Exploring different interaction among features for CTR prediction

Leilei Yang · Wenguang Zheng · Yingyuan Xiao

Abstract

Advertising Click-Through Rate (CTR) prediction is a relatively successful application in the field of recommendation system. Improving the accuracy of advertising click-through rate can not only make the user experience better, but also give more benefits to advertising platforms and advertisers. It can be seen from the research status that the interaction between learning features has become a very important part of the advertising CTR prediction model. Although the existing CTR prediction models based on deep learning have achieved good results, some models only consider a single interaction mode and lack the diversity of feature interaction. To resolve this problem, this paper proposes a CTR prediction model based on multi-feature interaction, called EDIF, which aims to enhance the diversity of feature interaction. Firstly, the model learns multiple different embedding vectors for each feature in the embedding layer, which reflects the correlation between features; secondly, in the interaction of high-order features, the embedding vectors of each feature are added and pooled to form the aggregation vector of the whole feature as the input, which reflects the integrity of the feature; finally, after the feature embedding operation, the model introduces two layers in parallel: compressed excitation network layer and explicit high-order interaction layer, which improves the ability of feature interaction. We have done a lot of experiments on two public data sets, Avazu and Criteo. The results show that the model effect of this paper has great advantages over the latest model.

Keywords CTR prediction · Second-order interaction · Explicit feature interactions · Attention mechanism

1 Introduction

At present, we have entered the era of big data, the remarkable feature of the era of big data is huge. Using this large amount of data, we can obtain useful information through data mining technology (Abualigah et al. 2019). The recommendation system can help users find the information they want, make producer production information more targeted displayed in front of users. In the advertising field of recommendation system application, CTR prediction is significant. There are three prominent roles in the advertising ecology: users, advertisers and advertising platforms. Users are the group that advertisers want to influence through the advertising platform, and advertisers can promote their products and improve their reputation through the advertising platform. Advertising platform can provide advertising space for advertisers to display advertisements. Advertising is an important business model for Internet companies. For advertisers, advertising can promote products and services, provide channels to obtain users and achieve rapid user growth. Advertising can increase revenue and bring economic value to advertising platforms, so it is of great practical significance to study advertising CTR prediction and improve the accuracy of CTR prediction.

Logistic regression (LR) is one of the proposed methods in the development process of CTR (Juan et al. 2016; Wang et al. 2017; Guo et al. 2017; Zhou et al. 2018b; Feng et al. 2019; Ouyang et al. 2019a; Huang et al. 2019; Zhou et al. 2019; Pi et al. 2019; Ouyang et al. 2019b; Lyu et al. 2020; Xu et al. 2020; Liu et al. 2020). The LR uses one-hot coding, which converts categorical features into vectors as input. However, the features after one-hot are too sparse and lead to a larger feature space. Moreover, logical regression relies on artificial feature engineering. This problem is solved by Factorization Machines (FM) (Rendle 2010), which achieves the purpose of learning combination feature weights by performing inner product operation on two latent vectors, while the Field-aware Factorization Machines (FFM) add the concept of the field in FM. Each feature uses different latent
vectors for different fields; thus, each feature corresponds to a group of latent vectors that enhance the expression of the model. Since it is imperative to learn combined features in the CTR prediction, Product-based Neural Networks (PNN) (Qu et al. 2016) learn feature interactions in the form of inner product. Nevertheless, the PNN does not use different vectors when one feature interacts with other features. The Operation-aware Neural Networks (ONN) (Yang et al. 2020) performs inner product operation after the operation-aware embedding, solving the problem of using different vectors when the same feature interacts with other features.

Some models only consider a single interaction mode and lack the diversity of feature interaction. To resolve this problem, this paper proposes a model called EDIF, which aims to enhance the diversity of feature interaction. We know that the correlation between each feature and different features is different. When a feature uses the same vector when interacting with other features, this correlation will be ignored. Therefore, our EDIF model takes into account the different embedding vectors when interacting with other features (Juan et al. 2016) and adds and pools the embedding vector of each feature to form the aggregation vector of the feature whole. In addition, during high-order feature interaction, we add up the embedding vectors of each feature to form the aggregation vector of the integral feature as an input, which reflects the integrity of the feature. After the feature embedding operation, two layers are introduced in parallel in our model: SENet layer (Hu et al. 2018) and Cross-layer (Wang et al. 2017). In different scenarios, users pay extra attention to different features. If we give the same weight to different cross-features, much valuable information will be ignored. The SENet layer gives different weights to different interactive features in our model. Compared with the previous deep neural networks, the Cross-layer can explicitly carry out high-order interaction and learn more nonlinear relationships.

