IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 18, 2020, accepted October 27, 2020, date of publication November 2, 2020, date of current version April 1, 2021.

Digital Object Identifier 10.1109/ACCESS.2020.3035162

Score Prediction Algorithm Combining Deep Learning and Matrix Factorization in Sensor Cloud Systems

JIBING GONG[1,2,4], WEIXIA DU[1,2], HUANHUAN LI[1,2], QING LI[1,2], YI ZHAO[1,2], KAILUN YANG[1,2], AND YING WANG[3]

[1]School of Information Science and Engineering, Yanshan University, Qinhuangdao 066004, China
[2]Key Laboratory for Computer Virtual Technology and System Integration of Hebei Province, Yanshan University, Qinhuangdao 066004, China
[3]STTC, Ministry of Science Technology, Beijing 10045, China
[4]Key Laboratory for Software Engineering of Hebei Province, Yanshan University, Qinhuangdao 066004, China

Corresponding author: Weixia Du (duweixia@stumail.ysu.edu.cn)

ABSTRACT In this era of exponential growth in the scale of data, information overload has become an urgent problem, and the use of increasingly flexible sensor cloud systems (SCS) for data collection has become a mainstream trend. Recommendation algorithms can search massive data sets to uncover information that meets the needs of users based on their interests. To improve the accuracy of recommendation scoring, this article proposes a score prediction algorithm that combines deep learning and matrix factorization. To address the problem of sparse scoring data, our study employs a sensor cloud system to collect data information, preprocesses the collected information, and then uses a deep learning model combined with explicit and implicit feedback to generate recommendations. The proposed algorithm, MF-NeuRec, combines fusion matrix decomposition and the NeuRec model score prediction algorithm. The algorithm employs user-based and item-based NeuRec algorithms to extract the feature vectors of users and items under implicit feedback data. The obtained user and item feature vectors are integrated in a certain ratio through the use of matrix decomposition under the display feedback data. The user and item feature vectors obtained by the algorithm are merged and analyzed to predict how users will rate items. Experiments demonstrate that the algorithm can improve the accuracy of recommendations.

INDEX TERMS Deep learning, collaborative filtering, score prediction, recommendation system, sensor cloud system.

I. INTRODUCTION

The vigorous development of sensor cloud systems (SCS) in recent years has earned them significant attention from academia and industry. SCS is often used for monitoring and controlling applications, including the emerging Internet of Things (IoT), machine-to-machine, and cyber-physical systems. Applications, which ease the task of data collection, have also exponentially increased the scale of data, and information overload has become an increasingly serious problem [1]–[3]. In order to solve the problem of information overload and facilitate effective data processing, a sensor cloud system can be used to collect data, and machine learning methods can subsequently be used to process the data.

The associate editor coordinating the review of this manuscript and approving it for publication was Md. Zakirul Alam Bhuiyan.

To make full use of such data, researchers have proposed many solutions; among them, the representative methods are classified catalogs and search engines. However, with the continuous expansion of the Internet, classified catalog websites can only cover a small number of popular websites and often do not meet the needs of users. Search engines search for relevant information according to keywords entered by users, which alleviates the problem of information overload to a certain extent; however, search engines are ineffective when users cannot find keywords that accurately describe their needs. Mining massive amounts of data to uncover information that meets the specific preferences and needs of users has become an urgent task [4]. Recommendation systems have been created to address this problem; their algorithms can actively explore user interests and help users find information that meets their needs. Scoring prediction is one of

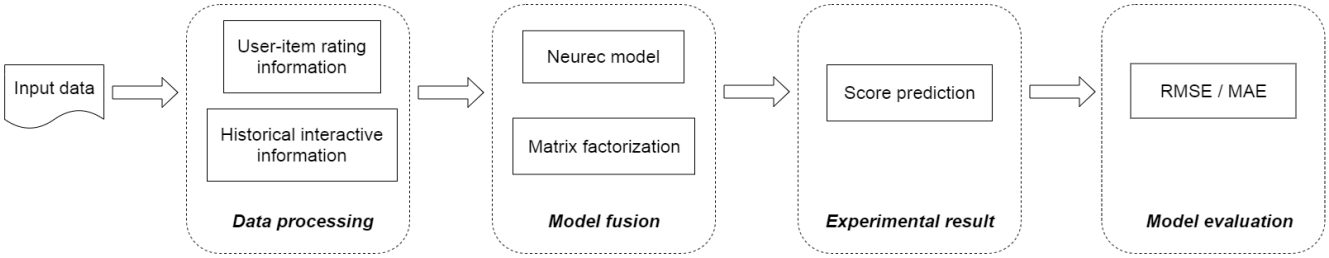


FIGURE 1. MF-NeuRe model technology roadmap.

the most commonly used methodologies in recommendation algorithms.

Recommendation algorithms are widely used in numerous applications and are a familiar fixture of online life. *Amazon*, *taobao.com*, *JD.com*, and other e-commerce companies analyze user preferences and needs based on historical browsing, collection, purchase, evaluation, and other information, in order to personalize recommendations for the user; such recommendations help these companies increase revenue [5]. Music playback services such as *QQ music*, *KuGou*, *Netease cloud music* recommend new songs for users based on their historical listening information, which includes songs, artists, and other information [6], [7]. Based on user browsing history and location information, news applications such as *Google News* and *Toutiao* recommend news sources to enhance user retention and user traffic [8], [9] [10]. Video sites such as *YouTube*, *Tencent Video*, and *iQiyi*, and short video apps such as *Tik Tok* and *Watermelon*, recommend videos that may interest a user by analyzing their browsing, likes, and comments [11], [12] [13]. Recommendation algorithms have also played an important role in online services such as *Tencent*, *Meituan*, *Douban*, and *Zhihu*. Recommendation algorithms are routinely encountered by users from all walks of life [14].

Traditional recommendation algorithms include content-based recommendations, collaborative filtering-based recommendations, and hybrid recommendations [15], [16]. Although content-based recommendations do not have the cold start problem (i.e., few interactions), certain difficulties may be encountered when obtaining the characteristics of an item [17]. Different acquisition methods must be designed for different data sets. Only the user's historical interactive item information is analyzed, and information that may be of interest to the user is recommended. Because there is no new historical interactive item information, it is impossible to generate recommendations for new users [18]. The most widely used type of recommendation algorithm is based on collaborative filtering. Such algorithms can automatically learn the hidden features of users and items, which can help identify potential interests of users [19], [20]. However, collaborative filtering algorithms present data sparseness and cold start problems. When the data are sparse, recommendation performance decreases significantly. New users do not receive recommendations and new items are not recommended because there are no interactive data [21], [22]. Hybrid recommendation algorithms are a fusion of many different recommendation algorithms, and recommendation results obtained by these different algorithms can be fused through voting mechanisms, linear combination, and other methods [23].

In recent years, deep learning has been widely used in fields such as computer vision, natural language processing, and speech recognition, owing to its flexible network structure and powerful feature learning capabilities. Deep learning has also attracted widespread attention in the recommendation field [24], [25]. There is now significant interest in studying the use of deep learning and collaborative filtering to construct recommendation algorithms. This article presents a recommendation algorithm that combines deep learning and collaborative filtering to address data sparseness and cold start problems [26], [27].

The overall technical route of our proposed solution includes data processing using cloud sensors, model fusion, relationship prediction, and model evaluation of four parts. The specific technical roadmap is shown in Figure 1:

The main contributions of this study are as follows:

- A cloud sensor collects data signals, performs signal preprocessing, performs signal feature extraction and screening, and establishes a score prediction model.
- A deep learning model composed of a convolutional neural network and a multilayer perceptron obtains non-linear feature vectors of item attributes to complete recommendations and solve data sparseness and cold start problems;
- The use of user-based and item-based NeuRec algorithms is proposed to separately learn the feature vectors of users and items under implicit feedback data, and to better learn the characteristics of users and items by combining implicit feedback information and display feedback information. Vectors can improve the accuracy of scoring predictions.
- The fusion model MF-NeuRec is proposed, and the user and item embedding vectors obtained using the NeuRec module and matrix decomposition module are fused according to a certain ratio to improve score prediction accuracy.
- In order to facilitate further research on this task, we publicly provide the source code and data of the

model on the GitHub community as contributions to the community.[1]

The remainder of this article is organized as follows. We review the relevant research related to our task in Section 2, and Section 3 provides details on executing our proposed fusion model framework. In Section 4, we describe extensive experimental evaluations and analyze the validity of the classification experiment results. Finally, conclusions and future work will be described in Section 5.

