Music personalization imputation method based on deep transfer transfer learning

Jing Wang¹,†

Abstract

The current music recommendation is less efficient and cannot meet the needs of most people. This paper combines deep learning and migration learning to research music recommendations. It mainly utilizes the hybrid recommendation method of label recommendation and noise reduction autoencoder to achieve accurate music recommendations after extracting the features of music and the user’s preference features. In addition, in order to explore the recommendation effect of the model in this paper, the deviation between the measured score and the actual score of this model is compared and analyzed as well as the similarity difference of music with different features is explored in terms of HR, Recall, and NDCG index performance. The results show that the number of songs with a deviation of 2 or less in the predicted scores of this recommender system all account for 88% of the total number of songs, which is better than other models. In terms of HR, Recall, and NDCG, this paper’s model has better recommendation performance than other models, and the similarity of music recommendation between the same album and different albums is more than 0.65. This study is of great significance for eliminating the information barriers to enhance and explore the value of music data information.

Keywords: Migration learning; Personalized recommendation; Similarity difference; Autoencoder; Preference analysis.

AMS 2010 codes: 97P10
1 Introduction

Music is one of the most important forms of entertainment, and it is an important expression and support of emotional pursuit in people’s lives. With the development of digital multimedia technology, digital music has been rapidly promoted, compared with the traditional way of music dissemination, digital music is inexpensive, easy to disseminate and obtain, and is popular among people [1-3]. Both online and mobile music have been developed rapidly, and various kinds of music websites and music radio stations have emerged, such as Spotify, Pandora, Douban music radio, etc., which provide users with thousands of songs [4-5]. The large number and variety of these songs make it difficult for users to find what they want to listen to, and even many users are not clear about their interests [6]. Therefore, music recommendation has essential significance.

On the one hand, for users, providing accurate recommendations for them according to their interests can save energy and time, thus obtaining a good user experience [7]. On the other hand, for music websites, providing good music recommendation services can attract new users and increase users’ loyalty to the website, which gives them an advantage in competition [8-10]. However, music recommendation has its unique characteristics: music listening time is short, low price, colossal quantity, and rich variety, listening with coherence and sequence, closely related to the contextual environment, etc. [11-12]. Traditional single recommendation algorithms do not solve the problem of music recommendation accuracy according to the characteristics of music recommendation [13]. Therefore, it is essential to carry out deep data mining based on users, songs, social tags, and users’ actual listening habits and apply multiple types of recommendation algorithms from various dimensions to recommend suitable music and listening lists with a continuity experience for users [14-16].

Current mainstream recommendation algorithms are categorized into two main groups: content-based recommendation and collaborative filtering-based recommendation, and most music websites or music radio stations at home and abroad usually adopt these two techniques [17]. Literature [18] based on the user’s music preference, the user’s implicit ratings of music calculated by deep convolutional neural network (DCNN) algorithm and weighted feature extraction (WFE) algorithm, achieved a high music recommendation accuracy rate after training on the million songs data set (MSD). Literature [19] proposed a hybrid music recommendation model based on personalized measurement and game theory, which solves the problem of user interest drift by improving the traditional collaborative filtering algorithm through two metrics: novelty (UPND) and (music popularity (MP). Literature [20] constructed a hybrid music recommendation system based on a genetic algorithm, which adopts the optimal individual retention strategy so that the maximum fitness value of the individuals in the population is equal to the global optimal fitness value and achieves a good music recommendation effect through the fusion of music genes. Literature [21] utilized a music-specific emotion model (GEMS) based on users’ music emotions to achieve contextual music recommendations for users.

