently by selecting the maximum (max pool) or average over a region of the feature map, Fig. 2. It gradually decreases the spatial dimension of the representation to decrease the number of parameters and computation in the network

III. The fully connected layer is the final layer. It is a multi-layer perceptron neural network that takes a one-dimensional vector of the feature maps of the previous layer as input. It can be zero, one, or more hidden layers. The output is a list of probabilities producing a multi-class classification. The class with the highest probability is the classification decision

Hybrid Deep CF Model (ConvSVD++)

A proposed model ConvSVD++ consists of four phases, as shown in Fig. 3, including the preprocessing phase, extracting latent item features by the CNN phase, rating prediction phase and evaluation phase. These phases will be discussed in detail in the following subsections.

Phase1: Preprocessing

The most commonly chosen data set is MovieLens that represent 21% of articles in the literature that are utilized in our experiments. MovieLens data set doesn't contain an item description. We extracted documents of plot summary corresponding items from IMDB.

The preprocessing phase is essential to clean text. In this section, three main steps are elaborated in dataset preprocessing, including word tokenization, stopword removal and normalization.

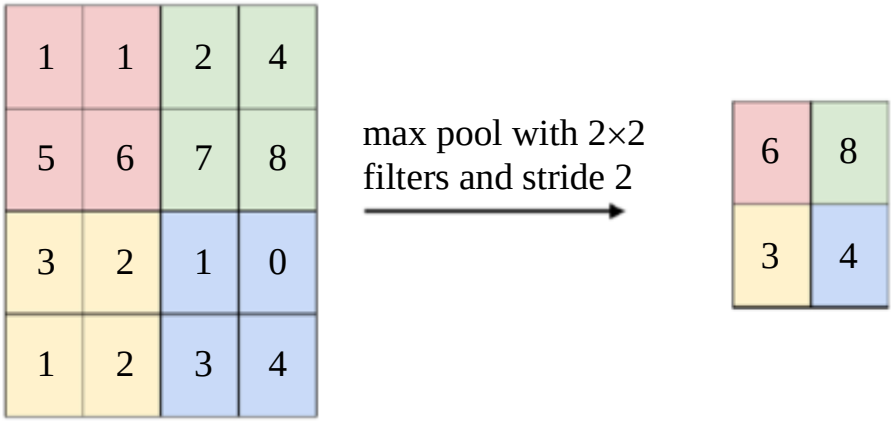

Fig. 2: The pooling layer

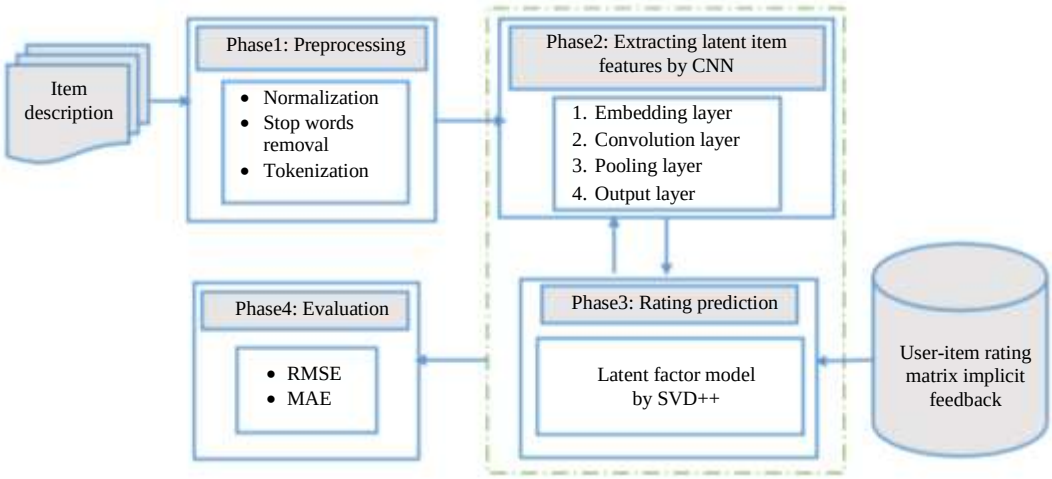

Fig. 3: The efficient ConvSVD++ model

Word Tokenization

Tokenization is the first step in natural language processing applications. It segments input text to several linguistic units, named tokens. It works based on white spaces, marks, white space and punctuation marks.

