Hindawi
Complexity
Volume 2019, Article ID 9476981, 11 pages
https://doi.org/10.1155/2019/9476981

Research Article

A Time-Aware CNN-Based Personalized Recommender System

Dan Yang,[1] **Jing Zhang**,[1] **Sifeng Wang**,[2] **and XueDong Zhang**[1]

[1]*School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan, Liaoning 114051, China*
[2]*School of Information Science and Engineering, Qufu Normal University, Rizhao, Shandong 276826, China*

Correspondence should be addressed to Sifeng Wang; sfwang@qfnu.edu.cn

Received 12 October 2019; Revised 17 November 2019; Accepted 28 November 2019; Published 18 December 2019

Guest Editor: Yuan Yuan

Recommender system has received tremendous attention and has been studied by scholars in recent years due to its wide applications in different domains. With the in-depth study and application of deep learning algorithms, deep neural network is gradually used in recommender systems. The success of modern recommender system mainly depends on the understanding and application of the context of recommendation requests. However, when leveraging deep learning algorithms for recommendation, the impact of context information such as recommendation time and location is often neglected. In this paper, a time-aware convolutional neural network- (CNN-) based personalized recommender system *TC-PR* is proposed. *TC-PR* actively recommends items that meet users' interests by analyzing users' features, items' features, and users' ratings, as well as users' time context. Moreover, we use Tensorflow distributed open source framework to implement the proposed time-aware CNN-based recommendation algorithm which can effectively solve the problems of large data volume, large model, and slow speed of recommender system. The experimental results on the MovieLens-1m real dataset show that the proposed *TC-PR* can effectively solve the cold-start problem and greatly improve the speed of data processing and the accuracy of recommendation.

1. Introduction

The explosive growth of information has brought great trouble to people's choices. As an effective tool to deal with "information overload," recommender system [1] has always attracted the attention of researchers. And the recommender system has been widely used in variable fields and domains such as medicine recommendation [2], citation recommendation [3], service recommendation [4–7], and big data analysis [8]. The current recommendation algorithms mainly include two kinds, one is traditional recommendation techniques and the other is popular deep learning [9] recommendation techniques. Traditional recommendation techniques mainly include content-based [10], association rule-based [11], collaborative filtering (CF) [12], and hybrid [13] recommendation algorithms. Content-based recommendation can recommend the same type of products according to the users' special interests. However, it cannot find new interesting products for the user, which is not personalized. Association rule-based recommendation algorithm can find new interest points of the user. However, the first step of the algorithm, i.e., the discovery of association rules, is the most critical and time-consuming, which has been the bottleneck of the algorithm. CF is the most classical and widely used recommendation algorithm, which mainly calculates the similarity among users according to users' historical records, then finds nearest neighbors for the target user, and finally uses the preferences of neighbors to recommend for the target user. CF is the most advantageous personalized recommendation algorithm in traditional recommendation techniques. It has a high degree of automation; however, there are some problems such as sparsity [14] and cold-start [15]. Hybrid recommendation algorithm combines the advantages of multiple recommendation algorithms, which improves the performance of single algorithm. Nevertheless, it is not effective for all problems and different applications. As the hottest research field at present, deep learning-based recommendation algorithms can deal with big data, and their speeds are higher than traditional recommendation techniques. Deep learning algorithms mostly used

in the field of recommendation include CNN algorithm [16] and Recurrent Neural Network (RNN) algorithm [17]. RNN mainly deals with tasks with sequential information, i.e., the former input is related to the latter output. For example, when we understand the meaning of a sentence, it is not enough to understand each word in isolation. We need to deal with the whole sequence connected by these words. CNN is a deep feedforward artificial neural network, which extends in space by using shared weights. CNN follows the structure of ordinary neural network, i.e., multilayer perceptron. The basic structure of CNN includes input layer, convolution layer, activation layer, pooling layer, full connection layer, and output layer. Through these layers, the CNN algorithm is realized. The flow chart of CNN-based recommendation algorithm is shown in Figure 1.

As shown in Figure 1, the recommendation algorithm based on CNN recommends items for the target user which mainly includes the following steps. Firstly, we input data and preprocess the attributes in data by the input layer. Then the embedded layer can be described as the feature extraction of the preprocessed data, and it can generate each attribute feature vector. And then full-connection operation is implemented after the embedding operation to connect attribute features and generate the user feature and item feature by the full-connection layer. After that, we use the user feature and item feature to obtain the prediction rating. Finally, the Top-k items with high prediction rating and not rated by users are selected for recommendation.

CNN is a multilayer perceptron. The key to its success lies in the way that it uses local connections and shares weights. On one hand, it reduces the number of weights to make the network easy to optimize. On the other hand, it avoids the risk of overfitting, considering that people often neglect the time context of the recommendation request, such as recommendation time, recommendation scenarios, and other temporal contextual features, when using deep neural network for recommendation. To solve the above problems, this paper proposes a time-aware CNN-based personalized recommender system TC-PR. By using local perception and weights sharing of CNN, the number of parameters of neural network is greatly reduced, the time is reduced, and the accuracy of recommendation is improved.

The major contributions of this paper can be summarized as

(i) We explore a comparative study of various existing time-aware recommender systems and CNN-based recommendation algorithms.

(ii) We propose a time-aware CNN-based personalized recommender system TC-PR by incorporating time context information into the CNN model which captures not only the temporal dynamics of users' interests over time but also the users' time context to recommend. TC-PR improves the quality of the CNN-based recommendation algorithm. When calculating the target user's similar neighbors, we introduce the user's time context information into the CNN. Moreover, we propose the time-aware item rating prediction function.

(iii) We do extensive experiments on real datasets to report comparative performance analysis of our proposed TC-PR with other baselines.

The remainder of the paper is structured as follows. Section 2 discusses related works reported in the literature. Section 3 presents the overview of TC-PR framework. Section 4 introduces our proposed time-aware CNN-based personalized recommendation algorithm in detail. The experimental results are presented in Section 5, and Section 6 concludes the paper.

