applied sciences

MDPI

Article

Deep Collaborative Recommendation Algorithm Based on Attention Mechanism

Can Cui [1,2], Jiwei Qin [1,2,*] and Qiulin Ren [1,2]

[1] School of Information Science and Engineering, Xinjiang University, Urumqi 830046, China
[2] Key Laboratory of Signal Detection and Processing, Xinjiang Uygur Autonomous Region, Xinjiang University, Urumqi 830046, China
* Correspondence: jwqin_xju@163.com

Abstract: Representation learning-based collaborative filtering (CF) methods address the linear relationship of user-items with dot products and cannot study the latent nonlinear relationship applied to implicit feedback. Matching function learning-based CF methods directly learn the complicated mapping functions that map user-item pairs to matching scores, which has limitations in identifying low-rank relationships. To this end, we propose a deep collaborative recommendation algorithm based on attention mechanism (DACR). First, before the user-item representations are input into the DNNs, we utilize the attention mechanism to adaptively assign different weights to the user-item representations, which captures the hidden information in implicit feedback. After that, we input the user-item representations with corresponding weights into the representation learning and matching function learning modules. Finally, we concatenate the prediction vectors learned from different dimensions to predict the matching scores. The results show that we can improve the expression ability of the model while taking into account not only the nonlinear information hidden in implicit feedback, but also the low-rank relationships of user-item pairs to obtain more accurate predictions. Through detailed experiments on two datasets, we find that the ranking capability of the DACR model is enhanced compared with other baseline models, and the evaluation metrics HR and NDCG of DACR are increased by 0.88–1.19% and 0.65–1.15%, respectively.

Keywords: attention mechanism; recommendation system; implicit feedback; representation learning; matching function learning

check for updates

Citation: Cui, C.; Qin, J.; Ren, Q. Deep Collaborative Recommendation Algorithm Based on Attention Mechanism. *Appl. Sci.* **2022**, *12*, 10594. https://doi.org/10.3390/app122010594

Received: 7 September 2022
Accepted: 3 October 2022
Published: 20 October 2022

1. Introduction

With the rapid growth of various network platforms, the explosive growth of information has made it difficult for users to rapidly seek targets. To alleviate the information overload problem, recommendation systems have developed rapidly. Classical CF recommendation method is widely used to alleviate information overload. CF takes the historical behaviour data of users as input to calculate the similarity between them, so as to predict the preference of users [1–3]. However, traditional CF cannot effectively solve the data sparsity. To this end, matrix factorization (MF) assumes that users have some latent factors with itemss, so establishes a low-dimensional dense representation space for users and items and then compares them in the common vector space. MF uses the dot product to acquire the similarity for user-items and then recommend items with high similarity to target users [4].

Generally, there are complex nonlinear mapping relationships between the original representation space and low-dimensional dense representation space for user-items. To better research the nonlinear relationships, researchers learn from the ideas of deep learning [5] and machine learning [6], and apply them to recommendation systems [7,8]. Huang et al. [9] proposed the Deep Structured Semantic Models (DSSM), which utilizes a DNN to express search keywords and clicked text titles as low-dimensional dense representation vectors and calculates the distance between the two dense representation vectors through cosine similarity. Inspired by DSSM, Xue et al. [10] proposed a Deep Matrix Factorization (DMF), which uses two-pathway DNNs instead of linear embedding as interaction function, learns the common low-dimensional dense representation space for user-items and makes use of the inner product as a predictive function to make rating predictions. Not only can DNNs learn more accurate user-item representations, but since DNNs can theoretically fit any continuous function, they are well-suited for learning complex matching functions [11,12]. For instance, He et al. proposed the NeuMF model which takes the concatenation of user-item representations as the input of MLP and uses a multilayer neural network as the interaction function [13]. Although DNNs are characterized by high capacity and nonlinearity, an MLP is difficult in capturing low-rank relationships as revealed by Beutel et al. [14]. The CF methods based on matching function learning use an MLP to explore user-item interaction.

Generally, humans cannot process the whole signal at once but can only selectively focus on part of the signal content, hence, the attention mechanism is derived [15,16]. Attention mechanisms have been widely applied to many fields [17] such as recommendation systems [18,19] that start from data processing. The models apply attention mechanism to automatically find the key points of the input vectors and distribute corresponding weights to the input to optimize the traditional recommendation model. For instance, Xiao et al. proposed the AFM model [20]. By introducing the attention mechanism into the enhanced FM model, AFM learns by assigning weights to different feature combinations and uses the attention network to update the weights to achieve performance improvements.

In brief, the methods based on MF are good at learning the low-rank relationships for user-items, but they are difficult to capture the nonlinear relationships for user-items due to using linear interaction functions. The methods based on MLP can capture the nonlinear relationship but have disadvantages in learning low-rank relationships. These two methods have different advantages in different perspectives. In order to effectively coordinate the flexibility of learning complex matching functions and user-item low-rank relationships, we propose a Deep Collaborative Recommendation Algorithm based on Attention Mechanism (DACR). Specifically, we use these two types of CF methods to obtain user-item representations from different perspectives, and then concatenate the representations learned from the two methods to integrate them into a common deep collaborative filtering framework, so as to obtain a more robust user-item pair joint representation for more accurate prediction. In addition, before inputting the user-item representation vectors into the MLP, we input them into the attention layer, which adaptively assigns weights through the attention mechanism to focus their attention on specific parts. When the latent representation vector of user-items is compressed into a prediction vector, different parts make different contributions. The higher the weight is, the more information the corresponding factor has to recommend. Hence, the attention layer can better capture the hidden information in implicit feedback and improve the expression ability of DNN.

Overall, our main contributions are as follows:

1. We propose a ranking recommendation model, DACR, with the characteristics of fusion MLP and traditional MF, which learns the low-rank relationship for user-items and the mapping functions between user-item representations and matching scores from different perspectives.
2. We integrate the attention mechanism to capture the hidden information within implicit feedback.
3. We research the effect of the dimension of the prediction vectors and the negative sampling ratio on personalized ranking performance through detailed experiments.
4. We conduct numerous and detailed experiments on two public datasets to demonstrate the effectiveness of the DACR model and discuss some possibilities for improvement.

The rest of the paper is structured as follows: We review the related works in Section 2 and present our method in Section 3, including details of the network structure and processing of the input data. Section 4 presents the experimental results, and Section 5 concludes the paper and discusses future work.

2. Related Work

In this part, we introduce different types of collaborative filtering and analyze the features of them. In order to understand, the second part illustrates the practical application of the CF algorithm with examples.