In recent years, many scholars have made improvements or proposed some new recommendation algorithms for the defects of traditional recommendation algorithms. Literature [22] proposed a music recommendation system (MRS) based on personality traits and physiological signals, which analyzes through a variety of information sources, uses user attributes and contextual data to reveal the essential indicators of some physiological characteristics, and realizes the improvement of recommendation accuracy. Literature [23] constructed a music genre classification system and a user music recommendation engine based on an improved deep neural network model, which extracts acoustic features from the network for classifying music genres and recommending music on the dataset. Literature [24] improves the traditional collaborative filtering recommendation algorithm based on the latent factor model and proposes a music recommendation algorithm that combines
clustering and latent factors. Literature [25] combined machine learning algorithms and deep learning algorithms to develop a personalized music recommendation system by combining both song popularity and rhythmic content through collaborative filtering algorithms. Literature [26] proposed an emotion-aware music recommendation algorithm based on a deep neural network (emoMR) using music emotion as a discrete representation, which utilized low-level audio features and music metadata to model and predict the user’s music emotion and music preference in a continuous form.

In this paper, deep learning is combined with migration learning. The weight matrix parameters are continuously optimized through iterative training, and the improved noise reduction autoencoder is used to downsize the data to improve the model performance. Then, based on the model, the audio features and lyrics features of music are extracted, the user’s music preference is analyzed in combination with the user data, and the music attribute labels are added on the basis of the noise reduction autoencoder to realize the personalized recommendation of music. Finally, the accuracy of the model recommendation is analyzed, comparing the superiority of this paper’s model over other models in terms of hit rate, recall rate, and NDCG evaluation indexes and exploring the similarity of this paper’s model in recommending songs from different albums and different musicians, which verifies the validity of this paper’s model in music personalized recommendation.

2 Methodology

2.1 Deep Transfer Transfer Learning

A popular research direction involves combining deep learning with migration learning to utilize datasets and trained models for new tasks. The learning effect on different tasks can be directly improved through deep migration learning. In addition, as deep learning is capable of learning and extracting directly from raw data, it can automate the extraction of more expressive features. The deep transfer migration learning model is shown in Figure 1.

Transfer learning is the process of transferring knowledge acquired in other domains to another domain, i.e., learning C++ can be analogized to learning Java. In 2010, Sinno et al. presented the initial formal definition of transfer learning, which has become a significant subfield of machine learning. The formal definition of transfer learning is: given a source domain \( S \) and a learning task \( T_s \) on the source domain, and a target domain \( T \) and a learning task \( T_t \) on the target domain, use the knowledge on \( S \) and \( T \) to learn a predictive objective function \( f(\cdot) \) on the target domain \( T_t \). Where \( S \neq T \) or \( T \neq T_t \), the domain consists of two parts: the feature space \( \mathcal{X} \) and the marginal probability distribution \( P(X), X = \{x_1, \ldots, x_n\} \in \mathcal{X} \), then the source domain refers to the domain of the existing knowledge, which can be denoted as \( D_s = \{(x_{s_1}, y_{s_1}), \ldots, (x_{s_n}, y_{s_n})\} \) where \( x_{s_i} \in \mathcal{X}_s, y_{s_i} \in \gamma_s \) and \( y_{s_i} \) are the corresponding category labels. The target domain refers to the domain where learning is to be performed and can be denoted as \( D_t = \{(x_{t_1}, y_{t_1}), \ldots, (x_{t_{n_t}}, y_{t_{n_t}})\} \), where \( x_{t_i} \in \mathcal{X}_t, y_{t_i} \in \gamma_t \) and \( y_{t_i} \) are the corresponding outputs, i.e. the category labels are predicted by the objective function. In most cases, \( 0 \leq n_t \leq n_s \).

The central assumption of traditional machine learning methods is that the data in the source domain and the data in the target domain are required to satisfy the data homogeneous distribution, i.e., They have to have the same feature space, distribution, and learning task. To learn a good model, it is
necessary to have sufficient training samples available. Migration learning relaxes these two restrictions. Not only its source domain and target domain data feature space is not the same or follow different data distributions, but also migration learning can migrate the acquired knowledge to different but similar domains, which can solve the learning problem of insufficient or even no available training samples in the target domain.

Traditional machine learning methods have been the mainstay of transfer learning research in the past. Deep transfer learning and its applications are of great significance in reality, as deep learning has become one of the fastest-growing machine learning fields. Tan et al. from the National Key Laboratory of Intelligent Technologies and Systems, Tsinghua University, defined the concept of deep transfer learning, which is to learn a predictive objective function \( f_T(\cdot) \) on a target domain \( D_T \) using knowledge on \( D_S \) and \( T_S \), given a transfer learning task defined by \(< D_S,T_S,D_T,T_T,f_T(\cdot) >\).