2. Related Work

In this section, we first introduce related works on time-aware recommender systems and then CNN-based recommendation algorithms.

Time-aware recommender systems are becoming more and more concerned in the field of personalized recommendation recently [18]. Early researches on recommender system focused on static recommender system which build time-independent recommendation model without considering the time context of users' historical behaviors. However, in real life, users' interests and preferences are closely related to time context factors, and users' tastes change over time. Therefore, the understanding and application of time context factors of recommendation requests have an important impact on the success of recommender system. Related work [19] considers that time plays an important role in point-of-interest (POI) recommendation, and most users tend to visit different places at different time in a day, e.g., visiting a restaurant at noon and visiting a bar at night. Therefore, a time-aware POI recommendation is proposed to recommend places where users have not visited before for a given user at a specified time in a day. Related work [20] considers the user-interest drifting and item popularity changing over a long period of time, and a time-aware collaborative filtering recommendation algorithm is proposed. Related work [21] proposes a probabilistic framework that utilizes temporal influence correlations of both weekdays and weekends for time-aware location recommendation, which not only recommends locations to users, but it also suggests when a user should visit a recommended location. Related work [22] leverages the product graph embedding model to do time-aware product recommendation. Related work [23] studies personalized Top-N sequential recommendation problem using a convolutional sequence embedding recommendation model.

CNN-based recommendation algorithms have some advantages that traditional recommendation techniques do not have, such as good fault tolerance, parallel processing, and self-learning ability. They can deal with the problems of complex environmental information, unclear background knowledge, and unclear reasoning rules. They allow the samples to have larger defects and distortions. They run fast and have good adaptive performance and high resolution. They integrate feature extraction function into multilayer perceptron by restructuring structure and reducing weight and omit the complex process of image feature extraction

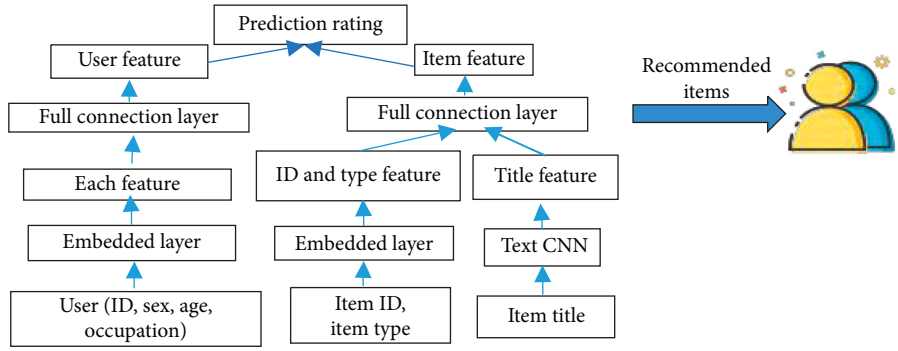

Figure 1: The flow chart of CNN-based recommendation algorithm.

before recognition. The generalization ability of CNN is better than other methods. Related work [24] proposes an automated CNN recommendation system for image classification task, which is able to evaluate the complexity of the classification task and the classification ability of the CNN model precisely. Related work [25] proposes a CNN-based approach for expert recommendation, which will reduce the questioner's waiting time and improve the quality of the answer. Related work [26] proposes a latent group recommendation (LGR) based on the dynamic probabilistic matrix factorization model integrated with convolutional neural network (DPMFM-CNN), which takes into comprehensively considering the relationships among users, groups, and services and improves the recommendation accuracy.

Most existing related studies often neglect the influence of time context factors on recommender system when using CNN for recommendation or incorporate time context factors into the model as common features. However, time context factors have a crucial impact on the success of recommender system in real-life applications, and integrating time context into CNN model can effectively improve the accuracy of recommender system and users' satisfaction.

3. Time-Aware Personalized Recommender System TC-PR

In this section, we first give some preliminaries related to our time-aware personalized recommendation algorithm and then present the framework of our proposed time-aware CNN-based personalized recommender system TC-PR.

3.1. Preliminary

3.1.1. User-Item Rating Matrix with Timestamp R^t. With m users $U = \{u_1, \ldots, u_m\}$ and n items $I = \{i_1, \ldots, i_n\}$, the user rating behaviors to item, i.e., $M \times N$ user-item matrix with timestamp R^t, is defined as follows:

$$R^t_{mn} = \begin{cases} \text{weekday}, & \text{if } (u_m, i_n) \text{interaction is observed}, \\ \text{weekend}, & \text{otherwise}, \end{cases} \tag{1}$$

where value weekday in R^t represents interactions at timestamp t (from Monday to Friday) between users and items, e.g., a user watched a movie and rated it on Tuesday. Similarly value weekend means interactions at timestamp t (Saturday and Sunday).

Each value in the matrix R^t can be seen as a 4-tuple of the form $<u, i, r, t>$, where u is the user, i is the item, r represents user u's rating on i, and t represents the timestamp of the opinion. For example, $<\text{user}_1, \text{item}_1, 3, \text{Tuesday}>$ represents the rating rated by user_1 for item_1 on Tuesday is 3. The recommendation task can be defined as a prediction problem which aims to infer the value of the interaction label of user-item pair $<u, i>$.

3.2. TC-PR Overview. The proposed framework of time-aware CNN-based personalized recommender system *TC-PR* is shown in Figure 2. It considers the time context information of users' behaviors in CNN model to improve prediction accuracy of personalized recommendation. By analyzing temporal features of users' behaviors, item features, and users' ratings on items in a specific time context, *TC-PR* can recommend more accurate results for the target user and at the same time improve the processing speed and recommendation accuracy.

