Hindawi
Journal of Electrical and Computer Engineering
Volume 2023, Article ID 9845450, 1 page
https://doi.org/10.1155/2023/9845450

Retraction

Retracted: CCPIN: Classification and Combine Parallel Interaction Network for CTR Prediction

Journal of Electrical and Computer Engineering

Received 19 December 2023; Accepted 19 December 2023; Published 20 December 2023

This article has been retracted by Hindawi following an investigation undertaken by the publisher [1]. This investigation has uncovered evidence of one or more of the following indicators of systematic manipulation of the publication process:

(1) Discrepancies in scope
(2) Discrepancies in the description of the research reported
(3) Discrepancies between the availability of data and the research described
(4) Inappropriate citations
(5) Incoherent, meaningless and/or irrelevant content included in the article
(6) Manipulated or compromised peer review

The presence of these indicators undermines our confidence in the integrity of the article's content and we cannot, therefore, vouch for its reliability. Please note that this notice is intended solely to alert readers that the content of this article is unreliable. We have not investigated whether authors were aware of or involved in the systematic manipulation of the publication process.

Wiley and Hindawi regrets that the usual quality checks did not identify these issues before publication and have since put additional measures in place to safeguard research integrity.

We wish to credit our own Research Integrity and Research Publishing teams and anonymous and named external researchers and research integrity experts for contributing to this investigation.

The corresponding author, as the representative of all authors, has been given the opportunity to register their agreement or disagreement to this retraction. We have kept a record of any response received.

References

[1] G. Tan, C. Yang, and J. Jiang, "CCPIN: Classification and Combine Parallel Interaction Network for CTR Prediction," *Journal of Electrical and Computer Engineering*, vol. 2022, Article ID 7093457, 12 pages, 2022.

Hindawi
Journal of Electrical and Computer Engineering
Volume 2022, Article ID 7093457, 12 pages
https://doi.org/10.1155/2022/7093457

Research Article

CCPIN: Classification and Combine Parallel Interaction Network for CTR Prediction

Guosheng Tan, Changchun Yang, and Jiaming Jiang

School of Computer Science and Artificial Intelligence, Changzhou University, Changzhou, Jiangsu 213164, China

Correspondence should be addressed to Changchun Yang; ycc@cczu.edu.cn

Received 24 April 2022; Revised 9 May 2022; Accepted 16 May 2022; Published 30 May 2022

Academic Editor: Xuefeng Shao

The study of feature interactions in deep neural network-based recommender systems has been a popular research area in industry and academic circles. However, the vast majority of parallel CTR prediction models do not classify the input features but instead feed them into the model. This way not only reduces the accuracy of the model but also ignores the effectiveness of learning individual feature interactions. In addition, the majority of parallel CTR prediction models only focus on the submodel intersections of their parallel models, ignoring the importance of the external intersection. To address the shortcomings, this paper proposes the CCPIN model on the basis of the XdeepFM model. In the CCPIN model, it can not only learn different category feature interactions but also learn individual feature interactions. Through the classification gate, adaptive features are maximized to improve the performance of the submodel. Through the Combine layer, the interaction of submodel results can be learned while retaining the original output. Through comparison experiments with other models on two datasets, it is demonstrated that the CCPIN model has an average increase of 0.93% in AUC and a decrease of 0.47% in Logloss compared to other models.

1. Introduction

With the rapid development of the Internet, the way people receive information has changed dramatically [1]. The way people get information has changed from active to passive access. The active reception of information has grown rapidly, reaching a peak in the last decade, such as Google search [2] and Baidu search [3]. Passive information acquisition is also known as a recommendation system, and it has grown dramatically in recent years.

Today, recommendation systems are one of the machine learning study topics, and they are a significant element of today's organizations and core businesses. The concept of the recommendation system was first proposed in the 1990s [4]. With the scholars' continuous research and development, up to now, the recommendation system can be divided into three categories: click-through rate (CTR) prediction [5], rating prediction [6], and top-N recommendation [7]. In this paper, the recommendation system study is CTR prediction.

Typically, the CTR prediction issue is considered a binary classification task. In the model, clicks are usually set to 1, and no clicks are set to 0. The most traditional CTR classification model is the logistic regression (LR) [8]. The LR model has occupied the traditional industrial recommendation system for a period of time because of its simplicity, speed, and certain accuracy. But since the LR model is too simple to learn no-linear features, it is quickly overwhelmed by the trend of neural networks. With the rise of neural networks, learning no-linear feature interactions and studying feature intersections have become a new wave of advancing CTR prediction problems. Scholars have found that deep neural networks (DNN) are very suitable for learning no-linear feature interactions. Based on this trend, Cheng et al. proposed the Wide and Deep [9] model and introduced a parallel model for the first time to solve the problem of click-through rate estimation. The Wide part helps to enhance the memory capability, and the Deep part helps to enhance the generalization capability, but the model still relies on manual feature engineering. On the basis of the Wide and Deep model, Guo et al. proposed the DeepFM [10] model. In the DeepFM model, the Factorization Machine (FM) [11] learns display features, and the DNN model learns implicit features. As a result, it achieved good performance. But this model can only automatically construct first-order and second-order features and cannot learn higher-order features. The DeepFM model can only construct second-order feature interactions at most, and the Deep and Cross Network (DCN) [12] model was proposed by Wang. In the DCN model, the Cross Net automatically constructs high-order feature interactions, which greatly improves the learning ability of high-order explicit features. But the DCN model uses bitwise feature interactions, which cannot learn the vectorwise feature interactions. Because of this, Lian et al. proposed a new compression model, which is named XdeepFM [13] model. It replaces bitwise with vectorwise to improve the model accuracy. But the XdeepFM models ignore the necessity to learn the intersection of individual features. So Yu et al. proposed the XCrossNet [14] model, which uses a three-layer innovative separate classification to learn separate features for subsequent cross processing, thereby improving the accuracy of the model. Although the XCrossNet model takes into account the necessity of learning features individually, it does not take into account the fact that feature inputs need to be classified and ignores the effect that features can cause noise in the model.

In this paper, Classification and Combine Parallel Interaction Network (CCPIN), a recommendation model, is proposed. The CCPIN model uses the classification gate layer. The classification gate layer is inspired by the MMOE model of the multitask recommender system. It can extract weights to classify features and fully maximize the power of the parallel model. At the same time, based on the XdeepFM model, this paper introduces a parallel model to explore the results of learning different types of feature pairs separately. Finally, merge the layer outputs through the model. This paper's contributions are summarized as follows:

(i) Inspired by the multitask recommendation system, this paper proposes a classification gate layer for feature classification, then uses a classification gate layer, and sends the features into suitable models adaptively. Not only that, the classification gate layer reduces the volume of useless feature input. Therefore, it may enhance the training data, which will effectively boost the model's generalization capabilities.

(ii) By adding a model to the parallel structure, the CCPIN model successfully made the model have the function of learning numerical feature intersection and categorical feature intersection independently. The newly proposed model significantly enhances the model's performance by adding a separate learning category for interactions [15].

(iii) By adding a Combine layer, the CCPIN model uses different parallel models to perform secondary output cross-merging and finally send them to the output. It can increase the model's breadth for feature learning. Experiments show that the Combine layer proposed in this paper has a certain improvement in performance.

(iv) By testing the model on two public datasets, we found that the proposed model in this paper outperforms the majority of CTR models on two evaluation metrics. Also, test the classification layer, the Combine layer, and the new parallel model's efficacy.

The remainder of this paper is structured as follows. Recent work pertaining to our suggested model is reviewed in Section 2. Each part of the CCPIN model is then detailed in Section 3. In Section 4, this paper does experiments on two datasets that are publicly available. Finally, in Section 5, this paper concludes with a brief conclusion and a suggestion for future work.

