Journal of Organizational and End User Computing
Volume 36 • Issue 1

An Empirical Study on Personalized Product Recommendation Based on Cross-Border E-Commerce Customer Data Analysis

Wanwan Li, Universiti Kebangsaan Malaysia, Malaysia & Taizhou Vocational & Technical College, China
https://orcid.org/0000-0002-3857-7804

Ying Cai, Taizhou Vocational & Technical College, China

Mohd Hizam Hanafiah, Universiti Kebangsaan Malaysia, Malaysia*

Zhenwei Liao, Universiti Kebangsaan Malaysia, Malaysia

ABSTRACT

Thanks to the rapid growth of cross-border e-commerce platforms, numerous cross-border items are now available to customers. Several serious issues with cross-border e-commerce platforms related to item promotion and consumer product screening have arisen. Particular importance should be placed on studying and implementing personalized recommendation systems based on international e-commerce. In light of the quick expansion of commodities, when making individualized suggestions, traditional recommendation algorithms have had to deal with issues such as scant data, a chilly start to the market, and trouble identifying user preferences. To automatically mine the implicit and latent relationships between users and objects in recommendation systems, this study employs deep learning with nonlinear learning capabilities, which resolves the challenges of user interest mining. The weaknesses of the existing global recommendation research are emphasized, the study of conventional recommendation algorithms mixed with deep learning technology is deep factorization machine (DeepFM) and neural matrix factorization (NeuMF) models. Both models excel in recommending implicit feedback data. The DeepFM model yields the lowest loss function values, while the NeuMF model outperforms the competing models in terms of HR@20 (a commonly used indicator to measure the recall rate) and loss functions. In summary, this research addresses critical issues in cross-border e-commerce by developing personalized recommendation systems and integrating deep learning with traditional recommendation algorithms to enhance global recommendations.

KEYWORDS:

Cross-Border E-Commerce, Recommendation Systems Automatically, Personalized Recommendation, Deep Learning Technology, Data Sparsity, Commodity Information

INTRODUCTION

With the rapid integration of the world economy, China's cross-border e-commerce has ushered in swift expansion. From 2014 to 2015, China's cross-border online shopping platforms experienced

DOI: 10.4018/JOEUC.335498

*Corresponding Author

explosive growth, a series of products focusing on a single market segment were released, and a professional cross-border online shopping platform was born to meet the more refined needs of domestic consumers. Due to the massive influx of overseas products, consumers encounter confusion when choosing their favorite products. At the same time, cross-border e-commerce platforms face increased difficulty in pushing suitable cross-border products to target consumer groups, motivating the development of a personalized recommendation system.

The concept of personalized recommendation systems based on cross-border e-commerce provides strong support for solving the above contradictions. The design goal of personalized recommendation systems is to help consumers discover products that interest them and that they can afford by presenting them with valuable information. At the same time, personalized recommendation systems can help commodity sales platforms quickly push and sell commodities, thereby effectively reducing inventory, improving the circulation speed of commodities, and ultimately achieving a win–win situation for consumers and commodity sellers (Balabanovic & Shoham, 1997; Li & Zhang, 2020; Liu et al., 2020).

The application research and implementation of personalized recommendation systems are vital and urgent because the traditional e-commerce personalized recommendation system does not account for some characteristics of cross-border e-commerce customer groups and cannot meet the needs of current mobile platform users for accurate, intelligent, and personalized information services based on cross-border e-commerce. Traditional recommendation algorithms mainly include demographic-based recommendation algorithms, content-based recommendation algorithms, collaborative filtering algorithms, hybrid recommendation algorithms, and model recommendation algorithms (Gao et al., 2019; Liu & Zhu, 2020; Wang & Li, 2017; Yan et al., 2020). Currently, the number of users and the types and quantities of products in e-commerce platforms are vast, resulting in a high degree of sparseness in the matrix formed by users' ratings of products. Traditional recommendation algorithms find it challenging to determine similar goods that attract users based on the sparse matrix when users and products grow simultaneously. This difficulty also contributes to the creation of the system cold start problem.

Another approach involves combining multiple recommendation algorithms for algorithm recommendation, filling in missing values in the scoring matrix, introducing auxiliary information, improving recommendation algorithms, and using deep learning-related technologies to alleviate data sparsity (Yue et al., 2020). Deep learning technology is widely utilized in the fields of image processing, speech recognition, and natural language processing, leading to remarkable achievements (Liu & Xia, 2014; Wang et al., 2020; Yue et al., 2015). Deep learning can process linear data, has stronger fitting ability for nonlinear data, and does not require feature engineering. Deep learning opens new research avenues for recommendation systems by combining low-level features to create richer high-level characteristics. Using a deep network structure to learn the unstructured data of products and users in the recommender system can mine the potential features of users and products and obtain the deep features of users and products, solving the problems of sparse data and difficulty in mining user interests in the recommender system (Fan, 2019). Therefore, applying deep learning in personalized product recommendation systems has research significance.

In this research, we selected the deep neural network (DNN) model and the factorization machine (FM), comprising the deep factorization machine (DeepFM) model and the neural matrix factorization model NeuMF, for the recommendation module of a personalized recommendation system (Huang et al., 2020; Li et al., 2019; Yu et al., 2019). Users can select from a variety of product suggestion models. The model uses one-hot encoding and the embedding layer to encode the data and embed it into the model to convert high-dimensional sparse data into low-dimensional dense data. This eliminates the need to perform feature engineering on the data, and the model effectively solves the problems of data sparsity and difficulty mining user interests.