The main contributions of this paper can be stated as follows:

- Our model does not lose the correlation between features when performing second-order interaction after embedding operation and does not lose the integrity of features when performing high-order feature interaction after embedding operation. Specifically, in the second-order feature interaction, we consider that a feature uses different embedding vectors when interacting with other features, reflecting the correlation between features. In high-order feature interaction, the embedding vectors of each feature are added to form the aggregation vector of the integral feature as the input, which reflects the integrity of the feature.
- Different interactive features are given different weights by introducing the SENet attention module after the second-order feature interactions.
- To improve the ability of feature crossing, second-order and explicit high-order interactions are also carried out to explore more nonlinear relationships between features.

The rest of this paper is structured as follows: the second section mentions the related work. In work related to our model, we have mentioned the investigation and research in four different directions. The third section introduces the model in detail, while in the fourth section, we describe the extensive experiments conducted on two open datasets, and we summarize our work in the fifth section.

2 Related work

2.1 Deep learning evolution of the FM model

The CTR is a very important parameter of the computational advertising and recommendation system. In CTR prediction, it is often necessary to combine multiple features. The FM solves the problem of high dimension and highly sparse input feature combinations well. A second-order part is added to the model based on the logistic regression to obtain the corresponding latent vector for each dimension. The latent vector inner product models the weight, and the feature combination, which has not appeared in the training set, can also be learned effectively. Due to the limitation of the combinatorial explosion, the model is not easy to be extended to the third-order feature crossing. FFM adds the concept of the “feature field” to the FM model so that each feature adopts different weights when crossing with features of different fields. Compared with the FM, the ability of feature interactions is further enhanced. By modifying the second-order part of FM, the Neural Factorization Machines (NFM) (He and Chua 2017) replace the feature crossing part of FM with a Deep Neural Network (DNN) with a Bi-interaction pooling layer. Bi-interaction pooling can be regarded as element-wise product embedding with different features. As compared to FM, the NFM has a more robust expression and feature crossing abilities. The FM model is adopted for the embedding layer of FNN (Zhang et al. 2016) to conduct the dimensional reduction with supervision for sparse features and transform them into dense, continuous features. The convergence speed of FNN becomes faster by using the FM initialization parameter. The Translation-based Factorization Machines (TransFM) (Pasricha and McAuley 2018) combines the ideas of FM and TransRec and applies them to sequential recommendations. The TransFM changed the inner product calculation method in the FM and used the
square Euclidean distance to improve the transitivity between sample features.

2.2 Combination model

Combining different models is a common method to build a recommended model to integrate the advantages of multiple models. Wide & Deep (Cheng et al. 2016) combined the Wide Linear Model and DNN for training. The advantage of this model is the memorization ability of the Wide Model and the generalization ability of the Deep Model are obtained at the same time. This has a significant impact on the subsequent development of the recommended deep learning model. However, the wide section needs to be screened manually for feature combination. The Deep & Cross-Network (DCN) can effectively capture the feature combination of specific order and learn highly nonlinear interaction, without artificial feature engineering. The xDeepFM (Lian et al. 2018) introduces the idea of convolutional neural network to achieve the effect of explicit learning of high-order feature interaction.

2.3 The combination of attention mechanism and recommendation model

The “Attention mechanism” is inspired by human habits. For example, when people browse the website or Taobao home page casually, they are attracted by specific areas. Therefore, if the attention mechanism is taken into account in modeling, the accuracy of the recommendation results can be significantly improved. By introducing the attention mechanism, Attentional Factorization Machines (AFM) (Xiao et al. 2017) are used to assign different importance to different feature combinations where the weights in the network can be learned automatically without introducing any additional field knowledge. Moreover, the Deep Interest Network (DIN) introduced the attention mechanism based on the traditional deep learning recommendation system model and calculated the attention score by using the correlation between the historical items of user behavior and the target advertising items, and thus, according to the different target advertising items, more targeted recommendations are made. However, the user interests are constantly evolving, while the DIN extracts user interests that are independent of each other, without capturing the dynamic evolution of interests. ATRank (Zhou et al. 2018a) proposed a general user behavior sequence modeling framework to integrate different types of user behaviors and conduct more detailed processing of the user heterogeneous behavior data. Momenta’s network architecture SENet, which is the champion of the ImageNet 2017 Challenge, can learn from the importance of different features, thereby weighting the important features and weakening the features that contain little information. The AutoInt (Song et al. 2019) studies the explicit learning problem of high-order feature interaction, which has good interpretability. The user behavior in each session is similar, but the difference between different sessions is significant. Therefore, the Deep Session Interest Network (DSIN) is proposed to model user behavior closely related to the session. The closer the user’s conversational interest is to the target item, the greater is the weight assigned by DSIN through the attention mechanism. Additionally, the Deep Spatio-Temporal Neural Networks (DSTN) consider both the spatial and temporal field information to estimate the CTR of advertising. The DSIN puts forward two Attention models, one is the Self-Attention Model, and the other is the Interactive Attention Model, in which the latter improves the former.