2. Related Work

Studies have shown that DNN-based parallel recommender systems models are often better than traditional ones, such as collaborative filtering [16] and Gradient Boosting Decision Trees (GBDT) [17]; therefore, the role of DNN for learning no-linear features in the parallel model is indispensable.

The Wide and Deep model was the first to use a deep neural network in a recommendation system, combining it with LR. In the Wide and Deep model, the Wide part uses the memory of LR, and the Deep part takes advantage of DNN's generalization capabilities to extract implicit feature relationships. But the model still relies on manual feature engineering, which consumes a lot of human resources, and in addition, the model is unable to learn feature interactions.

The DeepFM model can learn low-level features and high-level feature information at the same time. It uses FM and Deep parts to share the input layer and the embedding layer. In the DeepFM model, the FM part automatically constructs the 1-order and 2-order feature interactions. It can eliminate the tediousness of manually constructing feature interactions. But this model can only automatically construct first-order and second-order features and cannot construct higher-order features. This can result in underutilization of the sample, and the accuracy is not improved to the extreme.

The DCN model uses Cross Net to automatically construct high-order features. It can greatly improve the learning ability of high-order explicit features. The DNN is used to extract implicit features and then Combine them for output. This can greatly improve the accuracy of the model. The DCN model, on the other hand, employs bitwise feature intersection, which cannot learn vectorwise feature intersection.

The XdeepFM model proposes a new compression model to modularize the functional interaction network. It uses vectorwise to replace bitwise to improve the accuracy of the model, providing a new idea for subsequent scholars. But the above parallel models all ignore the need to learn the intersection of individual features.

The XCrossNet model learns individual features through a three-layer innovative separate classification and then performs cross processing. Through experiments, it was demonstrated that the learning crossover of individual features also has a certain impact on the accuracy of the model. Although the XCrossNet model takes into account the necessity of learning features individually, it does not take into account the fact that feature inputs need to be classified and ignores the feature noise effect.

The above work improves the accuracy of recommender system models by presenting several feature architectures and interaction techniques [18]. But they only consider how the features are built and how they interact. This leads to ignoring different features that are suitable for different models and does not classify the models, thereby reducing the accuracy of the model. In addition, since parallel models often directly send parallel results into the model, ignoring several parallel models can also learn features through cross-learning. Therefore, this paper proposes a CCPIN model with a classification layer and a Combine layer.

3. Classification and Combine Parallel Interaction Network

This section introduces the CCPIN model, which estimates user preferences for target clicks based on feature classification, feature merging, and feature intersection. The structure of CCPIN is shown in Figure 1. From the structural framework of the CCPIN model, there is a parallel deep neural network, and the CTR score is eventually determined by the CCPIN model. The next subsections will go through each section of the CCPIN in depth.

3.1. Item Embedding Layer. In order to better predict user behavior in complex display environments, recommender systems often collect a large amount of data, including users' personal information (age, gender, name, work, etc.), and even contextual information (workday, location, browsing history, etc.) will also be collected to construct a training dataset [19]. In the case of numerical features (bid, purchase quantity, etc.), in order to model processing, the usual method is to discretize and convert them into categorical features. Usually, the way is through one-hot encoding [20]. The following is an example (Gender = male, Age = 18, ..., Weekday = Monday):

$$\mathbf{x} = \underbrace{[1,0]}_{\text{Gender}}\ \underbrace{[0,1,0,\ldots,0]}_{\text{Age}} \cdots \underbrace{[1,0,0,\ldots,0]}_{\text{Weekday}}. \tag{1}$$

For the parallel-structured CTR model, the one-hot encoding often makes the features too sparse. Via feature embedding, each sparse vector is generally transformed into a low-dimensional dense vector [21]. The feature embedding can be obtained for the i-th categorical field by the following formula:

$$\begin{aligned} \mathbf{e}_i &= \mathbf{W}_{embed} \cdot \mathbf{x}_i, \\ \mathbf{E} &= \left[\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_f\right], \end{aligned} \tag{2}$$

where $\mathbf{e}_i$ is the embedding vector of the feature, $\mathbf{W}_{\text{embed}} \in \mathbf{R}^{v_i \times k}$, $\mathbf{W}_{\text{embed}}$ is the embedding matrix of the i-th feature domain, and v_i and k are the input dimension and the embedding vector dimension, respectively. $\mathbf{x}_i$ is the one-hot vector of the i-th feature, $\mathbf{E}$ represents the embedding, and f denotes the number of fields.

3.2. Classification Gate. The parallel model accepts the output from the embedding layer and then directly into the parallel model. But completely ignoring the features will have a negative effect on the model [22]. So, inputting suitable features for the model may have a positive effect. Based on this situation, this paper introduces the classification gate. It is inspired by the idea of a multitask model [23]. In the classification gate, it uses each fieldwise gating network to discriminate the feature distribution of the parallel network. The fieldwise gating network is based on the soft-select principle so that the model may completely learn appropriate features. Therefore, the classification gate layer $\mathbf{C}^m = [\mathbf{c}_1^m, \mathbf{c}_2^m, \ldots, \mathbf{c}_f^m]$ is proposed, where $\mathbf{m} = [\mathbf{D}, \mathbf{C}, \mathbf{N}]$ represents different parallel networks and $\mathbf{c}_i^m$ represents the weight of the i-th field of the respective classification gate. So the $\mathbf{c}_f^m$ formula is as follows:

$$\widehat{\mathbf{c}_i^m} = \frac{\mathbf{e}^{1/\tau}\mathbf{c}_i^m}{\sum_{j=1}^{f}\mathbf{e}^{1/\tau}\mathbf{c}_j^m}, \tag{3}$$

where τ is the classification gate coefficients to control classification, m represents the parallel network, and $\mathbf{c}_i^m$ represents the weight of the i-th field of the classification gate. So, the classification gate $\mathbf{E}^m$ for parallel network m is defined as

$$\mathbf{E}^m = \widehat{\mathbf{C}^m} \odot \mathbf{E} = \left[\widehat{\mathbf{c}_1^m}\mathbf{e}_1\widehat{\mathbf{c}_2^m}\mathbf{e}_2, \ldots, \widehat{\mathbf{c}_f^m}\mathbf{e}_f\right], \tag{4}$$

where $\mathbf{C}^m$ represents the m parallel network's classification gate layer, $\mathbf{E}$ represents the embedding, and $\mathbf{E}^m$ represents the feature weight of the classification gate layer.

As shown in Figure 1, the features input into the classification layer is selected and entered into three different models, so the model cannot be disturbed by a large number of unsuitable features. In this way, it can increase the model's learning efficiency and accuracy.

3.3. Item Parallel Layer. As shown in Figure 1, the parallel layer of CCPIN is based on the XdeepFM model. In order to fill the XdeepFM model's inability to learn different types of defects independently, the parallel layer of the CCPIN model consists of three models, namely, Double Cross Net, Compressed Interaction Network (CIN), and DNN. Next, it will be introduced separately.

3.3.1. Double Cross Net. Based on the inspiration of the XCrossNet [14] model, a double cross model is proposed. The model can learn numerical features and categorical features independently. This compensates for the previous XdeepFM model's inability to learn the interactions of separate types of features. The model structure of the Double Cross is shown in Figure 2. In this paper, the model will be divided into two structures, left and right, and will be explained in detail.

Cross Layer on Dense Feature. As can be seen from Figure 2, the dense features are interactions through the cross layer. This structure draws on the Cross Net in the DCN structure.