Normalization

The normalization process has two tasks stemming and lemmatization. Stemming is the process of eliminating suffix, prefixes and infixes from a word to get a word stem, for example, designing to design. Lemmatization captures canonical forms based on a word's lemma. For instance, the lemmatization of worse word returns bad.

Stop Word Removal

Stop word removal step is a significant step to get a cleaned dataset. Such a step removes all commonly occurring words that have no significant meaning, like of, the, in, some etc.

After accomplishing this step, each item document is represented as a vector of cleaned, normalized tokens.

Phase2: Extracting Latent Item Features by CNN

The main aim of CNN in a proposed model is to construct an item latent vector to be used in rating prediction of the recommender system. CNN capable of extracting sophisticated and effective feature representation. CNN architecture proposed in (Kim *et al*., 2016) is employed in the proposed model. CNN mainly consists of four layers as depicted in Fig. 4. These layers are described in details as follow.

The Embedding Layer

CNN does not understand the text. However, it recognizes only numbers. An embedding layer is a word embedding where individual words are represented as real-valued vectors in a predefined vector space. Words that have similar nearby words are very close vectors that point in the same direction. This representation helps to extract semantic information.

There are various word embedding models available. In a proposed model and embedding model is constructed using a simple python library, called "gensim", which is a simple toolkit developed for handling various NLP tasks. Moreover, it is based on Global Vectors for word representation, named GloVe. an algorithm is an extension to the word2vec which is introduced by (Mikolov *et al*., 2013).

There are n words in the item row vector obtained from the preprocessing phase. The embedding layer begins by mapping each word into a dense vector x_i and then concatenate these word vectors into a dense matrix to represent the item description document. The document matrix will be denoted by $D_i \in R^{P\times l}$ where l is the length of the document and P is the size of the embedding dimension for each word in the item document:

$$D_1 = x_{1:n} = x_1 \oplus x_2 ..... \oplus x_n \tag{11}$$

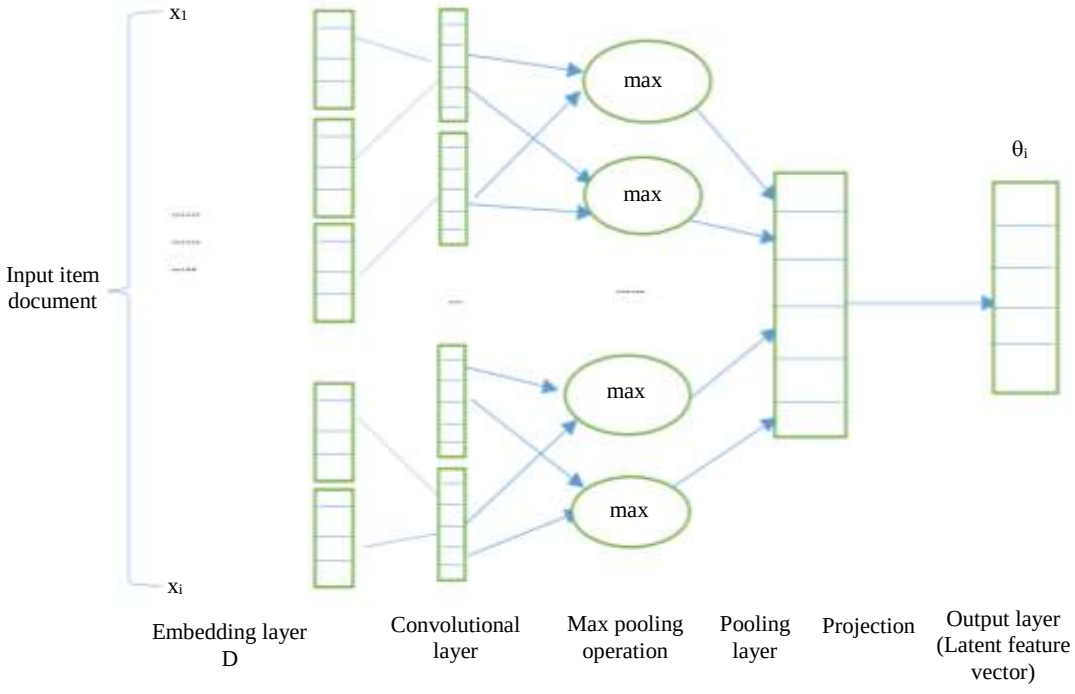

Fig. 4: CNN architecture

The Convolution Layer

The convolutional layer is responsible for extracting contextual content features from the item's description. A contextual feature $c_i \in R$ is extracted by jth shared weight $w \in R^{P \times WS}$ where WS is the window size that represents the number of nearby words. For example, creating contextual features c_i from $x_{(i:(i+ws-1))}$:

$$c_1 = f\left(w * x_{(i:(i+ws-1))} + b\right) \tag{12}$$

where, * is the convolution operator, $b \in R$ is a bias for w^j and f is the non-linear activation function, ReLU. Then, a contextual feature vector of a document is produced by jth shared weight w:

$$c^j = \left[c_1^j, c_2^j, ...c_i^j, ..., c_{l-ws+1}^j\right] \tag{13}$$

where, $c^j \in R^{l\text{-}ws+1}$ numerous contextual feature vectors should be produced n_c as many as various shared weights W_c are applied.

The Pooling Layer

The pooling layer is responsible for providing a high-level representation of item features that comes from the convolution layer. These contextual feature vectors obtained from the convolution layer have variable lengths (l-ws+1) so max-pooling operation extracts only the most important contextual feature vector as Equation 14 to construct features vectors in a fixed-length n_c:

$$d_f = \left[\max\left(c^1\right), \max\left(c^2\right), ..., \max\left(c^j\right)...., \max\left(c^{cn}\right)\right] \tag{14}$$

where, c^j is a contextual feature vector of length (l-ws+1) extracted by jth shared weight.

The Output Layer

At the output layer, conventional nonlinear projection is applied in order to get latent features vectors θ:

$$\theta = \tanh\left(W_{f_2}\left\{\tanh\left(W_{f_1} d_f + b_{f_1}\right)\right\} + b_{f_2}\right) \tag{15}$$

where, $W_{f_1} \in R^{f \times n_c}, W_{f_2} \in R^{k \times f}$ are projection matrices and $b_{f_1} \in R^f, b_{f_2} \in R^K$ is a bias vector for W_{f_1}, W_{f_2} with $\theta \in R^k$. The output of CNN is:

$$\theta_i = cnn\left(W, D_i\right) \tag{16}$$

where, θ_i latent vector of item i, W represents all the weight and bias variables which will be updated in the convolutional neural network and D_i item's description document.

Phase3: Rating Prediction

SVD++ model is an enhanced variant of biased SVD by considering the effect of the implicit feedback by adding the factor of $\left|R(u)\right|^{-\frac{1}{2}} \sum_{j \in R(u)} y_j$ as in the following Equation 17:

$$r_{ui}^- = \mu + b_i + b_u + q_i^T\left[p_u + \left|R(u)\right|^{-\frac{1}{2}} \sum_{j \in R(u)} y_j\right] \tag{17}$$

where, q_i is a vector of latent factors of item i and p_u is a vector of latent features of user u.

The proposed ConvSVD++ model employs the item's latent features extracted by CNN to the latent factor training process of SVD++, Fig. 5. Item latent vector q_i is divided into two parts (Zhang *et al*., 2017), $Cnn(W, D_i)$ part for the feature vector extracted from item's content, $\epsilon_i \in R^k (i = 1...n)$ part signifies the latent item-based offset vector. β is a hyper-parameter to normalize $Cnn(W, D_i)$. The predicted rate is estimated using Equation 18:

$$\begin{aligned} r_{ui}^- &= \mu + b_i + b_u \\ &+ \left(\beta \cdot Cnn\left(W, D_i\right) + \epsilon_i\right)^T\left[p_u + \left|R(u)\right|^{-\frac{1}{2}} \sum_{j \in R(u)} y_j\right] \end{aligned} \tag{18}$$

where, $R(u)$ is a set of items implicitly rated by user u. A user is modeled by $p_u + \left|R(u)\right|^{-\frac{1}{2}} \sum_{j \in R(u)} y_j$. where p_u is a user vector that is learned from the given explicit ratings. This vector is complemented by the sum $\left|R(u)\right|^{-\frac{1}{2}} \sum_{j \in R(u)} y_j$. Which represents the aspect of implicit feedback. Since the y_j's are centered around zero (by the regularization), the sum is normalized by $\left|R(u)\right|^{-\frac{1}{2}}$, in order to stabilize its variance across the range of observed values of $|R(u)|$.