Where \( D_S \) denotes a source domain, \( T_S \) denotes a learning task on a source domain, \( D_T \) denotes a target domain, \( T_T \) denotes a learning task on a target domain, \( f_T(\cdot) \) is a nonlinear function representing a deep neural network. Deep migration learning can be categorized into instance-based, mapping-based, network-based, and adversarial-based.

![Figure 1](image_url)  
**Figure 1.** Deep transfer learning

### 2.2 Recommendation Algorithms Based on Deep Migration Learning

In recent years, traditional collaborative filtering recommendation algorithms have become increasingly unable to meet the needs of users due to their shortcomings as well as the exponential growth of data volume. Many new music lacks labels and ratings, making it difficult to enter the public’s view through recommendation algorithms, which can be used to extract and categorize features of the music’s spectrum as well as audio time series. Traditional recommendation algorithms are unable to effectively process image and audio sequence information, making it challenging to classify new songs.

This paper’s focus is on training the neural network using the training set, continuously optimizing the weight matrix parameters, and finally predicting according to the model. In this paper, both the feature extraction model and the recommendation model use deep transfer learning models. The feature extraction model is a combination of a complete convolutional neural network and a recurrent
neural network. The recommendation model uses a hybrid recommendation model consisting of label-based recommendations and noise-reducing autoencoder-based recommendations, and label-based recommendations are one of the content-based recommendations.

2.2.1 Autoencoder

Autoencoder is a type of unsupervised learning neural network that involves autoencoding and auto-decoding and has a three-layer network structure that includes an input layer, hidden layer, and output layer. The input layer to the hidden layer belongs to the coding part, and the hidden layer to the output layer belongs to the coding part. The number of neurons in the input layer is equal to the number of neurons in the output layer, which is also more significant than the number of neurons in the hidden layer, and this neuron ratio can reduce the dimensionality of the input data, which is more conducive to feature extraction.

At runtime, the autoencoder is segmented into two phases: forward propagation and backpropagation.

For forward propagation, let the input sample be \( x \in D_k \), \( n \) neurons in the input and output layers and \( m \) neurons in the hidden layer, and the representation from the input layer to the hidden layer is shown in Equation (1):

\[
h(x) = \sigma_1(\omega_1 x + b_1)
\]

Where \( \omega_1 \) denotes the weight matrix between the input and hidden layers, order \( m \times n \), \( b_1 \) is the bias term, and order \( m \times 1 \), \( \sigma_1 \) denotes the activation function, usually chosen as the Sigmoid, Tanh, or Relu function. Encoding the input is how the process is represented.

Equation (2) depicts the transition from the hidden layer to the output layer:

\[
O(h(x)) = \sigma_2(\omega_2 h(x) + b_2)
\]

Where \( \omega_2 \) denotes the weight matrix between the hidden and output layers, order \( n \times m \), \( b_2 \) is the bias term, and order \( n \times 1 \), \( \sigma_2 \) denotes the activation function. This process can be embodied as decoding the previous operations to obtain the feature vectors.

At this point, the forward propagation process has been completed, after which the weight matrices \( \omega_1 \), \( \omega_2 \) and the bias vectors \( b_1 \), \( b_2 \) are updated by calculating the loss function using stochastic gradient descent. The loss function is defined as shown in Equation (3):

\[
loss(O(h(x)), x) = \frac{\lambda_1}{2} \sum_{i=1}^{k} \|O(h(x)) - x\|^2 + \frac{\lambda_2}{2} \|\omega_1\|^2 + \frac{\lambda_2}{2} \|\omega_2\|^2
\]

Where \( \|\| \) denotes the 2-parameter of the matrix, \( \frac{\lambda_1}{2} \|\omega_1\|^2 + \frac{\lambda_2}{2} \|\omega_2\|^2 \) is a regularization term added to prevent overfitting of the model, and \( \lambda_1 \) and \( \lambda_2 \) are generally denoted to be taken between 0 and 1.
2.2.2 Noise-canceling autoencoder

Autoencoders that add a certain amount of noise to the input data are used to reduce noise. This approach effectively improves the generalization and robustness of the network and can still recommend more accurately even in the presence of noise, and the rest of the process, including forward propagation and back propagation, is the same as that of a normal autoencoder. In this paper, we randomly select a small number of samples and add one to their input values.