2.4 Changing the way of features cross

The proposed model enriches the way of feature crossing in a deep learning network. The PNN learns the interaction of features in the form of inner product, outer product, inner product and outer product. Its product operation combines the features between different feature fields. The Neural Collaborative Filtering (NCF) (He and Chua 2017) replaces the traditional dot product operation by a neural network and only the ID features of users and items are used; no other features are added. The cross-network in the DCN uses a multi-layer residual network to feature cross each dimension of the feature vectors fully.

3 Model

Figure 1 is the structure diagram of our EDIF model. In EDIF model, the input is sorted as sparse and dense inputs. The sparse input is one-hot coding, but this coding method has the disadvantages of data sparseness and large space occupation, and hence, inspired by the FFM (Juan et al. 2016) and ONN (Yang et al. 2020), we embed the sparse features. After the embedding layer, shown on the left side of the model diagram, we perform the inner product operation for learning the second-order interaction between the features. After feature interactions, we go through the SENet layer (Hu et al. 2018) to learn the importance of the feature interactions. On the right side of the model diagram, each feature is added into an aggregation vector by sum pooling multiple embedding vectors generated by each feature. Then the aggregation vector and dense features are spliced into the explicit high-order interaction layer. Some symbol definitions in our paper are shown in Table 1.
3.1 Problem description

Advertising CTR prediction is defined as a two category problem: given the test sample feature \( x \) and the sample label \( y \), \( y \in \{0, 1\} \), the purpose is to predict the probability of the user clicking the test sample. According to the predicted probability of users clicking on advertisements, the advertisements are sorted from top to bottom. If the advertisements with greater probability are placed in the front, the users will pay more attention to the recommended advertisements to improve the income of advertising platform.

3.2 The proposed model

3.2.1 Input layer

In our model, the input is sorted as sparse input and dense input, denoted as \( x_{\text{sparse}} \) and \( x_{\text{dense}} \), respectively, where \( x_{\text{sparse}} = [x_1, x_2, \ldots, x_n] \), where \( n \) is the number of features. Sparse input is input to the model through one-hot coding, and dense features can be directly input to the model. The one-hot coding process is introduced as follows: The feature generates a new vector after the one-hot coding operation. Each dimension in the vector represents a category of the original feature, with the significant bit set to 1 and the others set to 0.

The values of “Click or not” in Table 2 are 0 and 1, respectively, indicating that the advertisement is not clicked and the advertisement is clicked. “Gender,” “Region” and “Advertisement Type” are the characteristics. After coding the above characteristics, it is shown in Table 3. It can be seen that the feature space is very sparse after the single hot coding operation. Therefore, our EDIF model uses the embedding method to turn sparse features into dense embedding vectors.

3.2.2 Embedding layer

The correlation between each feature is different, and this correlation is ignored when a feature interacts with other features if the same vector is used. Suppose we have three features: \( x_1, x_2, x_3 \) and \( v_1, v_2, v_3 \) are the embedding vectors of the three features, respectively. When \( x_1 \) interacts with \( x_2 \) and \( x_3 \), if different representations are not considered for different features interaction, then the weights of \( x_1 \) and \( x_2 \) and \( x_1 \) and \( x_3 \) are \( w_{x_1,x_2} = v_1 \cdot v_2 \) and \( w_{x_1,x_3} = v_1 \cdot v_3 \), respectively. As far as we know, When \( x_1 \) interacts with \( x_2 \) and \( x_3 \), the importance of each interaction is different. For example, if \( x_1 \) represents male users, \( x_2 \) represents basketball and \( x_3 \) represents lipstick, then in general, male users prefer to buy basketball rather than lipstick, and therefore, we should consider different representations of different features interactively. Then the weights of \( x_1 \) and \( x_2 \), \( x_1 \) and \( x_3 \) are \( w_{x_1,x_2} = v_1^2 \cdot v_2^2 \) and \( w_{x_1,x_3} = v_1^2 \cdot v_3^2 \), respectively. In this