The paper makes the following unique contributions:

- Identifies the current issues in the field of recommendation systems based on an exhaustive literature review.
- Explores the characteristics of the DeepFM and NeuMF models and recommends implicit feedback information.
- Includes the one-hot encoding technique, feeding the resultant data into the DeepFM and NeuMF models.

RELATED WORK

With the development of computer technology and artificial intelligence, more and more fields have begun to apply these technologies in cross-disciplinary research. Ankora and Aju (2022) integrate agile and design thinking practices to generate high-quality user stories for augmented reality (AR) application design. Sukte et al. (2022) propose a secure method called secure sharing PHR for exchanging personal health records in the cloud while preserving privacy and granting selective access. Tang and Zhang (2022), based on the theory of organizational information processing, examine the impact of virtual integration on product innovation speed in a competitive environment using samples of Chinese manufacturing enterprises. Li et al. (2022) propose a semantic similarity calculation method that integrates the word2vec model and TF-IDF to improve the accuracy of text similarity measurement. They apply this method to the density peak clustering of Chinese text data consulted by patients in an online medical community.

Recommender systems have undergone decades of development, during which many excellent studies have emerged (Choi et al., 2012; Hinton et al., 2006; Liu et al., 2013). The latent factor model (LFM) proposed by Koren et al. (2009) is a collaborative filtering algorithm based on matrix factorization (MF), which significantly improves the accuracy of score prediction. The algorithm was then further optimized to model temporal features based on the singular value decomposition algorithm of time series (Koren, 2010). However, the cold-start problem and the sparsity of the scoring data reduce the prediction accuracy of traditional collaborative filtering techniques (Adomavicius & Tuzhilin, 2005; Donahue et al., 2013).

Scholars have proposed a hybrid recommendation method that fuses the two methods to overcome the limitations of a single method, namely collaborative filtering and the problems of content-based recommendation diversity and low accuracy. This hybrid method has been verified to improve the overall effect of the recommender system in practice. However, traditional methods have low accuracy and slow computing speeds and can no longer meet the growing needs of users. As a result, many businesses are now interested in the rapidly evolving topic of deep learning, using it in tandem with recommendation systems for industrial applications. In the field, state-of-the-art results are expected (Zhang et al., 2019). Cho et al. (2014) employed a recommendation algorithm based on a recurrent neural network (RNN) in the news recommendation system, which uses a denoising autoencoder to generate a distributed expression of news and then uses a gated recurrent unit–based RNN model. A deeper user feature vector is learned by inputting user browsing behavior data. Online tests on the Yahoo News website show that the algorithm significantly improves recommendation performance compared to traditional methods. The recommendation algorithm proposed by Hochreiter and Schmidhuber (1997) improves the multilayer perceptron neural network model, which is divided into a candidate video generation network and a candidate video ranking network. The extracted video features are applied to YouTube website recommendations. Chorowski et al. (2014) built a recommendation system with both breadth and depth for app recommendation in Google Play. It enhances memory ability via the linear model and feature intersection method, which has good interpretability. At the same time, it uses deep feature mining to express hidden features by combining low-dimensional features.

The combination of linear models and DNNs balances memory and generalization. Therefore, a model with both memory and generalization abilities improves the performance of the APP recommender system. DNNs have played an essential role in the practical application of industrial recommendations, and repeated attempts have achieved good results, promoting the development of e-commerce platforms and continuously improving user satisfaction. The number of academic studies on the application of DNNs in recommender systems has increased, continuously promoting the development of this field (Luong et al., 2015; Mcauley et al., 2015; Wu et al., 2017). Several International Conferences on Recommendation Systems (RecSys) on deep learning have been held, with many scholars presenting excellent research results. The DeepFM model proposed by Guo et al. (2017) combines an FM and a DNN (Terveen et al., 1997). The deep part introduces an embedding layer to extract high-order features, and the factorization machine part combines low-order features. Kautz et al. (1997) proposed a collaborative deep learning (CDL) model that integrates a stacked denoising autoencoder into probabilistic matrix factorization. The recommendation system based on CDL is divided into perception and specific task components. The DNN is responsible for learning the content information of the text, and the collaborative filtering part processes the rating matrix to extract the latent variables of users and products. The accuracy and diversity of the recommender system are enhanced by the tight integration of the two to utilize a range of information sources fully. Whether in industry or academia, the use of deep learning technology in recommender systems has become a research hotspot, and research on hybrid recommendation algorithms based on DNN content recommendation combined with collaborative filtering algorithms has begun to attract attention.

To achieve the expected effect, the DNN must be able to mine deeper features from the commodity content and user auxiliary information. The general feedforward neural network cannot meet such needs, so using more sophisticated DNNs to extract features is necessary. Currently, the best performers in computer vision and text mining are CNNs and long short-term memory neural networks (Choi et al., 2020; Simonyan & Zisserman, 2014; Tang & Huang, 2021). Although they have achieved good results, the existing research on recommendation systems combined with DNN technology encounters two problems: First, the actual situation of the product life cycle is disregarded, and a single recommendation algorithm is employed. However, commodities in different life cycles have entirely different characteristics and data volumes. Although a single algorithm can alleviate some problems, it cannot be fully applied to different life cycles of commodities. Second, most recommendation algorithms use DNNs. The research only analyzes the historical interaction information of users and products. It does not fully utilize the intrinsic value of valuable information, such as images and comment texts, and the extracted features are ineffective or lack deep data mining. The performance of recommender systems is greatly enhanced by this content data. Therefore, this paper proposes a recommendation framework based on a DNN, which can use appropriate algorithms according to the different life cycles of products and use DNN technology to fully mine the content information of products and users according to needs.