Optimization

Our Proposed is a tightly integrated model that learns and optimizes parameters simultaneously. CNN and SVD++ integrated approaches are mutually affected by each other and allow two- way interaction between them. We utilized a backpropagation algorithm to optimize CNN weights. The objective function of the model is to minimize the regularized squared error on the training process while learning parameters:

$$\begin{aligned} &\min q*, p*, b* \sum_{u,i,t} \left(r_{ui} - r_{ui}^-\right)^2 \\ &+ \lambda\left[b_i^2 + b_u^2 + \left\|\epsilon_i\right\|^2 + \left\|p_u\right\|^2 + \sum_{j \in R(u)} \left\|y_j\right\|^2\right] \end{aligned} \tag{19}$$

where, λ is a constant that controls the degree of regularization and usually determined by cross-validation. Stochastic gradient descent is employed for optimization. Iterate all known ratings in K, K is the set of the (u,i) pairs for which r_{ui} is known the training set:

- $b_u \leftarrow b_u + \gamma_1 \cdot (e_{ui} - \lambda_1 \cdot b_u)$
- $b_i \leftarrow b_i + \gamma_1 \cdot (e_{ui} - \lambda_1 \cdot b_i)$
- $\epsilon_i \leftarrow \epsilon_i + \gamma_2 \cdot \left(e_{ui} \cdot \left(p_u + \left(|R(u)|^{-\frac{1}{2}} \sum_{j \in R(u)} y_j\right)\right) - \lambda_2 \cdot \epsilon_i\right)$
- $P_u \leftarrow P_u + \gamma_2 \cdot (e_{ui} \cdot (\beta \cdot Cnn(W, D_i)\ \epsilon_i) - \lambda_2 \cdot P_u)$
- $\forall\, j \in R(u)$: $y_j \leftarrow y_j + \gamma_2 \cdot (e_{ui} \cdot |R(u)|^{-\frac{1}{2}} \cdot (\beta \cdot Cnn(W, D_i)\ \epsilon_i) - \lambda_2 \cdot y_j)$

where, γ_1 and γ_2 are the learning rates, λ_1 and λ_2 are the regularization weights. Due to the high cost of updating the parameter y. an efficient training process applied to decrease the computation time of SVD++ using algorithme1 the same as (Zhang *et al*., 2017).

Algorithm 1 Efficient training algorithm for SVD++

Procedure UPDATE PARAMETERS
  for all user u do
    $p^{im} \leftarrow |R(u)|^{-\frac{1}{2}} \sum_{j \in R(u)} y_j$
    $p^{old} \leftarrow p^{im}$
    for all training samples of user u do
      update other parameters
      $p^{im} \leftarrow p^{im} + \gamma_2 \cdot (e_{ui} \cdot (\beta \cdot Cnn(W, D_i) + \epsilon_i) - \lambda_2 \cdot p^{im})$
    end for
    for all i in items rated by u do
      $y_i \leftarrow y_i + |R(u)|^{-\frac{1}{2}} \cdot (p^{im} - p^{old})$
    end for
  end for
end procedure

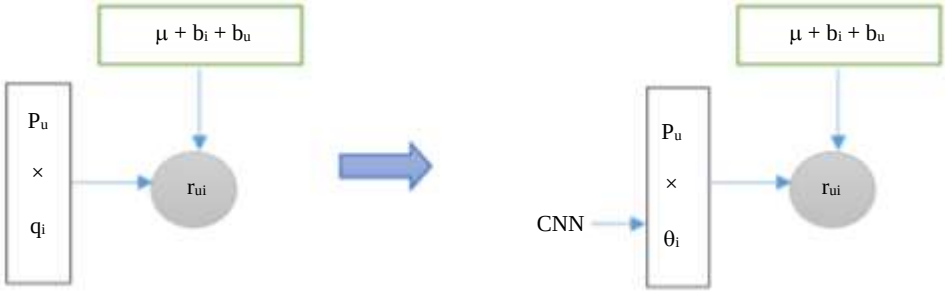

Fig. 5: The graphical modification of ConvSVD++

Fig. 6: Movie's plot summary

Phase4: Evaluation Phase

There are two measures for evaluating the accuracy of the ConvSVD++ model, Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left| p_i - r_i \right| \quad (20)$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left| p_i - r_i \right|^2} \quad (21)$$