2.1. Collaborative Filtering with Implicit Feedback

Since users do not have to express their preferences explicitly, most users will not rate items. The explicit feedback data (such as ratings) are harder to collect than implicit feedback. Therefore, implicit feedback of users such as clicks, viewing times, collects, or purchase history is easier to obtain on a large scale than explicit feedback. Based on this situation, it is especially significant to apply implicit feedback data to the recommendation algorithms [21,22]. Many basic algorithms have studied collaborative filtering with implicit feedback. ALS [23] and SVD++ [24] decompose the binary interaction matrix and suppose that unobserved interaction means the user does not like the item. Some other algorithms assume that users do not necessarily dislike unobserved items. For example, Rendle et al. believe that users prefer items they choose rather than unobserved items [25]. The model we propose in this paper is collaborative filtering without auxiliary data.

2.2. Collaborative Filtering Based on Representation Learning

The MF has been continuously optimized and developed since Funk-SVD was proposed [26–28]. The main idea of MF is to map users and items to the same latent space and then utilize linear interaction to calculate the similarity for user-items. Meanwhile, deep learning methods have acquired excellent results in recommendation systems. For instance, Wang et al. proposed the CDL model based on DNNs and CF, which represent content information and rating information, respectively [29]. Sedhain et al. proposed the AutoRec model which uses a single hidden layer auto-encoder to generalize user or item ratings and then learn user and item representations separately [30]. Xue et al. proposed the DMF comprehensively considering users' explicit ratings and implicit feedback for items. The DMF model maps user-item feature vectors to the common low-dimensional representation space through DNNs [10]. Ahmadian et al. [31] utilized a deep neural network for the representations of trust relationships and tag information and employed sparse auto-encoders to capture latent features from user–user trust relationships and user tag matrices. The similarity values among users are calculated with these latent features and make predictions and recommendations to target users. On the basis of this paper, in reference [32], deep sparse auto-encoders are applied to learn the latent features of users from three different input data (i.e., rating, trust relationships and tag), and the particle swarm optimization algorithm (PSO) is used to obtain the optimal weight when using these three latent features to calculate the similarity values. Yengikand et al. [14] used MLP and stacked auto-encoder network (SAN) to seek user and item latent factors from the historical interaction matrix and integrate the obtained latent user preferences and item features into the rating prediction module of the DSAPSO model. One study [33] utilized a probability model to determine whether user representation vectors are reliable and enhanced the unreliable user representation vectors by the implicit rating. The enhanced reliable rating matrix and trust matrix serve as the input of the deep sparse auto-encoder to obtain the latent features of users, so as to calculate the similarity values between users and make recommendations. The above methods obtain the user-item representations in different ways and apply linear interactions, such as dot products to predict that limits the model to learn nonlinear relationships between user-item representations and matching scores.

2.3. Collaborative Filtering Based on Matching Function Learning

Most MF methods use linear interactions, which limit the expression ability of the model, while recommendation models combined with neural networks directly learn interaction functions from historical data to obtain better recommendation prediction results. To overcome the shortcomings of MLPs in seizing low-rank relations, NeuMF [13] combines the MF and MLP in a unified framework and replaces the linear interaction in ordinary MFs with neural networks, which can directly learn the matching functions. Moreover, Bai et al. [34] proposed the NNCF model, which integrates neighborhood information into the neural collaborative filtering method to supplement user-item interaction data and enable the model to effectively capture local information. In addition to NeuMF, some models use auxiliary data learning matching functions. For example, Cheng et al. used the LR and MLP to learn matching functions from input feature vectors and user-item classification features [35]. Compared to the concatenation in NeuMF, He et al. [36] used the outer product and a two-dimensional convolutional layer to learn the joint representations of user-item pairs. Zhang et al. [37] proposed the CoupledCF model to solve the problem of existing collaborative filtering methods that ignore the rich coupling relationships of user-items and utilized the coupling relationship learning network based on a CNN to deeply explore the explicit and implicit coupling relationships of user-items. These methods use nonlinear interaction such as DNNs to fit the relationships between user-item representations and matching scores, but it is difficult to learn the low-rank relationship of the user-item interaction matrix.

2.4. Attention Mechanism in Recommendation System

Attention mechanisms have performed extremely well in many tasks, such as recommendation systems. Adding attention mechanisms to the traditional recommendation model has also achieved certain results. For instance, Chen et al. proposed the ACF model [38], which uses the attention mechanism in collaborative filtering. Cheng et al. assigned a corresponding attention weight vector to each user-item pair by using attention mechanism [39]. Xiao et al. designed an AFM model that utilizes the attention network to analyse the significance of each cross combination of features and reflects the model's different attention to different feature interactions by assigning different weights [20]. In [40], an attention mechanism is introduced in the deep convolutional neural network to extract latent features of users based on reviews, and the other is adopted in a user-item rating matrix to capture latent features of items. With these two valid attention mechanisms, users can acquire more precise recommendations. In [41], Xi et al. used a back-propagation network and an attention mechanism to learn latent features of target users and their nearest neighbors, so as to improve recommendation precision. The higher the weight assigned to a factor of the input vector by the attention mechanism, the greater the amount of information contained in the corresponding factors. In consequence, an attention mechanism can be utilized to capture the hidden information within implicit feedback.

3. Proposed DACR Model

In the third part, we first display the general process of the DACR framework. Second, we describe the attention representation learning and attention matching function learning modules. Finally, we introduce the implementation process and training details of the DACR model.

3.1. General Process

The general process of DACR is visualized in Figure 1. In the figure, the representation learning module and matching function learning module learn prediction vectors from the input layer separately. In advance, the attention layer is used to process the learned embedding vectors, respectively. We fuse the two modules by inputting the concatenated prediction vectors into the fully connected layer to obtain the matching scores. Finally, we map the matching scores to probabilities by which we rank the items in the list.

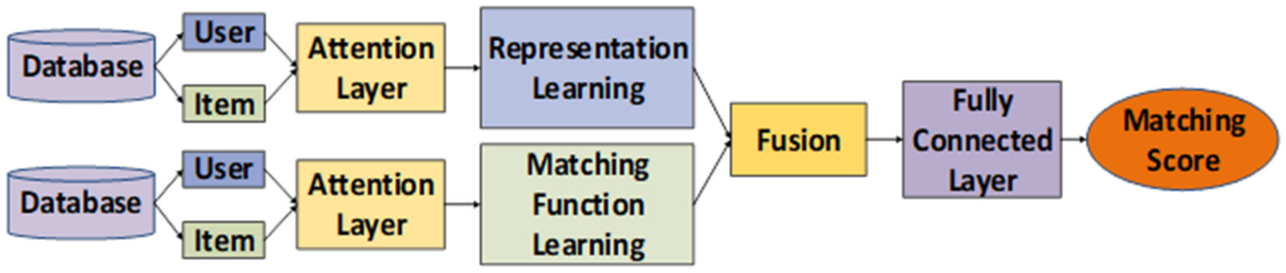

Figure 1. The general process of DACR.