II. RELATED WORK

In this section, we review related work in the following two aspects.

A. TRADITIONAL RECOMMENDATION ALGORITHMS

Traditional recommendation algorithms include content-based recommendation, collaborative filtering-based recommendation, and hybrid recommendation. The most widely used recommendation algorithms are those based on collaborative filtering. Tang *et al.* [8] first proposed collaborative filtering algorithms in 1992. Collaborative filtering-based recommendation algorithms include memory-based collaborative filtering algorithms and model-based collaborative filtering algorithms. [13]

Memory-based collaborative filtering first uses historical scores to obtain the similarity between users and items, and then further recommends and predicts based on the similarity. Deshpande and Karypis [28] use the cosine distance to calculate the similarity between items based on a user-item interaction matrix. Although this method can generate a recommendation model rapidly, it can only learn the similarity between items that have interacted, and lack of optimization for recommendation negatively impacts its performance. Simon Funk proposed a collaborative filtering recommendation algorithm based on matrix factorization (MF) in the first Netflix competition in 2006. Matrix factorization, which assumes that the relationship between users and items can be expressed using feature vectors, significantly improved recommendation performance. Subsequently, scholars continued to improve the research on matrix decomposition. Ruslan Salakhutdinov proposed probabilistic matrix factorization (PMF) [29], which introduced a probability model based on matrix factorization, assuming that the scores were normally distributed with Gaussian noise. Both user and item feature vectors follow a normal distribution. Ning and Karypis [30] proposed the Sparse Linear Method (SLIM) algorithm to estimate the model parameters by constructing an objective function, and obtained the similarity between items. Santosh Kabbur *et al.* [31] proposed the Factored Item Similarity Model (FISM), which uses historical interaction behavior as the feature vector to express user preferences. In addition, researchers have studied collaborative filtering algorithms that fuse the two models. The SVD++ algorithm proposed by Koren [32] combines the model-based implicit semantic model and the memory-based neighbor model to combine implicit feedback information and display feedback information. This fusion algorithm has performed well in the Netflix Million Dollar Recommendation Competition for three consecutive years. Rendle [33] proposed the Factorization Machine (FM) feature model, which not only uses the user-item interaction information but also introduces auxiliary information. Koren *et al.* [34] proposed a matrix factorization algorithm incorporating machine learning techniques. The matrix decomposition model maps all users and items to a low-dimensional joint latent factor space, and models the user-item interaction information through the inner product relationship. The unfilled portion of the matrix is predicted based on the existing ratings. After predicting the ratings, the ratings of all non-interactive items of the target user are sorted, and the items with high ratings are recommended to the user.

B. DEEP LEARNING AND RECOMMENDATION SYSTEMS

In recent years, significant deep learning breakthroughs have been achieved in many fields, and a substantial amount of research has been conducted on deep learning in the field of recommendation systems. Initially, such research mainly focused on combining encoders and recommendation algorithms. In 2015, Sedhain *et al.* [35] proposed the use of an autoencoder to reconstruct the scoring matrix. Wu *et al.* [36] proposed the CDAE (Collaborative Denoising Auto-Encoders) model to learn hidden feature vectors through the actions of the user and their implicit representation of items. In 2017, Xue *et al.* [37] proposed a deep matrix factorization (DMF) model, employing a two-way network architecture to replace the hidden feature vector representation in the original matrix factorization, and using dot products to simulate the interaction relationship between users and items. Based on FISM, He *et al.* [38] consider that the target item is affected by historical interaction items, and adopt an attention mechanism to assign personalized weights to each item, such that historical interaction information makes different contributions to the user's preferences. The outer product is used to replace the connection operation in the NeuMF model, and the NAIS model is proposed. Zhang *et al.* [39] proposed a neural network-based nonlinear recommendation model (NeuRec) based on neural network modeling of historical interactive information. First, the historical interaction matrix is mapped to a low-dimensional space through a multilayer perceptron, which reduces the number of model parameters and introduces non-linear features through conversion; the mapped non-linear feature vectors are then combined with latent feature vectors.

The continued development of deep learning has led to increased research on convolutional neural networks and the combining of neural networks and recommendation algorithms. Kim *et al.* [40] proposed the ConvMF algorithm to combine convolutional neural networks with probabilistic matrix decomposition, and introduced auxiliary information

[1]The code of our paper, experimental details, and source code of the model are publicly available at https://github.com/WeixiaDu/MF_NeuRec

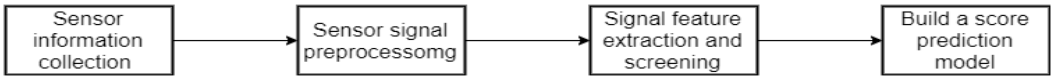


FIGURE 2. Sensor data processing flowchart.

by extracting text features to alleviate sparsity problems in the scoring matrix. Xue *et al.* [41] proposed the DeepICF algorithm, which not only models the second-order interaction between two items but also uses a nonlinear neural network to obtain interactions between all pairs of interactive items, effectively modeling the high-order relationship between items. Xiangnan *et al.* [42] used multilayer perceptrons instead of dot products to simulate the interaction between users and items. Fusion matrix decomposition and multilayer perceptron models compose the NeuMF framework. On this basis, a variety of variants have appeared. Ting *et al.* [43] introduced neighbor representations of users and items as auxiliary information for the NeuMF model to better express the user and item feature vectors. Deng *et al.* [44] proposed a collaborative filtering architecture containing joint representation learning and matching function learning, which can not only efficiently learn complex matching functions but also better learn the low-rank relationships between users and items. Kalchbrenner *et al.* [45] used dynamic CNNs and dynamic k-max pools in the training process. They provide powerful feature extraction capabilities in text modeling and can obtain the semantic vectors of sentences, leading to increased recommendation accuracy.

III. SENSOR SIGNAL ANALYSIS AND FEATURE EXTRACTION AND SCREENING

A. DATA ACQUISITION BASED ON SENSOR NETWORK

The scoring prediction algorithm mainly collects click information from the website, mines the correlation between sensor information and the user's rating of an item through related sensor signals and corresponding signal processing technology, and uses the algorithm model to predict the item's rating. The score prediction system is mainly divided into four steps: sensor signal acquisition, signal preprocessing, signal feature extraction and screening, and establishment of a score prediction model, as shown in Figure 2.

B. SENSOR SIGNAL PREPROCESSING AND FEATURE EXTRACTION

1) SENSOR SIGNAL PREPROCESSING

In practice, original signals that are collected will inevitably suffer from poor quality due to environmental noise, collection settings, and other factors. Because direct use in the subsequent algorithm model will affect the accuracy of the prediction, preprocessing operations are performed in order to obtain high-quality signals for subsequent model input. This section mainly focuses on improving collected signals through three preprocessing operations: removing invalid values, processing abnormal values, and sequential downsampling.

a: INVALID VALUE REMOVAL

As shown in Figure 3(a), the signal values at the beginning and end of the value showing how many users clicked on the website are very small. In order to prevent this signal from influencing the prediction model in subsequent calculations, the invalid values at the beginning and end must be removed. According to observations and calculations, the invalid data at the beginning and end account for 2.5% of the total sampling points; thus, in this study, 2.5% of the signal data at the beginning and end of extractions will be removed. Figure 3(b) shows the effect of removing the invalid values at the beginning and end.