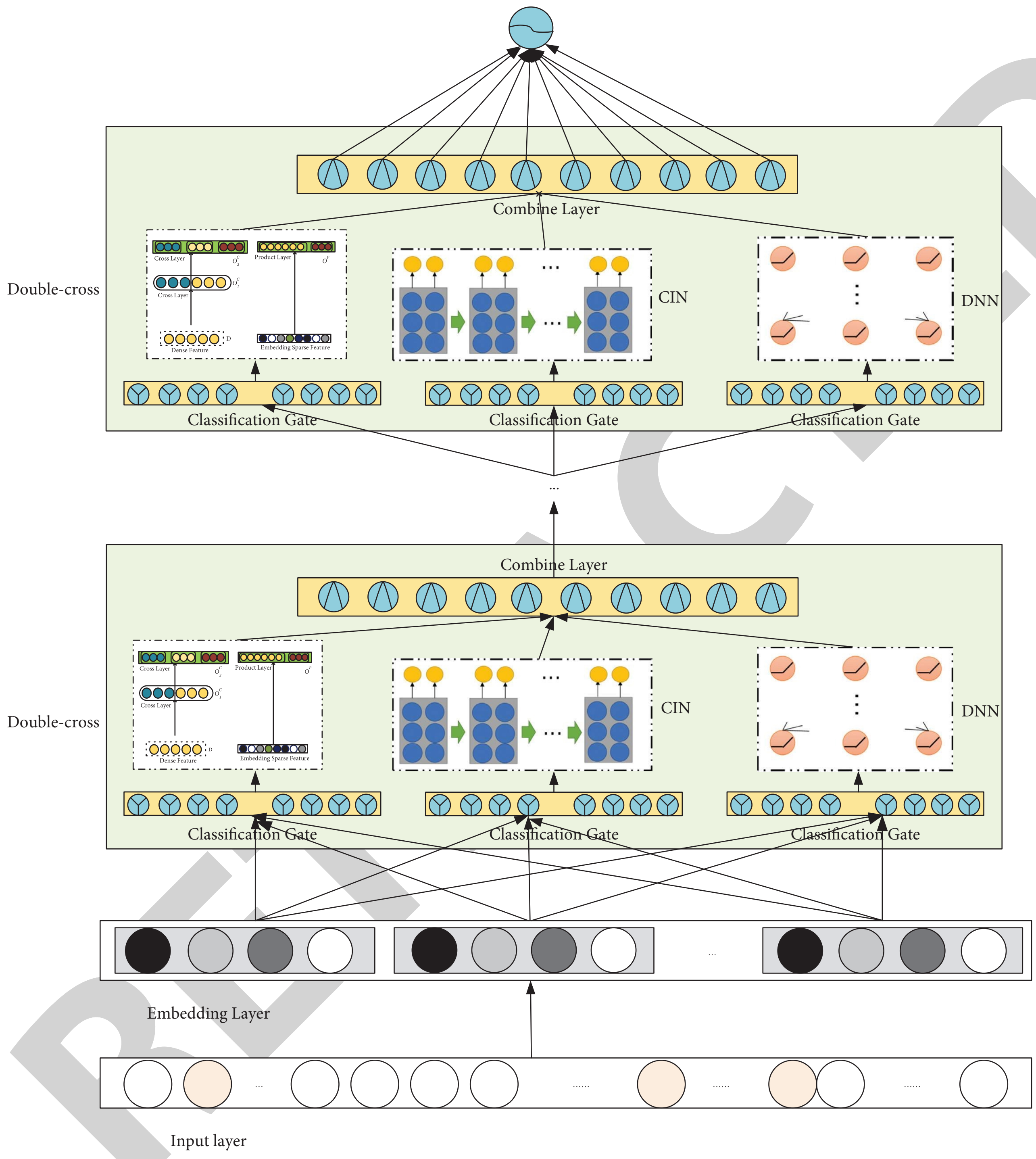

FIGURE 1: The CCPIN model enters three parallel submodels through the classification layer and then outputs the estimated results by merging and crossing.

The formula of the l layer of the cross layer can be observed from Figure 3 as follows:

$$\begin{aligned}\mathbf{C}_1 &= \mathbf{D}\cdot\mathbf{D}^{\mathrm{T}}\cdot\mathbf{W}_{\mathrm{C},0}+\mathbf{b}_{\mathrm{C},0}, \mathbf{O}_1^{\mathrm{C}}=[\mathbf{D};\mathbf{C}_1],\\ \mathbf{C}_{l+1} &= \mathbf{D}\cdot\mathbf{C}_l^{\mathrm{T}}\cdot\mathbf{W}_{C,l}+\mathbf{b}_{C,l}, \mathbf{O}_{l+1}^{C}=\left[\mathbf{O}_l^{C};\mathbf{C}_{l+1}\right],\end{aligned} \tag{5}$$

where $\mathbf{D}$ represents the dense feature of the input, $\mathbf{C}_1$ represents the 1-th layer of cross feature, and $\mathbf{W}_{\mathrm{C},0}$ and $\mathbf{b}_{\mathrm{C},0}$ are denoted as 1-th computational weight and bias parameters, respectively. Similarly, $\mathbf{C}_{l+1}$, $\mathbf{W}_{C,l}$, and $\mathbf{b}_{C,l}$ represent the l-th layer cross feature, weight, and bias parameters, respectively. $\mathbf{O}_1^{\mathrm{C}}$ and $\mathbf{O}_{l+1}^{C}$ are the outputs of the l-th and the $(l+1)$-th layers, respectively.

Product Layer on Embedding Sparse Features. As can be seen from Figure 2, the embedding layer converts the sparse

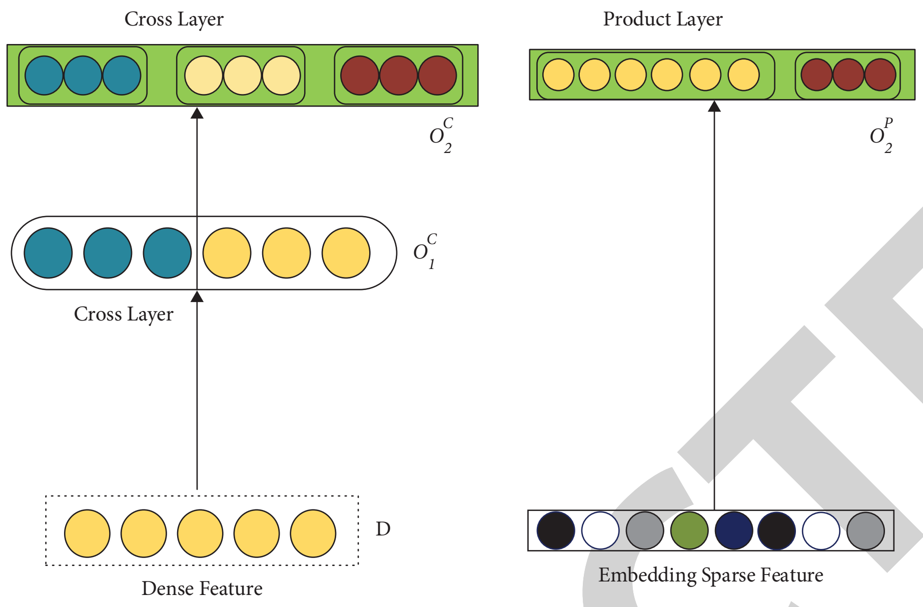

Figure 2: The network structure of Double Cross Net, including the cross layer and product layer.

