Noname manuscript No.
(will be inserted by the editor)

Edge Data Based Trailer Inception Probabilistic Matrix Factorization for Context-Aware Movie Recommendation

Honglong Chen · Zhe Li · Zhu Wang · Zhichen Ni · Junjian Li · Ge Xu · Abdul Aziz · Feng Xia

Corresponding author: Feng Xia. Email: f.xia@ieee.org.

Abstract The rapid growth of edge data generated by mobile devices and applications deployed at the edge of the network has exacerbated the problem of information overload. As an effective way to alleviate information overload, recommender system can improve the quality of various services by adding application data generated by users on edge devices, such as visual and textual information, on the basis of sparse rating data. The visual information in the movie trailer is a significant part of the movie recommender system. However, due to the complexity of visual information extraction, data sparsity cannot be remarkably alleviated by merely using the rough visual features to improve the rating prediction accuracy. Fortunately, the convolutional neural network can be used to extract the visual features precisely. Therefore, the end-to-end neural image caption (NIC) model can be utilized to obtain the textual information describing the visual features of movie trailers. This paper proposes a trailer inception probabilistic matrix factorization model called Ti-PMF, which combines NIC, recurrent convolutional neural network, and probabilistic matrix factorization models as the rating prediction model. We implement the proposed Ti-PMF model with extensive experiments on three real-world datasets to validate its effectiveness. The experimental results illustrate that the proposed Ti-PMF outperforms the existing ones.

H. Chen, Z. Li, Z. Wang, Z. Ni, and J. Li
College of Control Science and Engineering, China University of Petroleum, Qingdao 266580, China.
E-mail: chenhl@upc.edu.cn

H. Chen and G. Xu
College of Computer and Control Engineering, Minjiang University, Fuzhou 350108, China
E-mail: xuge@pku.edu.cn

A. Aziz
School of Software, Dalian University of Technology, Dalian 116620, China.

F. Xia
School of Engineering, IT and Physical Sciences, Federation University Australia, Ballarat, VIC 3353, Australia.
E-mail: f.xia@ieee.org

Keywords Neural image caption · Probabilistic matrix factorization · Recommender systems · Visual feature extraction

1 Introduction

In recent years, with the rapid development of Internet of Things (IoTs) [1–3], the number of mobile devices and applications deployed at the edge of the network to provide users with various services has increased significantly [4–8]. However, although they make full use of the computing resources of edge servers, which makes a great contribution to reducing network latency [9], they also generate a large amount of edge data, which aggravates the problem of information overload faced by the contemporary era, thus affecting the quality of various services and the user satisfaction [10]. Recommender systems can extract user preferences by using various user behaviors and application data generated on edge devices, so as to generate corresponding recommendations [11]. Unfortunately, the rapid growth of the number of users and the amount of related edge data makes data sparsity a challenging issue of the recommender systems [12–14], which severely deteriorates the recommendation performance. As an effective way, the contextual feature [15] can be utilized to customize its recommendation by adding additional information, such as visual and textual features, to alleviate the sparseness [16–19]. The original context-aware recommendation method is to get implicit feedbacks from users (such as the time spent on each page and click-through rate of each item) to infer whether the user prefers a certain item. In recent years, the deep learning-based algorithms have been well studied, for example, convolutional neural network (CNN) [20] and recurrent neural network (RNN) [21] have contributed greatly to extracting the visual features and textual features, respectively. Additionally, the movie trailer often contains a lot of important information of the whole movie. Therefore, in movie context-aware recommender system, extracting the features of movie trailers through deep learning-based algorithms would bring valuable and reliable additional information.

Most of the previous context-aware movie recommendation methods just used the static contextual features (such as the user attribute and movie attribute) to improve the recommendation performance. However, with the recent advances of deep learning algorithms, it is not hard to capture the deep features of images and videos by using deformations of CNN, such as AlexNet [22], GoogLeNet [23] and so on. Accordingly, after extracting the multiple features, the feature combination should be considered in the movie recommendation [13, 24]. Therefore, an end-to-end network neural image caption (NIC) generator combining the visual feature extraction network and textual generation network is proposed [25], which converts the movie trailer information into corresponding description texts. In the NIC model, the inputs are images, and the outputs are sentences, which are translated based on the visual features of the input images. The NIC method can speed up the trailer processing, making it feasible to utilize the visual features in a context-aware recommendation.

This paper focuses on utilizing the trailer information of movies in the context-aware recommendation to promote the performance of rating prediction. In order to avoid the coupling among different still frames, which may lead to a high similarity of the final visual features, we take 20 still frames evenly from one video track as the input of the NIC model. These still frames are input into the NIC model to generate the corresponding descriptive textual information, based on which the most accurate description texts can be extracted as the contextual text information. Note that the length of the descriptive texts of still frames extracted from movie trailers will be shorter than that of the users' review texts [26]. Moreover, it can not determine whether the user prefers the movie or not only with some review sentences, which results in low quality of textual features for the review texts. While for the descriptive text, the sentence is specifically used to describe the movie trailers, which is more concise and more valuable. Therefore, the image information description is more effective than the feature information of the review texts for the recommender systems. In this paper, we propose a trailer inception probabilistic matrix factorization model called Ti-PMF, which integrates the movie visual information to alleviate the sparsity of rating data and promote the performance of rating prediction. In the experiments, we utilize the advanced visual feature extraction network VGG and GoogLeNet respectively to evaluate the text conversion performance, and finally, adopt the texts generated by NIC to get the corresponding root mean squared error. The experimental results illustrate that the proposed Ti-PMF model significantly outperforms the existing schemes.

The main contributions of this paper are as follows:

- We propose a trailer inception probabilistic matrix factorization model called Ti-PMF, which integrates the movie visual information to alleviate the sparsity of rating data and promote the performance of rating prediction.
- We utilize the NIC model to automatically convert the video information of the movie trailers into the corresponding descriptive textual information and then embed the textual information into our recommendation model seamlessly.
- The training time of the proposed Ti-PMF method can be significantly shortened with a higher rating prediction accuracy.
- We implement the proposed Ti-PMF model with extensive experiments on three real-world datasets to validate its effectiveness.

The rest of the paper is organized as follows. Sect. 2 introduces the related work of context-aware recommendation and feature engineering in deep learning. Sect. 3 briefly reviews preliminaries on the probabilistic matrix factorization, the neural image caption generator, and textual feature extraction of recurrent convolutional neural network. Sect. 4 concretely presents our proposed model Ti-PMF. Sect. 5 shows the experiments about the proposed model and discusses the results of our model. Sect. 6 summarizes our work.