b: PROCESSING OF OUTLIERS

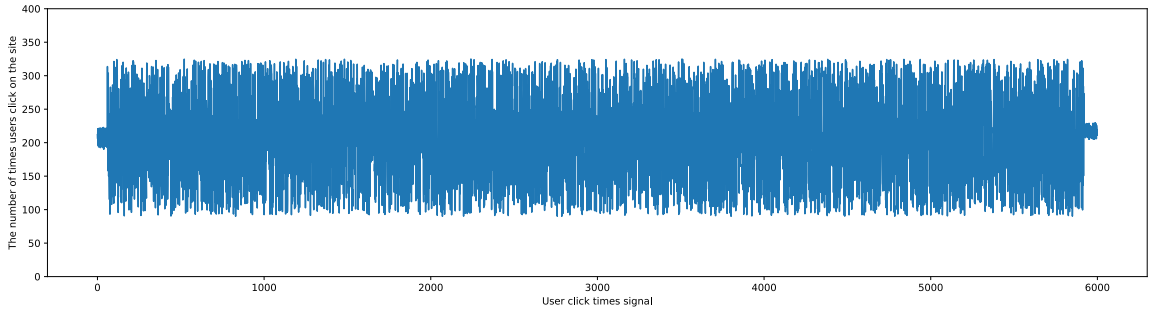
As shown in Figure 4(a), the data collection signal contains numerous abnormal points. In this study, abnormal points are replaced with the mean value of the adjacent 1000 points. Figure 4(b) shows the effect of outlier processing. It can be seen that after outlier processing, the signal value curve no longer contains clear outliers, and the curve fluctuates smoothly.

c: DOWNSAMPLING

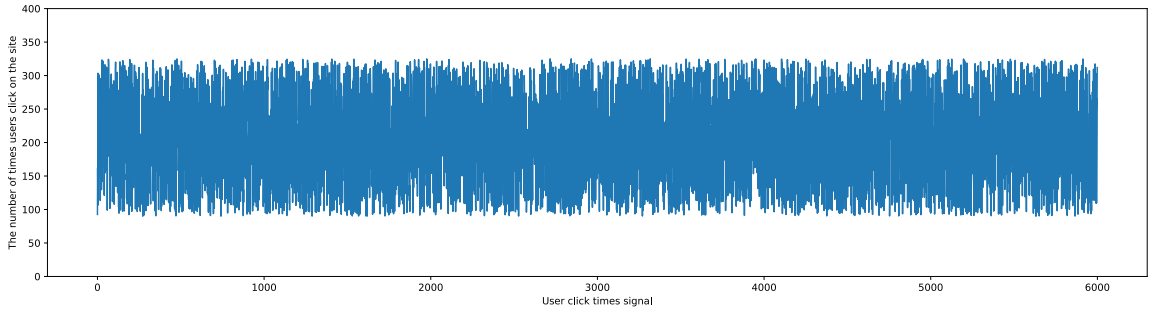
The signal acquisition frequency in the experiment is 50 kHz, and the average number of data elements sampled for a single signal is approximately 100,000. If all the signal data are used in the input of the subsequent model, the model training time will be excessive. Sensor information will not improve the prediction accuracy of the model. In order to accelerate the training speed of the subsequent model, under the premise of ensuring that all information can be collected, the collected sensor signal is downsampled at a downsampling ratio of 1/10. The signal frequency after the downsampling operation is reduced to 50 kHz / 10 = 5 kHz, and the data scale of a single signal sample is reduced to approximately 20,000. It is evident that the amount of signal data after downsampling is greatly reduced. Although the signal distribution is sparse, it contains all the pertinent information; this speeds up the training of subsequent models and does not affect the prediction accuracy of the model.

2) FEATURE EXTRACTION OF SENSOR SIGNAL

In the processing described in the previous section, each type of signal value was extracted separately, and each feature was extracted separately. For the score prediction experiment, a total of $(11+3+16)\times 6=180$ features were extracted. However, not all of these features need to be input to the scoring prediction model. On the one hand, because an excessive number of features will lead to dimensionality problems, the model training speed will decrease, and on the other hand,

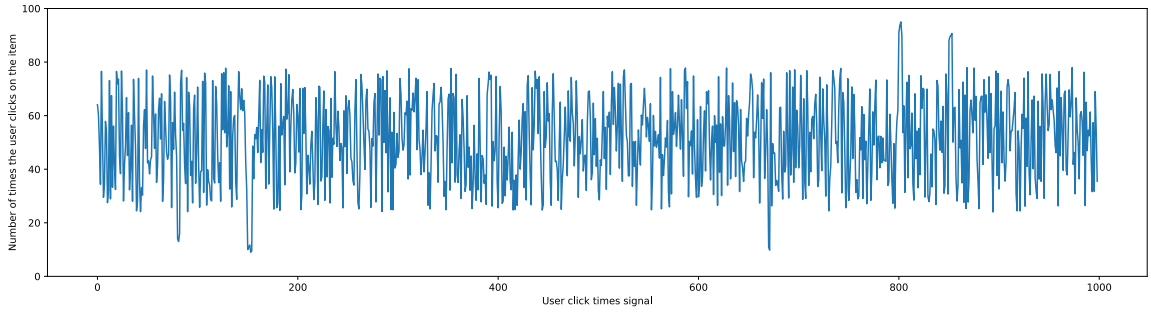


(a) Signal diagram of the number of times the original user clicks on the site

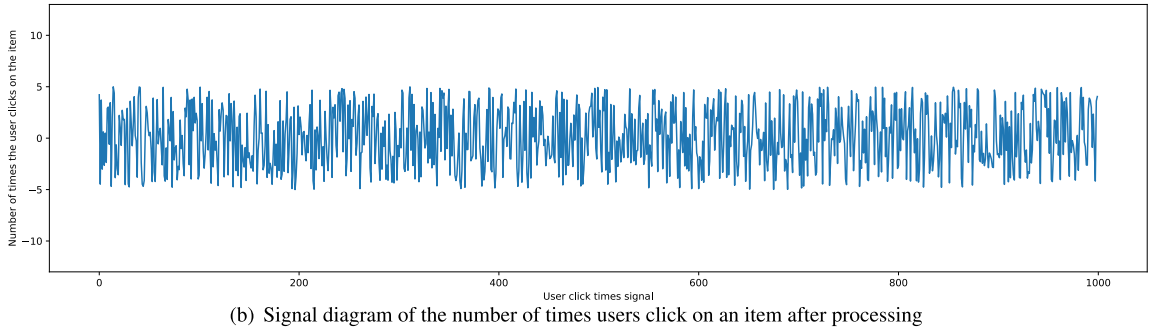


(b) Signal diagram of the number of times users click on the site after processing

FIGURE 3. **Signal diagram of the number of times users click on the website.**



(a) Signal diagram of the number of times the original user clicks on an item



(b) Signal diagram of the number of times users click on an item after processing

FIGURE 4. **Signal graph of the number of times an item has been clicked.**

there may be high-relevant redundancy in these features. Because additional features and features that are unrelated to score prediction will decrease the accuracy of the prediction model, the signal features extracted in the previous section must be filtered to obtain a meaningful feature input model. Feature selection methods are mainly divided into three types:

- **Filtering method:** selects the optimal feature according to the correlations between features.
- **Packaging method:** selects the best feature according to the prediction effect of each feature in the model.
- **Embedding method:** Obtains the importance weight of each feature through the model training results, and selects the optimal feature according to the weight order.

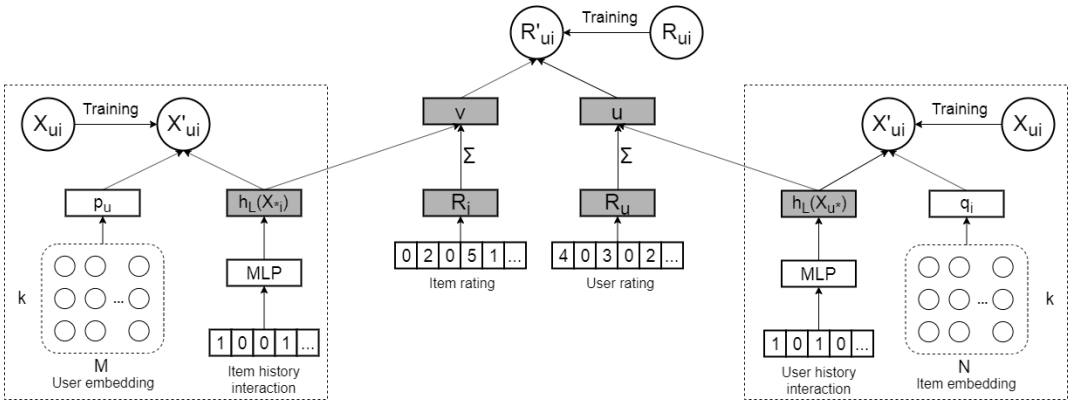


FIGURE 5. MF-NeuRec algorithm model diagram.