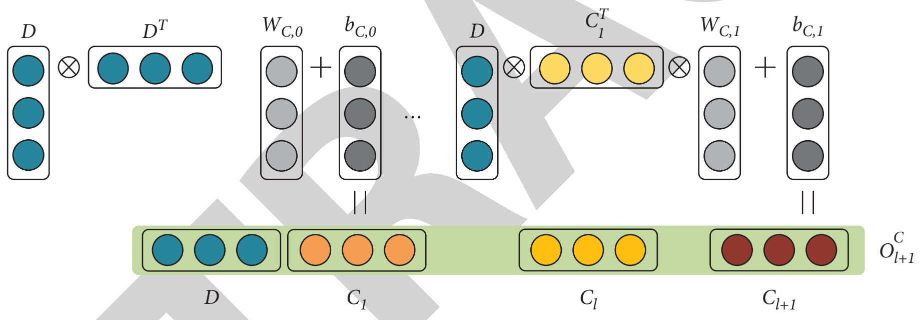

Figure 3: The structure of cross layer in Double Cross Net.

vector and then enters the product layers. In Figure 4, the two splicing processes are shown. $\odot$ means the inner product, $\mathbf{O}^P$ represents the output of the product layer, $\mathbf{O}^P = [\mathbf{P}_1; \mathbf{P}_2]$, and $\mathbf{P}_1$ and $\mathbf{P}_2$ represent the 1st-order and 2nd-order intersection embedding sparse features. The formula is as follows:

$$\begin{aligned}
\mathbf{P}_1 &= \left[\mathbf{P}_1^1, \ldots, \mathbf{P}_1^t, \ldots, \mathbf{P}_1^T\right], \\
\mathbf{P}_1^t &= \sum_{i=1}^{N} < \mathbf{W}_i^{1,t}, \mathbf{E}_i >, \\
\mathbf{P}_2 &= \left[\mathbf{P}^2, \ldots, \mathbf{P}_2^t, \ldots, \mathbf{P}_2^T\right], \\
\mathbf{P}_2^t &= \sum_{i=1}^{N}\sum_{j=1}^{N} \mathbf{W}_{i,j}^{2,t} < \mathbf{E}_i, \mathbf{E}_j >.
\end{aligned} \tag{6}$$

In the formula, the calculation process of $\mathbf{P}_1$ is that each feature vector and the weight vector are first inner products and then summed. After that, a single product layer can obtain a one-dimensional constant $\mathbf{P}_1^t$. In order to make the cross feature output as a vector, multiple sets of weights are taken. here, t is the number of product layers; The calculation process of $\mathbf{P}_2$ is that features are combined in pairs, the inner product is calculated, and then the weighted summation is performed to obtain a one-dimensional constant $\mathbf{P}_2^t$, and multiple sets of weights are also adopted to make the feature output as a vector.

3.3.2. CIN. The CIN model is a part of the model XdeepFM. It is an improvement to the high-order feature intersection in the DCN network. In the CIN model, the output of each layer is the input of the next layer, and the input of each layer will interact with the initial input X of the model. Through the interaction, the model obtained an intermediate result and then convolved to obtain the last output of the layer. The general structure is shown in Figure 5.

The first step of its CIN is explained separately in the form of Figure 6. After the embedding layer, $\mathbf{X}^0$ is obtained, and the shape size is $m \times D$, where m is composed of multiple field vectors obtained after embedding and D is the size of the field feature. Suppose the CIN structure has k layers, the output result of each layer is $\mathbf{X}^k$, the result of $\mathbf{X}^k$ is related to $\mathbf{X}^0$ and $\mathbf{X}^{k-1}$, and the calculation formula is

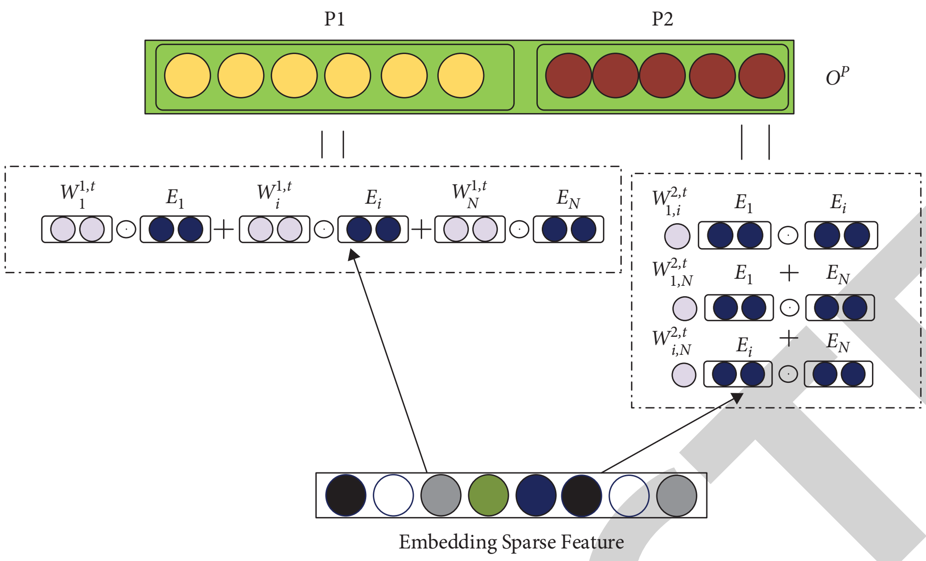

Figure 4: The structure of product layer in Double Cross Net.

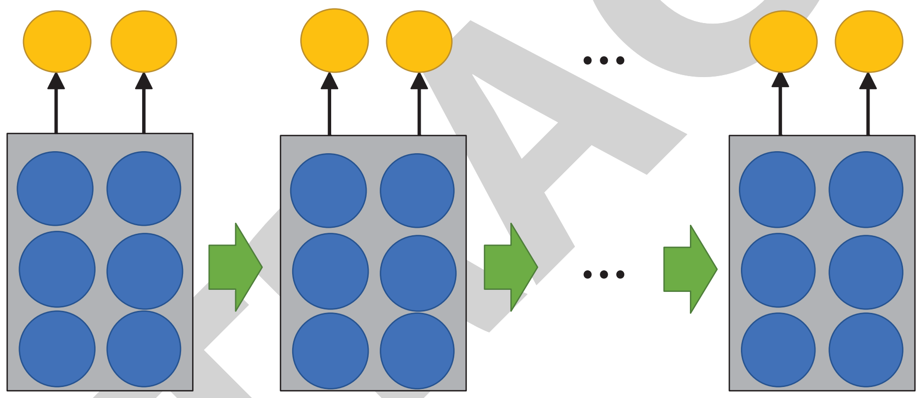

Figure 5: The structure of CIN.

$$\mathbf{X}^k_{h,*} = \sum_{i=1}^{H_{k-1}} \sum_{j=1}^{m} \mathbf{W}^{k,h}_{ij} \left(\mathbf{X}^{k-1}_{i,*} \bigcirc \mathbf{X}^0_{j,*}\right), \tag{7}$$

where $\mathbf{X}^k_{h,*}$ represents the output of the k-th layer, $W^{k,h} \in \mathbf{R}^{H_{k-1}*m}$ represents the h-th vector weight matrix of the k-th layer, and $\bigcirc$ represents the Hadamard product.

3.3.3. DNN. The DNN accepts the vector output from the embedding layer. The DNN is mainly used to learn implicit features. The l layer of the DNN layer's formula is

$$F_l(\mathbf{h}_l) = \mathrm{ReLU}(\mathbf{h}_l \mathbf{W}_l + \mathbf{b}_l). \tag{8}$$

$F_l(\mathbf{h}_l)$, $\mathbf{W}_l$, $\mathbf{b}_l$, and $\mathbf{h}_l$ represent the output vector, weight matrix, bias vector, and input vector of the l-th layer, respectively.

3.4. Combine Layer. Existing parallel deep CTR models learn explicit and implicit features separately through parallel submodels, but often the networks are executed independently and simply spliced to the final output layer. This type of model output processing significantly weakens the correlation between various models.