The Top-N recommendation list is generated among the items that the user does not rate, and the network parameters are trained based on the User-Item rating scale. In order to avoid the data sparsity problem, two encoders can be trained for the user and the item, respectively, and the activation function is chosen as the tanh function, see Equation (4):

\[
\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}
\]  

Noise-canceling autoencoder using Pytorch version 1.11.0 to construct the sample \( x \in D_k' \), indicates that there are a total of \( k \) sets of randomly noisy data, training, input \( x_i, i \in (0, k) \), that is, a set of user rating vectors, UserItem rating matrix in each row of data. The current output \( O(h(x)) \) is obtained by forward propagation of Eq. (1) and Eq. (2). The loss function between the current output and the input \( \text{loss}(O(h(x)), x) \) is calculated according to Eq. (3), and finally, the loss function is biased and fine-tuned to complete the gradient descent. All \( x_i \) of the samples are fed into the training network through a loop, and the training is completed.

After the training network step, according to the new User-Item scoring matrix output from the network, using the traditional collaborative filtering method, the similarity of the items is obtained by calculating the modified cosine similarity calculation between the items, and the output value of the modified cosine similarity is in the range of \([-1, 1]\), with the negative value indicating negative correlation, a positive value indicating positive correlation, and 0 representing irrelevance. The modified cosine similarity formula can be seen in equation (5):

\[
\text{sim}(x, y) = \frac{\sum_{i \in I_{xy}} (R_{x,i} - \bar{R}_x)(R_{y,i} - \bar{R}_y)}{\sqrt{\sum_{i \in I_x}(R_{x,i} - \bar{R}_x)^2} \sqrt{\sum_{i \in I_y}(R_{y,i} - \bar{R}_y)^2}}
\]  

Where \( \text{sim}(x, y) \) denotes the modified cosine similarity between user \( x \) and user \( y \), \( I_{xy} \) denotes the concatenation set of items rated by user \( x \) and user \( y \), \( R_{x,i}, R_{y,i} \) denote the ratings of user \( x \) or user \( y \) for item \( i \), respectively, and \( \bar{R}_x, \bar{R}_y \) denote the average of ratings of user \( x \) or user \( y \) for the item, respectively.

In addition, for new users, due to the lack of a rating matrix to train the data, it is also necessary to incorporate a recommendation algorithm based on music attribute labels, containing the labels obtained through extensive data analysis, audio feature attribute labels extracted through FCGRNN, and self-labeled attributes in the dataset, as mentioned in the previous section.
2.3 Feature extraction

The hybrid recommendation model uses a total of four features: two audio features MFCC feature $M_j$ and Chroma feature $C_j$, lyrics feature $L_j$, and rating vector $S^{(u)}$ (which represents the rows of the rating matrix).

2.3.1 Audio Characterization

Audio Features We use $M_j$ to denote the MFCC feature of the $j$th music and $C_j$ to denote the Chroma feature of the $j$th music.

2.3.2 Characterization of the lyrics

The lyrics’ words were vectorized using the word embedding technique. The best-known word embedding is Word2Vec.

Therefore, we converted all the words to lowercase and pre-trained them with FastText to produce word vectors for each word $v$. However, for each song, we need to obtain the feature vectors for the lyrics of the song as a whole. We built a bag-of-words model by multiplying the word vectors of each word of each song by its TF-IDF value and then summing them up to take the average value, which is used as the feature vector of the lyrics of each of our music.

The importance of a word in a document in the corpus is determined by its TF-IDF value. The significance of the word is proportional to how often it appears in that document, i.e., word frequency. However, at the same time, it is inversely proportional to the frequency of its occurrence in the entire corpus, i.e., Inverse document frequency.