4. Time-Aware CNN-Based Personalized Recommendation

CNN-based recommendation algorithm can recommend items that meet users' interests by analyzing users' features, items' features, and users' ratings information. When using CNN model for recommendation, it often ignores the users' time context factors or incorporates time context factors into the model as a common feature. However, the success of the recommender system often depends on its understanding and application of the context of the recommendation requests. Therefore, time context factors have a significant impact on the efficiency of recommendation. *TC-PR* effectively alleviates the cold-start problem and improves the speed of data processing and the accuracy of recommendation by introducing discrete time parameters to capture the real-world temporal dynamic information for recommendation. In this section, we first give a brief description of model design of *TC-PR* in Section 4.1. Then in Section 4.2, we give the detail algorithm description of the time-aware CNN-based personalized recommendation algorithm.

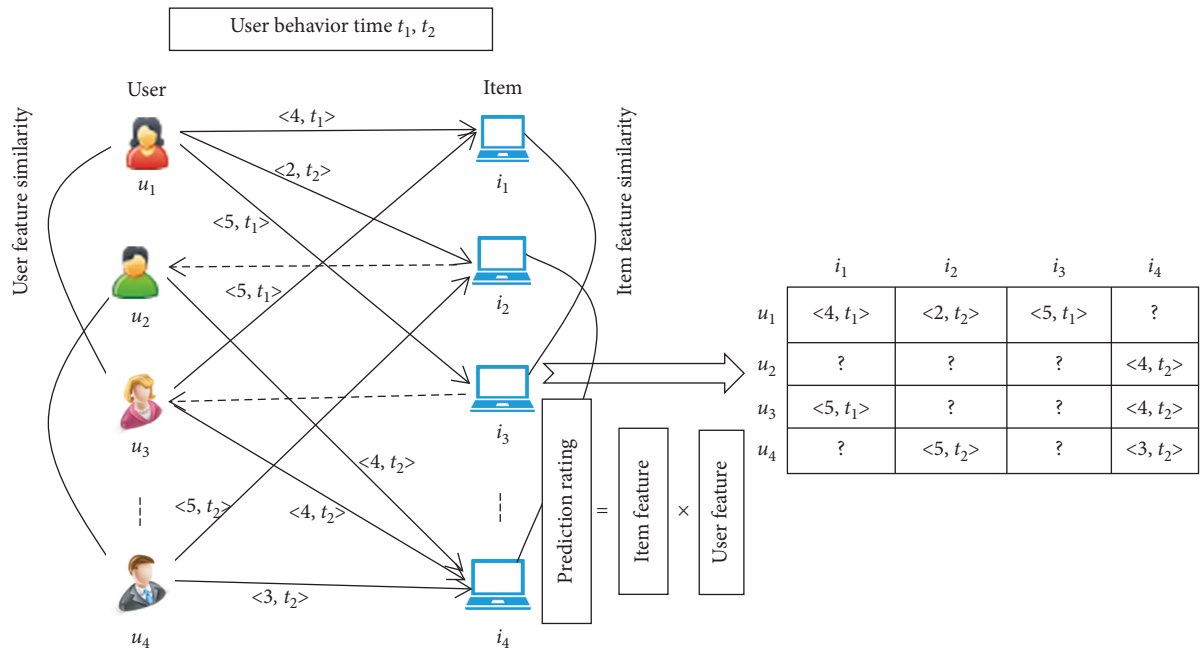

Figure 2: The framework of time-aware CNN-based personalized recommender system TC-PR.

4.1. Model Design. In this section, we first give a brief description of the time-aware CNN-based recommendation model used in *TC-PR* in Section 4.1.1 and then explain the time-aware item rating prediction and recommendation in Section 4.1.2.

4.1.1. Time-Aware CNN-Based Recommendation Model. The key idea of the *TC-PR* is to calculate the prediction ratings of the target user and recommend interesting items for the target user by analyzing the temporal dynamic features and the item features under the user's time context. That is to say, similar users with the same time context will have common preferences. Therefore, it is very important for the time-aware CNN-based recommendation performance to choose similar users with the same time context for the target user accurately.

Firstly, *TC-PR* captures the temporal information, i.e., time context associated with the users' behaviors. Then feed the user features, item features, and temporal information to the input layer of CNN as input data and obtain the original matrix. And extract features of the matrix by the convolution layer; the formula for calculating the output results is shown in formula (2). After that, the output results are obtained, because the calculation between adjacent layers of the neural network can be simulated by linearity, but only linear operation, multiple convolution layers are equivalent to the operation of one convolution layer, so it is necessary to use the activation function in the activation layer to carry out nonlinear operation to enable the neural network to simulate more complex models. There are mainly two activation functions used in traditional neural networks, i.e., $\sigma(x)$ and $\tanh(x)$. The two activation functions are exponential operations, which are inefficient. In the process of using these two activation functions, there are problems of small interval range and disappearance of gradient. In order to solve these two problems, Relu functions are mainly used, as shown in formula (3), which is a linear operation with high efficiency. And the pooling layer is used for down-sampling, sparse processing of feature data to reduce the amount of data computation. Typical pooling methods include average pooling and max pooling. The formula of pooling is shown in formula (4). In order to reduce the loss of feature information, we adopt the full connection method to refit in the tail of CNN. Finally, the results are output by the output layer.

$$N_2 = \frac{(N_1 + 2P - F)}{\text{stride} + 1}, \tag{2}$$

where N_2 is the size of output, N_1 is the size of the input data, F is the size of the convolution kernel, stride is the sliding step of the convolution kernel, and P is to fill in the input data in order to be divisible when the stride is greater than 1.

$$f(x) = \max(0, x), \tag{3}$$

where the actual number takes 0 when the gradient is less than 0, and the actual number is taken when the gradient is greater than 0, which avoids the problem of the disappearance of gradient.

$$N_2 = \frac{(N_1 - F)}{\text{stride} + 1}, \tag{4}$$

4.1.2. Time-Aware Item Rating Prediction. Firstly, the attributes in each information table are processed, and the fields whose attributes are categories are converted into numbers, and these numbers are used as the index of embedded matrix. Then the embedded layer is used in the first layer of the network. After that, utilizing the output features

of the embedded layer, the features are transferred to the full-connection layer, and the output of this layer is input to the full-connection layer again. Finally, the user features and item features are obtained. The prediction ratings of the items are obtained by training the user feature vector and item features vector, and the Top-k items with higher rating and not rated by the target user are selected according to the ranking of the ratings, that is, the recommendation list *RL* is recommended to users.