### Table 1: Notation

| Notation | Explanation |
|----------|-------------|
| \( x_{\text{sparse}} \) | Sparse input features |
| \( x_{\text{dense}} \) | Dense input features |
| \( x_i \) | The \( i \)th feature |
| \( n \) | The number of sparse features |
| \( e \) | Aggregation vector |
| \( v_i \) | The embedding vector matrix corresponding to the \( i \)th feature |
| \( v_i^j \) | The embedding vector of the \( i \)th feature interacting with the \( j \)th feature |
| \( P \) | The inner product of aggregation vector \( e \) |
| \( S \) | The result of the average pooling of vector \( P \) |
| \( A \) | The result of vector \( S \) after excitation |
| \( Q \) | The vector after reweighting \( P \) |
| \( c \) | The vector generated by merging aggregation vectors and dense features |
| \( l \) | The layers of cross-network |

### Table 2: Example of advertisement click

| Click or not | Gender | Region | Advertisement Type |
|--------------|--------|--------|--------------------|
| 1            | Female | China  | Movie              |
| 0            | Male   | Germany| Game               |
| 1            | Female | Korea  | Movie              |
Table 3  Advertising click One-Hot coding example

| Click or not | Gender = male | Gender = female | Region = China | Region = Germany | Region = Korea | Advertising type = movie | Advertising type = game |
|--------------|---------------|-----------------|---------------|-----------------|---------------|-------------------------|-------------------------|
| 1            | 0             | 1               | 1             | 0               | 0             | 1                       | 0                       |
| 0            | 1             | 0               | 0             | 1               | 0             | 0                       | 1                       |
| 1            | 0             | 1               | 0             | 0               | 1             | 1                       | 0                       |

Fig. 2  Embedding process

Fig. 3  The vector merging process

case, two different vectors are used when \( x_1 \) interacts with \( x_2 \) and \( x_3 \).

As shown in Fig. 2, the figure shows the embedding process of feature \( x_i \). Feature \( x_i \) generates \( n - 1 \) embedding vectors in the embedding process. These \( n - 1 \) embedding vectors are the vectors that feature \( x_i \) interacts with other \( n - 1 \) features. As shown in Fig. 3, combining these \( n - 1 \) embedding vectors together to obtain the aggregation vector \( e_i \) of the feature whole \( x_i \). \( v_i \) shown on the left side of Fig. 3 represents the embedding vector matrix of the \( i \)th feature, \( v_i = [v_{i,1}, v_{i,2}, \ldots, v_{i,j}, \ldots, v_{i,n}] \), where \( j \neq i \), \( v_{i,j} \) represents the embedding vector of feature \( x_j \) interacting with feature \( x_i \), that is, each feature will learn the corresponding embedding vector for other \( n - 1 \) features.

\( e_i = v_i \cdot x_i \) on the right side of Fig. 3 represents the \( i \)th aggregation vector. The aggregation vector is the result of adding and pooling the embedded vector matrix \( v_i \), where \( i \in [1, \ldots, n] \), \( j \neq i \). The sum pooling process is shown in formula 1.

\[
e_i^k = \sum_{j=1}^{n-1} v_{i,j}^k
\]

(1)

Where \( e_i \) is the aggregation vector after the sum pooling of matrix \( v_i \), \( e_i^k \) represents the \( k \)th dimension of \( v_i \), and \( e_i^k \) is obtained by adding the \( k \)th row of matrix \( v_i \). After the aggregation vectors of \( n \) features are spliced, the vector \( e = [e_1, e_2, \ldots, e_i, \ldots, e_n] \) is obtained.