Word frequency refers to the frequency of a given word in the document. In order to prevent the frequency will being biased with the size of the document, the value will be normalized to the number of words, for word $t_i$ and document $d_j$, $tf_{ij}$ as shown in equation (6):

$$tf_{ij} = \frac{n_{ij}}{\sum_k n_{kj}}$$

(6)

In the above equation, $n_{ij}$ denotes the number of times a word $t_i$ appears in document $d_j$ and denominator $\sum_k n_{kj}$ denotes the total number of all words in document $d_j$.

The inverse document frequency indicates the importance of the word, $idf_i$ for a particular word $t_i$, by dividing the total number of documents by the total number of documents containing the word, and subsequently after taking the quotient in logarithmic form, as shown in Equation (7):

$$idf_i = \log \frac{|D|}{|D:i|+1}$$

(7)

$|D|$ denotes the total number of documents in the corpus, $|D:i|$ denotes the number of documents containing the word, and $|D:i|+1$ is used in the denominator to prevent the word from not being in the corpus, which would result in a denominator of zero.
Finally, the $tfidf_{ij}$ value of word $t_i$ for document $d_j$ can be expressed as shown in equation (8):

$$tfidf_{ij} = tf_{ij} \times idf_i$$  \hspace{1cm} (8)

As we can see from the formula, $tfidf_{ij}$ tends to keep the words that are more important to the document and filter to the familiar words.

We have computed the value of $tfidf$ for each word in the dataset of Million Song Dataset.

Moreover, now, the lyrics of the songs are our documents, and for lyrics $d_j$, their lyrics feature vector $L_j$ can be represented as shown in Eq. (9):

$$L_j = \sum_{i} tfidf_{ij} \times v_i / |d_j|$$  \hspace{1cm} (9)

$|d_j|$ denotes the number of words in the $j$th lyrics, and it can be seen by Eq. The feature vector of that lyric has a higher weight for the significant words in it.

However, the pre-trained corpus is not the same as our training data, so there are cases where words do not exist in the lyrics, but since there are fewer words that do not exist, we ignore the non-existent words. Finally, we created feature vectors that depict the lyrics for each song.

### 2.4 User Preferences and Music Latent Characterization Acquisition

The obtained dataset contains records related to the user’s listening to music, which can be regarded as implicit feedback; the number of times the same user repeatedly plays a particular song can be considered as the user’s favorite degree of the song, and the playing behavior can be regarded as an implicit rating. Using this method prevents the display of ratings to users and ensures that ratings are based on the user’s experience with the music, not just on the release date or singer. The reasons why a user never listens to a song are very varied, e.g., the user does not like it or has not discovered it yet.

In order to minimize the influence of non-subjective factors, such as the number of times a user plays a song because they are relatively free and the number of times another user plays a song because they are busy, this paper collects data on the number of times a user plays a song, and collects data on the number of times a user plays a song. In this paper, after gathering information about the number of times a user plays a song, we first do some pre-processing and convert it into a user rating for the music based on a specific way. The process is, first of all, the number of times the collected data is normalized, i.e., the music that is played the most often is taken as the benchmark, and the number of times other music is played is divided by the number of times the benchmark is played. The result of the previous step is multiplied by 30 and rounded up (the number of times 0 is still recorded as 0), which ensures that the value corresponding to the number of times played will only be in the range of 0 to 30. Table 1 shows the scoring rules table. The corresponding scores are formed by fully converting the play counts using the specific rules shown in Table 1.
A music personalization imputation method based on deep transfer learning

<table>
<thead>
<tr>
<th>Table 1. Grading criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Playback frequency</strong></td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1-2</td>
</tr>
<tr>
<td>3-6</td>
</tr>
<tr>
<td>7-13</td>
</tr>
<tr>
<td>14-20</td>
</tr>
<tr>
<td>21-30</td>
</tr>
</tbody>
</table>