2 Related Work

2.1 Context-aware Recommendations

With the advent research on context-aware processing and becoming a hot-spot research topic in the field of recommendation, it is considered that when more contextual information is provided, better recommendation accuracy can be achieved

[27,28]. The contextual information used for the context-aware recommendations includes time, location, entity or event. The context-aware recommender system modifies the existing model to a scene in the specific dimension, realizing the context directly embedded in the recommendation process. The method provides a flexible and generalizable context-aware recommendation, which overcomes the obstacles of two-dimensional algorithms [29]. Specifically, the context-aware recommendation method can easily embed the contextual information. This method is applied to each item to make a precise recommendation for a given user u with context t, and then the top-k item recommendation is accomplished. The methods for calculating neighborhood are existing, and collaborative filtering and content-based methods are very commonly utilized in recommender systems [30, 31]. The latent factor model (LFM) recommends items with similar item features to target users based on the element features in the users' contexts. Another segment of the application of LFM is the factorization machine (FM) [32], the ratings are modeled as linear combinations of the interactions between input variables of the model. In addition, machine learning algorithms are utilized in content-based recommender systems to extract attributes associated with users, items, and contexts [33]. Moreover, in the context-aware recommender system, selecting appropriate attributes is also an important process. Common contextual information is utilized in recommender systems including: time, location, and social information [34,35]. Zarzour *et al.* [36] proposed the conception of multidimensional attributes, which uses dimensionality reduction and clustering techniques to integrate multidimensional attributes and reduce their dimensionality, thereby obtaining attributes with higher accuracy. The LSIC model proposed in [37] explores context-aware information (movie posters) and uses GAN framework to leverage the matrix factorization (MF) and RNN approaches for top-N recommendation to further improve the performance of movie recommendation. Recently, a deep learning recommendation framework that incorporates contextual information into neural collaborative filtering recommendation approaches is proposed in [38], which models contextual information in various ways for multiple purposes, such as rating prediction, generating top-k recommendations, and classification of users' feedback. Chen *et al.* [39] proposed that the existing methods suffer from context redundancy, and proposed a context-aware recommendation method based on embedded feature selection, which eliminates context redundancy by generating a minimum subset of all contextual information, and allocates weight to each context appropriately to achieve performance improvements.

2.2 Feature Engineering in Deep Learning

Feature engineering in deep learning embeds additional information into context-aware recommender system, which helps to alleviate the sparsity problem of rating data [39]. The data presented to the algorithm by feature engineering has the relevant structure or attributes of the basic data of the corresponding task [40,41]. Since there are many types of existing attribute features, the single feature [42] and the multi-feature fusion context-aware recommender system can be used [43]. Besides, feature fusion between texts is often used in text classification tasks and multiple feature weighting [44]. By preprocessing the data structure through feature engineering, the algorithm can reduce noise interference and find data trends. Specifically, in the preprocessing method based on dimensionality reduction, an item can be divided into several fictitious items by using several corresponding contexts in order to determine its attribute features [36]. Zhang *et al.* proposed a method of using context to establish user portrait features for recommendation [45]. In the movie context-aware recommender system, multiple attribute features extracted from movie descriptive texts and user review texts are utilized to obtain the personalized recommendations [46]. To select high-quality feature representations, Goldberg *et al.* discussed feature selection metrics for data classification [47]. Dense matrix data features utilize cross-feature and feature fusion technology to project attributes onto the fixed dimensional data feature spaces. Therefore, in the field of feature engineering, fewer data dimensions can be utilized to express more attributes [48,49].

3 Preliminary

3.1 Probabilistic Matrix Factorization

Probabilistic Matrix Factorization (PMF) [50] can obtain a relatively accurate prediction based on a few specific scores in the rating matrix. PMF aims to improve the rating prediction accuracy of the conventional matrix factorization by using the probabilistic method. Specifically, it is supposed that there are M movies and N users. The element R_{ij} in rating matrix $R \in \mathbb{R}^{N\times M}$ represents the rating of user i on movie j. The number of latent features is expressed as D, user matrix $U \in \mathbb{R}^{D\times N}$ and movie matrix $V \in \mathbb{R}^{D\times M}$ are both latent feature matrices, and their column vectors U_i and V_j represent the user-specific and the movie-specific latent feature vectors, respectively. Then PMF is based on the following two assumptions: 1) the observed errors follow the Gaussian distribution, 2) the user matrix U and movie matrix V follow the Gaussian distribution. The conditional distribution over the observed ratings based on the above two assumptions is:

$$p\Big(R|U,V,\sigma^2\Big) = \prod_{i=1}^{N}\prod_{j=1}^{M}\Big[N\Big(R_{ij}|U_i^T V_j,\sigma^2\Big)\Big]^{I_{ij}}, \tag{1}$$

where $N(x \mid \mu,\sigma^2)$ is the probability density function, which conforms to the Gaussian distribution with mean μ and variance σ^2. I_{ij} is an indicator function, if user i rated movie j, the function is equal to 1, otherwise it is equal to 0. The zero-mean spherical Gaussian priors are considered on the user and movie feature vectors and can be formulated as:

$$p\Big(U|\sigma_U^2\Big) = \prod_{i=1}^{N} N\Big(U_i|0,\sigma_U^2 I\Big), \tag{2}$$

$$p\Big(V|\sigma_V^2\Big) = \prod_{j=1}^{M} N\Big(V_j|0,\sigma_V^2 I\Big). \tag{3}$$

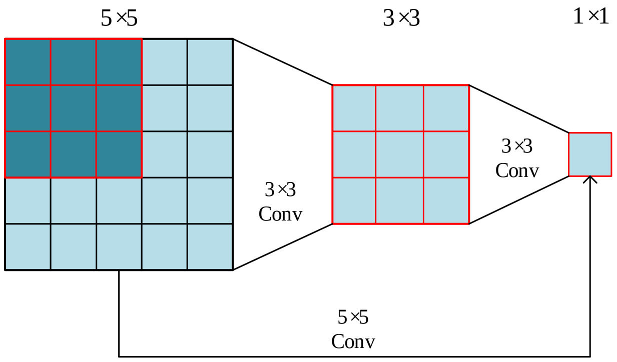


Fig. 1 A 5*5 filter is replaced with double 3*3 filters (stride = 1).

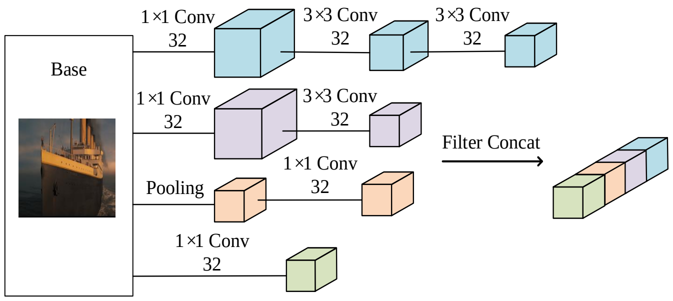


Fig. 2 The architecture of basic inception model with multiple convolutional kernels (inception-a).

Note that I in the above equation is not an indicator function, it represents a diagonal matrix. The $L2$ regularization term is applied to avoid over-fitting, and the loss function can be formulated as:

$$\mathcal{L}(U,V) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}\left(R_{ij} - U_i^T V_j\right)^2 + \frac{\lambda_U}{2}\sum_{i=1}^{M}\left\|U_i\right\|_F^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M}\left\|V_j\right\|_F^2, \tag{4}$$

where $\lambda_U = \sigma^2/\sigma_U^2$, $\lambda_V = \sigma^2/\sigma_V^2$, and $\|\cdot\|_F^2$ denotes the frobenius norm.

3.2 Neural Image Caption Generator

Neural image caption (NIC) generator is an end-to-end network with images as the input and text sequences as the output. Specifically, NIC integrates GoogLeNet for extracting the images' visual features and RNN for converting the visual features into sequential texts.