4.2. Algorithm Description. The flow chart of time-aware CNN-based personalized recommendation algorithm is shown in Figure 3. First, preprocess the original dataset. Then input the processed data with time information, train the network by constructing a neural network (NN) and a calculation graph, and obtain the number of samples to be trained. After that, the training parameters are saved into a file. And evaluate the model trained by the neural network by using Mean Square Error (MSE) value, the MSE value can be minimized by constantly adjusting the parameters, and the performance of the model is the best at this time. Finally, the user features and item features are obtained by the model. We adopt the model to calculate the prediction ratings, and select the Top-k items with the higher rating and not rated by target user to recommend for the target user.

The pseudocode of a time-aware CNN-based personalized recommendation algorithm is shown in Algorithm 1.

5. Experiments

In this section, we first introduce the experimental dataset in Section 5.1, then introduce time information retrieval and data preprocessing in Section 5.2, and give a brief description of the experimental setup in Section 5.3. The evaluation methods and the effects of parameters in *TC-PR* are shown in Sections 5.4 and 5.5, respectively. Finally, the experimental results and the performance comparison of different methods are shown in Section 5.6.

5.1. Experimental Dataset. In this paper, we use the MovieLens-1m (https://grouplens.org/datasets/movielens/lm/) experimental dataset collected by the GroupLens Research Project at the University of Minnesota as experimental data, which includes 1000000 ratings from 6040 users on 3952 different movies. The experimental dataset mainly contains the following information, user statistics are shown in Figure 4(a), which includes several attributes, i.e., UserID, Gender, Age, Occupation, and Zip-code, where Gender is represented by M, F, Age is divided into several age groups, and Zip-code is of no use. The rating statistics are shown in Figure 4(b), which includes four attributes, i.e., UserID, MovieID, Rating, and Timestamp. And the movie statistics are shown in Figure 4(c), which includes three attributes, i.e., MovieID, Title, and Genre.

As shown in Figure 4(b), the users' ratings range from 1 to 5. The higher the ratings, the more the users' preferences for the movie. As shown in Figure 4(c), a movie may have more than one genre, e.g., crime and adventure, at the same time. Table 1 lists the summarized information about experimental dataset.

5.2. Time Information Retrieval and Data Preprocessing. In order to process the data more quickly and smoothly in the later stage, we first preprocess the data to make the data format more standard and more suitable for operation. We deal with user information table, movie information table, and user-item rating information table, respectively.

The data preprocessing algorithm is presented in Algorithm 2 which mainly includes three procedures. For the attributes in user information table *users.dat*, we need to convert the gender values "F" and "M" into numbers 0 and 1 and represent several segments of age groups with continuous numbers 0~6. For the attributes in the movie information table *movies.dat*, the movie title and the movie genre are both category fields, and they also need to be converted to numbers. For the movie genre attribute, firstly the values of movie genre attribute are converted into strings and stored in a digital dictionary, and then the corresponding movie genre values of each movie is converted into a digital list because most movies have multiple genres. For the movie title attribute, the key idea is the same as the attribute of movie genre. It is just to create a text to a digital dictionary, then convert the description of movie title into a digital list, and remove the year from the movie title. For attributes in user-item rating information table, the timestamp attribute needs to be converted into specific time information.

5.3. Experimental Setup. *TC-PR* is implemented with the Tensorflow [27] framework which built under Windows with the pip 10.0.1 and python 3.5. Tensorflow can implement not only parallel computing on many CPUs or GPUs on a single machine, but also distributed computing [28, 29] which greatly improves the running speed of the algorithm. The experiments are conducted on a 1.8 GHz four-core processor Windows machine with 4 GB memory and 700 GB hard disk.

5.4. Evaluation Metrics. We evaluate the *TC-PR* with Mean Square Error MSE (formula (5)) and Root Mean Square Error RMSE (formula (6)) which are popular metrics in CNN as follows:

$$\mathrm{MSE} = \frac{\sum_{u,i \in T} \left(r_{ui} - \widehat{r}_{ui}\right)^2}{|T|}, \tag{5}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{T} \sum_{u,i \in T} \left(r_{ui} - \widehat{r}_{ui}\right)^2}, \tag{6}$$

where u denotes the user, i denotes the item, r_{ui} denotes user u's true rating on item i, $\widehat{r}_{ui}$ denotes user u's prediction rating on item i, and the $|T|$ represents the total number of the items. A smaller MAE value or a smaller RMSE value means a better performance of recommendation algorithm.

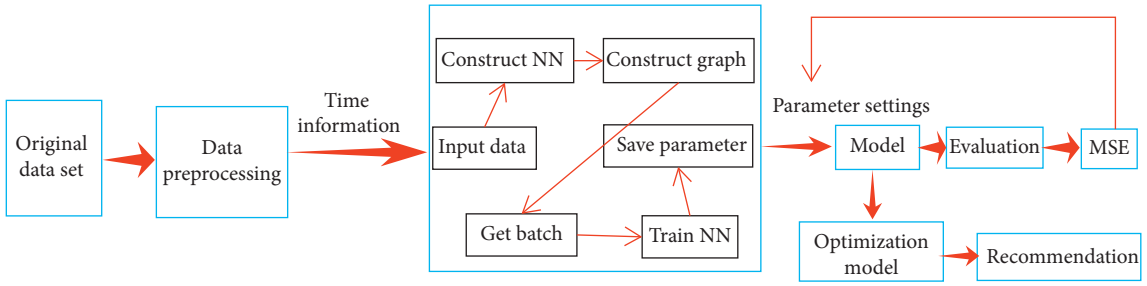

Figure 3: Flow chart of time-aware CNN-based personalized recommendation algorithm.