3.2. Attention Representation Learning

Y = interaction matrix of user-item
u = user
I = vector

CF methods based on representation learning pay more attention to learning the representation function; meanwhile, the matching function part is usually set to an ordinary linear function. Assume that users and items are mapped to a common latent vector space so that user and item representations can be compared in the common space. We do not add auxiliary data, use only implicit feedback data, apply the interaction matrix Y of user-item as input to the model. The original representation vector of user u and item i are the u-th row and the i-th column vector of Y, respectively, i.e., $v_u^U = Y_{u*}$, $v_i^I = Y_{*_i}$.

m = dimension of the encoder of the attention weight layer

user u utilizes the decoder of the attention layer to calculate the vector vd ∈ Rm×1 for Yu∗ ∈ R1xN

input = SoftMax(MLP layer transposed * input + bias vector) * vd

The attention representation learning module combines the MLP layer and the attention layer to learn the latent representations for users-items. Assume that the dimension of the encoder of the attention weight layer is m, that is, user u utilizes the decoder of the attention layer to calculate the vector $v_d \in R^{m\times 1}$ for $Y_{u*} \in R^{1\times N}$, as shown the following equation:

$$a_0 = W_0^T Y_{u*}^T \alpha = \delta(W^T a_0 + b) v_d = \alpha \odot a_0 \tag{1}$$

where $W_0 \in R^{N\times m}$ and $W \in R^{m\times m}$ separately denote the weight matrix of the encoder of the attention layer and the MLP layer, a_0 denotes the input of the attention layer, δ denotes the softmax activation function, b denotes the bias vector of the MLP layer, α denotes the attention ratio of the attention layer. We implement the representation learning part based on MLP as follows:

a'0 = concatenation a0 and vd
It's the first MLP input

Wx = weight matrix
ax = output of the x-th layer
bx = bias vector
teta = ReLU function

ax = ReLU(weight matrix transposed * ax-1 + bias)

$$\begin{aligned} a_0' &= \begin{bmatrix} a_0 \\ v_d \end{bmatrix} \\ a_1 &= \theta(W_1^T a_0' + b_1) \\ a_2 &= \theta(W_2^T a_1 + b_2) \\ &\cdots\cdots \\ p_u = a_X &= \theta(W_X^T a_{X-1} + b_X) \end{aligned} \tag{2}$$

where $a_0' \in R^{2m\times 1}$ denotes the MLP input, W_x, b_x, and a_x denote the weight matrix, bias vector, and output of the x-th layer, respectively, $\theta(\cdot)$ denotes the activation function ReLU. The latent representation q_i of item i is derived as above. Compared to the existing collaborative filtering method based on representation learning, the implementation of the user-item interaction part is as follows:

$$\begin{aligned} a_Y^{rl} &= p_u \odot q_i \\ \hat{y}_{ui} &= \sigma(W_{out}^T a_Y^{rl}) \end{aligned} \tag{3}$$

where, and denote the prediction vector, weight matrix, and sigmoid activation function, respectively. The element-wise product method and parameter neural network layer are used to replace the dot product or cosine similarity. Not only can it capture the low-rank relationship for user-items, but the method is also more comprehensive because of the different potential dimensions and nonlinear mapping function. In summary, the attention representation learning module is implemented by Formulas (1)–(3), which are reflected in the left dashed box in Figure 2.

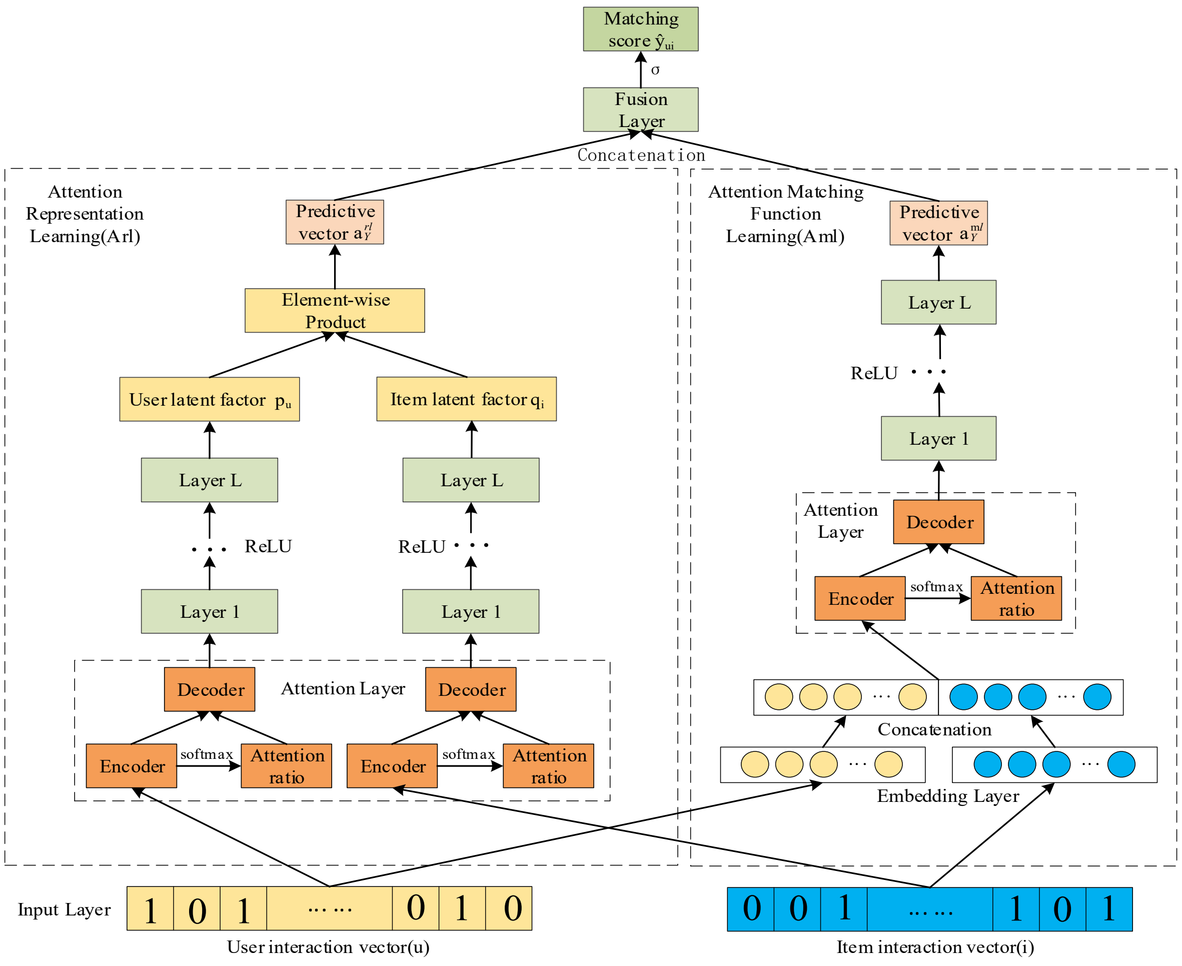
Matching score $\hat{y}_{ui}$
σ
Fusion Layer
Concatenation
Attention Representation Learning(Arl)
Predictive vector a_Y^{rl}
Element-wise Product
User latent factor p_u
Item latent factor q_i
Layer L
ReLU
Layer 1
Decoder
Attention Layer
Encoder
softmax
Attention ratio
Attention Matching Function Learning(Aml)
Predictive vector a_Y^{ml}
Concatenation
Embedding Layer
Input Layer
User interaction vector(u)
Item interaction vector(i)

Figure 2. The framework of DACR.