METHODS

Analysis of System Functional Requirements

The personalized product recommendation system mainly targets users' shopping needs, recommending products of interest to users and improving their purchasing power and shopping experience. The following section examines the system functions that are fundamentally relevant to users.

1) User functional requirements. User registration function: A new user entering the system must fill in basic personal information, including username, password, phone number, email address, shipping address, age, and gender. User login function: After correctly entering their username and password during registration, the user may sign in and use the system. If the user password is incorrect, it can be retrieved by email or phone number. If the user does not exist, they can

re-register. Comment information function for purchased products: Users can browse products in the recommended list when purchasing products. They can collect and continue to learn more about products that interest them, including product sales, ratings and evaluations, number of user views, and place of origin, and then payment can be made. In addition, after receiving the goods, the user can evaluate them. This evaluation can help other users by providing important reference information about the goods. User search function: Users can search for goods according to their needs. Viewing the recommended list: Users can browse the product list recommended by the system, select products they like or find interesting, and click the product detail function to learn more about it. Product purchase function: Users can purchase and pay for products on the product detail page or in the shopping cart. Recommendation feedback function: The user provides feedback on the recommendation results, thus changing the recommendation strategy. User rating and comment function: Users can evaluate and rate purchased products.

2) User's information collection function. This function collects information about the user's behavior in the system, including their purchased products, product evaluations, product browsing behavior, and shopping cart collection.
3) User data preprocessing function requirements. The system collects user behavior data in the system in real time, performs data preprocessing, such as data cleaning, data missing value processing, data normalization processing, data integration, and filtering of the originally collected data, and then stores the data in the database for recommendation modules to improve data support.
4) Recommended functional requirements. In the personalized recommendation module, users can choose different recommendation models and use data to help users recommend product lists. Model recommendations based on deep learning can solve the problems of data sparsity and cold start.

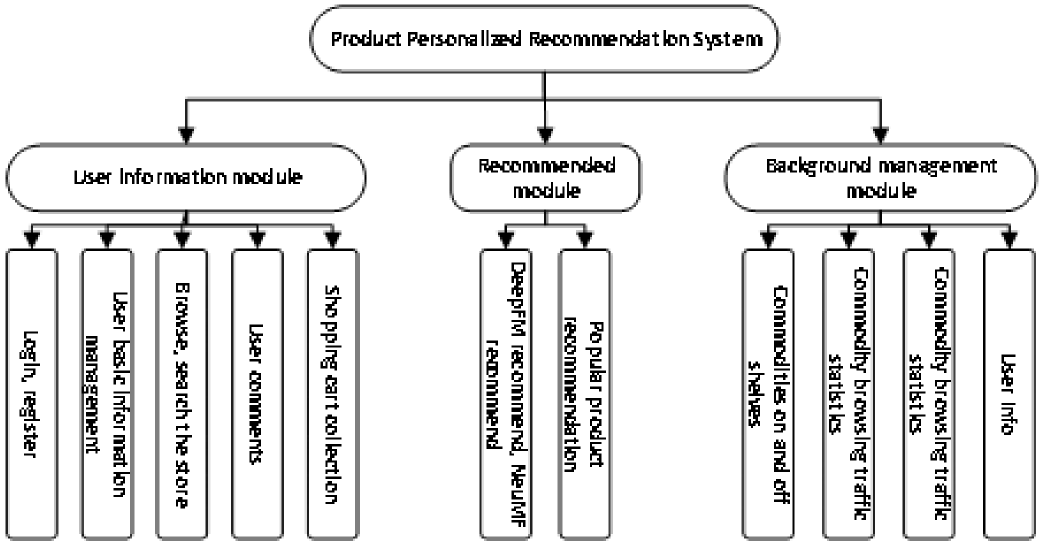
Overall System Framework Design

The overall architecture of the product personalized recommendation system presented in this paper was designed based on the classic B/S architecture and using the Django framework and MTV mode. The framework comprises the Web service layer, Django framework core layer, and data storage layer. The model layer includes the application logic, database, and recommendation model layers. In the Web server, uWSGI is a full-featured HTTP server that implements the WSGI protocol, uWSGI protocol, and HTTP protocol by converting the HTTP protocol into a network protocol supported by the language. For example, the HTTP protocol is converted to the WSGI protocol so that Python can use it directly. WSGI is a specification protocol that describes how Web servers communicate with Web applications. Nginx is a Web server in which the HTTP server function is similar to the uWSGI function; however, Nginx is most commonly applied as a reverse proxy.

The Web project development framework Django adopts the MTV model and is divided into three layers. The model layer interacts with the database MySQL and can perform operations such as adding, deleting, and querying the data stored in the database. The view receives requests, performs business logic processing, and then responds. The HTML built to be returned must be contained within the template layer. The MySQL database is selected for the data storage layer to store the system's data. The recommendation system stores many product pictures, serial numbers, prices, user-related data, and other information. Data access support is provided for the system, user data is collected via the database, and recommendation calculations are performed to generate a product recommendation list. According to the requirement analysis system of the recommendation system, we designed three modules, and the functions under each module are shown in Fig. 1.

1) User information module: This module includes the user's login and registration information. When the user logs in to the system, their input password is compared with the original password

Figure 1. Overall design framework of the system



to judge whether it is correct. The user's basic information includes hobbies, height and weight, occupation, age, and place of residence. Regarding product management, the products the user has searched for are most likely to be those the user recently needed to buy; thus, the searched products can serve as essential data for making recommendations. The products collected in the shopping cart are those the user has browsed for in the system. The user can purchase products of interest, and the system can recommend more suitable products for the user according to product type, price, or rating, for example. Ratings and reviews provide vital information to help the system recommend products to other users.