3.2.3 Second-order interaction layer

As shown in Fig. 4, each feature learns \( n - 1 \) vectors. Thus, each feature has a unique embedding vector corresponding to other features, so Hadamard product operation on the corresponding two embedding vectors can be performed, such as \( v_{i}^j \), \( v_{j}^i \), \( i, j \in [1, \ldots, n] \), \( j \neq i \). For Fig. 4, the Hadamard product is calculated in the following formula:

\[
v_{i}^j \odot v_{j}^i = [v_{i,j}^1 v_{j,j}^1, v_{i,j}^2 v_{j,j}^2, \ldots, v_{i,j}^d v_{j,j}^d]
\]

(2)

Where \( d \) is the dimension of the embedding vector. After the second-order interaction, we get \( m \) new vectors, which are represented as \( P = [p_1, p_2, \ldots, p_m] \), where \( m = (n \ast (n - 1))/2 \) and vector \( P \) is the input of SENet.
When people usually buy goods, they may prefer to buy one kind of goods. Thus while forecasting these kinds of goods, we should give more weight. For example, for people who usually like to buy skirts rather than lipstick should be given more weight to skirts than lipstick. To achieve this goal, we introduce the SENet, first applied in the image field, which can increase the weight of important features and weaken the weight of relatively unimportant features.

The SENet block is divided into three-step processes: squeeze step, explanation step, and reweight step, as shown in Fig. 5:

Let’s start with the squeeze step: In this step, the input is the vector \( P = [p_1, p_2, \ldots, p_m] \). Then, we compress each of the dimensions of the vector into one dimension and compress the input into vector \( S = [s_1, s_2, \ldots, s_i, \ldots, s_m] \), where \( s_i \) is a scalar, \( i \in [1, \ldots, m] \). The calculation of the squeeze is represented as follows:

\[
s_i = F_{sq}(p_i) = \frac{1}{k} \sum_{i=1}^{k} p_i^{(i)}
\]  

Next, we introduce the excitation step: In this step, we use two fully connected layers to learn the weight of vector \( S \). In the first layer, we reduce the dimension with parameter \( W_1 \) and then through the activation function \( \sigma_1 \). In the second layer, \( W_2 \) is used to restore the dimension before the dimension reduction and then through the activation function \( \sigma_2 \). Then we can learn the new vector \( A = [a_1, a_2, \ldots, a_m] \), calculated as:

\[
A = F_{ex}(S) = \sigma_2(W_2\sigma_1(W_1S)) = [a_1, a_2, \ldots, a_m]
\]  

Where \( A \in \mathbb{R}^m, W_1 \in \mathbb{R}^{m \times \overline{r}}, W_2 \in \mathbb{R}^{m \times \overline{r}}, r \) is the dimension reduction ratio.

Finally, we introduce the reweight step: In this step, we multiply the vector \( P \) with the corresponding position elements of \( A \), i.e., reweighting \( P \) to get the final output of SENet: \( Q = [q_1, q_2, \ldots, q_m] \). The calculation of \( Q \) is given as:

\[
Q = F_{ReWeight}(A, P) = [a_1 \cdot p_1, \ldots, a_m \cdot p_m] = [q_1, q_2, \ldots, q_m]
\]  

Here, \( a_i \in \mathbb{R}, i \in \mathbb{R}^d, q_i \in \mathbb{R}^d \). \( d \) is the embedding dimension.

3.2.5 Explicit high-order interaction layer

This section introduces the cross-network to learn the high-order interaction between features (Wang et al. 2017). The inputs of the cross-network are aggregation vector and dense feature. Firstly, we need to compress the aggregation vector \( c \) into a line and splice it with the dense feature to generate a new vector \( c \), while the merging process is shown in Fig. 6.

After merging into a vector, we input \( c \) into the cross-network. The interaction process of cross-network is shown in Fig. 7.

From Fig. 7, we can see that each layer interacts with the input \( c \), where deeper the layers are, greater is the degree of interaction. The interaction process from the first layer to the \( l \)th layer is calculated as follows:

\[
c_1 = cc_1^TW_0 + b_0 + c
\]

\[
c_2 = cc_1^TW_1 + b_1 + c_1
\]
Exploring different interaction among features for CTR prediction

The process of cross-network interaction

\[ c_l = c c_l^{T} W_{l-1} + b_{l-1} + c_{l-1} \]  \hspace{1cm} (6)

Here, \( W_{l-1} \) and \( b_{l-1} \) are the weights and bias of layer \( l \), respectively.

3.2.6 Output layer

We combine the linear part, the output of the cross-network, and the output of SENet as the input of the output layer. The formula for the output layer is as follows:

\[ \hat{y} = \sigma(W_{\text{linear}} x_{\text{linear}} + [c_l; Q] W_o + b_o) \]  \hspace{1cm} (7)

Here, \( W_o \) and \( b_o \) are the weights and bias terms of the output layer, respectively. We train the model by minimizing the log loss:

\[ L(y, \hat{y}) = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y}) \]  \hspace{1cm} (8)

Where \( y \in \{0,1\} \) represents whether the user has clicked or not.