3.2.1 GoogLeNet

Before GoogLeNet, a deep learning structure, was proposed [51], AlexNet, VGG and other structures used the method of increasing the depth of the network to achieve better training results, which would bring negative effects such as over-fitting, gradient disappearance, gradient explosion and so on. Inception is the basic component of the GoogLeNet network. The introduction of inception can make effective use of computing resources and obtain more features under the same amount of computation, which can improve the training effect. Specifically, various sizes of convolutional kernels (1×1, 3×3, 5×5) are utilized, which can perceive different perceptive fields and obtain more comprehensive and richer visual feature information. Since the receptive field of the 5×5 convolutional kernel is the same as the receptive field of two 3×3 convolutional kernels, the number of training parameters, i.e., 18, will be smaller than that of previous training parameters, i.e., 25. Therefore, one 5×5 convolutional kernel can be replaced by two 3×3 convolutional kernels [52], as shown in Fig. 1. And Fig. 2 shows the structure of inception [23]. Besides, 1×1 convolutional kernel and pooling operation are used to compute reductions before the expensive 3×3 and 5×5 kernels. In the GoogLeNet model, 1×1 convolutional kernel is mainly utilized for dimensionality reduction for image data. Although the large convolutional kernel is very helpful for extracting visual features, it will cause a parameter explosion in the deep neural network. Therefore, Szegedy *et al.* decomposed the large convolutional kernel asymmetrically to reduce the number of parameters [23]. Asymmetrical convolutional structure splitting is better than symmetrical convolutional structure splitting in processing more and richer spatial features and increasing feature diversity when reducing the amount of calculation. Specifically, an $n\times n$ convolutional kernel can be replaced by two convolutional kernels, a $1\times n$ kernel followed by an $n\times1$ one. In addition, the computational cost saved by the replacement can increase significantly with the increase of n. Finally, the CNN structure is equivalently replaced by the mini factorization form. The specific value of n is determined according to the size of the input images. For instance, using a 3×1 convolutional kernel followed by a 1×3 one is equivalent to sliding one layer network with the same receptive field as in a 3×3 convolutional kernel. Accordingly, the number of parameters in training will be reduced from $3\times3=9$ to $3+3=6$, which is shown in Fig 3. The second and third forms of inception are to factorize part of big convolutional kernels (3×3) [23], as shown in Fig. 4(a) and Fig. 4(b). Finally, the above three inception blocks (Fig. 2, Fig. 4(a), Fig. 4(b)) are combined to the final visual neural network. The network structures of VGG16 [53], VGG19 [53] and GoogLeNet [23] are shown in TABLE 1.

3.2.2 Long-Short Term Memory

In order to effectively process the long sequences and solve the problem of gradient disappearance, long-short term memory (LSTM) is proposed on the basis of traditional RNN. When LSTM is utilized to process the information of each neuron in sequences, the true meaning of the current word in the sequence is inferred by the understanding of a previously seen word. The memory cell c is the core of the LSTM

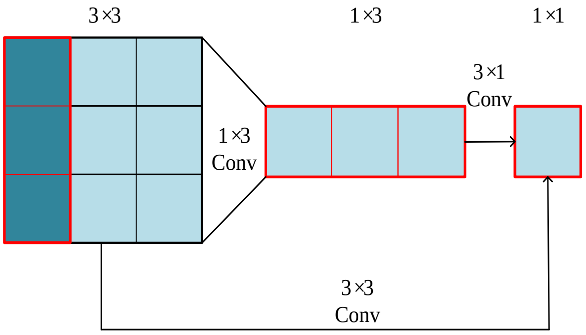
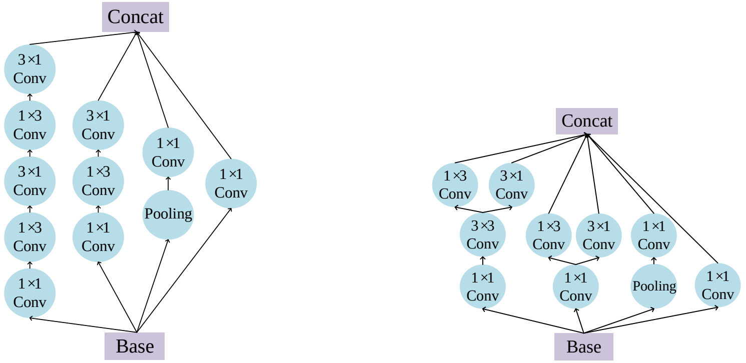


Fig. 3 A 3*3 filter is factorized by a 1*3 filter and a 3*1 filter (stride = 1).



(a) Inception-b

(b) Inception-c

Fig. 4 The architectures of the Inception-b and Inception-c.

model, which encodes the information of the observed inputs at every time step. The cell's behavior is controlled by three different gates: input gate, output gate, and forget gate. When the value of forget gate is set as 1, the information in LSTM will be maintained, and 0 means that the information will be forgotten. In particular, the three gates are used to control whether to forget the current cell value (forget gate f), whether to read the input (input gate i), and whether to output the new cell value (output gate o). The definitions of the gates, cell update, and output are as follows:

$$i_t = \sigma(W_{ix}x_t + W_{im}m_{t-1}), \tag{5}$$

$$f_t = \sigma(W_{fx}x_t + W_{fm}m_{t-1}), \tag{6}$$

$$o_t = \sigma(W_{ox}x_t + W_{om}m_{t-1}), \tag{7}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot h(W_{cx}x_t + W_{cm}m_{t-1}), \tag{8}$$

$$m_t = o_t \odot c_t, \tag{9}$$

Table 1 The network structures of VGG16, VGG19 and GoogLeNet.

Input		
VGG16	VGG19	GoogLeNet
2×conv3-64	2×conv3-64	conv3-32
pool2-64	pool2-64	conv3-32
2×conv3-128	2×conv3-128	pool3-64
pool2-128	pool2-128	conv3-64
3×conv3-256	4×conv3-256	conv3-80
pool2-256	pool2-256	conv3-192
3×conv3-512	4×conv3-512	3×inception-a
pool2-512	pool2-512	5×inception-b
3×conv3-512	4×conv3-512	2×inception-c
pool2-512	pool2-512	pool8-2048
2×fc1-4096	2×fc1-4096	fc1-2048
fc1-1000	fc1-1000	softmax1-1000
Output		

$$p_{t+1} = Softmax(m_t), \tag{10}$$

where $\odot$ represents the product with a gate value. The nonlinearities are sigmoid $\sigma(\cdot)$ and hyperbolic tangent $h(\cdot)$. The matrices W_{ix}, W_{im}, W_{fx}, W_{fm}, W_{ox}, W_{om}, W_{cx}, and W_{cm}, are the trained parameters. In Eq. (10), m_t is fed to the softmax function, resulting in a probability distribution p_t over all words.