2) Recommendation module: After logging in, the user sees a list of popular suggested items, along with choices for DeepFM and NeuMF recommendation systems, on the system page. Selecting a different method will display a different list of recommended products.
3) Background management module: This module addresses the listing and delisting of commodities. For many seasonal commodities, previous commodities will be delisted, and new commodities will be placed on the shelves, depending on the season. Statistical processing is performed on user and product information, statistics on the number of times users browse products and view detailed product information, and user information management.

DeepFM Algorithm Module

In 2016, Google proposed the Wide&Deep model, a hybrid model consisting of a single-layer wide part, an embedding layer, and a deep part consisting of multiple hidden layers, which are jointly input to the final output layer. The wide part gives the model a strong "memory ability." The deep part gives the model a strong "generalization ability." This structural element enables the model to swiftly process and retain a large number of past behavior variables while considering the benefits of logistic regression and DNNs. Due to the model's strong expressive ability, it has become the application's mainstream model, and numerous hybrid models have been derived based on the Wide&Deep model. The wide part of the model uses a linear model formula:

$$y = w_0 + P^T x \tag{1}$$

where x represents a sample of a d-dimensional feature and P represents a model parameter. The sigmoid function is used to transform the output value in the [0, 1] interval.

The deep part is a standard feedforward neural network. The form of each layer is shown in Equation 2. The input high-dimensional sparse original features must be converted to a low-dimensional dense vector, and the dimension range is usually 10 to 100. The vector is randomly initialized, the model is trained through the minimum anytime function, and the activation function is ReLU. The formula for each layer is expressed as follows:

$$R^{(l+1)} = f\left(W^{(l)}R^{(l)} + V^{(l)}\right) \tag{2}$$

where W, R, and V are the activation values, biases, and weights, respectively, of the Lth layer, and f is the activation function.

DeepFM is a neural network framework that integrates an FM and a DNN. The idea is similar to Wide&Deep; both include wide and deep parts. The wide part of DeepFM is the FM, and the deep part of both is a DNN layer. The DeepFM model represents an improvement because it addresses the defect that the wide part of the Wide&Deep model cannot automatically combine features, and the FM is used for feature combination. The DeepFM architecture has the following features: (a) there is no need to pre-train the FM to obtain latent vectors; (b) artificial feature engineering is not needed; (c) the architecture can simultaneously learn low-order and high-order combined features; and (d) the FM module and DNN module share the feature embedding part for faster training and more accurate training learning. The prediction result formula is:

$$\hat{y} = sigmoid\left(y_{FM} + y_{DNN}\right) \tag{3}$$

where y_{FM} is the output of the FM part and y_{DNN} is the output of the DNN part.

FM Components

The FM algorithm is improved on the basis of LR so that the model's cross-feature expression ability is stronger. The FM learns a latent vector for each feature and uses the inner product of the two feature latent vectors as the weight of the cross feature when the feature is crossed. The output function formula of the FM is:

$$\tilde{y} = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} u_i u_j x_i x_j \tag{4}$$

where $w_0 \in R$, $w \in R^n$, $u \in R^{n\times k}$, and n represents the number of features of the sample.

The output of the FM component model is the sum of the addition unit and the inner product unit. The formula is:

$$y_{FM} = w,x + \sum_{i=1}^{d} \sum_{j=i+1}^{d} U_i, U_j x_i x_j \tag{5}$$

where the addition unit w,x reflects the first-order feature extraction characteristics, and the inner product unit reflects the second-order feature extraction characteristics.

DNN Components

DNNs are feed-forward networks employed in DeepFM models to learn higher-order feature combinations. The DNN and FM parts share the embedding layer, and the output prediction formula of the DNN part is:

$$\hat{y}_{DNN} = W^{|H|+1} \cdot R^{|H|} + V^{|H|+1} \tag{6}$$

where $|H|$ is the number of hidden layers.

Neural Collaborative Filtering Framework

The neural collaborative filtering framework replaces the simple inner product operation in the matrix factorization model with the structure "multilayer neural network + output layer." It combines the generalized matrix factorization (GMF) and multilayer perceptron (MLP) models. The advantage of this approach is that the user vector and item vector can be more fully intersected, and more valuable feature combination information can be obtained. More nonlinear features are introduced to make the model more expressive.

Generalized Matrix Factorization

The input layer of the GMF model is that the user and the item are separately one-hot encoded, and then the embedding vector is the latent vector of the user and the item. We use $P^T v_u^U$ to represent the user's hidden vector p_u and $Q^T v_i^I$ to represent the item's hidden vector q_i. The mapping function of the first layer of the neural network is defined as:

$$\phi_1\left(p_u, q_i\right) = p_u \odot q_i \tag{7}$$

where $\odot$ means that each element of the vector is multiplied correspondingly.

Then, the vector obtained in the first layer is mapped to the output layer, where R_{out} represents the activation function of the output layer, and h represents the link weight of the output layer:

$$\hat{y}_{ui} = R_{out}\left(h^T\left(p_u \odot q_i\right)\right) \tag{8}$$

The nonlinear activation function R_{out} enables the GMF to capture nonlinear relationships; that is, more complex relationships can be captured relative to the MF.

Multilayer Perceptron

NCF separately models users and items through two branches. Thus, it is necessary to combine the respective features of the two branches at the end and use a standard MLP to learn the interaction between users and item latent features so that the model is more flexible in nonlinear modeling capabilities. The MLP model in the framework is defined as follows:

$$z_1 = \phi_1\left(p_u, q_i\right) = \begin{bmatrix} p_u \\ q_i \end{bmatrix} \tag{9}$$

$$\phi_2\left(z_1\right) = R_2\left(W_2^T z_1 + V_2\right) \tag{10}$$

$$\phi_L\left(z_{L-1}\right) = R_L\left(W_L^T z_{L-1} + V_L\right) \tag{11}$$

$$\hat{y}_{ui} = \lambda\left(h^T \phi_L\left(z_{L-1}\right)\right) \tag{12}$$

where W_x, R_x, and V_x in the formula represent the weight matrix, activation function, and bias vector, respectively, of the xth layer in the neural network.