In order to enhance the correlation between submodels' outputs, the Combine layer is proposed. The Combine layer is inspired by the Cross Net network in the DCN model and continues the output feature interactions while at the same time retaining the original input fed together to the output. The Combine layer makes submodels' outputs crossed twice to supply the output layer for learning [24]. By retaining the splicing vector $\mathbf{X}_0$ of the original submodel, a secondary submodel cross vector $\mathbf{X}_1$ is added, and the formula is as follows:

$$\begin{gathered}\mathbf{X}_0 = \left[\mathbf{O}^D; \mathbf{O}^C; \mathbf{O}^N\right],\\ \mathbf{X}_1 = \mathbf{X}_0 \cdot \mathbf{X}_0^{\mathrm{T}} \cdot \mathbf{W}_{X,0} + \mathbf{b}_{X,0}, \mathbf{H}_0 = \left[\mathbf{X}_0; \mathbf{X}_1\right],\end{gathered} \tag{9}$$

where $\mathbf{X}_0$ represents the connection of the output of the double cross output, the CIN output, and the DNN output. $\mathbf{X}_1$ represents the three parallel model features obtained through feature Combine. $\mathbf{H}_0$ represents the output of the combined layer. $\mathbf{W}_{X,0}$ and $\mathbf{b}_{X,0}$ represent the weights and bias parameters for the Combine layer, respectively.

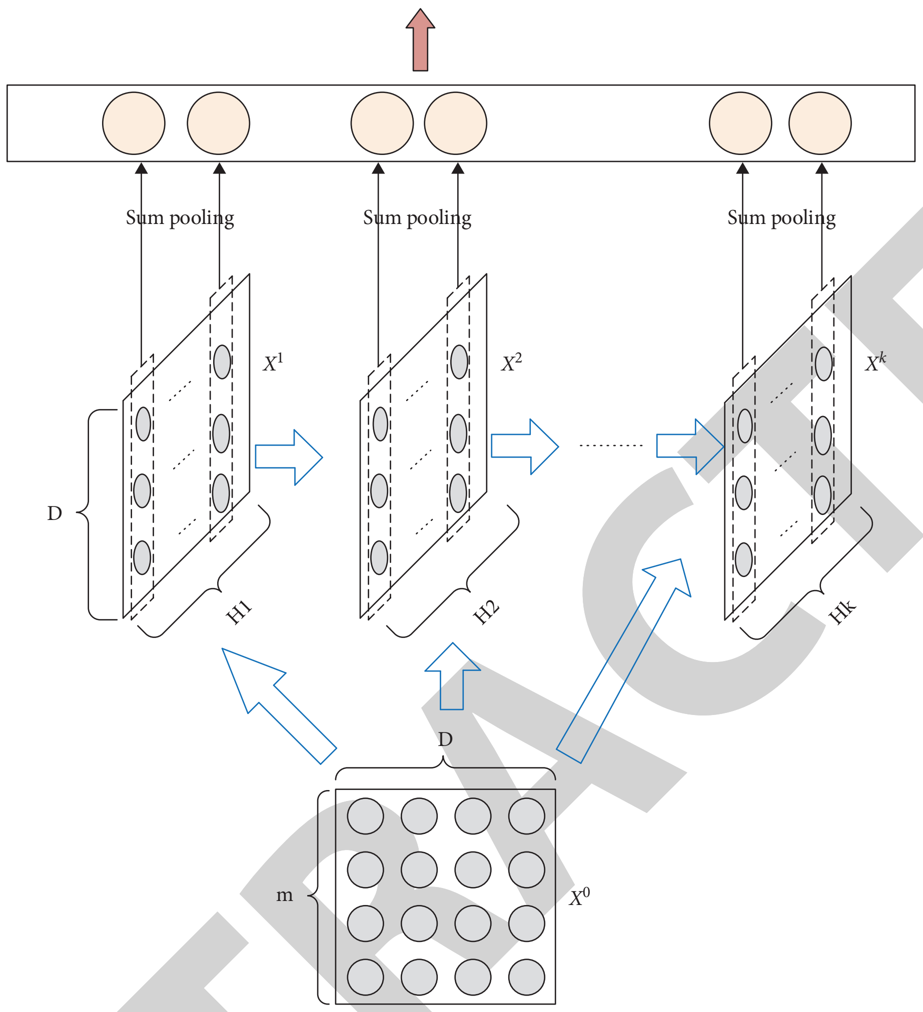

Figure 6: CIN first step calculation process.

3.5. Output Layer. The output from the Combine layer to the output layer is estimated to be the click-through rate. The formula is as follows:

$$\mathbf{O}^G = \text{Sigmoid}\left(\mathbf{W}_G \cdot \mathbf{H}_0 + \mathbf{b}_G\right), \tag{10}$$

where $\mathbf{O}^G$ represents the predicted click-through rate, $\mathbf{W}_G$ and $\mathbf{b}_G$ represent the calculated weight and bias coefficient of this output layer, respectively, and $\mathbf{H}_0$ represents the input vector.

The following is the formula for the loss function:

$$\mathscr{L} = -\frac{1}{N}\sum_{i=1}^{N} y_i \log\left(\hat{y}_i\right) + \left(1 - y_i\right)\log\left(1 - \hat{y}_i\right) + \lambda\|\Theta\|_2, \tag{11}$$

where y_i and $\hat{y}_i$ represent the true label and predicted label (click or not) of the i-th row, respectively. N is the total number of training instances, λ is the $L2$ regularization parameter, and Θ is the trainable parameter set for the entire model.

4. Experiments

In this section, two public datasets will be introduced, and the CCPIN model will be compared with different models on these two public datasets.

4.1. Datasets and Experimental Settings.

(1) The Criteo dataset contains click records of 45 million users, with a total of 13 numerical features and 26 categorical features. In this work, the dataset's missing value has been filled, and data labeling has been operated. During the experiment, in order to facilitate training, 10 million datasets were randomly selected and divided into two parts. 80% of the dataset was a trained dataset, and the remaining 20% was a tested dataset.

(2) The MovieLens-1M dataset has 1,000,209 ratings records. It consists of about 3,900 movies by 6,040 users. In order to make it suitable for the CTR prediction, this paper converts it into a binary classification dataset. The raw user ratings of movies are discrete values from 0 to 5. The samples designated with 4 and 5 in this dataset are marked as positive, and the others are labeled as negative samples. According to the user ID, 130,000 users are randomly selected from it. The data is divided into training and test sets. 100,000 users are randomly selected for training, and the remaining 30,000 users are test sets (about 5.02 million samples). Predict whether a user will rate a given movie higher than 3.

In this model, the batch size is uniformly set to 62500, the learning rate is set to 0.0001, the regularization coefficient is 0.00001, the dimension of the embedding layer is a fixed value of 16, the learner uses Adam [25], the training epochs are 30, and the number of parallel network layers is 2. In the Double Cross Net, the cross layer and the product layer of the Double Cross Net are set to 4. In the CIN, the CIN layer is set to 2 layers. In the DNN, the DNN layer is set to 2 layers, and the number of neurons in each hidden layer is 200. The size of Dropout [26] is set to 0.5 to prevent overfitting. For Wide and Deep, DeepFM, DCN, XdeepFM, and XCrossNet, the DNN layer is set to 2 layers, the number of neurons in each hidden layer is 200, the CIN and cross networks are also set to 2 layers, and the layer in the first layer of XCrossNet is set to 4.

For model evaluation, this paper employs two metrics: AUC (area under the ROC curve) [27] and Logloss (cross-entropy) [28]. These two measures assess performance from two distinct perspectives: AUC takes into account the order of predicted instances, it is unaffected by class imbalance issues, and it can calculate the likelihood that a positive instance will rank higher than a randomly selected negative instance. However, Logloss measures the difference between the predicted score of each instance and the true label.

4.2. Model Comparison. We validate the efficacy of our proposed model by comparing distinct experimental outcomes from the two datasets. We provide a brief overview of these recommendation systems' methods as follows:

Wide and Deep. It is composed of a Wide part and a Deep part. The Wide part uses the memory of LR, and the Deep part uses the generalization ability of DNN to extract the relationship between implicit features.