Where:
n: Unrated items used for the test.
P_i: The prediction on the ith test instance.
r_i: The corresponding rating value is given by the user

Experimental Result and Evaluation

Experimental Environment

The proposed model is coded in python 3.7 and runs on a personal laptop with the following specification: Windows 10 operating system, 64-bit Operating System, x64-based processor, an Intel® Core(TM) i7-9750U CPU @ 2.80 GHz, 16.00 GB RAM, 1 TB disk space.

Data Description

An experiment is conducted on the most common dataset MovieLens 1 M. MovieLens 1 M: Consists of 1,000,209 ratings of 3900 movies made by 6040 MovieLens users as shown in Table 1. Ratings are discrete values from 1 to 5. Such a dataset is downloaded from an online webpage at https://grouplens.org/datasets/movielens. Plot summaries are available at http://www.imdb.com/.plot example in Fig. 6. Data set contains three files users, items and ratings. User information is in the file "users.dat" and is in the following format: UserID::Gender::Age::Occupation::Zip-code.All ratings are contained in the file "ratings.dat" and are in the following format: UserID::MovieID::Rating:: Timestamp.User information is in the file "users.dat" and is in the following format: UserID::Gender::Age::Occupation::Zip-code. Movie's summary is stored in "plot.dat" and is in the format: MoviesID:: Plot. We preprocessed the movie's plot for all dataset elements as follows:

- Set maximum length of raw documents to 300
- Eliminate stop words
- Determine tf-idf score for each word
- Eliminate corpus-specific stop words that have the document frequency higher than 0.5
- Pick up the top 8000 diverse words as a vocabulary
- Eliminate all non-vocabulary words from raw documents

Experimental Settings

Configuration setting for CNN and SVD++ algorithms are set as follows:

1) The maximum document length is set to 300
2) The word latent vectors are initialized by pre-trained word embedding models with a dimension size of 200. They will be trained in the optimization method
3) Several window sizes (3, 4 and 5) employed in the convolution layer to consider the different lengths of the surrounding words
4) We apply 100 shared weights per window size
5) We employ dropout and set dropout rate to 0.2 to avoid CNN from over-fitting
6) We set gamma1 = 0.007, gamma2 = 0.007, Lambda1 = 0.005, Lambda2 = 0.015, Beta = 0.01, Number of iterations 30 and Latent factor dimension = 20

Baseline Algorithms

- PMF (Mnih and Salakhutdinov, 2008): Probabilistic Matrix Factorization is a standard rating prediction model that only uses ratings for collaborative filtering
- CDL (Wang *et al*., 2015b): Collaborative Deep Learning is another state-of- the-art recommendation model, which enhances rating prediction accuracy by analyzing documents using SDAE
- CTR (Wang and Blei, 2011): Topic Regression is a state-of-the- art recommendation model that join collaborative Filtering (PMF) and topic modeling (LDA) to use both ratings and item's content
- ConvMF+ (Kim *et al*., 2016): Convolutional Matrix Factorization with pre-trained word embedding model
- TmTx (Xiong *et al*., 2019): A model of temporal dynamics and text
- TmTI (Xiong *et al*., 2019): A model integrates temporal dynamics, comments and item correlations

Table 1: The movielens dataset description

	Domain	Users	Items	Range	Total ratings	Density (%)
MovieLens 1 M	Movie	6040	3952	5-stars	1,000209	4.19

Table 2: RMSE and MAE for the different ratio of the training set

	Training set (50%)	Training set (70%)	Training set (80%)
RMSE	0.87220	0.85606	0.84288
MAE	0.68535	0.67068	0.66093

DOI: 10.3844/jcssp.2020.1697.1708

Table 3: Average RMSE for Movielens- 1 M from compared models

Model	RMSE
PMF	0.8971
CTR	0.8969
CDL	0.8879
ConvMF+	0.8549
Biased SVD	0.876
SVD++	0.855
TmTx	1.0574
TmTI	0.9739
ConvSVD++	0.84288