3.3 Textual Feature Extraction in RCNN

3.3.1 Recurrent Structure in Convolutional Layer

Recurrent convolutional neural network (RCNN) [54] model embeds the recurrent structure into the convolutional layer. On the one hand, CNN is utilized to extract the textual features. On the other hand, it can also use RNN structure to memorize the full-textual information. Meanwhile, the recurrent structure can obtain the contextual information as much as possible, which means that less noise may be introduced than the window-based neural networks. The features extracted by RCNN are used as a part of the mean of Gaussian distribution in the item latent models. Specifically, in the RCNN model, a word is combined with both of the left and right contexts to represent itself in the word representation. Consequently, a word representation of RCNN contains much richer information with its associated contextual information than that of CNN. The specific context expressions are as follows:

$$c_l(w_i) = ReLU\Big(W^{(l)} c_l\big(w_{i-1}\big) + W^{(sl)} e\big(w_{i-1}\big)\Big), \tag{11}$$

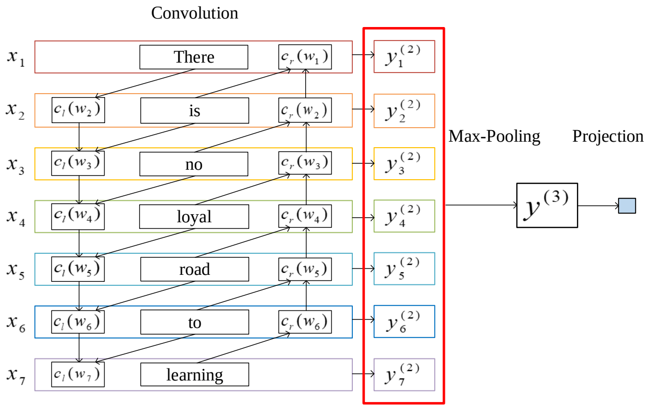


Fig. 5 RCNN model used in this paper.

$$c_r(w_i) = ReLU\Big(W^{(r)}c_r\big(w_{i+1}\big) + W^{(sr)}e\big(w_{i+1}\big)\Big), \tag{12}$$

where $c_l(w_i)$ and $c_r(w_i)$ represent the left and the right contexts of the word w_i respectively, $e(w_i)$ represents the word representation of word w_i, $W^{(sl)}$ and $W^{(sr)}$ represent the matrices, which are used to connect the semantics of the current word with the left and right adjacent words respectively. Furthermore, $W^{(l)}$ and $W^{(r)}$ represent the matrices, which combine all of the left and right context hidden layers, respectively. Then the context information and the word representation are cascaded as the whole word embedding model. Specifically, the word representation x_i of word w_i with its context information can be expressed as follows:

$$x_i = \Big[c_l\big(w_i\big), e\big(w_i\big), c_r\big(w_i\big)\Big]. \tag{13}$$

Note that different context window sizes can be utilized to capture different contextual information so as to investigate the performance more comprehensively. For instance, the word representation of w_i is represented by $\big[x(w_{i-1}); x(w_i); x(w_{i+1})\big]$ when the context window size is set to 3. Furthermore, an activation function, i.e., tanh, is applied to transform x_i into $y_i^{(2)}$ as follows:

$$y_i^{(2)} = \tanh\Big(W^{(2)}x_i + b^{(2)}\Big). \tag{14}$$

The RCNN model used in this paper is shown in Fig. 5.

3.3.2 Recurrent convolutional matrix factorization

The generated texts are input into the recurrent convolutional matrix factorization (RConvMF) recommender system, which combines RCNN with PMF. Thereby, the movie latent model with visual trailer features is obtained by the following equations:

$$V_j = rcnn(W, X_j) + \varepsilon_j, \tag{15}$$

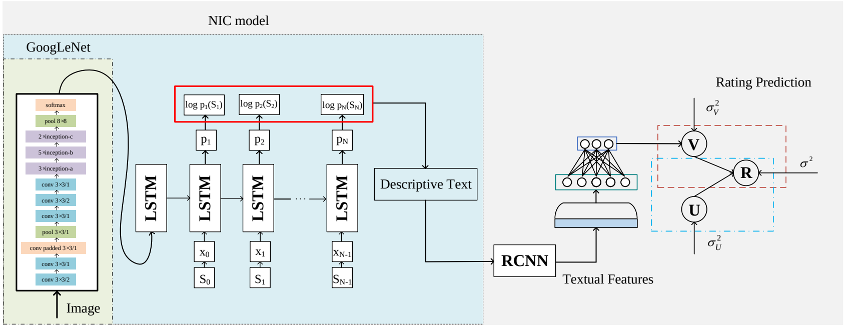


Fig. 6 The architecture of the proposed Ti-PMF model.

$$\varepsilon_j \sim N(0, \sigma_V^2 I), \tag{16}$$

where X_j represents the descriptive texts converted from the visual information extracted from the movie trailers. In Eq. (15), for each weight w_k in W, the zero-mean spherical Gaussian prior is shown as follows:

$$p(W|\sigma_W^2) = \prod_k N(w_k|0, \sigma_W^2). \tag{17}$$

Accordingly, the conditional distribution over item latent models is given by:

$$p(V|W, X, \sigma_V^2) = \prod_j^M N(V_j|rcnn(W, X_j), \sigma_V^2 I), \tag{18}$$

where X is the set of descriptive documents of items obtained through the NIC model from the movie trailers. A document latent vector obtained from the RCNN model is taken as the mean of Gaussian distribution, and the Gaussian noise of the item is taken as the variance of the gaussian distribution. In this way, the NIC, RCNN, and PMF model can be connected seamlessly.

3.3.3 Optimization Methodology

To optimize the variables such as the weights and bias of RCNN, the maximum a posteriori (MAP) estimation is utilized, which can be expressed as follows.

$$\begin{aligned}
&\max_{U,V,W} p\left(U, V, W \middle| R, X, Y, \sigma^2, \sigma_U^2, \sigma_V^2, \sigma_W^2\right) \\
&= \max_{U,V,W} \left[p\left(R \middle| U, V, \sigma^2\right) p\left(U \middle| \sigma_U^2\right) p\left(V \middle| W, X, Y, \sigma_V^2\right) p\left(W \middle| \sigma_W^2\right)\right].
\end{aligned} \tag{19}$$

Coordinate descent optimization method is adopted in training. It optimizes a latent variable iteratively while fixing the other ones. The optimal solution of U_i and V_j can be obtained as follows:

$$U_i \leftarrow \left(V I_i V^T + \lambda_U I_K\right)^{-1} V R_i, \tag{20}$$

$$V_j \leftarrow \left(UI_jU^T + \lambda_V I_K\right)^{-1}\left(UR_j + \lambda_V \mu_j\right), \tag{21}$$

where I_i is a diagonal matrix with I_{ij}, $j = 1, \cdots, n$ as its diagonal elements and R_i is a vector for user i with $(r_{ij})_{j=1}^{n}$. The back-propagation algorithm is applied to optimize W. In the whole optimization process, the unobserved rating of user i on movie j can be predicted as: $\widehat{R_{ij}} \approx E\left[R_{ij}\middle|U_i^T V_j, \sigma^2\right] = U_i^T V_j = U_i^T\left(\mu_j + \varepsilon_j\right)$.

4 Ti-PMF Model

A probabilistic neural framework is proposed in this paper to generate the descriptive documents from images. The mean idea behind the proposed model, named trailer-inception probabilistic matrix factorization (Ti-PMF), is to convert the images extracted from the movie trailers into the corresponding description texts, which will be used in the context-aware recommender system. It is possible to obtain the precise descriptive sentences of the corresponding images by directly maximizing the likelihood $p(S|I)$ of generating a target sequence of words S. The above description can be summarized as follows:

$$\theta^* = arg \max_{\theta} \sum_{(I,S)} \log p(S|I;\theta), \tag{22}$$

where θ is the parameters of the proposed model, I is a set of images, and S is the set of correct transcriptions of I. Since S represents a sentence with an unfixed length, it will be appropriate to use the chain rule to model the joint probability over $S_0, \cdots, S_N$. Specifically, the textual information before time t and the input image information can be used to predict the textual information at t:

$$\log p(S|I) = \sum_{t=0}^{N} \log p(S_t|I, S_0, \cdots, S_{t-1}). \tag{23}$$

The LSTM model can be trained to predict every word in the sentence with the prerequisite that the images and the preceding words are known, and the probability of correct prediction is defined by $p(S_t|I, S_0, \cdots, S_{t-1})$. Specifically, as for each word in a sentence, LSTM share the same parameters in all blocks. The output m_{t-1} at time $t-1$ will be fed to the LSTM at time t. The architecture of the proposed Ti-PMF model with GoogLeNet is shown in Fig. 6. Note that in the proposed Ti-PMF model, the GoogLeNet can also be replaced with VGG, the performance comparison is shown in the next section. In the Ti-PMF model, as shown in Fig. 6, the NIC model combines the unrolling LSTM and GoogLeNet. Then the RCNN model is utilized to extract features of the textual information generated by NIC. After that, the RConvMF algorithm in Sect. 3.3.2 can be used to predict the ratings. The procedure of the NIC model is represented as follows in Eqs. (24), (25) and (26):

$$x_{-1} = CNN(I), \tag{24}$$

$$x_t = W_e S_t, \tag{25}$$

$$p_{t+1} = LSTM(x_t), \tag{26}$$

Table 2 Data statistics of pre-trained datasets.