3.3. Matching Function Learning

Matching function learning-based CF methods not only include the matching function learning part but also have the representation learning part. As the initial representations of users and items given by v_u^U and v_i^I, respectively, have very high dimensionality and sparseness, it is difficult for the model to explore the matching function. Hence, collaborative filtering methods based on matching function learning generally apply linear embedding layers to obtain low-dimensional latent representations for users-items, which is more effective in learning the matching function.

The attention matching function learning module of the DACR model combines the MLP and the attention layer and takes the interaction matrix Y as input. Assuming that the dimension of the encoder of the attention layer is h, we apply the attention layer to the initial representations $Y_{u*} \in R^{1\times N}$ and $Y_{*_i} \in R^{M\times 1}$ to calculate the decoding vector as shown in Equation (4):

$$\begin{aligned} p_u &= P^T Y_{u*} \\ q_i &= Q^T Y_{*i} \\ a_0 &= \begin{bmatrix} p_u \\ q_i \end{bmatrix} \\ \alpha &= \delta(W^T a_0 + b) \\ v_d &= \alpha \odot a_0 \end{aligned} \tag{4}$$

where p and Q denote the parameter matrix of the linear embedding layer, p_u and q_i denote the low-dimensional latent representation vectors for users and items, respectively. Overall, the derivation of this part can be defined as Equation (5):

$$
\begin{aligned}
a'_0 &= \begin{bmatrix} a_0 \\ v_d \end{bmatrix} \\
a_1 &= \theta(W_1^T a'_0 + b_1) \\
a_2 &= \theta(W_2^T a_1 + b_2) \\
&\cdots\cdots \\
a_Y^{ml} = a_Y &= \theta(W_Y^T a_{Y-1} + b_Y) \\
\hat{y}_{ui} &= \sigma(W_{out}^T a_Y^{ml})
\end{aligned}
\tag{5}
$$

where a'_0 denotes the input of the MLP layer, W_x, b_x, a_x denote the weight matrix, bias vector, and output of the MLP layer, respectively, and a_Y^{ml} denotes the prediction vector. In summary, the matching function learning module implements the representation learning functions $f(\cdot)$ and $g(\cdot)$ through the linear embedding layer and learns the latent representation vectors p_u and q_i for the user and item, respectively. We concatenate the latent representation vectors of users and items and then input the aggregated vectors into the attention layer. Next, the output prediction vector is passed through the fully connected layer to calculate the matching scores. This process is realized by Equations (4) and (5), that is, the right dashed box in Figure 2.

3.4. Fusion

Sections 3.2 and 3.3 of this chapter introduce the two sub-modules Arl and Aml of the DACR model, as well as the details of the implementation of the two modules. To enhance the two modules, it is crucial to determine a suitable strategy to fuse the modules. The most commonly used fusion strategy is concatenation and inputting the concatenated joint representation to the fully connected layer. As described in Sections 3.2 and 3.3, the Arl module and Aml module respectively generate prediction vectors. The two sub-modules are experts in learning information of different dimensions. Therefore, we concatenate the two prediction vectors learned from different dimensions and obtain a more comprehensive joint representation of user-item pairs. In summary, the output after fusing the two sub-modules is defined by Equation (6):

$$
\hat{y}_{ui} = \sigma(W_{out}^T \begin{bmatrix} \mathrm{a}_Y^{rl} \\ \mathrm{a}_Y^{ml} \end{bmatrix}) \tag{6}
$$

3.5. Other Details

3.5.1. Inout Data of DACR

Assuming that there are M users and N items in the system [10], the interaction matrix for user and item $Y \in R^{M \times N}$ is established by extracting implicit feedback (such as watching or purchasing) based on the historical behaviour data as follows, where $p_{ui} = 1$ denotes u interacting with i; otherwise, $p_{ui} = 0$.

$$
y_{ui} = \begin{cases} 1, \; u \; interact \; with \; i; \\ \quad 0, \; others \end{cases} \tag{7}
$$

While implicit feedback is easier to collect, there are two main problems. Firstly, user interactions with items ($p_{ui} = 1$) can only indirectly reflect user preferences; that is, they cannot identify the user's preference extent. Secondly, the unobserved interaction ($p_{ui} = 0$) is not equivalent to the user disliking the item at all. This is simply because the number of items in the system is large, and the user does not have an opportunity to interact with the item. In particular, with the second problem, the model faces a huge challenge in extracting information from implicit data. To avoid the bias caused by the problem, there are two solutions. One is to treat all interactions with a value of 0 as negative instances, and the other is to sample some interactions with a value of 0 as negative instances [42]. We adopt the second method, sampling from interactions that have never been observed as negative instances.

3.5.2. Processing Input Data

Recommendation problems with explicit feedback are usually rating prediction problems. This kind of problem usually predicts the missing value of the rating matrix and recommends items with high ratings to the user based on the predicted value. Recommendation problems with implicit feedback are usually expressed as interaction prediction problems, i.e., calculating the missing value of the interaction matrix to predict whether an unobserved interaction will occur. However, implicit feedback data are binary and discrete. The binary classification problem cannot be further evolved into a prediction and recommendation problem. Therefore, the DACR module addresses the probability of Y and assumes that y_{ui} obeys a Bernoulli distribution:

$$P(y_{ui} = k|p_{ui}) = \begin{cases} 1 - p_{ui}, \ k = 0; \\ p_{ui}, \ k = 1 \end{cases} = p_{ui}^{k}(1 - p_{ui})^{1-k} \tag{8}$$

where $\hat{P}_{ui}$ is the probability that y_{ui} is equal to 1, which can also be understood as the probability that u matches i, i.e., $\hat{P}_{ui} = 1$ means that u matches i completely, and $\hat{P}_{ui} = 0$ means that u does not matches i completely. We do not model the binary and discrete y_{ui} but model the continuous value p_{ui} and transform the binary interactive prediction into a matching score prediction.

3.5.3. Loss Function

Model-based methods usually assume that the data can be generated by the underlying model as $\hat{y}_{ui} = f\,(u, i|\theta)'$, where $\hat{y}_{ui}$ denotes the prediction of y_{ui}, θ denotes model parameters, and f denotes the mapping function. The definition of the function f and the estimation of the parameters θ are related to the learning ability of the model.