Because of the need to use multiple prediction models for comparison, this study uses the mutual information method in the first filtering method to screen features. The sensor collects data on movie websites and shopping websites. Movie websites mainly include rating data, user data, and movie data. The rating data include user ID, movie ID, a rating value between 1 and 5, and a timestamp. Movie data include information such as movie ID, movie title, and movie category; user data include information such as user ID, gender, age, occupation, and zip code. In this model, implicit feedback data are combined with display feedback data, implicit feedback is utilized in the NeuRec model, and explicit feedback is utilized in the MF model. When converting the displayed feedback from the data set into implicit feedback, 0/1 is used to indicate whether the user rates the movie. Shopping websites include review data and product metadata. The review data include information such as ratings, review text, and votes; product metadata include information such as description, category information, price, brand, and image characteristics.

The movie website contains data for 1 million movie ratings, 6040 users, and 3089 movies. Users and movies with minimal feedback information are filtered out. After filtering, the number of users is 6040, the number of movies is 3706, the number of historical user interactions is 1,000,209, and the data sparsity is 4.46%.

The shopping website contains data for 2 million item ratings, 8030 user reviews, and 4230 products. Users and items with minimal feedback information are filtered out. After filtering, the number of users is 7900, the number of items is 4000, the number of historical user interactions is 3,030,109, and the data sparsity is 3.76%. The details regarding the data used in the study are listed in table1.

IV. PROPOSED APPROACH

A. MF-NeuRec MODEL

The model diagram of MF-NeuRec, which combines matrix decomposition and the NeuRec score prediction algorithm, is shown in Figure 5. It mainly includes the user-based NeuRec module, item-based NeuRec module, matrix decomposition module, and fusion model MF-NeuRec.

- **User-based NeuRec module:** The virtual box on the right is the user-based NeuRec model; input is the user ID and item ID. The user ID is used to obtain the user's embedding vector from the embedding matrix, and the item ID is extracted through the multilayer perceptron. The user-item rating matrix obtains the implicit embedded vector representation of the item.
- **Item-based NeuRec module:** The virtual box on the left is the item-based NeuRec model; the input is the user ID and item ID. The item ID is used to obtain the item embedding vector from the embedding matrix, and the user ID is used to pass the multilayer perceptron. The item interaction matrix is represented by the user's implicit embedded vector.
- **Matrix decomposition module:** The middle section is the matrix decomposition module; the input is the user ID and item ID, which are used to obtain the embedding vector representations of the user and item from the user-item rating matrix.
- **Fusion model MF-NeuRec:** The user and item embedding vectors based on the NeuRec module and matrix decomposition module are fused through the model layer to obtain the fusion implicit feedback and the hidden feature representations of the users and items displaying the feedback information. The obtained user and item hidden features are used to predict user ratings of items.

1) NeuRec MODULE

The NeuRec model is a nonlinear recommendation model based on a neural network. The interaction matrix X is mapped to a low-dimensional space through a multilayer perceptron. Subject-based differences are divided into user-based and item-based NeuRec models. Two models are introduced:

a: USER-BASED NeuRec MODEL

MLP is used to obtain the nonlinear representation of the user from the user-item interaction matrix. The specific

TABLE 1. Data set features of cloud sensor system.

Features	Description	Type
Movie site data set		
***Rating data**		
User ID	Unique code for each user commenting on the movie	id
Movie name	Movie names appearing in the data set	string
Rating value	Rating of each movie	int
Timestamp	User comment time on the movie	timestamp
***User data**		
User ID	Unique code for each user in the data set	id
Sex	Gender of each user in the data set	boolean
Age	Age of each user in the data set	int
Occupation	Occupation of each user in the data set	string
Zip code	Zip code of each user in the data set	string
***Movie data**		
Movie ID	Unique encoding of each movie in the data set	id
Movie name	All movie names in the data set	string
Movie category	Categories of the movies in the data set	string
Shopping site data set		
***Rating data**		
User ID	Unique codes for all users in the data set	id
Product ID	Unique codes for all products in the data set	id
Timestamp	Timestamp of each user's product review	timestamp
Number of votes	Number of votes for each product	int
Comment text	Review text for each product	string
score	Average rating of each product in the data set	int
***Product metadata**		
Sales ranking	Sales ranking of each product based on sales volume	int
description	Merchant description of each product	string
Category information	Category information for each product	string
Price	Price of each product	double
Brand	Brand of each product	string
Image features	Image features of each product	string

calculation formula is shown in Formula 1:

$$\begin{aligned} h_1\left(X_{u^*}\right) &= f\left(W_1 X + b_1\right) \\ h_j\left(X_{u^*}\right) &= f\left(W_j h_{j-1} + b_j\right) \\ h_L\left(X_{u^*}\right) &= f\left(W_{\text{out}} h_{L-1} + b_L\right) \end{aligned} \tag{1}$$

where X_{u^*} is a row in the interaction matrix X representing a user's interaction, $f(\cdot)$ is the activation function, L is the number of layers in the neural network MLP, W_j is the weight of the unit of the MLP layer j, and b_j is the regularization term of the MLP layer j. $h_L\left(X_{u^*}\right)$ represents the user's implicit feature vector.

Assuming that the output dimension is k, a representation of each user can be obtained. The recommendation score is obtained from the inner product of two hidden vectors, and the calculation formula is shown in Formula 2:

$$\hat{X}_{ui} = h_L\left(X_{u^*}\right) \cdot q_i \tag{2}$$

Above, q_i represents the embedded vector of items embedded from the embedding matrix and $\hat{X}_{ui}$ represents the predicted value of the model.

The model is trained by the mean square error function with regular terms, and the loss function is shown in Formula 3:

$$L = \sum_{u,i} \left(X_{ui} - \hat{X}_{ui}\right)^2 + \lambda \left(\|W\|_F^2 + \|Q\|_F^2\right) \tag{3}$$

where λ is the regularization coefficient, obtained using Frobenius norm regularization weight matrix W and item hidden vector Q. The model is trained by minimizing the loss function, and the vector extracted by the multilayer perceptron is used as the user's implicit embedding vector $h_L\left(X_{u^*}\right)$, which is the input of the MF-NeuRec model.

b: ITEM-BASED NeuRec MODEL

Similar to the user-based NeuRec model, MLP is used to obtain a nonlinear representation of the item from the user-item interaction matrix. The specific calculation formula is shown in Formula 4:

$$\begin{aligned} h_1\left(X_{*_i}\right) &= f\left(W_1 X_{*_i} + b_1\right) \\ h_j\left(X_{*_i}\right) &= f\left(W_j h_{j-1} + b_j\right) \\ h_L\left(X_{*_i}\right) &= f\left(W_{\text{out}} h_{L-1} + b_L\right) \end{aligned} \tag{4}$$

Above, X_{*_i} denotes a row in interaction matrix X, and represents the interaction of an item. $h_L\left(X_{*_i}\right)$ represents the implicit feature vector of the item. The meanings of the

remaining symbols are the same as those in the user-based NeuRec model, and will not be repeated here.

The recommendation score is obtained from the inner product of two hidden vectors, and the calculation formula is shown in Formula 5:

$$\hat{X}_{ui} = p_u \cdot h_L\left(X_{*i}\right) \tag{5}$$

where $\hat{X}_{ui}$ represents the model prediction value and the embedded vector p_u represents the hidden vector of the user.

The mean square error function with regular terms is also used to train the model. The loss function is shown in Formula 6:

$$L = \sum_{u,i}\left(X_{ui} - \hat{X}_{ui}\right)^2 + \lambda\left(\|W\|_F^2 + \|P\|_F^2\right) \tag{6}$$

Above, P represents the user's hidden vector; the meanings of the remaining symbols are the same as those in the user-based NeuRec model, and will not be repeated here.

The model is trained by minimizing the loss function, and the vector extracted by the multilayer perceptron is used as the item's implicit embedding vector $h_L\left(X_{*i}\right)$, which is the input of the MF-NeuRec model.