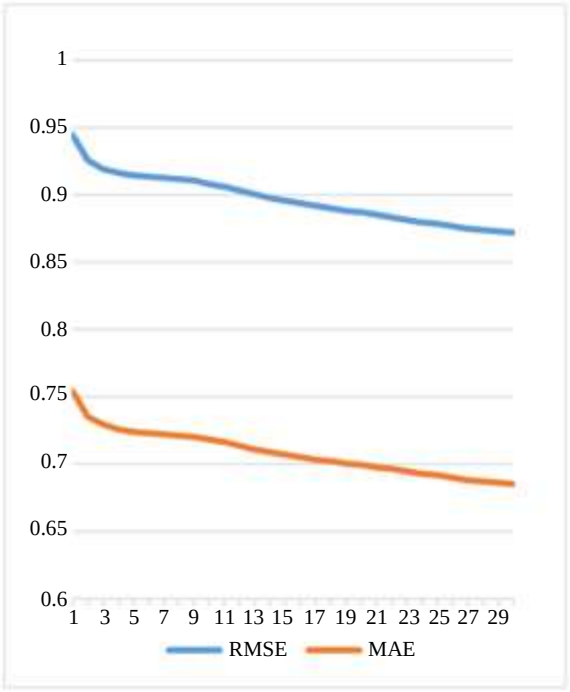

Fig. 7: RMSE and MAE for each iteration (training set 50%)

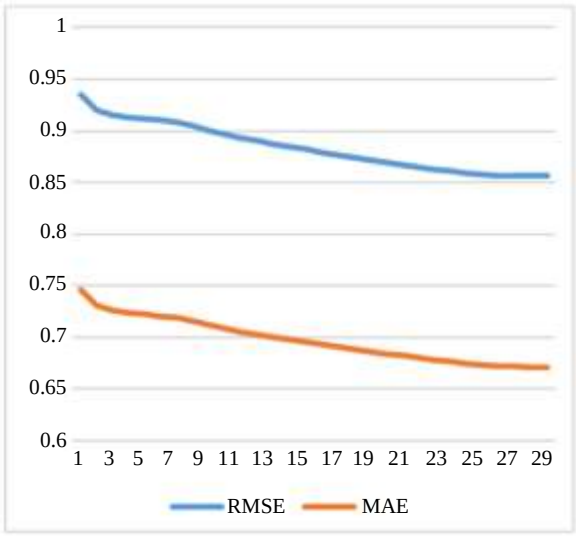

Fig. 8: RMSE and MAE for each iteration (training set 70%)

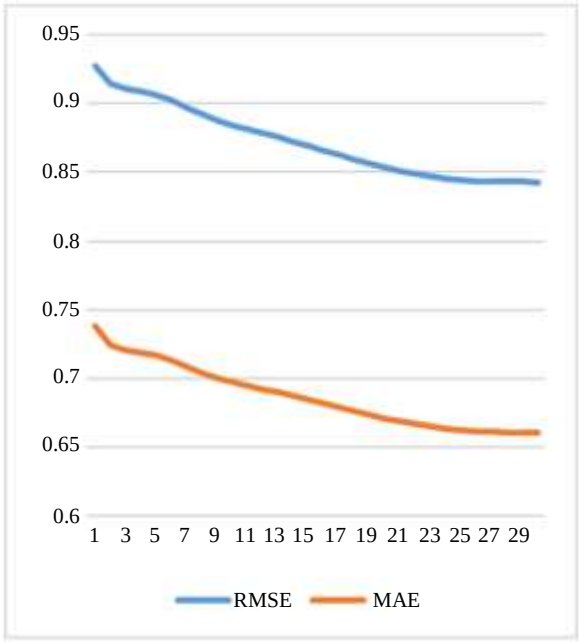

Fig. 9: RMSE and MAE for each iteration (training set 80%)