After completing the above processing, a class of user-music matrices $R_{m \times n}$, denoted as (10), that derive the ratings corresponding to the music on the basis of the relative number of plays and with a specific level of sparsity can be obtained:

$$
R = \begin{bmatrix}
  r_{11} & r_{12} & \cdots & r_{1n} \\
  r_{21} & r_{22} & \cdots & r_{2n} \\
  \vdots & \vdots & \ddots & \vdots \\
  r_{m1} & r_{m2} & \cdots & r_{mn}
\end{bmatrix} = \begin{bmatrix}
  5 & 3 & \cdots & 1 \\
  2 & 0 & \cdots & 2 \\
  \vdots & \vdots & \ddots & \vdots \\
  0 & 5 & \cdots & 4
\end{bmatrix}
$$

(10)

The matrix decomposition method mentioned in the previous section is used to carry out specific processing of the matrix $R$, in order to ensure that the resulting regular objective function can be optimized, the stochastic gradient descent method is used, that is, the derivatives of the parameters are derived to determine the direction in which the objective function decreases the fastest, and then the variables are allowed to move along the direction until the point of the minimum value. Based on previous experience, the learning rate $\alpha$ and regularization parameters are set to 0.0002 and 0.02. After continuous optimization iterations, the user-potential factor matrix $P$ and the music-potential factor matrix $Q$ are obtained. Represented by (11) below:

$$
P = \begin{bmatrix}
  p_{11} & p_{12} & \cdots & p_{1k} \\
  p_{21} & p_{22} & \cdots & p_{2k} \\
  \vdots & \vdots & \ddots & \vdots \\
  p_{m1} & p_{m2} & \cdots & p_{mk}
\end{bmatrix}
Q^T = \begin{bmatrix}
  q_{11} & q_{12} & \cdots & q_{1n} \\
  q_{21} & q_{22} & \cdots & q_{2n} \\
  \vdots & \vdots & \ddots & \vdots \\
  q_{k1} & q_{k2} & \cdots & q_{kn}
\end{bmatrix}
$$

(11)

As can be seen, the decomposition operation can effectively reduce the matrix dimension, $P$ is the user preference model, which mainly reflects the user’s interest in the different characteristics of the music respectively. $Q^T$ is the music feature model, each column represents the characteristics of a particular piece of music.

3 Results and analysis

3.1 Music Recommendation Accuracy Analysis

In this paper, under the premise of rating prediction, for all 200 songs in the dataset, the predicted rating deviations of the multimodal music recommender system and audio-based unimodal music recommender system are compared.

When a user inputs the instant ratings of some songs, the user’s instant ratings are mapped to that user’s ratings for the model. Then, the user’s interest in all 200 songs is finally obtained by similarity
analysis with the multimodal feature matrix of all 200 songs in the data set. The user’s interest degree for all 200 songs is converted into the user’s predicted rating for all 200 songs for accuracy validation using the premise of rating prediction.

Here, linear regression is used to convert interest degree to predicted ratings, and the actual operation is done by building a linear regression model using linear_model in the sci-kit-learn library. When constructing the dataset, the user’s absolute ratings of 200 songs are required to use cross-validation, and only a portion of the values are retained as rated songs in each round of cross-validation. The data in the training set is the user’s interest degree in rated songs, the labels are the user’s ratings of the rated songs, and the test set is the user’s interest degree in all 200 songs. The fit method of the model uses the training set as a parameter to finish the training. Finally, the prediction method is used to predict the ratings of all 200 songs in the test set to convert interest levels into predicted ratings. Figure 2 shows the deviation of predicted ratings from the actual ratings.

After obtaining the predicted user ratings for all 200 songs, it is possible to determine the difference between the predicted ratings and the actual ratings of the 200 songs. As shown in Fig. 2, this figure plots the deviation of predicted ratings from accurate ratings for the recommendation method of this paper. It can be seen that the predicted ratings of the music recommendation method in this paper are almost fitted to the actual ratings, and the deviation of their ratings basically stays within (0, 2.5) points, which is almost without much loss in accuracy.