DeepFM. The FM and Deep parts share the input layer and the embedding layer, and the FM part automatically constructs the first-order and second-order feature interactions. This can remove the tediousness of manually constructing feature interactions.

DCN. This model uses a Cross Net to automatically construct high-order features and, at the same time, uses a DNN to extract implicit features, and then the two parts are combined for output.

XdeepFM. This model proposes a new compression model to modularize the functional interaction network, replacing bitwise with vectorwise to improve model accuracy.

XCrossNet. This model learns separate features for subsequent cross processing. It reflects the learning cross of separate features and finally improves the accuracy of the model to a certain extent.

4.3. Performance Evaluation. This subsection will compare the performance of several models on two datasets in this section.

Table 1: Performance evaluation of different models on the dataset.

Model	Criteo		MovieLens-1M	
	AUC	Logloss	AUC	Logloss
Wide and Deep	0.8029	0.4456	0.7991	0.4562
DeepFM	0.8059	0.4454	0.7995	0.4451
DCN	0.8042	0.4463	0.7993	0.4459
XdeepFM	0.8069	0.4445	0.8011	0.4447
XCrossNet	0.8098	0.4370	0.8066	0.4439
CCPIN	**0.8182**	**0.4354**	**0.8084**	**0.4445**

In the above table, in order to see clearly, the metrics of the CCPIN model are bloated. The AUC is 0.8182 and 0.8084 on the Criteo dataset and MovieLens-1M dataset, respectively, and the Logloss is 0.43540 and 0.4445, respectively.

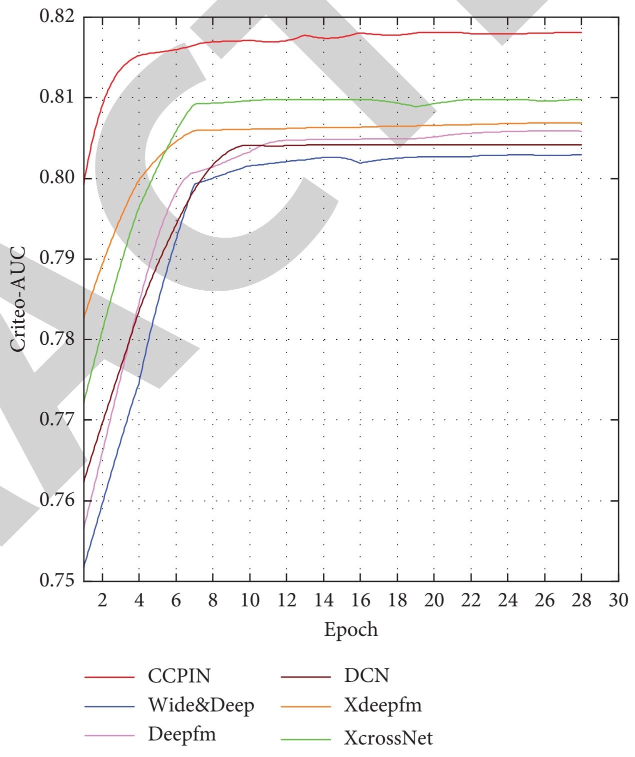

Figure 7: The CTR prediction performance of different models on the Criteo dataset.

4.3.1. Effectiveness Comparison of Different Models. Table 1 shows the performance of different CTR models on the two datasets. The CCPIN model outperforms the rest of the models, surpasses the state-of-the-art XCrossNet model by 0.84% and 0.18% in terms of AUC, and significantly reduces Logloss by 0.16% in the Criteo dataset. But in the MovieLens-1M dataset, the Logloss has a small increase because of the high number of parallel network layers, and this will cause overfitting. Compared with our basic XdeepFM model, the AUC is increased by 1.13% and 0.73%, and the Logloss is decreased by 0.91% and 0.02%, respectively. This shows that the CCPIN model has better feature learning ability than the existing XdeepFM model.

Combined with Figures 7 and 8, in terms of feature interactions, the CCPIN model can quickly learn and quickly reach the optimal value. But as far as Figure 7 is concerned, from the first epoch, the CCPIN model outperforms the

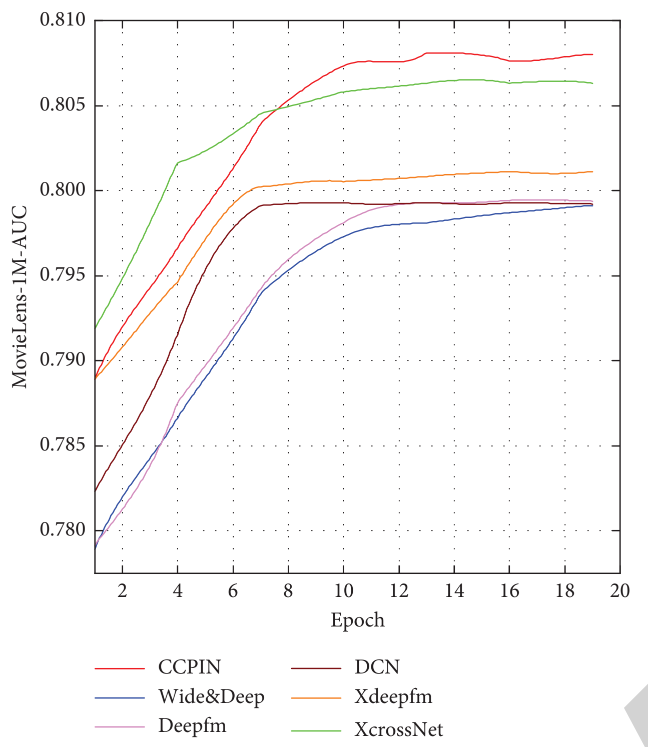

FIGURE 8: The CTR prediction performance of different models on the MovieLens-1M dataset.

TABLE 2: Performance evaluation of parallel models on the dataset.

Model	Criteo		MovieLens-1M	
	AUC	Logloss	AUC	Logloss
DNN + CIN	0.8069	0.4445	0.8011	0.4447
DNN + Double Cross	0.8057	0.4452	0.8006	0.4448
CIN + Double Cross	0.8055	0.4451	0.7997	0.4453

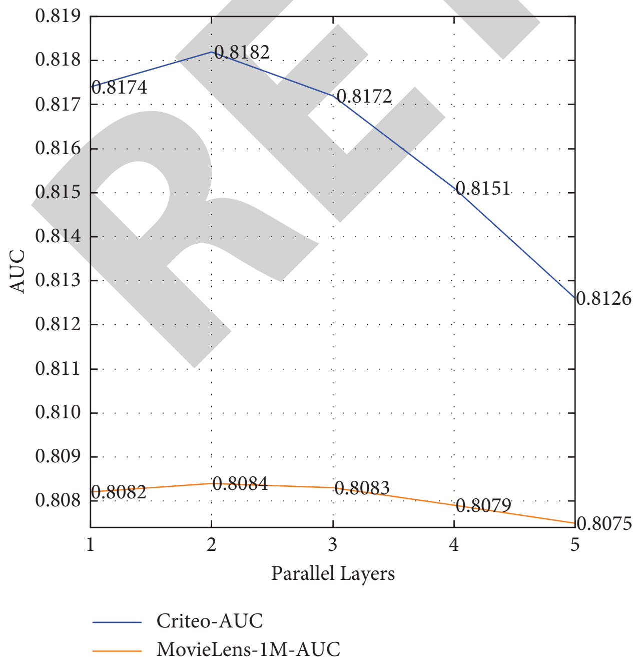

FIGURE 9: The parallel layers of AUC in model CCPIN.