Input: users.dat, items.dat, ratings.dat with timestamp R^t, target user u, user u's time context t
Output: recommended item list for u *RL*
Step 1: process data and save processed data to preprocess.p;
Step 2: open preprocess.p and set parameters;
Step 3: construct NN and generate users' features and items' features;
Step 4: construct graph to calculate prediction rating by user similarity calculation, update parameter settings according to MSE;
Step 5: randomly split dataset into training set and test set, and then train NN;
Step 6: save trained model and parameters;
Step 7: load saved model to recommend for target user u according to t;
Step 8: generate recommendation list *RL* from time-aware CNN-based personalized recommendation algorithm;
Step 9: *return RL*.

Algorithm 1: Time-aware CNN-based personalized recommendation algorithm.

```
1::F::1::10::48067
2::M::56::16::70072
3::M::25::15::55117
4::M::45::7::02460
5::M::25::20::55455
```

(a)

```
1::1193::5::978300760
1::661::3::978302109
1::3408::4::978300275
2::1213::2::978298458
2::21::1::978299839
```

(b)

```
1::Toy Story (1995)::Animation|Children's|Comedy
2::Jumanji (1995)::Adventure|Children's|Fantasy
3::Grumpier Old Men (1995)::Comedy|Romance
4::Waiting to Exhale (1995)::Comedy|Drama
5::Father of the Bride Part II (1995)::Comedy
6::Heat (1995)::Action|Crime|Thriller
7::Sabrina (1995)::Comedy|Romance
8::Tom and Huck (1995)::Adventure|Children's
```

(c)

Figure 4: The samples of experimental dataset.

5.5. The Effects of the Parameters. In this section, we will show the effects of key parameters, i.e., filter_number (the size of convolution kernel), stride (the size of the sliding window), learning_rate (learning rate), and batch_size (batch size) in *TC-PR* based on Tensorflow framework. The values of the Training loss and the Test loss are shown in Figure 5.

5.5.1. Parameter Settings for TC-PR. As shown in Figure 5, we can observe that the Training loss and the Test loss change with parameter settings. The minimize MSE is achieved according to adjusting parameters in *TC-PR*. Table 2 gives the specific parameter settings and MSE value.

5.6. Experimental Results and Analysis

5.6.1. The Performance of TC-PR. In this subsection, the movie prediction ratings are calculated and the movies are recommended for the target user by the proposed *TC-PR*. Table 3 gives example of the recommended movie list of our proposed *TC-PR*, and Table 4 gives the example of the

Table 1: The statistics of experimental dataset.

Dataset	#Users	#Movies	#Ratings	Discrete time (day)	Sparsity (%)
MovieLens-1m	6040	3952	1,000,209	Weekend, weekday	4.1

```
(1) procedure users.dat
(2)    read users.dat.
(3)    get gender and age attributes in users.dat.
(4)    update gender = {"F": 0, "M": 1}.
(5)    foreach age in enumerate(set(users["Age"])
(6)       users["Age"] = users["Age"].map(age)
(7)    end foreach
(8) end procedure
(9) procedure movies.dat
(10)   read movies.dat.
(11)   get movie genre attribute in movies.dat.
(12)   genres_set = set()
(13)   foreach genre in movies["Genres"].str.split("|")
(14)      genres_set.update(genre)
(15)   end foreach
(16)   foreach genreint in enumerate(genres_set)
(17)      genres_map = {genreint in enumerate(set(movies["Genres"]))}
(18)   end foreach
(19) end procedure
(20) procedure ratings.dat
(21)   read ratings.dat.
(22)   get timestamp attribute in ratings.dat.
(23)   foreach timestamp in ratings
(24)      timestamp = datetime.fromtimestamp(int(timestamp)).weekday()
(25)   end foreach
(26) end procedure
```

Algorithm 2: The data preprocessing algorithm.

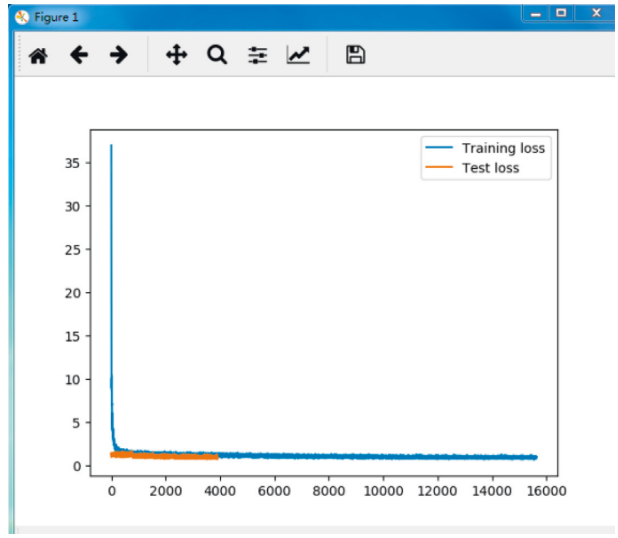

Figure 5: Training loss and test loss with different parameter settings.

recommended movie list of general CNN algorithm. In our experiments, we use two categories of time intervals, i.e., weekday and weekend. And we analyze examples of recommended movie list in Tables 3 and 4 and draw pie charts (Figure 6) based on the data in the tables.

In the experiments, we use *TC-PR* implemented with Tensorflow framework, and general CNN-based recommendation algorithm to recommend the Top-k movies with the higher prediction ratings and not rated by the target user for the target user. The recommendation results for the

Table 2: The information of parameter settings.

Filter_number	Stride	Batch_size	Learning_rate	MSE
8	2, 3, 4, 5	256	0.0001	0.740

Table 3: The example of the recommended movie list of TC-PR.