2) MATRIX FACTORIZATION MODULE

The matrix decomposition model maps all users and items to a low-dimensional joint latent factor space, and models the user-item scoring matrix through the inner product relationship. In the user-item rating matrix $R^{m\times n}$, m represents the number of users, and n represents the number of items. In matrix decomposition, the scoring matrix is decomposed into matrices $R_i^{m\times k}$ and $R_u^{m\times k}$, which represent the dimensions of the mapped space. The prediction of item u by user u is the score $\hat{r}_{ui}$, and the calculation formula is shown in Formula 7.

$$\hat{r}_{ui} = R_i^T R_u \tag{7}$$

The vector R_i contains the hidden information of the items obtained after mapping the items, which indicates the degree to which the items possess these characteristics. R_u is the user's hidden information, which indicates the degree to which the users are interested in the characteristics of the corresponding items.

To consider the influence of bias in the scoring prediction, bias is added to the score prediction calculation formula as shown in 8.

$$\hat{r}_{ui} = \mu + b_i + b_u + R_i^T R_u \tag{8}$$

Above, μ represents the global average score, b_i represents the item rating bias, and b_u represents the user rating bias.

3) FUSION MODEL MF-NeuRec

Fusion model MF-NeuRec integrates user and item embedding vectors obtained based on the NeuRec module and matrix decomposition module according to a certain ratio. The user's implicit vector extracted from the user-based NeuRec module and the user's embedded vector in matrix decomposition are summed to obtain a user vector fused with implicit feedback and displaying feedback information. The calculation formula is shown in Formula 9.

$$u = R_u + \beta h_L\left(X_{u^*}\right) \tag{9}$$

where β is a hyperparameter used to regularize the results of implicit feedback.

The item implicit vectors extracted from the item-based NeuRec module and the item embedding vectors in matrix decomposition are summed to obtain item vectors that merge implicit feedback and display feedback information. The calculation formula is shown in Formula 10:

$$v = R_i + \beta h_L\left(X_{*i}\right) \tag{10}$$

where β is a hyperparameter used to regularize the results of implicit feedback.

The vector representation of users and items that incorporates implicit feedback and display feedback information is used to predict the user's item ratings. The calculation formula is shown in Formula 11:

$$\hat{r}_{ui} = \mu + b_i + b_u + v^T u \tag{11}$$

B. MF-NeuRec ALGORITHM ANALYSIS

This section analyzes the time complexity of the MF-NeuRec algorithm, which combines matrix decomposition and the NeuRec algorithm. The time complexity of the matrix decomposition and the NeuRec algorithm were analyzed separately. The time complexity of the NeuRec algorithm based on items is $O\left(D_L + k^*M\right)$, where D_L represents the time complexity of the multilayer perceptron, K represents the length of the embedding vector, and M represents the number of items. The user-based NeuRec algorithm has a time complexity of $O\left(D_L + k^*N\right)$, where D_L represents the time complexity of the multilayer perceptron, k represents the length of the embedded vector, and N represents the number of users. The time complexity of matrix decomposition is $O\left(k'\right)$, where k' represents the length of the matrix decomposition embedding vector. Finally, it can be concluded that the time complexity of the MF-NeuRec algorithm is $O\left(2 * D_L + k^*(N + M + 1)\right)$.

V. EXPERIMENTS

A. EVALUATION METRICS

The MF-NeuRec algorithm was implemented using the Python 3.6 programming language, and toolkits such as Pandas, Numpy, Scipy, and Sklearn were used to process the data sets. The experiments were conducted on a PC equipped with Windows 10 and 8.00 GB of memory. The TensorFlow 1.15 deep learning framework was adopted to implement the model.

B. EVALUATION INDEX

In score prediction, we use the point-by-point loss method to train the recommendation algorithm by minimizing the gap between the predicted score and the actual score, and

then evaluate the recommendation results in the test set. Root mean square error (RMSE) and mean absolute error (MAE) are common evaluation indicators in scoring prediction algorithms [46], thus we use RMSE and MAE to evaluate the performance of different methods.

1) RMSE

In score prediction methods, RMSE is a commonly used prediction error indicator. RMSE is the square root of the ratio of the squared deviation between the observed value and the true value to the number of observations m. It is used to measure the deviation between the observed value and the true value. The calculation formula is as follows:

$$RMSE = \sqrt{\frac{\sum_{(\omega,l)\in T}\left(r_{ui} - \hat{r}_{wi}\right)}{|T|}} \tag{12}$$

where T is the test set, $\boldsymbol{r}_{ui}$ is the actual score, and $\hat{r}_{ui}$ is the predicted score. RMSE represents the error value. The smaller the value of RMSE, the closer the predicted score value and the true score value, and the more accurate the prediction results of the algorithm.

2) MAE

Represents the average value of the absolute error, which can better reflect the actual situation of the predicted value error. Its calculation formula is

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|\hat{y}_i - y_i\right| \tag{13}$$

where $\hat{y}_i$ is the predicted score value and y_i is the true score value. The range of MAE is [0, +). When the predicted value and the true value are completely coincident, it is equal to 0, that is, a perfect model, indicating that the algorithm's prediction is highly accurate; the larger the error, the larger the value.

C. COMPARED BASELINE METHODS

In order to evaluate our approaches more comprehensively, we compare them against a suite of classical and state-of-the-art baselines, including the following:

——**Biasd FM [32]:** This algorithm is a recommendation algorithm based on matrix factorization. It considers user bias, item bias, and global bias on the basis of traditional matrix factorization.

——**PMF [29]:** This algorithm introduces a probability model based on matrix factorization. First, it is assumed that the scores are normally distributed with Gaussian noise, and that the user and item feature vector matrices are normally distributed. Then, based on the existing scores of the scoring matrix, the user and item feature vector matrices are obtained through the maximum posterior probability and maximum likelihood estimation, and finally, the user and item feature vectors are used to predict the unknown score in the scoring matrix.

——**NNMF(3HL) [47]:** This algorithm combines traditional matrix decomposition and a neural network with three hidden layers. The input layer of the neural network includes three parts, which are the user embedding vector, the item embedding vector, and the inner product of the embedding vector. The output is the predicted score value.

——**SVD++ [32]:** This algorithm combines the model-based implicit semantic model and the memory-based neighbor model, and combines implicit feedback information and display feedback information to improve the recommendation performance of the algorithm.

——**AutoSVD [48]:** This algorithm introduces eigenvectors of obtained items by compressing automatic encoders on the basis of SVD.

——**AutoSVD++ [48]:** Based on SVD++, this algorithm introduces item feature vectors obtained by compressing automatic encoders.

——**U-AutoRec [35]:** This is a recommendation algorithm that combines collaborative filtering and a self-encoder to predict scores.

The experiments mainly included the following tasks:

(1) Compare the proposed MF-NeuRec algorithm against the above-mentioned baseline algorithm on the basis of the Movie site and Shopping site data sets, and evaluate the accuracy of the predicted scores through RMSE and MAE in order to demonstrate the effectiveness of the proposed algorithm.

(2) Analyze how different parameters influence the accuracy of the MF-NeuRec algorithm.