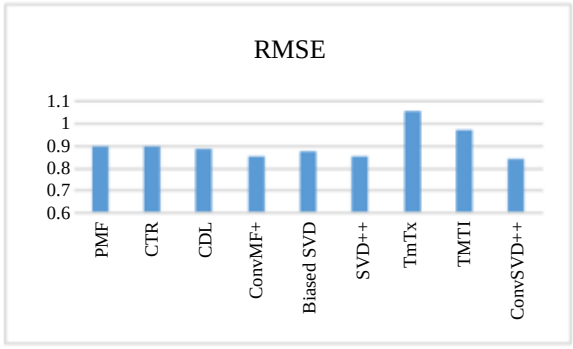

Fig. 10: Average RMSE for Movielens- 1M from compared models

Experimental Result

To evaluate the performance of the proposed model and all baseline models based on Movielens- 1 M datasets, we randomly split the dataset into a training set (80%) and a test set (20%). The training set contains at least a rating on every user and item to deal with all users and items. As the evaluation measure, we used Root Mean Squared Error (RMSE), which is directly related to an objective function of a conventional rating prediction model. The proposed ConvSVD++ model is trained for 30 iterations. Figure 7 to 9 show the change of RMSE and MAE for each iteration for Movielens- 1 M data set for training set 50, 70 and 80% respectively. Table 2 shows the RMSE and MAE for different ratio of the training set to the entire dataset.

Table 3 and Fig. 10 present the RMSE of ConvSVD++ and the baseline models for Movielens-1M dataset. There is an improvement of ConvSVD++ compared to the state of the art models.

Conclusion

In this study, we present a hybrid deep CF model, called ConvSVD++. It utilizes latent features from items' descriptions through CNN and SVD++ that joins the user's implicit feedback. CNN considers the contextual information of the input that leads to a significant performance. The result approves that the proposed model ConvSVD++ provides better accuracy compared to baseline models. In the future, we further can incorporate temporal dynamics factors to the proposed ConvSVD++ model to get more accurate results. Additionally, further experiments should be conducted in various datasets with different density levels to ensure that the proposed model handles sparsity data and gives an accurate recommendation.

Acknowledgment

The author(s) recieved no specific funding for this work.

Author's Contributions

Mohamed Grida: Designed the research plan, coordinated the data analysis and organized the study.

Lamiaa Fayed: Participated in all experiment, coordinated the data analysis.

Mohamed Hassan: Revised the manuscript, designed the research plan.

Ethics

The authors declare that there is no conflict of interests regarding the publication of this article/paper.

References

Barragáns-Martínez, A. B., Costa-Montenegro, E., Burguillo, J. C., Rey-López, M., Mikic-Fonte, F. A., & Peleteiro, A. (2010). A hybrid content-based and item-based collaborative filtering approach to recommend TV programs enhanced with singular value decomposition. Information Sciences, 180(22), 4290-4311.

Batmaz, Z., Yurekli, A., Bilge, A., & Kaleli, C. (2019). A review on deep learning for recommender systems: challenges and remedies. Artificial Intelligence Review, 52(1), 1-37.

Bauer, J., & Nanopoulos, A. (2014). Recommender systems based on quantitative implicit customer feedback. Decision Support Systems, 68, 77-88.

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. Knowledge-based systems, 46, 109-132.

Celdrán, A. H., Pérez, M. G., Clemente, F. J. G., & Pérez, G. M. (2016). Design of a recommender system based on users' behavior and collaborative location and tracking. Journal of Computational Science, 12, 83-94.

Costa-Montenegro, E., Barragáns-Martínez, A. B., & Rey-López, M. (2012). Which App? A recommender system of applications in markets: Implementation of the service for monitoring users' interaction. Expert systems with applications, 39(10), 9367-9375.

Cremonesi, P., Garzottto, F., & Turrin, R. (2012, September). User effort vs. accuracy in rating-based elicitation. In Proceedings of the sixth ACM conference on Recommender systems (pp. 27-34).

Ghazanfar, M. A., & Prügel-Bennett, A. (2014). Leveraging clustering approaches to solve the gray-sheep users problem in recommender systems. Expert Systems with Applications, 41(7), 3261-3275.

Herlocker, J. L., Konstan, J. A., Borchers, A., & Riedl, J. (2017, August). An algorithmic framework for performing collaborative filtering. In ACM SIGIR Forum (Vol. 51, No. 2, pp. 227-234). New York, NY, USA: ACM.

Kim, D., Park, C., Oh, J., Lee, S., & Yu, H. (2016, September). Convolutional matrix factorization for document context-aware recommendation. In Proceedings of the 10th ACM conference on recommender systems (pp. 233-240).