User ID	The sort of recommended list	Recommended movie title	Time interval	Recommended movie genres
234	1	Tigrero: A Film That Was Never Made (1994)	Weekday	Documentary|Drama
	2	The Maltese Falcon (1941)		Film-Noir|Mystery
	3	The Gate of Heavenly Peace (1995)		Documentary
	4	Schindler's List (1993)		Drama|War
	5	The Wrong Trousers (1993)		Animation|Comedy
	6	Talk of Angels (1998)		Drama
	7	Sunset blvd. (a.k.a. Sunset boulevard) (1950)		Film-Noir
	8	Make Mine Music (1946)		Animation|Children's|Musical
	9	The Shawshank Redemption (1994)		Drama
	10	Raiders of the Lost Ark (1981)		Action|Adventure
234	1	Star Wars: Episode IV—A New Hope (1977)	Weekend	Action|Adventure|Fantasy|Sci-Fi
	2	Casablanca (1942)		Drama|Romance|War
	3	Apocalypse Now (1979)		Drama|War
	4	Saving Private Ryan (1998)		Action|Drama|War
	5	Schindler's List (1993)		Drama|War
	6	Hangmen Also Die (1943)		Drama|War
	7	Lifeboat (1944)		Drama|Thriller|War
	8	Stand By Me (1986)		Adventure|Comedy|Drama
	9	Aliens (1986)		Action|Sci-Fi|Thriller|War
	10	The Manchurian Candidate (1962)		Film-Noir|Thriller

Table 4: The example of the recommended movie list of general CNN-based recommendation algorithm.

User id	The sort of recommended list	Recommended movie title	Time interval	Recommended movie genres
234	1	City Lights (1931)	No division	Comedy|Drama|Romance
	2	Rear Window (1954)		Mystery|Thriller
	3	Schindler's List (1993)		Drama|War
	4	The Godfather (1972)		Action|Crime|Drama
	5	The Usual Suspects (1995)		Crime|Thriller
	6	Notorious (1946)		Film-Noir|Romance|Thriller
	7	The Great Escape (1963)		Adventure|War
	8	Raiders of the Lost Ark (1981)		Action|Adventure
	9	Seven Samurai (The Magnificent Seven) (Shichinin No Samurai) (1954)		Action|Drama
	10	Zachariah (1971)		Western

target user whose ID is 234 are described in Tables 3 and 4, respectively. As shown in Table 3, the time context is divided into two dispersed time intervals, i.e., weekday and weekend. The Top-10 movies are recommended for the target user according to different time intervals, respectively. However, Table 4 recommends Top-10 movies for the target user without dividing time interval. We can clearly note that dividing the time interval has obvious influence on the genres of recommended movies for the target user by observing the table.

The recommended movie lists in Tables 3 and 4 are represented by pie charts, as shown in Figure 6.

From Figure 6, we can observe that *TC-PR* and the general CNN-based recommendation algorithm recommend movies for the target user whose user ID is 234 have obvious differences in the genres of recommended movies. The experimental results of Figures 6(a) and 6(b) are analyzed in detail as follows:

(i) The genres of movies recommended by *TC-PR*: the genres of movies recommended are shown in Figure 6(a). In time-aware CNN-based algorithm, we divide time into two kinds of time intervals, i.e., weekday and weekend. The left half of Figure 6(a) shows the genres of movies recommended for the target user during the working day, while the right half of Figure 6(a) shows the genres of movies recommended for the target user at the weekend. In the left half of Figure 6(a), we can see that 20% of the 10 movies