4 Experiment

4.1 Experimental setup

4.1.1 Datasets

We used two datasets to evaluate our model, as described below.

- Avazu dataset: Avazu is an Internet advertising company. Avazu dataset comes from the CTR prediction dataset of the Kaggle data mining competition. The purpose is to predict whether users will click on advertisements. In this chapter, 1 million pieces were randomly selected from the data set, 800000 pieces in the training set and 200000 pieces in the test set, respectively. The dataset has 24 attribute columns.

- Criteo dataset: Criteo is a marketing technology company with a reach across the globe. The Kaggle dataset shows an advertising challenge, which aims to predict the CTR of ads. The label attribute in the data indicates whether the advertisement has been clicked or not. I1-I13 are the numerical features, and C1-C26 are the category features. We have intercepted the first one million data points, and the training and the test sets were 80% and 20% of the intercepted data, respectively.

4.1.2 Evaluation metrics

We used Log loss and AUC as evaluation indicators, described as:

- Log loss: It is often used in the offline evaluation in which the convergence of the model can be observed. According to this indicator, the smaller the loss value, the better is the model effect.
- AUC: It can be used to evaluate the ranking model of the recommendation system, in which the higher the AUC, the better is the model effect.

4.1.3 Implementation details

In our experiment, the parameters are set as follows: The embedding dimension is 2, the SENet dimension reduction ratio is 3, and the batch size is 256. For the Avazu dataset, the number of cross-layers considered is 4. For the Criteo dataset, the number of cross-levels considered is 3. Among many excellent optimization algorithms (Agushaka et al. 2022; Oyelade et al. 2022; Abualigah et al. 2022, 2021a, b), we find that Adam optimization algorithm (Kingma and Ba 2014) is the most suitable for our model.

All the methods mentioned above are implemented on Intel i7-6700 3.4GHz CPU and 16GB RAM, with Python version of 3.7 and Tensorflow version of 1.14.

4.2 Model comparison

We have compared our model to the following models: FM (Rendle 2010), FFM (Juan et al. 2016), DCN (Wang et al. 2017), AutoInt (Song et al. 2019), FiBiNET (Huang et al. 2019), ONN (Yang et al. 2020), and the results are shown in Table 4, Figs. 8 and 9. From the perspective of experimental results, our model performs better in terms of Log loss or AUC. This is because our EDIF model enhances the diversity of feature interactions and improved the accuracy of advertising click rate prediction. Specifically, there are following points: First, our EDIF model takes into account the correlation between each feature and different features: not only uses different embedding vectors when embedding when the same feature interacts with other features, The embedding...
**Table 4** Comparison of model effects

| Model | Avazu Logloss | Avazu AUC | Criteo Logloss | Criteo AUC |
|-------|---------------|----------|----------------|-----------|
| FM    | 0.38003       | 0.75707  | 0.48483        | 0.75540   |
| FFM   | 0.37683       | 0.76293  | 0.48467        | 0.75677   |
| DCN   | 0.38303       | 0.75943  | 0.48297        | 0.75960   |
| AutoInt | 0.38477    | 0.75713  | 0.48753        | 0.75100   |
| FiBiNET | 0.38080    | 0.76023  | 0.48583        | 0.75407   |
| ONN   | 0.37930       | 0.76023  | 0.48583        | 0.75407   |
| EDIF  | **0.37617**   | **0.76463** | **0.48067**   | **0.76147** |

Bold is used to highlight the excellence of our model experimental results compared with other model experimental results.

**Fig. 8** The performance of the Log loss of different models on Avazu and Criteo datasets

**Fig. 9** The performance of the AUC of different models on Avazu and Criteo datasets

vectors for each feature are also added and pooled, The aggregation vector that forms the characteristic whole; next, After the embedding operation, The EDIF model performs explicit higher-order interactions, Learn more about the nonlinear relationships; last, The EDIF model gives different weights to the different feature interactions with the SENet, The importance of feature interactions is dynamically learned.

1. **FM**: The FM achieves second-order interaction by inner product of hidden vectors.
2. **FFM**: The FFM model adds the idea of field to FM model, which can learn a set of hidden vectors for each feature.
3. **DCN**: The DCN can learn feature combinations of the specific order and carry out feature crossover explicitly.
4. **AutoInt**: The AutoInt uses the multi-head attention mechanism to model feature interactions explicitly.
5. **FiBiNET**: FiBiNET refers to the SENet structure, highlighting the importance of dynamic learning features and using a bilinear function to enhance the model cross-features.
6. **ONN**: The ONN model can learn different representations of different operations instead of sharing an embedding structure among all operations.