For parameter estimation, the parameters are usually estimated by optimizing the objective function. There are generally two types of objective functions, namely point-wise loss [23] and pair-wise loss [25]. Point-wise loss, especially the squared loss function, is mostly used in CF with explicit feedback [43]. However, the squared loss does not work with implicit feedback because it assumes that the error between the real rating and predicted rating obeys a normal distribution. We assume that y_{ui} obeys a Bernoulli distribution in implicit feedback, i.e., $y_{ui} \sim Bern(p_{ui})$. Since p_{ui} is also understood as the probability that u matches i, the interactive prediction is simplified into a matching score prediction. Therefore, using the maximum likelihood estimation to determine the model parameters θ is the same as minimizing the error function between y_{ui} and $\hat{y}_{ui}$. By replacing p_{ui} with $\hat{y}_{ui}$ in Equation (8), we can define the likelihood function [13] as:

$$\begin{aligned} \mathrm{L}(\theta) &= \prod_{(u,i)\in y^+ \cup y^-} P(y_{ui}|\theta) \\ &= \prod_{(u,i)\in y^+ \cup y^-} \hat{y}_{ui}{}^{y_{ui}}(1 - \hat{y}_{ui})^{1-y_{ui}} \end{aligned} \tag{9}$$

where y^+ represents the set with the value 1 in Y. y^- represents the set of samples with a value of 0 in Y, i.e., negative samples. After taking the negative logarithm of the likelihood function, we obtain the following.

$$L_{BCE} = -\sum_{(u,i)\in y^+ \cup y^-} y_{ui} \log \hat{y}_{ui} + (1 - y_{ui}) \log(1 - \hat{y}_{ui}) \tag{10}$$

In conclusion, we use the binary cross-entropy loss function which is good at learning parameters from implicit feedback as the objective function [44].

3.5.4. Training of DACR

We present the model proposed in this paper in order to better demonstrate the specific process shown in Algorithm 1.

Algorithm 1. The training of DACR

```
Input: $Y$: user- item interaction matrix; $y^+$ all the observed interactions in $Y$; $y^-$: the sampled
Unobserved interactions; n: epochs.
Output: θ: the model parameters
Randomly initialize the model parameters θ with a Gaussian distribution.
For all epochs = 1 to n do
For all $(u,i) \in y^+ \cup y^-$ do
            Calculate the predictive vector of $a_r^{rl}$ by Equations (1)–(3)
            Calculate the predictive vector of $a_r^{ml}$ by Equations (4) and (5)
            Calculate the predicted interaction $\hat{y}_{ui}$ by Equation (6)
      End for
      Obtain the loss $L_{BCE}$ by Equation (10)
      Optimize $L_{BCE}$
End for
Return θ
```

4. Experiment

In this chapter, we first introduce some hyperparameter settings for our experiments. Then, we answer the following questions with a large amount of real experimental data:

RQ1: How does DACR perform better than other methods?

RQ2: How does the negative sample ratio of the model affect DACR?

RQ3: How do the dimensions of prediction vectors affect DACR?

RQ4: How do different modules of the DACR model affect the recommend results?

4.1. Experimental Settings

4.1.1. Datasets

We perform experiments on three common available real-world datasets: MovieLens 1M (ml-1m), Lastfm and Amazon music (AMusic). The three datasets are acquired from the following sources: the MovieLens 1M is collected from the MovieLens website and has been widely used in movie recommendation, Lastfm is a set of sequences about users listening to songs from the online system of Last.fm, which comes from an internet radio and music community, and AMusic contains users' rating data in Amazon. All three datasets are preprocessed, the selected users are those who have rated at least 20 items, and the selected items are those that have been rated by at least five users. Table 1 lists the statistics of the three datasets, and the interaction matrix Y of each dataset is constructed by Equation (7).

Table 1. Statistics of the datasets.

Statistics	**MovieLens 1M**	**Lastfm**	**AMusic**
Of users	6040	1741	1776
Of items	3706	2665	12,929
Of ratings	1,000,209	69,149	46,087
Sparsity	0.9553	0.9851	0.9980

4.1.2. Comparison Methods

In the following, a brief description of methods compared with DACR is presented to facilitate analysis of the results in the following sections.

ItemPop is a non-personalized method that pays no attention to the characteristics of each user and historical behaviors, but simply and rudely recommends items with more sales to users. ItemPop is the baseline method of the recommendation system.

ItemKNN [45] is an item-based CF method that calculates the similarity between items based on historical data and then predicts user ratings.

BPR [25] concentrates on leveraging implicit feedback and using pairwise loss functions to improve personalized ranking performance.

DMF [10] proposes an MF model based on DNNs, establishes a user-item matrix with ratings and implicit feedback, proposes a deep learning framework to learn a common low-dimensional space for user-item representations, and designs a novel loss function based on binary cross-entropy.

DeepCF [46] combines the advantage of representation learning methods for dealing with low-rank relationships and the ability of matching function methods to learn nonlinear relationships to promote ranking performance.

DeepUCF+a [47] combines linear and nonlinear DNNs to build relationships between users and adds an attention network to distinguish the historical user importance of items.

4.1.3. Parameter Settings

Firstly, the learning rate for all methods is set to 0.001. Then, for each method, the weight of all interactions observed by each user is set to 1, that is, the weight of each observed interaction value in the input interaction matrix is the same. Finally, we sample 3–6 negative instances for each positive instance, that is, the negative sample ratio ρ is set between 3–6, and the prediction vector dimension of the three datasets is set to 64.

4.1.4. Evaluation Protocols

Following study [13], we adopt the common leave-one-out evaluation method instead of using all the user's historical interactions for testing; only the most recent interaction of each user is applied for verification, and the rest of the data are used for training. Obviously, this method is very time-consuming. To improve efficiency, we randomly sample 100 unobserved interactions for each user. Then, the 100 items and test items were sorted according to the predicted results. The predicted ranking results are evaluated on the test set by two evaluation metrics commonly found in recommendation systems, namely hit rate (HR) and normalized discounted cumulative gain (NDCG) separately. The rankings list of these two indicators is truncated to 10. Intuitively, the HR evaluates the accuracy of the predicted top-10 list, and NDCG highlights the top items by giving higher scores to them. The greater the value of these two evaluation indicators, the higher the corresponding prediction accuracy.

We performed 10 repeated experiments for each result and took the average of the ten experimental results. In this way, the instability of the DACR model was reduced so that the data error of the experiment could be reduced.

4.2. Overall Performance (RQ1)

We executed comprehensive experiments on both datasets and then compared the performance with several given baseline methods. Table 2 shows the results of comparing different algorithms with the same evaluation indicators. According to the data analysis in the table, our model outperforms other models on both datasets, and the evaluation performance is obviously stable.

The best results for the data in Table 2 are shown in bold. Overall, the HR and NDCG values obtained by our experiments are superior to other methods. Compared with these methods, DACR shows better performance ranking. According to the performance comparison, we have the following analysis: DACR model combines attention representation learning and attention matching function learning modules. In experiments on the ml-1m dataset, DACR has a significant performance improvement over other algorithms and is better than the DeepUCF+a model. In the music dataset Lastfm, both the HR and NDCG of DACR reached the maximum value, which shows that our model not only performs well on datasets with low sparsity but still has good capability on datasets with high sparsity. In addition, when implicit feedback is used as input, the performance of DMF is severely degraded, and the performance of our DACR model is better than DMF, which shows that using neural networks as interaction functions can learn nonlinear information that cannot be learned by linear interaction, thereby enhancing the prediction accuracy of the model.