NeuMF Model

The NeuMF model is a combination of GMF and MLP models. The GMF and MLP embedding layers are separate because the embedding vectors of the two models are different, and using a shared embedding layer would limit the effect of the model. To make the fusion model more flexible, GMF and MLP can learn independent embeddings and combine the two models by connecting their final hidden layer outputs. The model function formula is expressed as follows:

$$\phi^{GMF} = p_u^G \odot q_i^G \tag{13}$$

$$\phi^{MLP} = R_L\left(W_L^T\left(R_{L-1}\left(\ldots R_2\left(W_2^T\begin{bmatrix} p_u^M \\ q_i^M \end{bmatrix} + V_2\right)\ldots\right)\right) + V_L\right) \tag{14}$$

$$\hat{y}_{ui} = \alpha\left(h^T\begin{bmatrix} \phi^{GMF} \\ \phi^{MLP} \end{bmatrix}\right) \tag{15}$$

The general collaborative filtering framework NeuMF is used to solve the top-K item recommendation problem with implicit feedback data, so the loss functions used by GMF, MLP, and the fusion model NeuMF are binary cross-entropy loss functions. The function formula is:

$$-\frac{1}{m}\sum_{i=1}^{m} y_i \log\left(h_\theta\left(x_i\right)\right) + \left(1 - y_i\right)\log\left(1 - h_{\theta_{\left(x_i\right)}}\right) \tag{16}$$

The advantages of the NeuMF model compared with the latent semantic model LFM are listed as follows:

1. An LFM connects users and items by latent features, which maximizes the feature expression of the latent space connecting users and items, while NeuMF uses an NN to learn the interaction function between users and items, which includes the solution process of latent features.
2. An LFM is a latent semantic model calculated by the matrix decomposition method, and the solution is the user-hidden class matrix and item-hidden class matrix. In contrast, the NeuFM solution obtains a model.
3. An LFM can only calculate the scores of users and items that have appeared but are not rated according to two matrices and can only fill in the blanks of the original user–item matrix. The NeuMF model is universal and can calculate new users and new things.
4. An LFM only considers the linear relationship between two hidden classes, and the relationship between two hidden classes is generally nonlinear, especially when the number of hidden classes is small. NeuMF combines GMF, which considers the linear relationship between two hidden classes, and MLP to learn the nonlinear relationship among latent features between users and items, which is very flexible.

EXPERIMENT AND ANALYSIS

Experimental Data Selection and Processing

The data for the experiment are based on the research content of this paper for product recommendation via deep model training, and the data should be selected from a large e-commerce platform dataset. E-commerce platforms such as Alibaba's cross-border trading platform have a large user base, and the active users of daily shopping malls have been stable. In addition, information such as pictures of products purchased by users, product clicks, product ratings and comments, product views, shopping cart products, and store attention are more comprehensive. The deep learning model has specific requirements for the data volume of the experimental dataset. Choosing a dataset with a particular scale and avoiding a dataset with a small data scale and incomplete information will cause significant differences in the product recommendation results. The dataset in this paper was used to obtain user behavior data on a cross-border e-commerce platform using crawler technology as a training set and test set due to the many active users, a complete range of commodities, and a large amount of commodity transaction information in the mall. Crawling user data includes user name, user shopping cart product number, user search product information, user favorite products, user purchased product ratings, user browsing product names and entering product detail pages, product sales, product ratings, data parameters such as product browsing time, the province where the user purchased the product, and the source of the product, as shown in Table 1.

We divided the data required for the experiment into training and test sets in time order. The time of the data volume of 2.5 million served as the time cutoff, and the data before the time cutoff served as the training dataset. A small part of the data was truncated as the test set of 500,000 pieces of data. Part of the data obtained by the crawler was incomplete, and the data were cleaned to remove any damaged samples, such as short text, highly distorted images, false output labels, and a large amount of null-valued, feature data-related information. The acquired information was encoded using the one-hot method. The encoded data were converted to dense low-dimensional features through the embedding layer, and the feature values were input into the trained model.

DeepFM Experimental Analysis

We used the open-source tool TensorFlow to train the FM, DNN, DeepFM, and Wide&Deep models. The evaluation index of the experimental results was the log-likelihood loss function LogLoss, and the training set and test set evaluation index of the area under the curve (AUC) were the logarithmic loss function LogLoss and AUC value. The definition formula of LogLoss in the evaluation index is:

$$LogLoss = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i logp_i + \left(1 - y_i\right)\left(1 - p_i\right)\right) \tag{17}$$

where y_i is the actual category of the sample x_i, and p_i is the probability that x_i is predicted to be clicked. The smaller the LogLoss, the better the prediction effect of the model on the data.

AUC is the area under the ROC curve, in which the vertical axis is the recall rate, and the horizontal axis is the false-positive rate. This performance indicator measures the pros and cons of the learner. The larger the AUCm, the better the model effect. The formulas are:

Table 1. Experimental data sheet

Number of Users	Related Data Volume	Large Category	Small Category	Browsing Quantity	Purchase Quantity
20,000	3,000,000	25	875	2,200,000	200,000

$$AUC = P(P_{positive\ sample} > P_{negative\ sample}) \tag{18}$$

$$Rec = \frac{TP}{TP + FN} \tag{19}$$

$$Pre = \frac{TP}{TP + FP} \tag{20}$$

where Rec denotes recall, Pre indicates precision, TP represents true positive, FN denotes false-negative, and FP indicates false-positive.