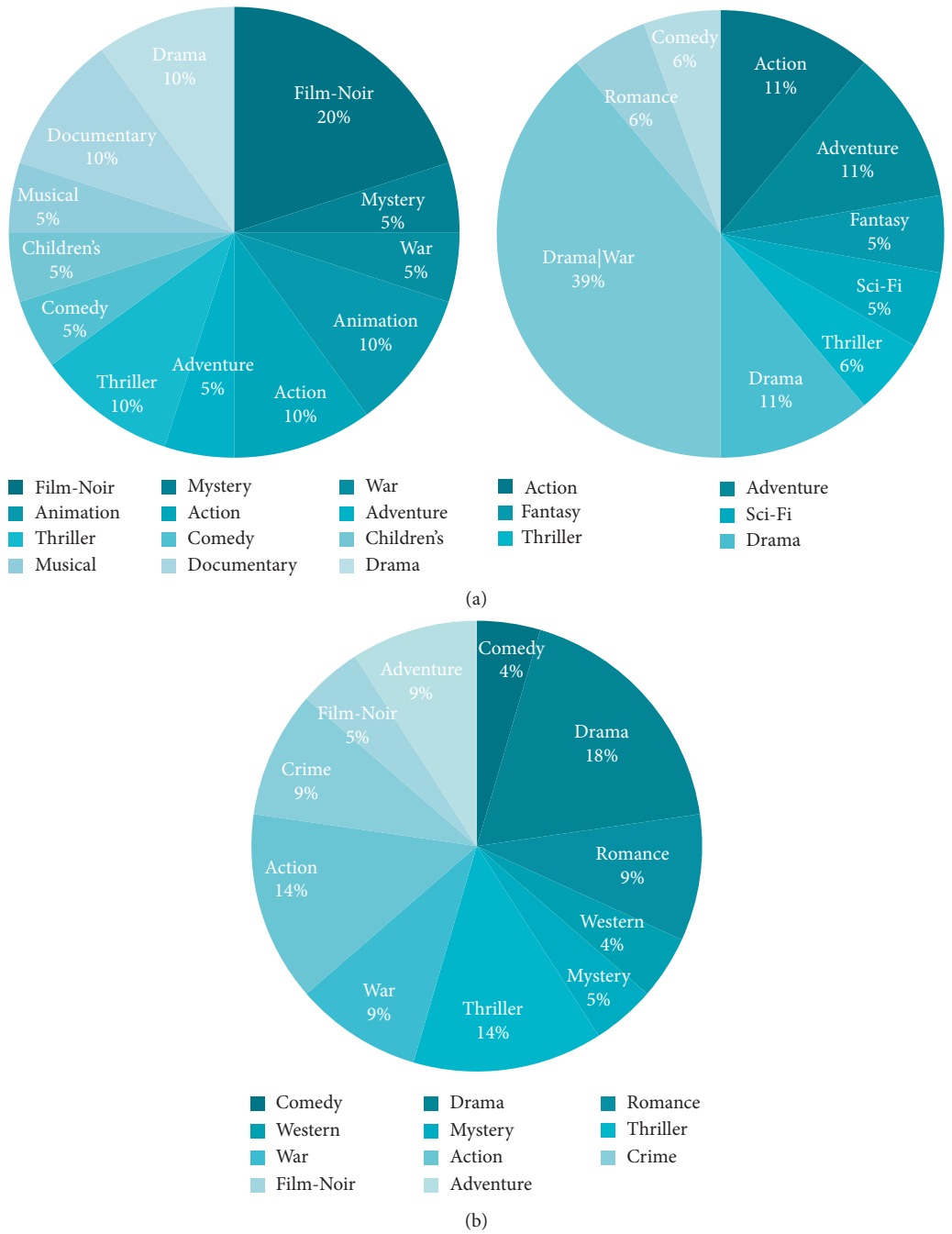

Figure 6: Recommended movie list with different algorithms. (a) The movie genres recommended by TC-PR. (b) The movie genres recommended by general CNN-based recommendation algorithm.

recommended are with horror genre, it is the largest proportion, and the movies with horror, adventure, and mystery are obviously more than movies with comedy and drama. In the right half of Figure 6(a), we can see that the proportion of the 10 recommended movies with drama and war genres is the largest, which is 39%. And we can find that movies with comedy and drama genres are obviously more than movies with horror and adventure genres by observing. In a word, we can see from Figure 6(a) that *TC-PR* can recommend more accurate items (movies) for target user.

(ii) The movie genres recommended by general CNN recommendation algorithm: the genres of movies recommended are shown in Figure 6(b). The general CNN recommendation algorithm recommends movies directly for the target user without considering the time context information of the target user. Figure 6(b) shows the genres of movies recommended for the target user. In Figure 6(b), we can see that among the ten movies recommended, the genres of movies are relatively wide and the proportion is not significantly different. It is

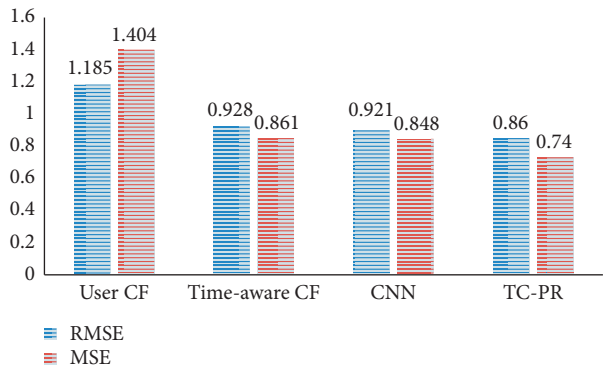

Figure 7: The performance comparison of four recommendation algorithms.

difficult to recommend movies that meet the users' preferences.

TC-PR can more accurately recommend movies that meet users' preferences. In the case of sparse user rating matrix, it can avoid the disadvantage of low recommendation accuracy of the traditional CF algorithm. In the case of huge amount of data, it can be processed in parallel and greatly improve the speed of operation.

5.6.2. Performance Comparison. We compare the TC-PR proposed in this paper with the following baselines on the same dataset:

(i) Baseline1: user-based CF

(ii) Baseline2: time-aware CF

(iii) Baseline3: CNN-based recommendation algorithm

The detailed comparison results of TC-PR and three baselines are shown in Figure 7.

It can be observed that among all the compared methods, TC-PR achieves best performance, which has the highest accuracy and the best recommendation effect.

5.6.3. Further Discussion. We discuss challenges and opportunities for our proposed time-aware CNN-based personalized recommender system. Currently, *TC-PR* is only realized with Tensorflow framework but without distributed realization. With the rapid development and research of edge computing [30], a real-time and distributed time-aware CNN-based personalized recommender system in edge computing environment can greatly improve the performance and make it more applicable.

6. Conclusion and Future Work

This paper proposes a time-aware CNN-based personalized recommender system TC-PR based on Tensorflow framework. TC-PR enables us to capture the temporal information of users' historical behaviors and improves recommendation accuracy. Compared with the traditional recommendation algorithm, when using TC-PR for recommendation, the larger the dataset, the higher the stability of training, and the higher the accuracy of recommendation. The TC-PR overcomes the limitations of traditional approaches and improves system performance. The experimental results on MovieLens-1m real dataset show that the proposed TC-PR can effectively alleviate the cold-start problem and greatly improve the speed of recommendation processing and accuracy. As for the future work, we will continue to explore more accurate time division models and consider other deep learning models to realize the time-aware personalized recommender system.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by the Natural Science Foundation of Liaoning Province, China (grant no. 20170540471), General Scientific Research Projects of Liaoning Province, China (grant no. 2019LNJC07), and University of Science and Technology Liaoning Talent Project (grant no. 601011507-22).

References

[1] J. Dong, X. Li, and B. Fang, "A recommendation system based on multi-attribute," in *Proceedings of the 2016 9th International Conference on Service Science*, October 2016.

[2] M. Wang, M. Liu, J. Liu, S. Wang et al., "Safe medicine recommendation via medical knowledge graph embedding," 2017, https://arxiv.org/abs/1710.05980.

[3] H. Liu, H. Kou, C. Yan, and L. Qi, "Link prediction in paper citation network to construct paper correlation graph," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, no. 1, p. 233, 2019.

[4] L. Qi, X. Zhang, W. Dou, and Q. Ni, "A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2616–2624, 2017.