D. EXPERIMENTAL RESULTS

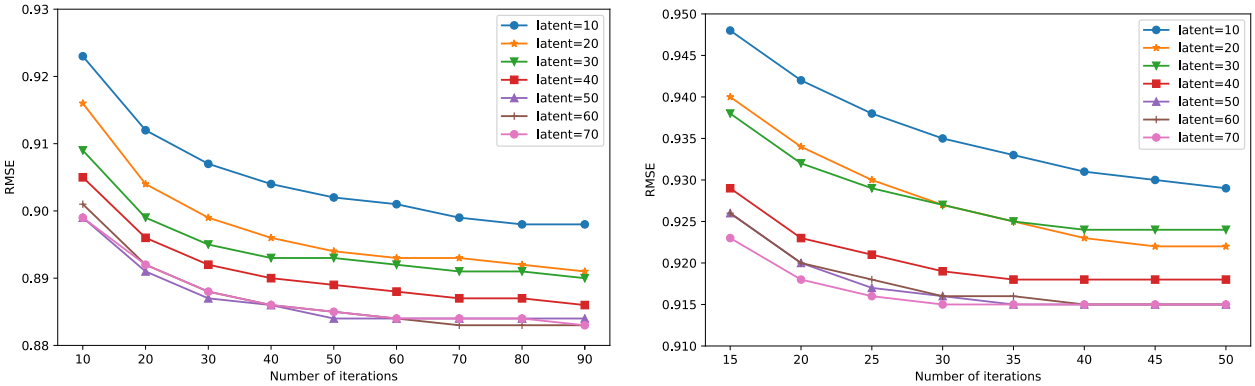
1) ALGORITHM RESULT ANALYSIS

The MF-NeuRec algorithm can better obtain the hidden features of users and items through the combination of implicit feedback, display feedback, and deep learning. The relevant parameters of MF-NeuRec were set as follows: The learning rate was set to 0.005, the regularization coefficient was set to 0.001, and the batch size was set to 512. The embedding vector settings were different for each dataset. The embedding vector dimension was set to 10 for the Movie site data, and set to 60 for the Shopping site data. The embedding layer used a normally distributed initialization vector with a mean of 0 and a standard deviation of 0.005. In the U_NeuRec module and I_NeuRec module, the learning rate was set to 0.005, the regularization coefficient was set to 0.001, the hidden layer was set to 5 layers, and the batch size was set to 3000.

The recommendation performance of the proposed algorithm was verified using the Movie site and Shopping site data sets. From these data sets, 90% of the data were randomly selected for training and 10% of the data were randomly selected for testing. We conducted a comparative experiment using baseline methods. Each baseline method was trained to obtain the optimal solution. Table 2 shows the comparison results under the evaluation indicators RMSE and MAE. The left side of the table shows the comparison results of each

TABLE 2. Comparison of RMSE and MAE results of each recommended algorithm.

Methods	Movie site		Methods	Shopping site	
	RMSE	MAE		RMSE	MAE
Biasd FM	O.911	0.659	Biasd FM	0.931	0.765
PMF	0.952	0.660	PMF	0.952	0.877
U_AutoRec	0.907	0.701	NNMF(3HL)	0.967	0.877
SVD++	0.913	0.684	SVD++	0.933	0.830
AutoSVD	0.901	0.717	AutoSVD	0.921	0.825
AutoSVD++	0.904	0.705	AutoSVD++	0.934	0.806
MF-NeuRec	**0.885**	**0.647**	**MF_NeuRec**	**0.917**	**0.750**
Improve	**1.78%**	**0.82%**	**Improve**	**0.43%**	**1.96%**



(a) RMSE results for different embedding vector lengths with Movie site data set

(b) RMSE results for different embedding vector lengths with Shopping sited data set

FIGURE 6. RMSE results for different embedding vector lengths with each data set.

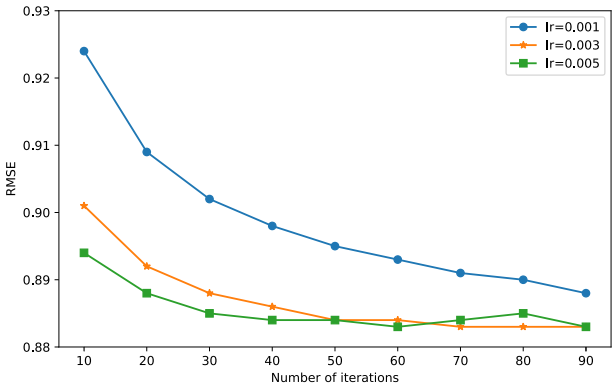
recommendation algorithm using the Movie site data set, and the right side shows the comparison results using the Shopping site data set. Each row represents the RMSE and MAE values of a recommendation algorithm. The smaller the value, the better the prediction performance.

The results listed in the table show that with the Movie site data set, the MF-NeuRec algorithm produced a 1.78% reduction in RMSE and a 1.82% reduction in MAE compared to the best baseline algorithm; with the Shopping site data set, the MF-NeuRec algorithm produced the best results. A good algorithm reduces RMSE by 0.43% and MAE by 1.96%. With both data sets, the proposed algorithm produced lower RMSE and MAE values than the other recommendation algorithms, which proves that the proposed algorithm has a greater ability to predict accurate scores.

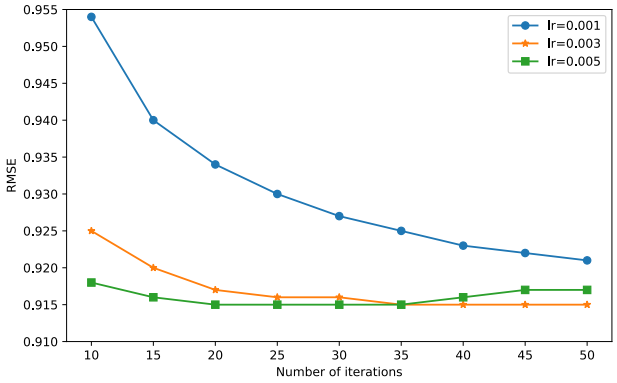
2) ALGORITHM PARAMETER ANALYSIS

This subsection analyzes how the experimental results are influenced by different embedding vector lengths, learning rates, batch sizes, and training data. We incremented the embedding vector length from 10 to 70 and kept other parameters unchanged, and conducted experiments using the Movie site and Shopping site data sets. Figure 6(a) shows the RMSE obtained when 90% of the data in the Movie site data set was used for training. Figure 6(b) shows the RMSE obtained when 50% of the data in the Shopping site data set was used for training.

We selected the RMSE results from the first 90 iterations of the Movie site experiment and the RMSE results from the first 50 iterations of the Shopping site experiment. Figure 6(a) shows the results of iterations 10 through 90 for the Movie site data set. At the beginning of the iterations, the models with larger embedding vector lengths outperformed those with smaller embedding vector lengths, and the algorithm results improved as the embedding vector length increased. This may have occurred because an embedding vector dimension that is too low will lead to poor program learning ability, resulting in under-fitting. However, we also found that as the length of the embedding vector increased, the training time also gradually increased; moreover, when the number of iterations was close to 90, the RMSEs of the embedding vectors with lengths of 70, 60, and 50 were similar, because the dimensions of the embedded vector tended to be saturated. An excessive number of dimensions does not further improve the accuracy of the experiment and prolongs the running time of the program. Figure 6(b) shows the results of iterations 15 through 50 for the Shopping site data set, which are similar

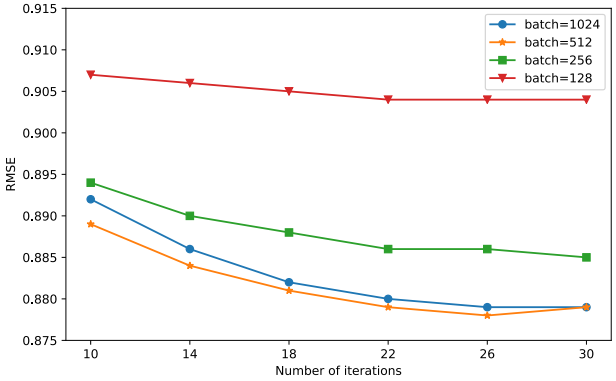


(a) RMSE results for different learning rates with Movie site data set

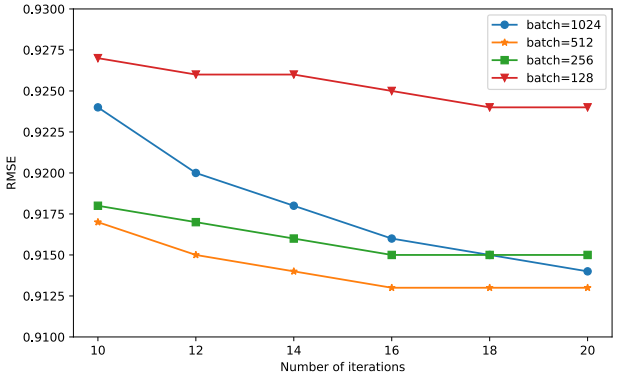


(b) RMSE results for different learning rates with Shopping site data set

FIGURE 7. RMSE results for different learning rates with each data set.



(a) RMSE results for different batch sizes with Movie site data set



(b) RMSE results for different batch sizes with Shopping site data set

FIGURE 8. RMSE results for different batch sizes with each data set.