Quantitatively, on the ml-1m and Lastfm datasets, compared with DeepUCF+a, the HR and NDCG metrics of DACR are improved by 0.88–1.19% and 0.65–1.15%, respectively.

Table 2. Comparison of evaluation for different models.

Statistics	ml-1m		Lastfm		AMusic	
	HR@10	**NDCG@10**	**HR@10**	**NDCG@10**	**HR@10**	**NDCG@10**
ItemPop	0.4535	0.2542	0.6628	0.3862	0.2483	0.1304
ItemKNN	0.6624	0.3905	0.8673	0.5617	0.3510	0.1989
BPR	0.6725	0.3908	0.6249	0.3466	-	-
DMF	0.6565	0.3761	0.8702	0.5804	0.3744	0.2149
DeepCF	0.7220	0.4378	0.8995	0.6186	0.4116	**0.2601**
DeepUCF+a	0.7264	0.4420	0.9014	0.6193	-	-
Ours(DACR)	**0.7352**	**0.4485**	**0.9133**	**0.6308**	**0.4189**	0.2564
Improvement	0.88%	0.65%	1.19%	1.15%	0.73%	−0.37%

As mentioned before, BPR and DMF methods only contain shallow data interactions and lack complex and deep multiangle data interactions. DeepCF learns the model using the MLP structure in a way that ignores shallow interactions that are as important as the deep interactions learned by MLP. Therefore, the model has limited performance on the datasets. In other words, the DACR method solves the above two shortcomings and is more superior than the comparison methods.

4.3. Impact of the Negative Sampling Ratio (RQ2)

We explore the effect of negative sampling ratio ρ on the accuracy of DACR recommendation results with extensive experiments, that is, the influence of the number of sampled un-interacted instances corresponding to each real interaction instance on the experimental results. The experimental results of different negative sampling ratios are tested on the Lastfm dataset. Figure 3 shows the experimental results.

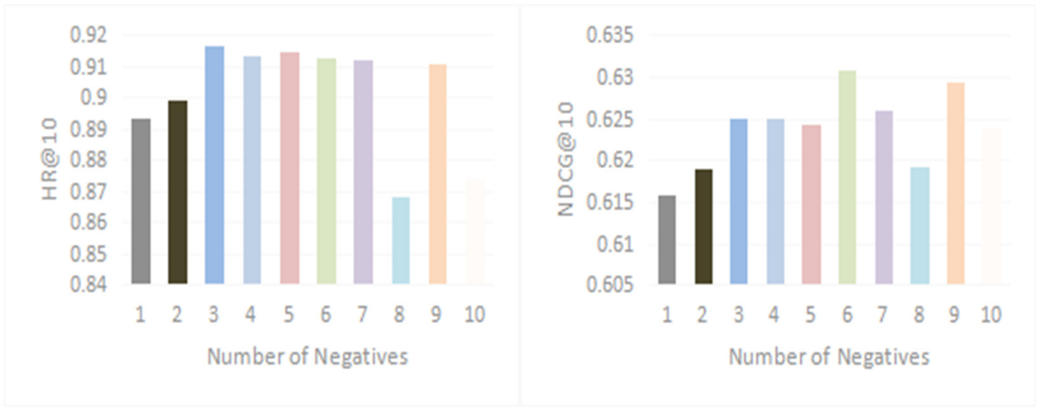

Figure 3. The effect of ρ on Lastfm.

From Figure 3, we first observe that when the value of ρ is set to 3, HR@10 achieves maximum performance, and when ρ is equal to 6, NDCG@10 achieves maximum performance. Increasing the negative sampling ratio of the instance will enhance the predictive performance of the DACR. However, the quantity of model parameters is also increased, resulting in a substantial increase in training time. This will reduce the prediction performance [13]. In summary, we believe that the best negative sampling ratio should be between 3–6, which can not only guarantee the prediction and recommendation results of the model but also reduce the parameters that the model needs to train, thereby saving training time.

4.4. Impact of the Dimension of Prediction Vectors (RQ3)

To adequately realize the maximum probability of the DACR model, we verify the performance of DACR with prediction vectors of multiple dimensions. In the experiments, we set the number of dimensions of the prediction vectors to 8, 16, 32, and 64 and set the recommended list length to 10. As Tables 3 and 4 show, increasing the dimension of the prediction vectors can enhance the precision of the recommendation results. When the prediction vector dimension is 64, DACR works best on both datasets. Generally, the higher the dimension of the prediction vectors, the better the experimental results. The higher the dimension is, the stronger the ability to express hidden information of the DACR model.

Table 3. The impact of different dimensions of prediction vectors on ml-1m.

Evaluation	**Dimension of Prediction Vectors**			
	8	**16**	**32**	**64**
HR@10	0.6925	0.7087	0.7262	0.7352
DNCG@10	0.4032	0.4174	0.4391	0.4485

Table 4. The impact of different dimensions of prediction vectors on lastfm.

Evaluation	**Dimension of Prediction Vectors**			
	8	**16**	**32**	**64**
HR@10	0.8852	0.8895	0.8975	0.9133
DNCG@10	0.6072	0.6134	0.6167	0.6308

4.5. Impact of Different Modules (RQ4)

In this part, we conducted some ablation experiments on LastFM and MovieLens-1M to explore the influence of representation learning module, matching function learning module and attention module on recommendation results. We divided the experiments into six scenarios, and the details of these scenarios are discussed below:

Scenario 1 (CFNet-rl): In this case, we use only the representation learning module, that is, we use only the dot product to calculate the matching scores for user-items.

Scenario 2 (CFNet-ml): In this case, we use only the matching function learning, that is, only the MLP is used to fit the function between the user-item representations and the matching scores.

Scenario 3 (CFNet): We combine scenario 1 and scenario 2, that is, we take into account these two methods of learning interaction.

Scenario 4 (Arl): Adding an attention mechanism to the user-item latent representations in scenario 1 before calculating the matching scores.

Scenario 5 (Aml): Adding an attention mechanism to the concatenated user-item latent representations for scenario 2, which is inputted to the MLP.

Scenario 6 (DACR): In this scenario, we combine two modules from scenario 4 and scenario 5.

The experiment results are shown in Figure 4. It can be seen that the result of scenario 2 is the worst, while that of scenario 1 is slightly better. This is because of the data sparsity of the dataset. Scenario 2 only uses The MLP to fit the nonlinear interaction between user-item representations and matching scores, which may result in over-fitting of results and make the recommendation result not ideal. Scenario 3 combines the two methods and takes into account the low-rank relationship of the user-item interaction matrix, so it is beneficial to the recommendation results. Scenario 6 performs best because the model can distinguish the contribution of different parts of the user-item representations with the attention mechanism, thus capturing more hidden information in the implicit feedback, so as to enhance the quality of recommendations.