![Figure 2. The deviation between the prediction score and the actual score](image)

In order to reflect the value of the results, Table 2 gives an analysis of the deviation of the predicted scores from the actual scores for the two recommender systems mentioned above. It can be seen that the number of songs with a deviation of the predicted scores of the recommender system of this paper of 2 points or less all account for 88% of the total number of songs. Through the analysis of the latter two experiments, it can be concluded that the present method performs excellently in terms of accuracy, regardless of the form of recommendation.

<table>
<thead>
<tr>
<th>Deviation</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>≤2</th>
<th>&gt;3</th>
</tr>
</thead>
<tbody>
<tr>
<td>CB</td>
<td>0.24</td>
<td>0.25</td>
<td>0.13</td>
<td>0.26</td>
<td>0.21</td>
</tr>
<tr>
<td>CF</td>
<td>0.39</td>
<td>0.37</td>
<td>0.11</td>
<td>0.37</td>
<td>0.13</td>
</tr>
<tr>
<td>Context-Pre</td>
<td>0.32</td>
<td>0.31</td>
<td>0.18</td>
<td>0.19</td>
<td>0.19</td>
</tr>
<tr>
<td>Context-Post</td>
<td>0.28</td>
<td>0.29</td>
<td>0.22</td>
<td>0.26</td>
<td>0.18</td>
</tr>
<tr>
<td>Depth learning</td>
<td>0.19</td>
<td>0.15</td>
<td>0.07</td>
<td>0.12</td>
<td>0.12</td>
</tr>
</tbody>
</table>
3.2 Performance Analysis of Hit Rate, Recall Rate, and NDCG Metrics

The model of this paper is compared with CB, CF, Context-Pre, and Context-Post models in terms of HR, Recall, and NDCG performance. Fig. 3 shows the comparison between the recommendation method for this topic and other recommendation methods in terms of the Hit Rate (HR) metric. Because the hit rate is calculated as the proportion of users hit to the total number of users, the longer the recommendation list is, the higher the probability of hit, so increasing the list length N will improve the HR value. When the length of the recommended list N is taken as 5, 7, 10, and 15, the hit rate is 0.1178, 0.1545, 0.1889, and 0.1981, respectively, as can be seen from the bar chart. This paper’s recommendation method maintains a relatively better level in the evaluation index of the hit rate, which indicates that the technique can provide users with a relatively accurate personalized recommendation list. At the same time, the deep transfer learning recommendation method makes full use of the information in the review text. It adopts the deep transfer learning technique not only to better portray the user preference characteristics and song characteristics, which makes the recommendation method in this paper perform well in the hit rate. In particular, when the length of the recommendation list N is taken as 5, the recommendation method in this paper improves by 41.72% over CB. When the length of the recommendation list is set to 7, it improves by 42.8% over Context-Pre.

![Figure 3. HR evaluation indicators for each recommendation](image)

Secondly, Fig. 4 shows the comparison of this paper’s recommendation method with other recommendation methods in terms of recall evaluation metrics. The coverage rate is 0.1742, 0.2332, 0.2985, and 0.4565 when the length of recommendation list N is taken as 5, 7, 10, and 15, respectively. The coverage rate is relatively low when the length of recommendation list N is taken as 5. This is because the coverage rate is positively correlated with the length of the recommendation list. It improves as the length of the recommendation list increases. Still, in general, this paper’s recommendation method maintains the leading edge under all recommendation list lengths, indicating that the recommendation list of the deep transfer migration learning recommendation method covers more songs. List lengths indicate that the recommendation lists of the deep-pass migration learning recommendation method cover more songs. In particular, there is an improvement of 263% over CF when the recommendation list length N is taken as 15. From the table, we can see that Context-Post and Context-Pre methods perform poorly in terms of recall, which is due to the fact that both types of methods adopt the filtering idea. The filtering means is simple and rough, leading to the fact that the songs can only appear in the recommendation list if they meet specific scenarios, which leads to low recall. The recommendation method in this paper not only can take into account having a better hit rate but, at the same time, ensure that it has a high coverage rate, indicating that the method has a clear, comprehensive advantage, that is, in as many users as possible to be predicted hits at the same
time, but also make the recommendation results exposed to a more significant amount of the song. Therefore, the advantage of the coverage rate makes this paper’s recommendation method more commercially viable.