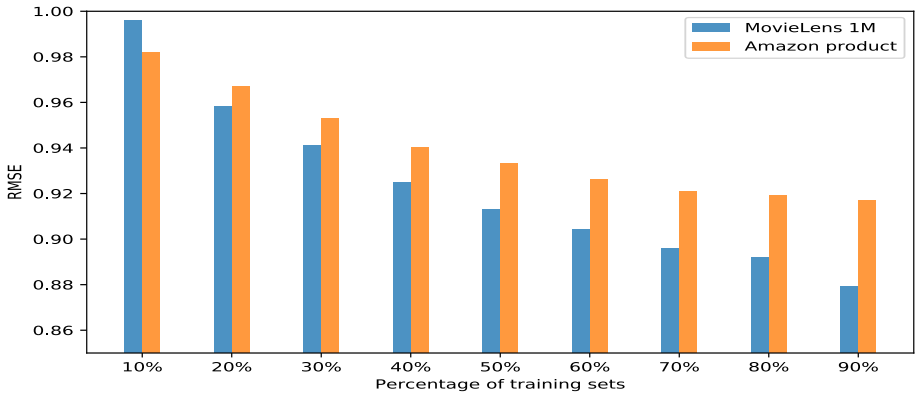


FIGURE 9. Results for different percentages of training data from Movie site and Shopping site data sets.

to the results shown in Figure 6(a). When the number of iterations is close to 40, the RMSEs of the embedding vectors with lengths of 70, 60, and 50 are similar.

To study how the algorithm results were affected by using different learning rates, we conducted experiments with learning rates of 0.001, 0.003, and 0.005. Figures 7(a) and 7(b) show the experimental results obtained using the training data in the Movie site and Shopping site data sets, respectively. Figure 7(a) shows that the learning rate increased and the model converged faster after the learning rate was increased. When using training data from the Movie site data set and learning rates of 0.003 and 0.005,

the model generally converges at approximately 50 iterations. Figure 7(b) shows experimental results obtained with training data from the Shopping site data set and learning rates of 0.003 and 0.005; here, the model generally converges at 30 iterations.

Furthermore, we studied how different batch sizes affected the algorithm results. In the experiments, batch sizes of 128, 256, 512, and 1024 were used. Figure 8(a) shows that when testing with the Movie site data set, the model performs best when the batch size is 512. Figure 8(b) shows that when testing with the Shopping site data set, the model performs best when the batch size is 512. This occurred because MF-NeuRec requires a suitable dimension to encode semantic information; larger dimensions may introduce additional redundancy, while smaller dimensions do not allow a sufficient amount of useful information to be obtained.

Finally, we studied how the algorithm was affected by using different percentages of training data from the Movie site and the Shopping site data sets. The training data percentages ranged from 10% to 90%. The experimental results shown in Figure 9 indicate that as the percentage of training data increases, the RMSE value decreases; that is, the experimental results of the algorithm improved. This occurs because more detailed information is obtained during the training process, which increases the effectiveness of the training and significantly improves overall test results.

VI. CONCLUSION AND FUTURE WORK

To address the sparseness of scoring data in scoring prediction, this article proposes MF-NeuRec, a scoring prediction algorithm that combines matrix factorization and the NeuRec model. The algorithm uses the cloud sensor system to collect data; after preprocessing, the data are used as the input data set for the score prediction model. The NeuRec algorithm is then used to effectively obtain the feature representations of items and users under the implicit feedback data. The implicit feedback and display feedback data are fused into a matrix decomposition framework; this combination of implicit feedback and display feedback can better learn the hidden features of users and items, and to a certain extent completes the effect of solving the rating data sparsity problem. The combination of matrix factorization and NeuRec described in this article is innovative. The algorithm proposed in this article was verified using MovieLens 1M and data from Amazon. The experimental results prove that deep learning and the combination of explicit and implicit feedback can improve the accuracy of score prediction. The algorithm takes advantage of deep learning's ability to effectively extract features and implicit feedback to effectively supplement display feedback. The combination of matrix factorization and NeuRec has a certain novelty.

In future work, we will first propose improvements to the data processing portion of the cloud sensor signal, with the goal of improving the collected signal information and increasing its suitability as the data set of our scoring algorithm. We will then aim to improve the interpretability of recommendations while ensuring the accuracy of the algorithm.

ACKNOWLEDGMENT

The authors would like to thank my mentor for providing our team with helpful guidance and suggestions during the process of writing this article. They also would like to thank my alma mater, Yanshan University, for providing me with more learning platforms and the students for their help in writing my thesis.

REFERENCES

[1] B. D. Deebak and F. Al-Turjman, "A novel community-based trust aware recommender systems for big data cloud service networks," *Sustain. Cities Soc.*, vol. 61, Oct. 2020, Art. no. 102274. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2210670720304959
[2] L. He, C.-T. Lu, G. Ma, S. Wang, L. Shen, P. S. Yu, and A. B. Ragin, "Kernelized support tensor machines," in *Proc. Mach. Learn. Res.*, vol. 70, D. Precup and Y. W. Teh, Eds. Sydney, NSW, Australia: PMLR, Aug. 2017, pp. 1442–1451.
[3] M. Z. A. Bhuiyan, G. Wang, J. Wu, J. Cao, X. Liu, and T. Wang, "Dependable structural health monitoring using wireless sensor networks," *IEEE Trans. Dependable Secure Comput.*, vol. 14, no. 4, pp. 363–376, Jul. 2017.
[4] J. Tang, "AMiner: Toward understanding big scholar data," in *Proc. 9th ACM Int. Conf. Web Search Data Mining (WSDM)*, 2016, p. 467.
[5] M. F. Aljunid and M. Dh, "An efficient deep learning approach for collaborative filtering recommender system," *Procedia Comput. Sci.*, vol. 171, pp. 829–836, Jan. 2020.
[6] J. S. Lee and J. C. Lee, "Context awareness by case-based reasoning in a music recommendation system," in *Proc. Int. Symp. Ubiquitious Comput. Syst.* Springer, 2007, pp. 45–58.
[7] R. Wang, X. Ma, C. Jiang, Y. Ye, and Y. Zhang, "Heterogeneous information network-based music recommendation system in mobile networks," *Comput. Commun.*, vol. 150, pp. 429–437, Jan. 2020.
[8] J. Tang, "Computational models for social network analysis: A brief survey," in *Proc. 26th Int. Conf. World Wide Web Companion (WWW Companion)*, 2017, pp. 921–925.
[9] G. Ma, C.-T. Lu, L. He, P. S. Yu, and A. B. Ragin, "Multi-view graph embedding with hub detection for brain network analysis," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2017, pp. 967–972.
[10] M. Zihayat, A. Ayanso, X. Zhao, H. Davoudi, and A. An, "A utility-based news recommendation system," *Decis. Support Syst.*, vol. 117, pp. 14–27, Feb. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167923618301970
[11] H. Cheng, X. Yan, J. Han, and P. S. Yu, "Direct discriminative pattern mining for effective classification," in *Proc. IEEE 24th Int. Conf. Data Eng.*, Apr. 2008, pp. 169–178.
[12] J. Li, C. Li, J. Liu, J. Zhang, L. Zhuo, and M. Wang, "Personalized mobile video recommendation based on user preference modeling by deep features and social tags," *Appl. Sci.*, vol. 9, no. 18, p. 3858, Sep. 2019.
[13] J. Tang and W. Hall, "Cross-domain ranking via latent space learning," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017.
[14] K. Wu, J. Tang, Z. Shao, X. Xu, B. Gao, and S. Zhao, "CareerMap: Visualizing career trajectory," *Sci. China Inf. Sci.*, vol. 61, no. 10, pp. 247–249, 2018.
[15] H. Peng, J. Li, Y. Song, and Y. Liu, "Incrementally learning the hierarchical softmax function for neural language models," in *Proc. 31st AAAI Conf. Artif. Intell.* Palo Alto, CA, USA: AAAI Press, 2017, pp. 3267–3273.
[16] H. Peng, J. Li, Y. He, Y. Liu, M. Bao, L. Wang, Y. Song, and Q. Yang, "Large-scale hierarchical text classification with recursively regularized deep graph-CNN," in *Proc. World Wide Web Conf. World Wide Web (WWW)*, 2018, pp. 1063–1072.
[17] H. Peng, J. Li, Q. Gong, Y. Song, Y. Ning, K. Lai, and P. S. Yu, "Fine-grained event categorization with heterogeneous graph convolutional networks," in *Proc. 28th Int. Joint Conf. Artif. Intell.* Palo Alto, CA, USA: AAAI Press, Aug. 2019, pp. 3238–3245.
[18] J. Gong, S. Wang, J. Wang, W. Feng, H. Peng, J. Tang, and P. S. Yu, "Attentional graph convolutional networks for knowledge concept recommendation in MOOCs in a heterogeneous view," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 79–88.