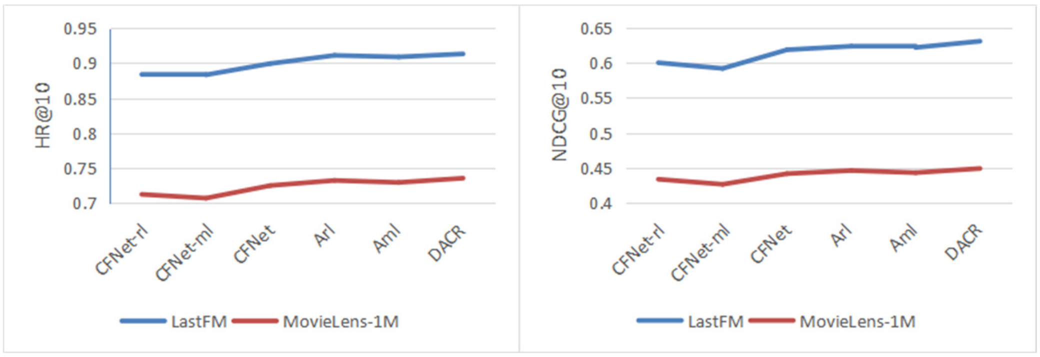

Figure 4. The impact of different modules on recommendation results on LastFM and MovieLens-1M.

5. Conclusions and Future Work

Overall, we propose a ranking recommendation model, DACR, with the features of fusion MLP and traditional MF, which learns the low-rank relationships for user-items and the mapping functions between user-item representations and matching scores from different perspectives. We integrate the attention mechanism to capture the hidden information within implicit feedback. To demonstrate the effectiveness of the DACR model, we conduct numerous and detailed experiments on three public datasets and discuss some possibilities for improvement.

We envision some innovative work in the near future. First, due to the problem of data sparseness, the application of the model to datasets with high data sparseness will result in overfitting. In the future, the overfitting problem can be alleviated by increasing the weight of the linear part or other regularization methods. Then, we will increase common auxiliary information, such as trust or review information to improve the superiority of the proposed method.

Author Contributions: Conceptualization, C.C.; methodology, C.C.; software, C.C.; validation, C.C.; formal analysis, J.Q.; writing—original draft preparation, C.C.; writing—review and editing, Q.R.; supervision, C.C.; funding acquisition, J.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Science Foundation of China under grant no. 61867006. This work was supported by the Science Fund for Outstanding Youth of Xinjiang Uygur Autonomous Region under grant no. 2021D01E14; the Key Laboratory Open Project of Science and Technology Department of Xinjiang Uygur Autonomous Region named Research on video information intelligent processing technology for Xinjiang regional security; the Major science and technology project of Xinjiang Uygur Autonomous Region under grant no 2020A03001.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: We evaluated our algorithm on two datasets: MovieLens 1M, and LastFM. http://www.grouplens.org/node/73 (accessed on 1 May 2021); http://ocelma.net/MusicRecommendationDataset/lastfm-1K.html (accessed on 1 May 2021).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhao, Z.L.; Huang, L.; Wang, C.D.; Huang, D. Low-rank and sparse cross-domain recommendation algorithm. In Proceedings of the International Conference on Database Systems for Advanced Applications, Gold Coast, QLD, Australia, 21–24 May 2018; Springer: Cham, Switzerland, 2018; pp. 150–157.
2. Cai, Y.; Leung, H.F.; Li, Q.; Min, H.; Tang, J.; Li, J. Typicality-based collaborative filtering recommendation. *IEEE Trans. Knowl. Data Eng.* **2013**, *26*, 766–779. [CrossRef]
3. Wang, H.; Li, W.J. Relational collaborative topic regression for recommender systems. *IEEE Trans. Knowl. Data Eng.* **2014**, *27*, 1343–1355. [CrossRef]