Figure 4. Recall evaluation index comparison

Fig. 5 shows the comparison of NDCG evaluation indexes between the recommended method of this paper and other recommended methods. The NDCG is 0.1389, 0.2332, 0.3289, 0.4169, respectively when the length of recommendation list N is taken as 5, 7, 10, 15. The CF recommendation method performs better in the NDCG evaluation index compared to HR and Recall, and the Context-Post recommendation method and Context-Pre recommendation method perform about the same. The Deep Transfer Migration Learning recommendation method maintains its advantage in each recommendation list length because the feature vectors extracted by this method using deep learning technology are more accurate, and the integration of situational features into the training model can provide users with more personalized recommended songs and increase the gain of recommended songs. In particular, the recommendation method in this paper improves by 75.5% over CB when the recommendation list length N is taken as 10.

Figure 5. NDCG evaluation index comparison

In conclusion, the experimental results illustrate that the deep transfer migration learning recommendation method obtains the relative best evaluation results in HR, Recall, and NDCG compared with other recommendation methods, indicating that the technique has relatively good advantages in algorithmic accuracy, result diversity, and system robustness. In particular, the benefit of the deep transfer migration learning recommendation method is more apparent when the recommendation list is longer.
3.3 Analysis of differences in musical similarity

In this paper, we compare the features of different music using Matlab to calculate their similarity. We compared 20 songs from the same album with the same musicians to calculate their similarity. Then we selected 20 songs from different albums with different musicians to calculate the similarity between them. Table 3 shows the comparison data for music.

It can be seen that the mean value of the comparison between the identical albums is 0.702559, while the mean value of the comparison between different albums is 0.666933. There is a significant difference between the two. It can be observed that the music similarity of a single album is significantly higher than the result of comparing different albums. If a user likes this person’s songs, he is likely to enjoy the songs of the same musician in the same album. Still, suppose a user likes the songs of the same musician. In that case, the user has likely appreciated the musician’s works, so it is proved that the similarity between some of the music that the user likes is relatively high, so we can recommend the songs that have not been appreciated by the user to him, and achieve the purpose of humanized music recommendation. This also confirms the effectiveness of the music recommendation method in this paper.

<table>
<thead>
<tr>
<th>Table 3. Music comparison data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Same album comparison</td>
</tr>
<tr>
<td>------------------------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>13</td>
</tr>
<tr>
<td>14</td>
</tr>
<tr>
<td>15</td>
</tr>
<tr>
<td>16</td>
</tr>
<tr>
<td>17</td>
</tr>
<tr>
<td>18</td>
</tr>
<tr>
<td>19</td>
</tr>
<tr>
<td>20</td>
</tr>
<tr>
<td>Mean</td>
</tr>
</tbody>
</table>

4 Conclusion

This paper constructs a music-personalized recommendation algorithm based on the deep transfer migration learning model and studies the superiority of this paper’s model over other models in music
recommendation performance based on accuracy, hit rate, recall rate, rating bias, similarity, and other metrics. Here are the main conclusions:

The predicted scores of this paper’s music recommendation method are almost fitted with the actual scores, and the number of songs whose predicted scores deviate within 2 points in this paper’s recommendation system all account for 88% of the total number of songs, which is superior to other models.

In terms of hit rate, when the recommended list length N is taken as 5, 7, 10, 15, its hit rate is 0.1199, 0.1591, 0.1809, and 0.2021, respectively, and this paper’s recommendation method maintains a relatively better level in the evaluation index of hit rate. On the evaluation index of recall rate, when the recommended list length N is taken as 5, 7, 10, 15, its coverage rate is 0.1742, 0.2332, 0.2985, and 0.4565, respectively, which maintains a leading advantage over other models. The length of each recommendation list has been retained by the deep transfer migration learning recommendation method when it comes to NDCG evaluation metrics.

In terms of the similarity difference of music, the average value of the comparison of the same album is 0.702559, while the average value of the comparison between different albums is 0.666933, which all maintain a high similarity, verifying the effectiveness of the method of this paper in music recommendation.

References