**4.3 Study of the parameters**

In this section, we study the influence of different hyperparameters on the results of the model. Table 5 shows our comparison results:

**4.3.1 Embedding dimension**

We have performed experiments in the range of 2–12. From Figs. 10 and 11, we can see that the model works best when the embedding dimension is 2, in both the Avazu and the Criteo datasets. Moreover, the larger the embedding dimension is, the more complex the model is, which may lead to overfitting.

**4.3.2 Cross-layers**

We have performed experiments in the range of 2–5 cross-layers. From Figs. 12 and 13, we can see that in the Avazu dataset, when the number of cross-layers changes from 2 to 4, the effect of the model is improved steadily. In the Criteo dataset, when the number of cross-layers changes from 2 to 3, the model significantly improves. However, when the number of layers is too small, the model training is not enough, and the effect cannot be achieved. When the number of cross-layers changes from 4 to 5, the effect will not be improved due to overfitting.
The influence of different hyperparameters on the model results

| Parameter          | Avazu Logloss | Avazu AUC | Criteo Logloss | Criteo AUC |
|--------------------|---------------|-----------|----------------|-----------|
| Embedding dimension|               |           |                |           |
| 2                  | 0.37617       | 0.76463   | 0.48067        | 0.76147   |
| 4                  | 0.37663       | 0.76183   | 0.48137        | 0.75960   |
| 8                  | 0.37895       | 0.76185   | 0.48240        | 0.75650   |
| 12                 | 0.37790       | 0.76125   | 0.48415        | 0.75735   |
| Cross-layers       |               |           |                |           |
| 2                  | 0.37840       | 0.76177   | 0.48210        | 0.75933   |
| 3                  | 0.37670       | 0.76290   | 0.48067        | 0.76147   |
| 4                  | 0.37617       | 0.76463   | 0.48170        | 0.75877   |
| 5                  | 0.37823       | 0.76170   | 0.48267        | 0.75867   |
| Reduction ratio    |               |           |                |           |
| 2                  | 0.37630       | 0.76177   | 0.48095        | 0.75920   |
| 3                  | 0.37617       | 0.76463   | 0.48067        | 0.76147   |
| 4                  | 0.37630       | 0.76420   | 0.48090        | 0.76085   |
| 5                  | 0.37660       | 0.76190   | 0.48280        | 0.75898   |

Bold values indicate the process of parameter tuning, when the parameter takes which value, the model performance is the best.

Fig. 10 The Log loss performance in different embedding dimensions

Fig. 11 The AUC performance in different embedding dimensions

Fig. 12 The Log loss performance in different cross-layers

Fig. 13 The AUC performance at different cross-layers
4.3.3 Reduction ratio

We performed experiments with the dimension reduction ratio between 2 and 5. From Figs. 14 and 15, we can see that, in the Criteo dataset, the model achieves the best effect when the dimension reduction ratio is 3 and 4, but when the dimension reduction ratio is 5, the model does not perform very well. The reason can be stated as the dimension is reduced too much, and the learning ability of the model is not enough. In the Avazu dataset, the model has a good effect when the dimension reduction ratio is 3 and 4.

After the comparative experiments on the above parameters, it is found that each parameter can find a parameter with the best effect, which requires many experiments to be found. We also found that too small or too large number of parameters could not achieve the best effect because a too small number of parameters may lead to underfitting, and a too large number of parameters will lead to overfitting.

5 Conclusion

This paper proposes an EDIF model, which enhances the ability of feature interaction and improves the accuracy of CTR prediction. Firstly, the model learns several different embedding vectors for each feature to enhance the correlation between features; secondly, in the process of second-order feature interaction, the model obtains the interaction results of different embedding vectors and uses compressed excitation network to dynamically learn the importance of feature interaction and improve the quality of feature interaction; thirdly, the model uses the explicit high-order interaction layer to explicitly learn the high-order interaction of features and learn more nonlinear relationships between features; finally, the results of the model on two public data sets, Avazu and Criteo, are better than the latest model.

In this paper, attention mechanism and other structures are used to capture the cross-relationship between features and improve the effect of advertising CTR prediction. In order to maintain the effect of advertising CTR prediction and make the model lighter, the method of knowledge distillation can be used in the future. Transfer knowledge to lighter student networks through teacher networks, so that students can learn from the experience of teacher networks, so as to achieve similar or better results.