Kurucz, M., Benczúr, A. A., & Csalogány, K. (2007, August). Methods for large scale SVD with missing values. In Proceedings of KDD cup and workshop (Vol. 12, pp. 31-38).

Li, S., Kawale, J., & Fu, Y. (2015, October). Deep collaborative filtering via marginalized denoising auto-encoder. In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (pp. 811-820).

Lu, J., Wu, D., Mao, M., Wang, W., & Zhang, G. (2015). Recommender system application developments: a survey. Decision Support Systems, 74, 12-32.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

Mnih, A., & Salakhutdinov, R. R. (2008). Probabilistic matrix factorization. In Advances in neural information processing systems (pp. 1257-1264).

Moreno, M. N., Segrera, S., López, V. F., Muñoz, M. D., & Sánchez, Á. L. (2016). Web mining based framework for solving usual problems in recommender systems. A case study for movies′ recommendation. Neurocomputing, 176, 72-80.

Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).

Qiu, T., Chen, G., Zhang, Z. K., & Zhou, T. (2011). An item-oriented recommendation algorithm on cold-start problem. EPL (Europhysics Letters), 95(5), 58003.

Ricci, F., Rokach, L., & Shapira, B. (2011). Introduction to recommender systems handbook. In Recommender systems handbook (pp. 1-35). Springer, Boston, MA.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2000). Application of dimensionality reduction in recommender system-a case study. Minnesota Univ Minneapolis Dept of Computer Science.

Shambour, Q., & Lu, J. (2015). An effective recommender system by unifying user and item trust information for B2B applications. Journal of Computer and System Sciences, 81(7), 1110-1126.

Shin, D., Cetintas, S., Lee, K. C., & Dhillon, I. S. (2015, October). Tumblr blog recommendation with boosted inductive matrix completion. In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (pp. 203-212).

Vartak, M., Thiagarajan, A., Miranda, C., Bratman, J., & Larochelle, H. (2017). A meta-learning perspective on cold-start recommendations for items. In Advances in neural information processing systems (pp. 6904-6914).

Vozalis, M. G., & Margaritis, K. G. (2005, September). Applying SVD on item-based filtering. In 5th International Conference on Intelligent Systems Design and Applications (ISDA'05) (pp. 464-469). IEEE.

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

Wang, H., Shi, X., & Yeung, D. Y. (2015a, February). Relational stacked denoising autoencoder for tag recommendation. In Twenty-ninth AAAI conference on artificial intelligence.

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

Wang, X., & Wang, Y. (2014, November). Improving content-based and hybrid music recommendation using deep learning. In Proceedings of the 22nd ACM international conference on Multimedia (pp. 627-636).

Wei, J., He, J., Chen, K., Zhou, Y., & Tang, Z. (2017). Collaborative filtering and deep learning based recommendation system for cold start items. Expert Systems with Applications, 69, 29-39.

Xie, F., Xu, M., & Chen, Z. (2012, March). RBRA: A simple and efficient rating-based recommender algorithm to cope with sparsity in recommender systems. In 2012 26th International Conference on Advanced Information Networking and Applications Workshops (pp. 306-311). IEEE.

Xiong, M. T., Feng, Y., Wu, T., Shang, J. X., Qiang, B. H., & Wang, Y. N. (2019). TDCTFIC: a novel recommendation framework fusing temporal dynamics, CNN-based text features and item correlation. IEICE Transactions on Information and Systems, 102(8), 1517-1525.

Xu, M., Berkovsky, S., Ardon, S., Triukose, S., Mahanti, A., & Koprinska, I. (2013, October). Catch-up TV recommendations: show old favourites and find new ones. In Proceedings of the 7th ACM conference on Recommender systems (pp. 285-294).

Zhang, S., Yao, L., & Xu, X. (2017, August). AutoSVD++ An Efficient Hybrid Collaborative Filtering Model via Contractive Auto-encoders. In Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval (pp. 957-960).

Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). Deep learning based recommender system: A survey and new perspectives. ACM Computing Surveys (CSUR), 52(1), 1-38.

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

Zheng, L., Noroozi, V., & Yu, P. S. (2017, February). Joint deep modeling of users and items using reviews for recommendation. In Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (pp. 425-434).