The parameters of the embedding layer of FM, DNN, DeepFM, and Wide&Deep in this experiment were fully connected weights. We encoded the experimental data with one-hot and input the encoded high-dimensional sparse data into the embedding layer for dimension reduction. The embedding dimension of the embedding layer in this experiment was k = 5, and the domain value was 14. We used Adam to optimize the model training, and the value of the experimental epoch was in the range of [0, 40]. When the loss function increases several times in the epoch, when the value of the function does not rise or fall, the training model can end the training. Neural networks often have overfitting problems during model training, and the appearance of overfitting on the training set reduces the model's generalization ability. When the model was tested on the test set, the experimental results differed significantly from the training results, and the model had no practical application value. To avoid overfitting the model, we increased the amount of data during data training to make the model's training data more complete and to enhance the model's generalization ability. Another method involves randomly dropping some neurons in the hidden layer of the model. The method of discarding entails randomly dropping them. When the dropout value is 0.8, the model's neurons should be evenly dropped. The absence of model neurons plays a role in solving model overfitting. The experimental results, particularly the AUC and LogLoss values, are shown in Table 2.

The experimental results are shown in Fig. 2 regarding the value of the loss function and Fig. 3 regarding the AUC value as the epoch value increases.

In the experimental findings, the DeepFM algorithm's loss function value—which was less than that of Wide&Deep and DNN—was also its greatest AUC value. This verified that the recommendation effect of the DeepFM model was better. The test results were also relatively stable. In the experiment, the FM model converged at a faster rate because it could only process second-order eigenvectors at the highest and could not be calculated for high-order eigenvectors. DNN had the slowest convergence, a long learning time for high-order features, and a significant difference between the test results and the training results, which was unstable. Most second-order features were more common.

NeuMF Experimental Analysis

The neural collaborative filtering paper had to conduct pretraining experiments before training NeuMF, initialize the model's parameters, use one-hot encoding to encode users and commodity items in the same experimental data, and separately embed them via embedding. The dimension of the user's latent factor was 32; that is, the model embedding size was 32 for model training. The number of

Table 2. AUC and LogLoss index values after each model converges

Algorithm	AUC	LogLoss
FM	0.782	0.109
DNN	0.778	0.114
Wide&Deep	0.787	0.105
DeepFM	0.796	0.101

Figure 2. Loss function values for each model

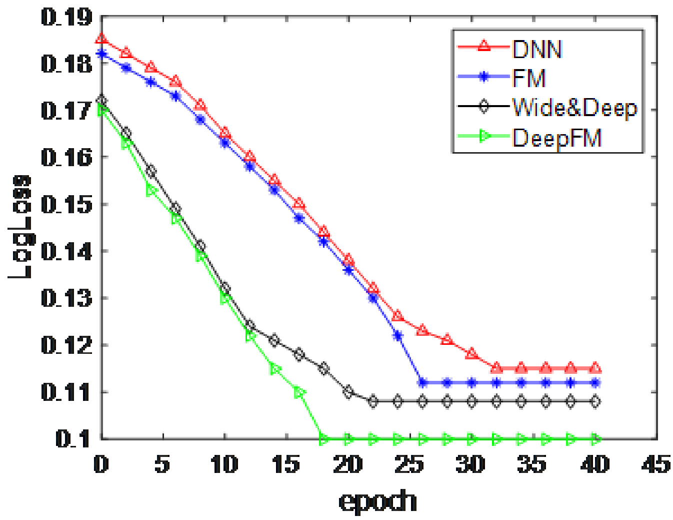
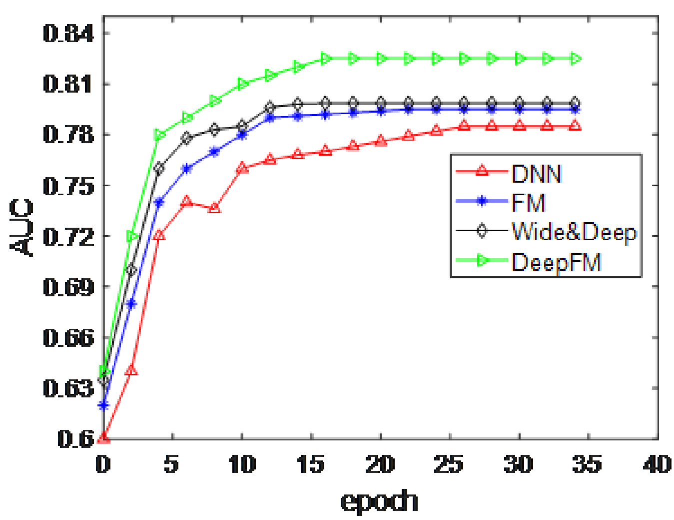


Figure 3. AUC values for each model



hidden layers of the MLP model was four with a stack structure of [200, 128, 64, 32], and the GMF model did not contain a hidden layer. We conducted accelerated training of experiments with an Adam optimizer. During the training process, the data were divided into batches for training. The size of each batch of data was 50,000. The value of the epoch was 40. The experimental recommendation structure was the top-K recommendation type. The number of recommended products K was 20, and the experimental evaluation index was compared with the loss function and the number of hits. In the top-k recommendation, HR is a commonly employed indicator to measure the recall rate. The calculation formula is:

$$HR@K = \frac{Number\,of\;Hits\;\;@K}{|GT|} \tag{21}$$

where $umber\,of\;Hits\;\;@K$ is all test sets and $|GT|$ is the sum of the number of test sets in each user's top-K recommendation list.