Datasets	Train	Test	Valid
Flickr8k	6000	1000	1000
Flickr30k	28000	1000	1000
MSCOCO	82783	40775	40504

where $t \in \{0, \cdots, N-1\}$, x_{-1} represents that the visual feature of image I, which is input into the NIC model at $t = -1$, S_t represents the textual information encoded by one-hot, W_e is the word2vec [48] model transition matrix to convert S_t into a dense numeric matrix. Finally, the textual features are input into the PMF model to get an accurate rating prediction. The negative log-likelihood of the correct word at each step can be expressed as:

$$L(I, S) = -\sum_{t=1}^{N} \log p_t(S_t). \tag{27}$$

The above loss function can be minimized over all the parameters in the LSTM model, the top layer of the image embedder CNN and the word embeddings W_e.

5 Experiments

5.1 Datasets

The datasets used in this paper are divided into the following two parts.

- **Model pre-training**: To avoid overfitting, the GoogLeNet, LSTM, and RCNN models need to be pre-trained initially. Specifically, the ImageNet dataset is utilized to pre-train the GoogLeNet and VGG models, which are used for comparative experiments, and the datasets consisting of images and English sentences describing these images (such as MSCOCO, Flickr8k, and Flickr30k) are also used. The statistics of the datasets are shown in TABLE 2.
- **Recommender system**: The main objective of the recommender system is to predict the target users' ratings of unknown movies. We validate the proposed Ti-PMF model on three different real-world datasets, including MovieLens-1m (marked as ML-1M), MovieLens-10m (marked as ML-10M), and Amazon Instant Video (marked as AIV). As shown in Fig. 7, we perform simple sentence length statistics about ML-10M dataset on the length of text sentences. Specifically, the main statistics of the three datasets are shown in TABLE 3.

Since there is no real-world dataset of movie trailers corresponding to user ratings, we crawl the video clips of the movie ID on Youtube and IMDB. For the few early movies without trailers, we randomly select a part of the video as the trailer from the movie. Before using the NIC model to extract the image features and generate the sentence texts, we need to extract the images from the trailers. We randomly

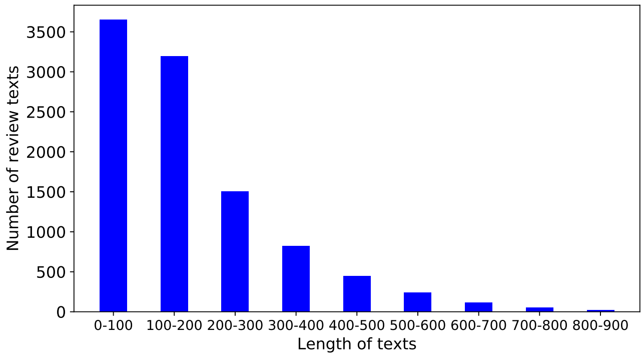


Fig. 7 Data statistics of reviews in ML-10M dataset.

Table 3 Data statistics on three real-world datasets.

Datasets	Users	Movies	Ratings	Sparsity
ML-1M	6040	3883	1000209	95.73%
ML-10M	71567	10681	10000054	98.69%
AIV	29757	15149	135188	99.97%

select 20 still frames as the input of the NIC model. In addition, since the item description documents are not contained in the MovieLens dataset, we get them from IMDB[1], Douban[2] and Youtube [55].

5.2 Experiment Settings

5.2.1 Model Settings

- **NIC model (VGG16/19)**: Each image in the Flickr8k and Flickr30k datasets has five reference captions. Accordingly, the part of the MSCOCO dataset that exceeds five captions is deleted. All of the pre-training images in the datasets are resized into 224 × 224 × 3. As the VGG network is extremely deep, the Batch-Normalization [52] method is adopted in each layer to avoid the internal covariate shift. The dropout rate of the VGG model is set as 0.3 during the training process, and the mini-batch is set as 128.
- **NIC model (Inception)**: The receptive field of the input image is cropped to 299 × 299 with stride 2. We put the max-pooling layer behind the first layer to reduce the parameters of the GoogLeNet, and the number of inception-a, b, c

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

[2] https://movie.douban.com/

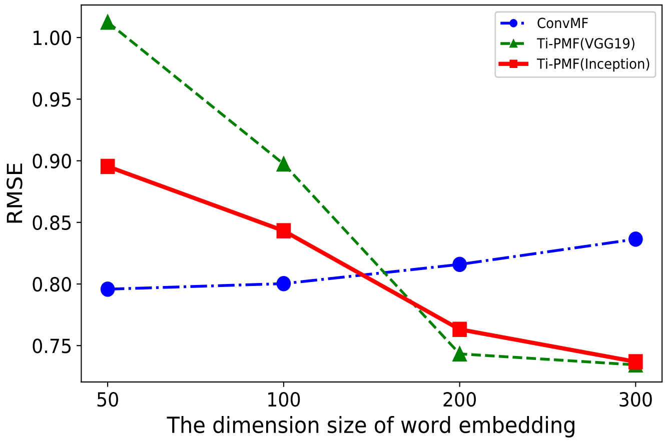


Fig. 8 The effect of the dimension size of word embedding on ML-10M.

is set as 3, 5, 2, respectively. In order to enhance the effect of low-dimensional feature representation, we set the depth of the convolutional kernel to 2048 as the layers increase. At the end of the network, a max-pooling layer with 8 × 8 patch size is set to compress the features to the single-dimension deep feature vector $(1 \times 1 \times 2048)$. Finally, the visual features are scaled into $(1 \times 1 \times 1000)$ through a fully connected layer.

- **NIC model (LSTM)**: We adopt the tokenization method in the NLTK library for the word labeling. The NIC model generates the texts, and the maximum raw text is set as 30. For the word embedding, the word2vec model is utilized to convert the natural language to machine language. And the word vector is initialized randomly with dimension size of 200. Fig. 8 shows the RMSE of Ti-PMF and ConvMF with the dimension of word embedding varying from 50 to 300. As the description texts generated by the NIC model are shorter than the comment texts, which are used as the textual information in the ConvMF model, Ti-PMF has a faster convergence speed. We will train these word latent vectors in the optimization process. Various convolutional window sizes (3, 4, 5) with 100 feature maps are utilized to obtain different contextual information.
- **Ti-PMF**: The dimensions of user latent vector (U) and item latent vector (V) are both set as 50, and we initialize U and V with each element randomly selected in the range of $(0, 1)$. The mini-batch size is set as 128. Thereafter, we put comprehensive features into the projection layer and fix the dimension to 50. We set the precision parameter of CTR and CDL to 1 when r_{ij} is observed, otherwise it is set to 0. The number of iterations is set as 15 in Ti-PMF. An average value of five repeated experiments is performed as the final result to reduce the random error.