4. Hu, Q.Y.; Huang, L.; Wang, C.D.; Chao, H.Y. Item orientated recommendation by multi-view intact space learning with overlapping. *Knowl.-Based Syst.* **2019**, *164*, 358–370. [CrossRef]
5. Heidari, A.; Navimipour, N.J.; Unal, M. Applications of ML/DL in the management of smart cities and societies based on new trends in information technologies: A systematic literature review. *Sustain. Cities Soc.* **2022**, *85*, 104089. [CrossRef]
6. Heidi, A.; Jafari Navimipour, N.; Unal, M.; Toumaj, S. Machine learning applications for COVID-19 outbreak management. *Neural Comput. Appl.* **2022**, 1–36.
7. Covington, P.; Adams, J.; Sargin, E. Deep neural networks for youtube recommendations. In Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, 15–19 September 2016; pp. 191–198.
8. Xia, H.; Li, J.J.; Liu, Y. Collaborative filtering recommendation algorithm based on attention GRU and adversarial learning. *IEEE Access* **2020**, *8*, 208149–208157. [CrossRef]
9. Huang, P.S.; He, X.; Gao, J.; Deng, L.; Acero, A.; Heck, L. Learning deep structured semantic models for web search using clickthrough data. In Proceedings of the 22nd ACM International Conference on INFORMATION & Knowledge Management, San Francisco, CA, USA, 27 October–1 November 2013; pp. 2333–2338.
10. Xue, H.J.; Dai, X.; Zhang, J.; Huang, S.; Chen, J. Deep matrix factorization models for recommender systems. In Proceedings of the IJCAI, Melbourne, Australia, 19–25 August 2017; Volume 17, pp. 3203–3209.
11. Xu, J.; He, X.; Li, H. Deep learning for matching in search and recommendation. *Found. Trends® Inf. Retr.* **2020**, *14*, 102–288. [CrossRef]
12. Hornik, K.; Stinchcombe, M.; White, H. Multilayer feedforward networks are universal approximators. *Neural Netw.* **1989**, *2*, 359–366. [CrossRef]
13. He, X.; Liao, L.; Zhang, H.; Nie, L.; Hu, X.; Chua, T.S. Neural collaborative filtering. In Proceedings of the 26th International Conference on World Wide Web, Perth, Australia, 3–7 April 2017; pp. 173–182.
14. Beutel, A.; Covington, P.; Jain, S.; Xu, C.; Li, J.; Gatto, V.; Chi, E.H. Latent cross: Making use of context in recurrent recommender systems. In Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, Los Angeles, CA, USA, 5–9 February 2018; pp. 46–54.
15. Niu, Z.; Zhong, G.; Yu, H. A review on the attention mechanism of deep learning. *Neurocomputing* **2021**, *452*, 48–62. [CrossRef]
16. Feng, J.; Feng, X.; Chen, J.; Cao, X.; Zhang, X.; Jiao, L.; Yu, T. Generative adversarial networks based on collaborative learning and attention mechanism for hyperspectral image classification. *Remote Sens.* **2020**, *12*, 1149. [CrossRef]
17. Wang, H.; Zhang, F.; Xie, X.; Guo, M. DKN: Deep knowledge-aware network for news recommendation. In Proceedings of the 2018 World Wide Web Conference, Lyon, France, 23–27 April 2018; pp. 1835–1844.
18. He, X.; He, Z.; Song, J.; Liu, Z.; Jiang, Y.G.; Chua, T.S. Nais: Neural attentive item similarity model for recommendation. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 2354–2366. [CrossRef]
19. Tay, Y.; Zhang, S.; Tuan, L.A.; Hui, S.C. Self-attentive neural collaborative filtering. *arXiv* **2018**, arXiv:1806.06446.
20. Xiao, J.; Ye, H.; He, X.; Zhang, H.; Wu, F.; Chua, T.S. Attentional factorization machines: Learning the weight of feature interactions via attention networks. *arXiv* **2017**, arXiv:1708.04617.
21. Oard, D.W.; Kim, J. Implicit feedback for recommender systems. In Proceedings of the AAAI Workshop on Recommender Systems, AAAI. Madison, WI, USA, 26–30 July 1998; Volume 83, pp. 81–83.
22. Ma, H. An experimental study on implicit social recommendation. In Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval, Dublin, Ireland, 28 July–1 August 2013; pp. 73–82.
23. Hu, Y.; Koren, Y.; Volinsky, C. Collaborative filtering for implicit feedback datasets. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, Washington, DC, USA, 15–19 December 2008; pp. 263–272.
24. Koren, Y. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, NV, USA, 24–27 August 2008; pp. 426–434.
25. Rendle, S.; Freudenthaler, C.; Gantner, Z.; Schmidt-Thieme, L. BPR: Bayesian personalized ranking from implicit feedback. *arXiv* **2012**, arXiv:1205.2618.
26. Koren, Y.; Bell, R.; Volinsky, C. Matrix factorization techniques for recommender systems. *Computer* **2009**, *42*, 30–37. [CrossRef]
27. Koren, Y. Collaborative filtering with temporal dynamics. In Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, 28 June–1 July 2009; pp. 447–456.
28. Hu, L.; Sun, A.; Liu, Y. Your neighbors affect your ratings: On geographical neighborhood influence to rating prediction. In Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval, Gold Coast, QLD, Australia, 6–11 July 2014; pp. 345–354.
29. Wang, H.; Wang, N.; Yeung, D.Y. Collaborative deep learning for recommender systems. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, Australia, 10–13 August 2015; pp. 1235–1244.
30. Sedhain, S.; Menon, A.K.; Sanner, S.; Xie, L. Autorec: Autoencoders meet collaborative filtering. In Proceedings of the 24th International Conference on World Wide Web, Florence, Italy, 18–22 May 2015; pp. 111–112.
31. Ahmadian, S.; Ahmadian, M.; Jalili, M. A deep learning based trust-and tag-aware recommender system. *Neurocomputing* **2022**, *488*, 557–571. [CrossRef]

32. Ahmadian, M.; Ahmadi, M.; Ahmadian, S.; Jalali, S.M.J.; Khosravi, A.; Nahavandi, S. Integration of Deep Sparse Autoencoder and Particle Swarm Optimization to Develop a Recommender System. In Proceedings of the 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Melbourne, Australia, 17–20 October 2021; pp. 2524–2530.
33. Ahmadian, M.; Ahmadi, M.; Ahmadian, S. A reliable deep representation learning to improve trust-aware recommendation systems. *Expert Syst. Appl.* **2022**, *197*, 116697. [CrossRef]
34. Bai, T.; Wen, J.R.; Zhang, J.; Zhao, W.X. A neural collaborative filtering model with interaction-based neighborhood. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, Singapore, 6–10 November 2017; pp. 1979–1982.
35. Cheng, H.T.; Koc, L.; Harmsen, J.; Shaked, T.; Chandra, T.; Aradhye, H.; Anderson, G.; Corrado, G.; Chai, W.; Ispir, M.; et al. Wide & deep learning for recommender systems. In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, Boston, MA, USA, 15 September 2016; pp. 7–10.
36. He, X.; Du, X.; Wang, X.; Tian, F.; Tang, J.; Chua, T.S. Outer product-based neural collaborative filtering. *arXiv* **2018**, arXiv:1808.03912.
37. Zhang, Q.; Cao, L.; Zhu, C.; Li, Z.; Sun, J. Coupledcf: Learning explicit and implicit user-item couplings in recommendation for deep collaborative filtering. In Proceedings of the IJCAI International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 13–19 July 2018.
38. Chen, J.; Zhang, H.; He, X.; Nie, L.; Liu, W.; Chua, T.S. Attentive collaborative filtering: Multimedia recommendation with item- and component-level attention. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Tokyo, Japan, 7–11 August 2017; pp. 335–344.
39. Cheng, Z.; Ding, Y.; He, X.; Zhu, L.; Song, X.; Kankanhalli, M.S. A^{3}NCF: An Adaptive Aspect Attention Model for Rating Prediction. In Proceedings of the IJCAI 2018, Stockholm, Sweden, 13–19 July 2018; pp. 3748–3754.
40. Da'u, A.; Salim, N.; Idris, R. An adaptive deep learning method for item recommendation system. *Knowl.-Based Syst.* **2021**, *213*, 106681. [CrossRef]
41. Xi, W.D.; Huang, L.; Wang, C.D.; Zheng, Y.Y.; Lai, J. BPAM: Recommendation Based on BP Neural Network with Attention Mechanism. In Proceedings of the IJCAI, Macao, 10–16 August 2019; pp. 3905–3911.
42. Pan, R.; Zhou, Y.; Cao, B.; Liu, N.N.; Lukose, R.; Scholz, M.; Yang, Q. One-class collaborative filtering. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, Washington, DC, USA, 15–19 December 2008; pp. 502–511.
43. Salakhutdinov, R.; Mnih, A. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland, 5–9 July 2008; pp. 880–887.
44. McCaffrey, J.D. Why you should use cross-entropy error instead of classification error or mean squared error for neural network classifier training. *Last Accessed Jan* **2018**.
45. Sarwar, B.; Karypis, G.; Konstan, J.; Riedl, J. Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th International Conference on World Wide Web, Hong Kong, 1–5 May 2001; pp. 285–295.
46. Deng, Z.H.; Huang, L.; Wang, C.D.; Lai, J.H.; Philip, S.Y. Deepcf: A unified framework of representation learning and matching function learning in recommender system. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 61–68.
47. Chen, J.; Wang, X.; Zhao, S.; Qian, F.; Zhang, Y. Deep attention user-based collaborative filtering for recommendation. *Neurocomputing* **2020**, *383*, 57–68. [CrossRef]