The LogLoss and HR@20 values of GMF, MLP, and NeuMF after model convergence are shown in Table 3.

The NeuMF model had the lowest loss function value, and its HR@20 value was higher than those of both GMF and an MLP. Fig. 4 shows the change trends of the log loss and HR@20 indicators as the number of epochs increased during the training processes of the GMF, MLP, and NeuMF models.

The values of LogLoss and HR@20 changed with the increase in epoch, as shown in Fig. 4. The loss function decreased rapidly before epoch 10 and decreased slowly after the 10th epoch. When the number of epochs exceeded 10, the model training process tended to be stable, and the loss function of the NeuMF continued to decrease as the number of iterations increased. The value of HR@20 could also be verified. When the number of epochs was greater than 10, it continued to increase slowly, and the NeuMF more effectively performed the recommendation process.

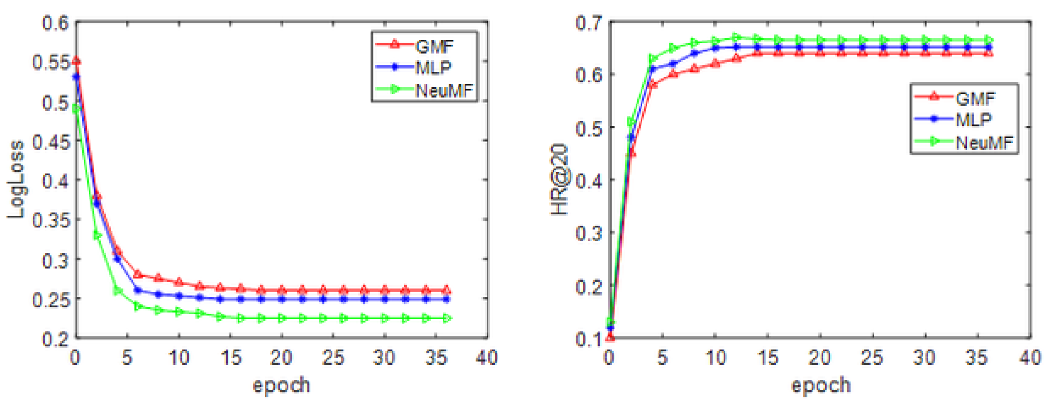
CONCLUSION

To enhance the overall performance of recommendation systems and more accurately deliver products to users based on their interests, this research has primarily focused on product recommendations within distinct life cycles. It established a recommendation framework designed to align with the specific characteristics of products at different stages of their life cycles, thus enabling the selection

Table 3. LogLoss and HR@20 values after model convergence

Algorithm Model	LogLoss	HR@20
GMF	0.283	0.636
MLP	0.264	0.652
NeuMF	0.232	0.688

Figure 4. Comparison chart LogLoss and HR@20



of suitable recommendation algorithms. Consequently, this study delves into the recommendation of various products throughout their life cycles.

Addressing shortcomings related to content information extraction, data sparsity, and cold start challenges within recommendation technology, this paper thoroughly assessed several diverse deep learning models and their potential applications in recommendation systems. The primary research objective was to harness deep learning technology and integrate it into recommender systems to address data sparsity issues and the difficulty of uncovering user preferences.

The research presented in this paper can be summarized as follows:

1) A comprehensive exploration of the theories related to deep learning and recommender systems was provided, emphasizing the introduction of traditional recommender system algorithms, deep learning–based development frameworks, and the Django development framework.
2) A thorough analysis of the DeepFM and NeuMF models was performed to resolve the data sparsity and information overload challenges within recommendation systems. Both models were evaluated to determine their ability to recommend implicit feedback information, employ one-hot encoding for model embedding purposes, and undergo rigorous model training processes. The DeepFM model exhibited the most favorable results, including the lowest loss function value and the highest AUC. The NeuMF recommendation model, while delivering lower loss values than the two individual models, achieved the highest HR@20.
3) An in-depth analysis of system requirements was performed, leading to the development of a personalized product recommendation system using the Django framework. The recommended approach involved the utilization of trained DeepFM and NeuMF recommendation models.

In summary, this research contributes significantly to the field of recommendation systems by introducing a framework capable of adapting recommendation algorithms to product life cycles, effectively addressing issues such as data sparsity and information overload. The incorporation of deep learning techniques, particularly through the DeepFM and NeuMF models, holds promise for enhancing the performance of recommendation systems. The study could be further enhanced by implementing the proposed model on a larger dataset and considering more evaluation metrics. Additionally, explainable AI concepts could be included to achieve more confidence, transparency, and explainability for the generated results.

AVAILABILITY OF DATA AND MATERIAL

The datasets used for the current study are available from the corresponding author upon reasonable request.

COMPETING INTERESTS

The authors of this publication declare that there are no competing interests.

FUNDING

This research was supported by the Scientific Research Fund of Zhejiang Provincial Education Department (Y202352165).

REFERENCES

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, *17*(6), 734–749. doi:10.1109/TKDE.2005.99

Ankora, C., & Aju, D. (2022). Integrating user stories in the design of augmented reality application. *International Journal of Information Technologies and Systems Approach*, *15*(1), 1–19. doi:10.4018/IJITSA.304809

Balabanović, M., & Shoham, Y. (1997). Fab: Content-based, collaborative recommendation. *Communications of the ACM*, *40*(3), 66–72. doi:10.1145/245108.245124

Choi, J. W., Yun, S. K., & Kim, J. B. (2020). Improvement of data sparsity and scalability problems in collaborative filtering based recommendation systems. *Applied Computing and Information Technology*, 17–31.