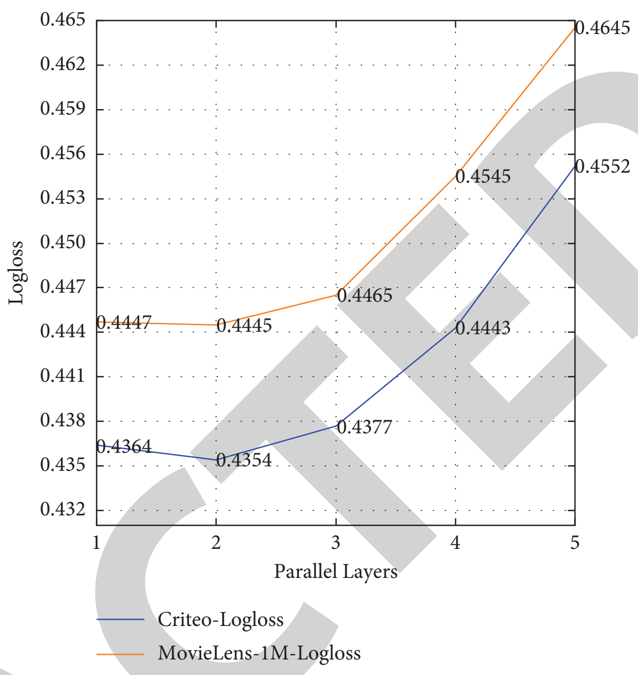

FIGURE 10: The parallel layers of Logloss in model CCPIN.

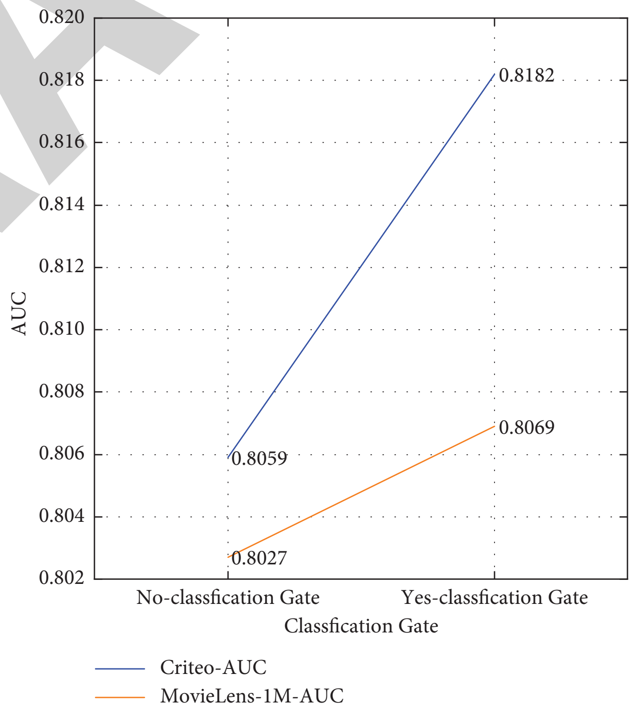

FIGURE 11: The classification gate of AUC in model CCPIN.

other models and decreases the fitting epoch. But in Figure 8, it is lower than the XCrossNet model in the early stage and quickly surpassed in the eighth epoch. This may be the result of the feature classification due to the preparallel layer.

4.3.2. Effectiveness Verification of Different Part of Parallel Model. The results listed in Table 2 show that the dual-parallel network composed of CIN and DNN has the best

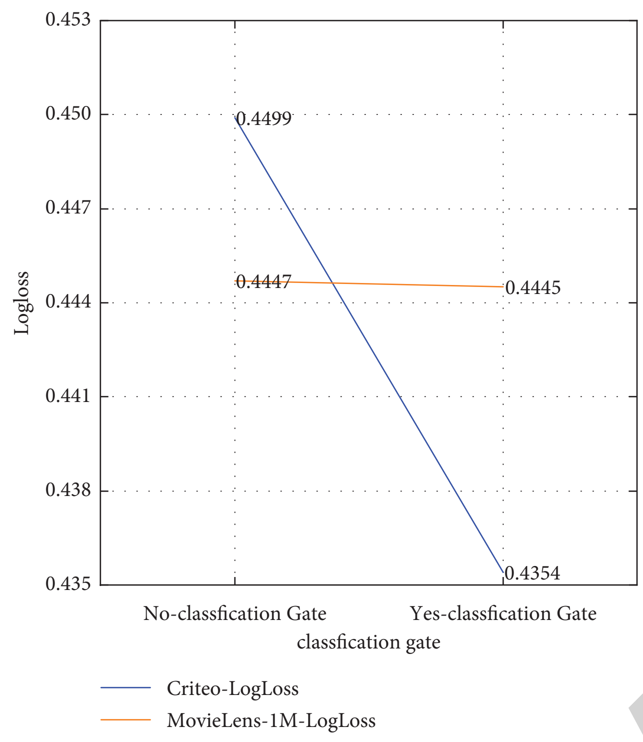

Figure 12: The classification gate of Logloss in model CCPIN.

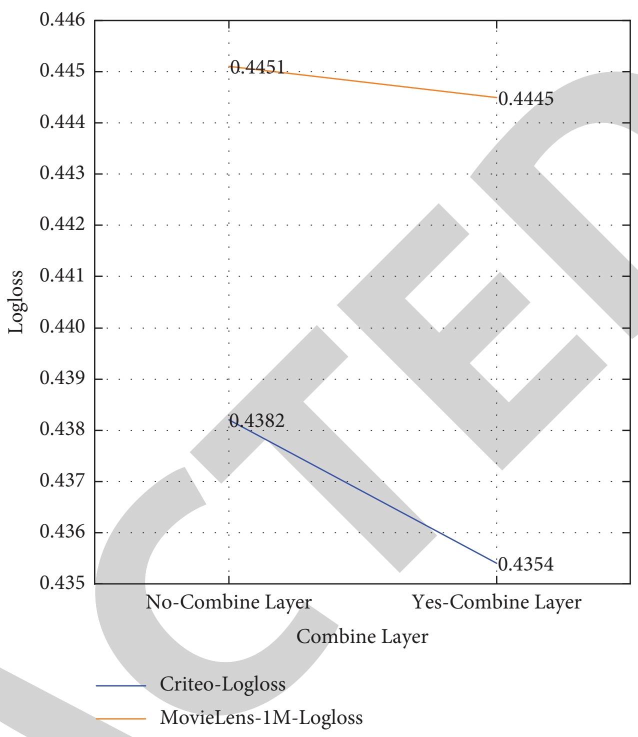

Figure 14: The Combine layer of Logloss in model CCPIN.

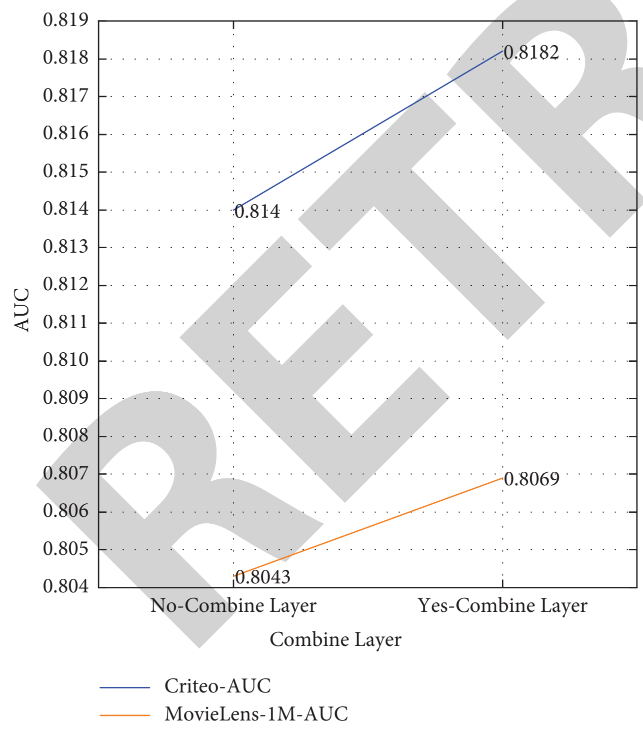

Figure 13: The Combine layer of AUC in model CCPIN.

performance. This indicates that the combined CIN and DNN have reached the optimal value based on the dual-parallel model. However, the results of the CCPIN model are far better than the results of the dual-parallel model, so after this section, the paper will lower the model module and then explore the impact of each module of the CCPIN model.