[5] L. Qi, Z. Zhou, J. Yu, and Q. Liu, "Data-sparsity tolerant web service recommendation approach based on improved collaborative filtering," *IEICE Transactions on Information and Systems*, vol. E100.D, no. 9, pp. 2092–2099, 2017.

[6] W. Gong, L. Qi, and Y. Xu, "Privacy-aware multidimensional mobile service quality prediction and recommendation in distributed fog environment," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 3075849, 8 pages, 2018.

[7] L. Qi, X. Zhang, W. Dou, C. Hu, C. Yang, and J. Chen, "A two-stage locality-sensitive hashing based approach for privacy-preserving mobile service recommendation in Cross-platform edge environment," *Future Generation Computer Systems*, vol. 88, pp. 636–638, 2018.

[8] X. Zhang, W. Dou, Q. He et al., "LSHiForest: A generic framework for fast tree isolation based ensemble anomaly analysis," in *Proceedings of the 33rd IEEE International Conference on Data Engineering, ICDE 2017*, vol. 4, San Diego, CA, USA, April 2017.

[9] L. W. Huang, B. T. Jiang, S. Y. Lv et al., "Summary of recommendation system research based on deep learning," *Chinese Journal of Computer*, vol. 41, no. 427, pp. 191–219, 2018.

[10] I. Cantador and D. Vallet, "Content-based recommendation in social tagging systems," in *Proceedings of the 2010 ACM Conference on Recommender Systems*, Barcelona, Spain, September 2010.

[11] C. Li, W. Liang, Z. Wu et al., "An efficient distributed-computing framework for association-rule-based recommendation," in *Proceedings of the 2018 IEEE International Conference on Web Services (ICWS)*, pp. 339–342, IEEE, San Francisco, CA, USA, July 2018.

[12] Y. Pang, Y. Jin, Y. Zhang, and T. Zhu, "Collaborative filtering recommendation for MOOC application," *Computer Applications in Engineering Education*, vol. 25, no. 1, pp. 120–128, 2017.

[13] P. Y. Lu, X. X. Wu, and D. N. Teng, "Hybrid recommendation algorithm for E-commerce website," in *Proceedings of the 2015 International Symposium on Computational Intelligence & Design*, December 2015.

[14] A. Y. Xue, J. Qi, X. Xie, R. Zhang, J. Huang, and Y. Li, "Solving the data sparsity problem in destination prediction," *The VLDB Journal*, vol. 24, no. 2, pp. 219–243, 2015.

[15] Y. H. Guo and G. S. Deng, "Hybrid recommendation algorithm of item cold-start in collaborative filtering system," *Computer Engineering*, vol. 34, no. 23, pp. 11–13, 2008.

[16] S. S. M. Salehi, D. Erdogmus, and A. Gholipour, "Auto-context convolutional neural network (Auto-Net) for brain extraction in magnetic resonance imaging," *IEEE Transactions on Medical Imaging*, vol. 36, no. 11, pp. 2319–2330, 2017.

[17] S. Wu, W. Ren, C. Yu et al., "Personal recommendation using deep recurrent neural networks in NetEase," in *Proceedings of the 2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, May 2016.

[18] W. J. Li, *Research for Time-Aware Recommendation Algorithm*, University of Electronic Science and Technology of China, Chengdu, China, 2017.

[19] Q. Yuan, G. Cong, Z. Ma et al., "Time-aware point-of-interest recommendation," in *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, ACM, Dublin, Ireland, July 2013.

[20] S. Wei, N. Ye, and Q. Zhang, "Time-aware collaborative filtering for recommender systems," in *Proceedings of the Chinese Conference on Pattern Recognition*, September 2012.

[21] J. D. Zhang and C. Y. Chow, "TICRec: a probabilistic framework to utilize temporal influence correlations for time-aware location recommendations," *IEEE Transactions on Services Computing*, vol. 9, no. 4, pp. 633–646, 2017.

[22] Y. Li, W. Chen, and H. Yan, "Learning graph-based embedding for time-aware product recommendation," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 2163–2166, ACM, Singapore, November 2017.

[23] J. Tang and K. Wang, "Personalized top-N sequential recommendation via convolutional sequence embedding," in *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*, pp. 565–573, ACM, Marina del Rey, CA, USA, February 2018.

[24] S. Wang, L. Sun, W. Fan et al., "An automated CNN recommendation system for image classification tasks," in *Proceedings of the 2017 IEEE International Conference on Multimedia and Expo (ICME)*, July 2017.

[25] J. Wang, J. Sun, H. Lin, H. Dong, and S. Zhang, "Convolutional neural networks for expert recommendation in community question answering," *Science China Information Sciences*, vol. 60, no. 11, pp. 19–27, 2017.

[26] H. Y. Wang and M. W. Dong, "Latent group recommendation based on dynamic probabilistic matrix factorization model integrated with CNN," *Journal of Computer Research and Development*, vol. 54, no. 8, pp. 1853–1863, 2017.

[27] M. Abadi, A. Agarwal, P. Barham et al., "Tensorflow: large-scale machine learning on heterogeneous distributed systems," 2016, https://arxiv.org/abs/1603.04467.

[28] X. Xu, X. Zhang, H. Gao, X. Yuan, L. Qi, and W. Dou, "BeCome: blockchain-enabled Computation offloading for IoT in mobile edge computing," *IEEE Transactions on Industrial Informatics*, 2019.

[29] X. Xu, Q. Liu, X. Zhang, J. Zhang, L. Qi, and W. Dou, "A blockchain-powered Crowdsourcing method with privacy preservation in mobile environment," *IEEE Transactions on Computational Social Systems*, vol. 6, no. 6, pp. 1407–1419, 2019.

[30] X. Xu, C. He, Z. Xu, L. Qi, S. Wan, and M. Z. A. Bhuiyan, "Joint optimization of offloading utility and privacy for edge computing enabled IoT," *IEEE Internet of Things Journal*, 2019.