Choi, S. M., Ko, S. K., & Han, Y. S. (2012). A movie recommendation algorithm based on genre correlations. *Expert Systems with Applications*, *39*(9), 8079–8085. doi:10.1016/j.eswa.2012.01.132

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., & Darrell, T. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the International Conference on Machine Learning* (pp. 647–655). PMLR.

Fan, R. X. (2019). *Research on personalized e-commerce product recommendation based on deep neural network* [Unpublished master's thesis]. Hefei University of Technology, China.

Gao, O. U., Wang, H., & Jiang, Y. (2019). *Hybrid recommendation algorithm based on deep learning*. Transducer and Microsystem Technologies.

Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, *18*(7), 1527–1554. doi:10.1162/neco.2006.18.7.1527 PMID:16764513

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. doi:10.1162/neco.1997.9.8.1735 PMID:9377276

Huang, G., Chen, Q., & Deng, C. (2020). A new click-through rates prediction model based on Deep&Cross network. *Algorithms*, *13*(12), 342. doi:10.3390/a13120342

Kautz, H., Selman, B., & Shah, M. (1997). Referral Web: Combining social networks and collaborative filtering. *Communications of the ACM*, *40*(3), 63–65. doi:10.1145/245108.245123

Koren, Y. (2009). Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 447–456). Association for Computing Machinery. doi:10.1145/1557019.1557072

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, *42*(8), 30–37. doi:10.1109/MC.2009.263

Li, C., & Zhang, Y. (2020). A personalized recommendation algorithm based on large-scale real micro-blog data. *Neural Computing & Applications*, *32*(15), 11245–11252. doi:10.1007/s00521-020-05042-y

Li, M., Bi, X., Wang, L., Han, X., Wang, L., & Zhou, W. (2022). Text similarity measurement method and application of online medical community based on density peak clustering. *Journal of Organizational and End User Computing*, *34*(2), 1–25. doi:10.4018/JOEUC.302893

Li, W. X., Wen, Y. J., & Tang, L. J. (2019). Research and implementation of personalized recommendation method for educational resources. *Computer Technology and Development*, *29*(6), 18–22.

Liu, N. N., He, L., & Zhao, M. (2013). Social temporal collaborative ranking for context aware movie recommendation. *ACM Transactions on Intelligent Systems and Technology*, *4*(1), 1–26. doi:10.1145/2414425.2414440

Liu, Y., & Zhu, W. H. (2020). A hybrid recommendation algorithm combination of content-based and tag weight. *Computer & Digital Engineering*, *366*(4), 44–48.

Liu, Y. J., & Xia, C. (2014). Learning neural network application in speech recognition deeply. *Network Security Technology & Application*, *12*, 28–30.

McAuley, J., Targett, C., Shi, Q., & Van Den Hengel, A. (2015). Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 43–52). Association for Computing Machinery.

Sukte, C. D., Mark, E., & Deshmukh, R. R. (2022). Efficient cryptographic protocol design for secure sharing of personal health records in the cloud. *International Journal of Information Technologies and Systems Approach*, *15*(1), 1–16. doi:10.4018/IJITSA.304810

Tang, J., & Huang, J. (2021). Research on content-based reciprocal recommendation algorithm for student resume. In *Proceedings of the 10th International Conference on Computer Engineering and Networks* (pp. 233–239). Springer Singapore. doi:10.1007/978-981-15-8462-6_27

Tang, Y., & Zhang, P. (2022). The impact of virtual integration on innovation speed: On the view of organizational information processing theory. *Journal of Organizational and End User Computing*, *34*(1), 1–20. doi:10.4018/JOEUC.298702

Terveen, L., Hill, W., Amento, B., McDonald, D., & Creter, J. (1997). PHOAKS: A system for sharing recommendations. *Communications of the ACM*, *40*(3), 59–62. doi:10.1145/245108.245122

Wang, Y., & Li, X. (2017). Study on improved clustering collaborative filtering algorithm based on demography. *Computer Science*, *44*(3), 63–69.

Wang, Z., Shi, P., & Wu, C. (2020). A fatigue driving detection method based on deep learning and image processing. *Journal of Physics: Conference Series*, *1575*(1), 012035. doi:10.1088/1742-6596/1575/1/012035

Wu, C. Y., Ahmed, A., Beutel, A., & Smola, A. J. (2017). Joint training of ratings and reviews with recurrent recommender networks. *Proceedings of the International Conference on Learning Representations*.

Xiaofei, L., Fei, Z., Yuchen, F., & Quan, L. (2020). Personalized recommendation algorithm based on user preference feature mining. *Computer Science*, *47*(4), 56–59.

Yan, W., Li, S., & Cheng, Y. (2020). Multiple similarity collaborative filtering recommendation among users. *IOP Conference Series. Materials Science and Engineering*, *768*(7), 072010. doi:10.1088/1757-899X/768/7/072010

Yu, B., Li, Z. H., & Zhang, K. (2019). Advertisement recommendation system based on DeepFM model. *Computer Applications and Software, 36*(7), 307–310, 316.

Yue, J., Zhao, W., Mao, S., & Liu, H. (2015). Spectral-spatial classification of hyperspectral images using deep convolutional neural networks. *Remote Sensing Letters*, *6*(4–6), 468–477. doi:10.1080/2150704X.2015.1047045

Yue, X., Tang, R., & Shu, H. P. (2020). Research on improvement of collaborative filtering recommendation algorithm based on data sparseness. *Journal of Sichuan University*, *52*(001), 198–202.

Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys*, *52*(1), 1–38. doi:10.1145/3158369