The NIC network is trained with stochastic gradient descent. We set the batch size as 64 for 50 epochs, and the model can be achieved using RMSProp with the decay of 0.9 and $\epsilon = 1.0$. We set the learning rate as 0.045, and the dropout rate as

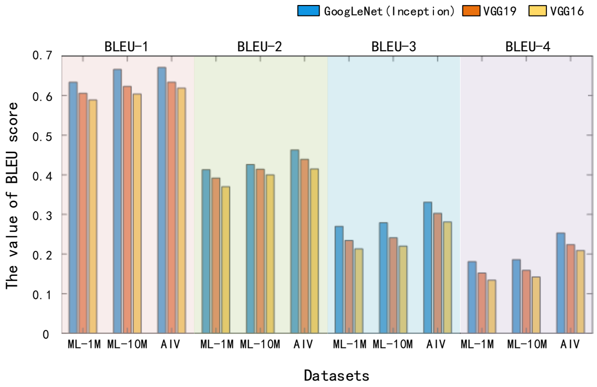


Fig. 9 The BLEU score of NIC model on three different real-world datasets.

0.2 to avoid over-fitting. The Ti-PMF network is trained with the coordinate descent method utilizing the Theano backend, and the dropout rate of Ti-PMF is 0.4.

5.2.2 Evaluation Metrics

We divide the performance evaluation of the proposed Ti-PMF model into two parts. The first part is to evaluate the precision of word n-grams. And the other part is to evaluate the rating prediction accuracy of recommender systems.

- **BLEU** [56]: Bilingual evaluation understudy (BLEU) is an auxiliary tool for bilingual translation quality assessment. It is a metric used to evaluate the quality of machine translation. Since manual processing is too time-consuming and laborious, the BLEU method is utilized to evaluate the quality of generated texts by machines. In our experiments, the NIC model is evaluated with four indicators (i.e., BLEU-1 to BLEU-4). Specifically, BLEU-n respectively represent the value of n in the $n - gram$. BLEU is a measure of the matching degree between the generated textual sequences and the texts in the ground truth.
- **RMSE**: Before training, each dataset is randomly split into a training set (80%), a test set (10%) and a validation set (10%). Then, to validate the performance of the proposed Ti-PMF model, we select the root mean squared error (RMSE) as the evaluation metric:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{m}\sum_{j=1}^{n}(r_{ij} - \widehat{r_{ij}})^2}{\left|N\right|}}, \tag{28}$$

where $\left|N\right|$ is the number of test ratings. We set the number of iterations to 30.

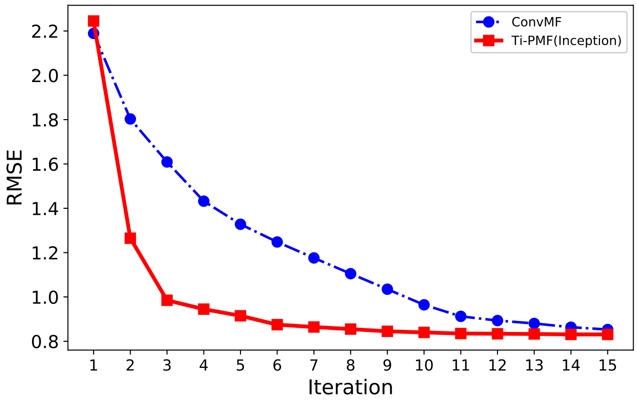


Fig. 10 Comparative experiments of the iterative process between ConvMF and Ti-PMF on the ML-1m dataset.

5.3 Compared Schemes

- **PMF** [50]: Probabilistic matrix factorization is a basic method of rating prediction, which uses the rating scores in the form of probability.
- **CTR** [57]: Collaborative topic regression is a recommendation algorithm, which combines PMF and the latent Dirichlet allocation. Both the ratings and textual documents are used in CTR.
- **CDL** [58]: Collaborative deep learning is a recommendation model that improves the recommendation performance by using the stacked denoising autoencoder.
- **ConvMF** [59]: Convolutional matrix factorization is a context-aware recommendation model, which combines CNN and PMF seamlessly to improve the rating prediction accuracy.

5.4 Results and Discussion

*5.4.1 **Comparison of VGG and Inception in NIC Model***

In the NIC model, comparative experiments are implemented by using GoogLeNet and VGG models to connect LSTM, respectively. VGG improves the performance of CNN from the perspective of changing the network depth, while GoogLeNet improves the performance by expanding the network width. In the final result, VGG performs a bit better than GoogLeNet. However, VGG requires more calculations during the training process. Therefore, in the application, the GoogLeNet is used to predict the ratings of the recommender systems. The experimental results compared with VGG are shown in Fig. 9. BLEU-1 to BLEU-4 represent the accuracy of the generated textual information under $1 - gram$ to $4 - gram$ evaluation metrics. Fig. 9 shows that GoogLeNet achieves better performance in generating the sentences by the

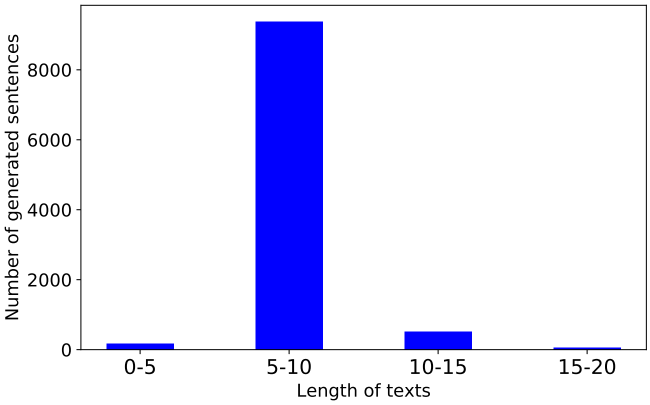


Fig. 11 Data statistics of descriptive text generated by NIC model in ML-10M dataset.

Table 4 RMSE of overall test sets.

	Dataset		
Model	ML-1M	ML-10M	AIV
PMF	0.8971	0.8311	1.4118
CTR	0.8969	0.8275	1.4594
CDL	0.8879	0.8186	1.3594
ConvMF	0.8531	0.7958	1.1337
Ti-PMF (VGG19)	**0.8120**	**0.7344**	**1.0125**
Ti-PMF (Inception)	0.8310	0.7369	1.0160
Improved	4.82%	7.40%	10.38%

NIC model. Moreover, in the NIC model, GoogLeNet is superior to VGG in terms of convergence speed. The superposition of multiple convolutional kernels (5×5, 3×3, 1×1) makes GoogLeNet obtain better performance than the VGG with the single-size convolutional kernel (3×3).

5.4.2 ***Impact of NIC in Ti-PMF Model***

Inception model in Ti-PMF is indeed faster than ConvMF model as shown in Fig. 10. Therefore, it can be considered that the textual information of texts generated by the NIC model is short and concise. As shown in Fig. 11, for the statistics of image description texts generated by the NIC model, 98% of the image description texts are with a length shorter than 10, which is much smaller than that of the users' comment texts. The generated textual information is finally input into the RConvMF model.