[19] Y. He, J. Li, Y. Song, M. He, and H. Peng, "Time-evolving text classification with deep neural networks," in *Proc. 27th Int. Joint Conf. Artif. Intell.* Palo Alto, CA, USA: AAAI Press, Jul. 2018, pp. 2241–2247.
[20] Q. Mao, J. Li, S. Wang, Y. Zhang, H. Peng, M. He, and L. Wang, "Aspect-based sentiment classification with attentive neural turing machines," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 5139–5145, doi: 10.24963/ijcai.2019/714.
[21] H. Peng, J. Li, Q. Gong, Y. Ning, S. Wang, and L. He, "Motif-matching based subgraph-level attentional convolution network for graph classification," in *Proc. 21st AAAI Conf. Artif. Intell.* Palo Alto, CA, USA: AAAI Press, 2020, pp. 5387–5394.
[22] G. Ma, L. He, C.-T. Lu, W. Shao, P. Yu, A. Leow, and A. Ragin, "Multi-view clustering with graph embedding for connectome analysis," in *Proc. ACM Conf. Inf. Knowledge Manage.*, 2017, pp. 127–136.
[23] Y. He, Y. Song, J. Li, C. Ji, J. Peng, and H. Peng, "HeteSpaceyWalk: A heterogeneous spacey random walk for heterogeneous information network embedding," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Nov. 2019, pp. 639–648.
[24] Z. Liu, Y. Dou, P. S. Yu, Y. Deng, and H. Peng, "Alleviating the inconsistency problem of applying graph neural network to fraud detection," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 1–4.
[25] X. Zhu, Y. Luo, A. Liu, W. Tang, and M. Z. A. Bhuiyan, "A deep learning-based mobile crowdsensing scheme by predicting vehicle mobility," *IEEE Trans. Intell. Transport. Syst.*, to be published. [Online]. Available: http://dx.doi.org/10.1109/tits.2020.3023446
[26] H. Peng, J. Li, S. Wang, L. Wang, Q. Gong, R. Yang, B. Li, P. Yu, and L. He, "Hierarchical taxonomy-aware and attentional graph capsule RCNNs for large-scale multi-label text classification," *IEEE Trans. Knowl. Data Eng.*, early access, Dec. 16, 2020, doi: 10.1109/TKDE.2019.2959991.
[27] B. Du, H. Peng, S. Wang, M. Z. A. Bhuiyan, L. Wang, Q. Gong, L. Liu, and J. Li, "Deep irregular convolutional residual LSTM for urban traffic passenger flows prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 972–985, Mar. 2020, doi: 10.1109/tits.2019.2900481.
[28] M. Deshpande and G. Karypis, "Item-based top-n recommendation algorithms," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 143–177, Jan. 2004, doi: 10.1145/963770.963776.
[29] A. Mnih and R. Salakhutdinov, "Probabilistic matrix factorization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 1257–1264.
[30] X. Ning and G. Karypis, "Slim: Sparse linear methods for top-n recommender systems," in *Proc. IEEE 11th Int. Conf. Data Mining*, Dec. 2011, pp. 497–506.
[31] S. Kabbur, X. Ning, and G. Karypis, "FISM: Factored item similarity models for top-N recommender systems," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*. New York, NY, USA: Association for Computing Machinery, 2013, p. 659, doi: 10.1145/2487575.2487589.
[32] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*. New York, NY, USA: Association for Computing Machinery, 2008, pp. 426–434, doi: 10.1145/1401890.1401944.
[33] S. Rendle, "Factorization machines," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2010, pp. 995–1000.
[34] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.
[35] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "Autorec: Autoencoders meet collaborative filtering," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 111–112.
[36] Y. Wu, C. Dubois, A. X. Zheng, and M. Ester, "Collaborative denoising auto-encoders for top-n recommender systems," in *Proc. Int. Conf. Web Search Data Mining*, Nov. 2016, pp. 153–162.
[37] H. Xue, X. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems," in *Proc. IJCAI*, 2017, pp. 3203–3209.
[38] X. He, X. Du, X. Wang, F. Tian, J. Tang, and T.-S. Chua, "Outer product-based neural collaborative filtering," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 2227–2233, doi: 10.24963/ijcai.2018/308.
[39] S. Zhang, L. Yao, A. Sun, S. Wang, G. Long, and M. Dong, "NeuRec: On nonlinear transformation for personalized ranking," 2018, *arXiv:1805.03002*. [Online]. Available: https://arxiv.org/abs/1805.03002
[40] D. Kim, C. Park, J. Oh, S. Lee, and H. Yu, "Convolutional matrix factorization for document context-aware recommendation," in *Proc. 10th ACM Conf. Recommender Syst.* New York, NY, USA: Association for Computing Machinery, Sep. 2016, p. 233, doi: 10.1145/2959100.2959165.
[41] F. Xue, X. He, X. Wang, J. Xu, K. Liu, and R. Hong, "Deep item-based collaborative filtering for top-n recommendation," *ACM Trans. Inf. Syst.*, vol. 37, no. 3, p. 33, 2019.
[42] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. Chua, "Neural collaborative filtering," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 173–182.
[43] T. Bai, J. Wen, J. Zhang, and W. X. Zhao, "A neural collaborative filtering model with interaction-based neighborhood," in *Proc. ACM Conf. Inf. Knowledge Manage.*, 2017, pp. 1979–1982.
[44] Z.-H. Deng, L. Huang, C.-D. Wang, J.-H. Lai, and P. Yu, "Deepcf: A unified framework of representation learning and matching function learning in recommender system," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019, pp. 61–68.
[45] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics*. Baltimore, Maryland: Association for Computational Linguistics, Jan. 2014, pp. 655–665.
[46] C. Shi, J. Liu, F. Zhuang, P. S. Yu, and B. Wu, "Integrating heterogeneous information via flexible regularization framework for recommendation," *Knowl. Inf. Syst.*, vol. 49, no. 3, pp. 835–859, Dec. 2016.
[47] G. Karolina Dziugaite and D. M. Roy, "Neural network matrix factorization," 2015, *arXiv:1511.06443*. [Online]. Available: http://arxiv.org/abs/1511.06443
[48] S. Zhang, L. Yao, and X. Xu, "AutoSVD++: An efficient hybrid collaborative filtering model via contractive auto-encoders," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Aug. 2017, pp. 957–960.

JIBING GONG received the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, China. He is currently a Professor with the School of Information Science and Engineering, Yanshan University, where he is the Head of the Knowledge Engineering Group (KEG) Research Team. His main research interests include big data analytics, heterogeneous information networks, machine learning, and data fusion.

WEIXIA DU is currently pursuing the master's degree with the Department of Information Science and Engineering, Yanshan University. Her research interests include heterogeneous information networks and recommendation systems.

HUANHUAN LI is currently pursuing the master's degree with the Department of Information Science and Engineering, Yanshan University. His research interest includes hot event discovery algorithms.

QING LI is currently pursuing the master's degree with the Department of Information Science and Engineering, Yanshan University. Her main research interests include machine learning and social networks.

YI ZHAO is currently pursuing the master's degree with the Department of Information Science and Engineering, Yanshan University. His research interests include machine learning and deep learning.

KAILUN YANG is currently pursuing the master's degree with the Department of Information Science and Engineering, Yanshan University. His research interest includes recommendation systems.

YING WANG is currently the Deputy Director of the Division of Talent Information Development, STTC, MOST, China. Her research interests include science and technology human resources and Chinese researcher policy.