From Figures 9 and 10, the indicators AUC and Logloss reach their maximum values in the two parallel layers, respectively. But with the increase of layers, it can be seen that AUC decreases rapidly and Logloss increases rapidly. This is because, as the number of parallel layers increases, the number of training parameters and neural network continues to rise, and gradient slopes and the possibility of model overfitting increase. Figures 11 and 12 are the analysis of the classification gate, and Figures 13 and 14 are the analysis of the Combine layer. On the whole, the classification gate has a large share of the improvement of the model CCPIN. The average AUC contribution is 0.825%, and the average Logloss contribution is 0.735%. The Combine layer's contribution to the model CCPIN is small, the average AUC contribution is 0.34%, and the average Logloss contribution is 0.17%.

5. Conclusion and Future Works

This paper proposes a parallel prediction model of click-through rate based on feature classification and Combine [29]. The experimental results show that the newly added model has an average increase of 0.93% in AUC and a decrease of 0.47% in Logloss compared to the benchmark model. However, in the study, it is found that adding a new parallel model leads to a large number of additional parameters to the model. This behavior not only increases the training time but also increases the risk of gradient skewing. In addition, although the effect of Combine layers has some improvement on the model performance, the Logloss is decreased compared with the latest model. This may not be worthwhile.

In the future, we plan to expand the CCPIN model in two aspects. In the first aspect, the classification gate can be improved [30]. The current classification layer only relies on the traditional softmax principle. In the future, we plan to improve the classification layer with a neural network. Secondly, the recommendation system not only includes feature intersection but also considers some other pieces of auxiliary information to gradually improve the practicability of the model, but how to expand its practicability is the next goal.

Data Availability

The data used to support the findings of this study are included within the paper.

Conflicts of Interest

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

Acknowledgments

This work was supported by the fund project of 2021 Jiangsu Postgraduate Research and Innovation Plan (no. KYCX21_2838).

References

[1] S. Q. Xie, Y. L. Tu, R. Y. K. Fung, and Z. D. Zhou, "Rapid one-of-a-kind product development via the Internet: a literature review of the state-of-the-art and a proposed platform," *International Journal of Production Research*, vol. 41, no. 18, pp. 4257–4298, 2003.

[2] J. Schalkwyk, D. Beeferman, F. Beaufays, B. Byrne, C. Chelba, and M. Cohen, "Your word is my command": Google search by voice: a case study," *Advances in Speech Recognition*, pp. 61–90, Springer, Berlin, Germany, 2010.

[3] Q. Yuan, E. O. Nsoesie, B. Lv, G. Peng, R. Chunara, and J. S. Brownstein, "Monitoring influenza epidemics in China with search query from Baidu," *PLoS One*, vol. 8, no. 5, Article ID e64323, 2013.

[4] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *IEEE Internet computing*, vol. 7, no. 1, pp. 76–80, 2003.

[5] C. Heng-Tze, K. Levent, H. Jeremiah, T. Shaked, T. Chandra, and H. Aradhye, "Wide & deep learning for recommender systems," in *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, ACM, Boston, MA, USA, September 2016.

[6] W. Hong, N. Zheng, Z. Xiong, and Z. Hu, "A parallel deep neural network using reviews and item metadata for cross-domain recommendation," *IEEE Access*, vol. 8, Article ID 41774, 2020.

[7] F. Xue, X. He, X. Wang, J. Xu, K. Liu, and R. Hong, "Deep item-based collaborative filtering for Top-N recommendation," *ACM Transactions on Information Systems*, vol. 37, no. 3, pp. 1–25, 2019.

[8] R. E. Wright, *Logistic Regression*, 1995.

[9] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, and H. Aradhye, "Wide & deep learning for recommender systems," in *Proceedings of the 1st workshop on deep learning for recommender systems*, F, Boston, MA, USA, [C], Boston, MA, USA, September 2016.

[10] H. Guo, R. Tang, Y. Ye, Z. Li, X. He, and Z. Dong, "Deepfm: an end-to-end wide & deep learning framework for CTR prediction," 2018, https://arxiv.org/abs/1804.04950.

[11] S. Rendle, "Factorization machines," in *Proceedings of the 2010 IEEE International conference on data mining*, IEEE, Sydney, NSW, Australia, December 2010.

[12] R. Wang, B. Fu, G. Fu, and M. Wang, "Deep & cross network for ad click predictions," in *Proceedings of the ADKDD*, pp. 1–7, Halifax NS Canada, August 2017.

[13] J. Lian, X. Zhou, F. Zhang, Z. Chen, X. Xie, and G. Sun, "XdeepFM: combining explicit and implicit feature interactions for recommender systems," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, F, London United Kingdom, August 2018.

[14] R. Yu, Y. Ye, Q. Liu et al., "Xcrossnet: feature structure-oriented learning for click-through rate prediction," in *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Springer, Virtual Event, May 2021.

[15] M. K. Dahouda and I. Joe, "A deep-learned embedding technique for categorical features encoding," *IEEE Access*, vol. 9, Article ID 114381, 2021.

[16] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*, F, 2001.

[17] X. He, J. Pan, O. Jin et al., "Practical lessons from predicting clicks on ads at facebook," in *Proceedings of the Proceedings of the eighth international workshop on data mining for online advertising*, F, New York, NY, USA, August 2014.

[18] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system," *ACM Computing Surveys*, vol. 52, no. 1, pp. 1–38, 2020.

[19] L. Huang, B. Jiang, and S. Lv, "Survey on deep learning based recommender systems," *Chinese Journal of Computers*, vol. 41, no. 7, pp. 1619–1647, 2018.

[20] J. Buckman, A. Roy, C. Raffel, and I. Goodfellow, "Thermometer encoding: one hot way to resist adversarial examples," in *Proceedings of the International Conference on Learning Representations*, F, 2018.

[21] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, https://arxiv.org/abs/1301.3781.

[22] M. Gong, H. Yang, and P. Zhang, "Feature learning and change feature classification based on deep learning for ternary change detection in SAR images," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 129, pp. 212–225, 2017.

[23] J. Ma, Z. Zhao, X. Yi, J. Chen, L. Hong, and E. H. Chi, "Modeling task relationships in multi-task learning with multi-gate mixture-of-experts," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, F, London United Kingdom, August 2018.

[24] C.-F. Tsai and Y.-C. Hsiao, "Combining multiple feature selection methods for stock prediction: union, intersection, and multi-intersection approaches," *Decision Support Systems*, vol. 50, no. 1, pp. 258–269, 2010.

[25] K. Da, "A method for stochastic optimization," 2014, https://arxiv.org/abs/1412.6980.

[26] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural

networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[27] J. M. Lobo, A. JIMéNEZ-Valverde, and R. Real, "AUC: a misleading measure of the performance of predictive distribution models," *Global Ecology and Biogeography*, vol. 17, no. 2, pp. 145–151, 2008.

[28] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of Operations Research*, vol. 134, no. 1, pp. 19–67, 2005.

[29] Y. Yang, X. Zhang, and Y. Song, "An effective and practical performance prediction model for parallel computing on nondedicated heterogeneous NOW," *Journal of Parallel and Distributed Computing*, vol. 38, no. 1, pp. 63–80, 1996.

[30] M. Oya, Y. Shi, M. Yanagisawa, and N. Togawa, "A score-based classification method for identifying hardware-trojans at gate-level netlists," in *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, IEEE, Grenoble, France, March 2015.