5.4.3 Overall Performance

TABLE 4 shows the overall rating prediction performance of the proposed Ti-PMF and other schemes. Compared with PMF, CTR and CDL, Ti-PMF achieves significant performance improvement. Compared with ConvMF, the improvements of Ti-PMF are 4.82%, 7.40%, and 10.38% on ML-1M, ML-10M and AIV datasets, respectively. The significant improvement of rating prediction performance is due to the model Ti-PMF combining RCNN and PMF to process the concise text generated by the visual information of movie trailers to extract richer contextual information, which verifies the effectiveness of the model. It is worth noting that in Ti-PMF, the training time of VGG is three times that of Inception, and better experimental results than Inception are obtained on three datasets. Moreover, the improvement of Ti-PMF on dataset AIV is much more significant than that on datasets ML-1M and ML-10M. The results in TABLE 4 show that Ti-PMF achieves a more remarkable improvement when the dataset is with a higher sparsity, indicating that Ti-PMF can effectively alleviate the problem of data sparsity in recommender systems.

6 Conclusions and Future Work

In this paper, we propose a trailer inception probabilistic matrix factorization model called Ti-PMF. The proposed Ti-PMF model combines the neural image caption, recurrent convolutional neural network, and probabilistic matrix factorization model as the rating prediction model of recommender systems. We implement the proposed Ti-PMF model and conduct extensive experiments on three real-world datasets to illustrate that the proposed Ti-PMF model outperforms the existing ones.

In future work, we will take into account the user attributes (such as gender, age, and occupations) to promote the performance of rating prediction. Furthermore, we also consider using the RNN model to extract the video features to improve the recommendation performance.

ACKNOWLEDGEMENT

This work was supported in part by NSFC grants (No.61772551), the Open Project of Minjiang University (No.MJUKF-JK202003) and the Major Scientific and Technological Projects of CNPC under Grant ZD2019-183-003.

References

1. X. Ai, H. Chen, K. Lin, Z. Wang, J. Yu, Nowhere to Hide: Efficiently Identifying Probabilistic Cloning Attacks in Large-Scale RFID Systems, IEEE Transactions on Information Forensics and Security 16 (2021) 714–727.
2. K. Lin, H. Chen, N. Yan, Z. Li, J. Li, N. Jiang, Fast and Reliable Missing Tag Detection for Multiple-Group RFID System, IEEE Transactions on Industrial Informatics, DOI: 10.1109/TII.2021.305895.
3. H. Chen, X. Ai, K. Lin, N. Yan, N. Jiang, Z. Wang, J. Yu, Fast and Reliable Missing Tag Detection for Multiple-Group RFID System, IEEE Transactions on Industrial Informatics 18 (2022) 345–355.

4. Q. He, B. Li, F. Chen, J. Grundy, X. Xia, Y. Yang, Diversified Third-party Library Prediction for Mobile App Development, IEEE Transactions on Software Engineering, DOI: 10.1109/TSE.2020.2982154.
5. G. Cui, Q. He, F. Chen, Y. Zhang, H. Jin, Y. Yang, Interference-aware Game-theoretic Device Allocation for Mobile Edge Computing, IEEE Transactions on Mobile Computing, DOI: 10.1109/TMC.2021.3064063.
6. H. Zhou, X. Chen, S. He, J. Chen, J. Wu, DRAIM: A Novel Delay-constraint and Reverse Auction-based Incentive Mechanism for WiFi Offloading, IEEE Journal on Selected Areas in Communication 38 (4) (2020) 711–722.
7. H. Zhou, T. Wu, H. Zhang, J. Wu, Incentive-driven Deep Reinforcement Learning for Content Caching and D2D Offloading, IEEE Journal on Selected Areas in Communication 39 (8) (2021) 2445–2460.
8. S. Liu, J. Yu, X. Deng, S. Wan, FedCPF: An Efficient-Communication Federated Learning Approach for Vehicular Edge Computing in 6G Communication Networks, IEEE Transactions on Intelligent Transportation Systems, DOI: 10.1109/TITS.2021.3099368.
9. P. Lai, Q. He, X. Xia, F. Chen, M. Abdelrazek, J. Grundy, J. G. Hosking, Y. Yang, Dynamic User Allocation in Stochastic Mobile Edge Computing Systems, IEEE Transactions on Services Computing, DOI: 10.1109/TSC.2021.3063148.
10. Y. Gong, Z. Jiang, Y. Feng, B. Hu, K. Zhao, Q. Liu, W. Ou, EdgeRec: Recommender System on Edge in Mobile Taobao, in: Proc. of ACM CIKM, 2020, pp. 2477–2484.
11. M. Altulyan, L. Yao, X. Wang, C. Huang, S. S. Kanhere, Q. Z. Sheng, Recommender Systems for the Internet of Things: A Survey, arXiv preprint arXiv:2007.06758.
12. G. Penhu, C. Hauff, What does BERT know about books, movies and music? Probing BERT for conversational recommendation, in: Proc. of ACM RecSys, Rio de Janeriro, Brazil, 2020, pp. 388–397.
13. H. Chen, S. Wang, N. Jiang, Z. Li, N. Yan, L. Shi, Trust-aware Generative Adversarial Network with Recurrent Neural Network for Recommender, International Journal of Intelligent Systems 36 (2021) 778–795.
14. Z. Li, H. Chen, K. Lin, V. Shakhov, L. Shi, J. Yu, From edge data to recommendation: A double attention-based deformable convolutional network, Peer-to-Peer Networking and Applications, DOI: 10.1007/s12083-020-01037-7.
15. S.-Y. Chou, J.-S. R. Jang, Y.-H. Yang, Fast Tensor Factorization for Large-Scale Context-Aware Recommendation from Implicit Feedback, IEEE Transactions on Big Data 6 (1) (2020) 201–208.
16. W. Wang, J. Liu, Z. Yang, X. Kong, F. Xia, Sustainable Collaborator Recommendation Based on Conference Closure, IEEE Transactions on Computational Social Systems 6 (2) (2019) 311–322.
17. A. Li, B. Yang, GSIRec: Learning with graph side information for recommendation, World Wide Web, DOI: 10.1007/s11280-021-00910-6.
18. Z. Wang, H. Chen, Z. Li, K. Lin, N. Jiang, F. Xia, VRConvMF: Visual Recurrent Convolutional Matrix Factorization for Movie Recommendation, IEEE Transactions on Emerging Topics in Computational Intelligence, DOI:10.1109/TETCI.2021.3102619.
19. S. Wan, S. Ding, C. Chen, Edge computing enabled video segmentation for real-time traffic monitoring in internet of vehicles, Pattern Recognition 121 (2022) 108146.
20. R. Guo, H. Xia, J. Li, D. Liu, DRCGR: Deep reinforcement learning framework incorporating CNN and GAN-based for interactive recommendation, in: Proc. of IEEE ICDM, Beijing, China, 2019, pp. 1048–1053.
21. Q. Cui, S. Wu, Q. Liu, W. Zhong, L. Wang, MV-RNN: A Multi-View Recurrent Neural Network for Sequential Recommendation, IEEE Transactions on Knowledge and Data Engineering 32 (2) (2020) 317–331.
22. J. Zhang, Y. Yang, L. Zhuo, Q. Tian, X. Liang, Personalized Recommendation of Social Images by Constructing a User Interest Tree With Deep Features and Tag Trees, IEEE Transactions on Multimedia 21 (11) (2019) 2762–2775.
23. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shelens, Rethinking the Inception Architecture for Computer Vision, in: Proc. of IEEE CVPR, 2016, pp. 2818–2826.
24. F. Xia, N. Y. Asabere, A. M. Ahmed, J. Li, X. Kong, Mobile Multimedia Recommendation in Smart Communities: A Survey, IEEE Access 1 (1) (2013) 606–624.
25. O. Vinyals, A. Toshev, S. Bengio, D. Erhan, Show and Tell: A Neural Image Caption Generator, in: Proc. of IEEE CVPR, 2015, pp. 3156–3164.
26. Z. Li, B. Wu, Q. Liu, L. Wu, H. Zhao, T. Mei, Learning the Compositional Visual Coherence for Complementary Recommendations, in: Proc. of IJCAI, Yokohama, Japan, 2021, pp. 3536–3543.

27. N. Y. Asabere, F. Xia, W. Wang, J. J. Rodrigues, F. Basso, J. Ma, Improving Smart Conference Participation through Socially-Aware Recommendation, IEEE Transactions on Human-Machine Systems 44 (5) (2014) 689–700.
28. H. Liu, F. Xia, Z. Chen, N. Y. Asabere, J. Ma, R. Huang, TruCom: Exploiting Domain-Specific Trust Networks for Multi-Category Item Recommendation, IEEE Systems Journal 11 (1) (2017) 295–304.
29. L. Baltrunas, B. Ludwig, F. Ricci, Matrix Factorization Techniques for Context Aware Recommendation, in: Proc. of ACM RecSys, Chicage, USA, 2011, pp. 301–304.
30. L. Zhao, Z. Lu, S. J. Pan, Q. Yang, Matrix Factorization+ for Movie Recommendation, in: Proc. of IJCAI, 2016, pp. 3945–3951.
31. X. Zhou, J. He, G. Huang, Y. Zhang, SVD-based incremental approaches for recommender systems, Journal of Computer and System Sciences 81 (2015) 717–733.
32. W. Pan, Z. Liu, Z. Ming, H. Zhong, X. Wang, C. Xu, Compressed knowledge transfer via factorization machine for heterogeneous collaborative recommendation, Konwledge-Based Systems 85 (2015) 234–244.
33. Q. Zhu, D. Wang, A SVM recommendation IoT model based on similarity evaluation and collaborative filtering of multi-angle knowledge units, International Journal of Computers and Applications 42 (3) (2020) 278–281.
34. F. Xia, A. M. Ahmed, L. T. Yang, J. Ma, J. J. Rodrigues, Exploiting Social Relationship to Enable Efficient Replica Allocation in Ad-hoc Social Networks, IEEE Transactions on Parallel and Distributed Systems 25 (12) (2014) 3167–3176.
35. F. Xia, A. M. Ahmed, L. T. Yang, Z. Luo, Community-Based Event Dissemination with Optimal Load Balancing, IEEE Transactions on Computers 64 (7) (2014) 1857–1869.
36. H. Zarzour, Z. Al-Sharif, M. Al-Ayyoub, Y. Jararweh, A New Collaborative Filtering Recommendation Algorithm Based on Dimensionality Reduction and Clustering Techniques, in: Proc. of ICICS, Wuhan, 2018, pp. 102–106.
37. W. Zhao, B. Wang, M. Yang, J. Ye, Z. Zhao, X. Chen, Y. Shen, Leveraging Long and Short-Term Information in Content-Aware Movie Recommendation via Adversarial Training, IEEE Transactions On Cybernetics 50 (11) (2019) 4680–4693.
38. M. Unger, A. Tuzhilin, A. Livne, Context-Aware Recommendations Based on Deep Learning Frameworks, ACM Transactions on Management Information Systems, DOI:10.1145/3386243 11 (2020) 1–15.
39. L. Chen, M. Xia, A context-aware recommendation approach based on feature selection, Applied Intelligence 51 (3) (2020) 865–875.
40. J. Liu, F. Xia, L. Wang, B. Xu, X. Kong, H. Tong, I. King, Shifu2: A Network Representation Learning Based Model for Advisor-advisee Relationship Mining, IEEE Transactions on Knowledge and Data Engineering, DOI: 10.1109/TKDE.2019.2946825.
41. X. Ding, L. Wang, Z. Shao, H. Jin, Efficient Recommendation of De-Identification Policies Using MapReduce, IEEE Transactions on Big Data 5 (3) (2019) 343–354.
42. A. Q. Macedo, L. B. Marinho, R. L. T. Santos, Context-Aware Event Recommendation in Event-based Social Networks, in: Proc. of ACM RecSys, Vienna, Austria, 2015, pp. 123–130.
43. K. R. Prabhakar, V. S. Srikar, R. V. Babu, DeepFuse: A Deep Unsupervised Approach for Exposure Fusion With Extreme Exposure Image Pairs, in: Proc. of IEEE ICCV, Italy, 2017, pp. 4714–4722.
44. C. C. Aggarwal, C. Zhai, A Survey of Text Classification Algorithms, Mining Text Data (2012) 163–222.
45. H. Zhang, X. Qin, H. Zheng, Research on Contextual Recommendation System of Agricultural Science and Technology Resource Based on User Portrait, Journal of Physics: Conference Series 1693 (1).
46. Y. Kim, Convolutional Neural Networks for Sentence Classification, in: Proc. of EMNLP, Doha, Qatar, 2014, pp. 1746–1751.
47. M. Goldberg, R. Goldberg, Data classification: algorithms and applications, Computer Reviews 56 (12) (2015) 724–725.
48. T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient Estimation of Word Representations in Vector Space, in: Proc. of IJCAI, 2013.
49. J. Pennington, R. Socher, C. D. Manning, GloVe: Global Vectors for Word Representation, in: Proc. of EMNLP, 2014, pp. 1532–1543.
50. R. Salakhutdinov, A. Mnih, Probabilistic Matrix Factorization, in: Proc. of NIPS, 2008, pp. 1257–1264.
51. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, Going deeper with convolutions, in: Proc. of IEEE CVPR, 2014.

52. S. Ioffe, C. Szegedy, Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, in: Proc. of IEEE ICML, Lille France, 2015.
53. K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, in: Proc. of ICLR, 2015.
54. S. Lai, L. Xu, K. Liu, J. Zhao, Recurrent Convolutional Neural Networks for Text Classification, in: Proc. of AAAI, Sydney, NSW, Australia, 2015, pp. 2267–2273.
55. S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, S. Vijayanarasimhan, YouTube-8M: A Large-Scale Video Classification Benchmark, arXiv preprint arXiv:1609.08675.
56. K. Papineni, S. Roukos, T. Ward, W. Zhu, BLEU: a Method for Automatic Evaluation of Machine Translation, in: Proc. of ACL, Philadelphia, 2002, pp. 311–318.
57. S. Purushotham, Y. Liu, C.-C. J. Kuo, Collaborative Topic Regression with Social Matrix Factorization for Recommendation Systems, in: Proc. of IEEE ICML, Edinburgh, Scotland, UK, 2012, pp. 759–766.
58. H. Wang, N. Wang, D.-Y. Yeung, Collaborative Deep Learning for Recommender Systems, in: Proc. of ACM SIGKDD, Sydney, NSW, Australia, 2015, pp. 1235–1244.
59. D. Kim, C. Park, J. Oh, S. Lee, H. Yu, Convolutional Matrix Factorization for Document Context-Aware Recommendation, in: Proc. of ACM RecSys, Boston, MA, USA, 2016, pp. 233–240.