Author Contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Leilei Yang, Wenguang Zheng and Yingyuan Xiao. The first draft of the manuscript was written by Leilei Yang, and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding No funding was received for conducting this study.

Data availability A data availability statement is mandatory for publication in this journal. Please confirm that this statement is accurate, or provide an alternative.

Declarations

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

Conflict of interest The authors declare that they have no conflict of interest.

Informed consent Informed consent was not required as no humans or animals were involved.

References

Abualigah L, Diabat A, Mirjalili S, Abd Elaziz M, Gandomi AH (2021) The arithmetic optimization algorithm. Comput Methods Appl Mech Eng 376:113609
Abualigah L, Yousri D, Abd Elaziz M, Ewees AA, Al-Qaness MA, Gandomi AH (2021) Aquila optimizer: a novel meta-heuristic optimization algorithm. Comput Ind Eng 157:107250
Abualigah L, Abd Elaziz M, Sumari P, Geem ZW, Gandomi AH (2022) Reptile search algorithm (rsa): A Nature-inspired meta-heuristic optimizer. Expert Syst Appl 191:116158
Abualigah LMQ et al (2019) Feature selection and enhanced krill herd algorithm for text document clustering. Springer, Cham
Agushaka JO, Ezungwu AE, Abualigah L (2022) Dwarf mongoose optimization algorithm. Comput Methods Appl Mech Eng 391:114570
Cheng HT, Koc L, Harmsen J, Shaked T, Chandra T, Aradhye H, Anderson G, Corrado G, Chai W, Ispir M, et al. (2016) Wide & deep learning for recommender systems. In: Proceedings of the 1st workshop on deep learning for recommender systems, pp 7–10
Feng Y, Lv F, Shen W, Wang M, Sun F, Zhu Y, Yang K (2019) Deep session interest network for click-through rate prediction. arXiv preprint arXiv:1905.06482
Guo H, Tang R, Ye Y, Li Z, He X (2017) Deepfm: a factorization-machin-based neural network for ctr prediction. arXiv preprint arXiv:1703.04247
He X, Chua TS (2017) Neural factorization machines for sparse predictive analytics. In: Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval, pp 355–364
Hu J, Shen L, Sun G (2018) Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 7132–7141
Huang T, Zhang Z, Zhang J (2019) Fibenet: combining feature importance and bilinear feature interaction for click-through rate prediction. In: Proceedings of the 13th ACM Conference on Recommender Systems, pp 169–177
Juan Y, Zhuang Y, Chin WS, Lin CJ (2016) Field-aware factorization machines for ctr prediction. In: Proceedings of the 10th ACM conference on recommender systems, pp 43–50
Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980
Lian J, Zhou X, Zhang F, Chen Z, Xie X, Sun G (2018) xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp 1754–1763
Liu B, Zhu C, Li G, Zhang W, Lai J, Tang R, He X, Li Z, Yu Y (2020) Autoifs: Automatic feature interaction selection in factorization models for click-through rate prediction. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp 2636–2645
Lyu Z, Dong Y, Hua C, Ren W (2020) Deep match to rank model for personalized click-through rate prediction. Proc AAAI Conf Artif Intell 34:156–163
Ouyang W, Zhang X, Li L, Zou H, Xing X, Liu Z, Du Y (2019a) Deep spatio-temporal neural networks for click-through rate prediction. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp 2078–2086
Ouyang W, Zhang X, Ren S, Li L, Liu Z, Du Y (2019b) Click-through rate prediction with the user memory network. In: Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data, pp 1–4
Oyelade ON, Ezungwu AE, Mohamed TI, Abualigah L (2022) Ebola optimization search algorithm: a new nature-inspired metaheuristic optimization algorithm. IEEE Access 10:16150–16177
Pasricha R, McAuley J (2018) Translation-based factorization machines for sequential recommendation. In: Proceedings of the 12th ACM Conference on Recommender Systems, pp 63–71
Pi Q, Bian W, Zhou G, Zhu X, Gai K (2019) Practice on long sequential user behavior modeling for click-through rate prediction. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp 2671–2679
Qu Y, Cai H, Ren K, Zhang W, Yu Y, Wen Y, Wang J (2016) Product-based neural networks for user response prediction. In: 2016 IEEE 16th International Conference on Data Mining (ICDM), IEEE, pp 1149–1154
Rendle S (2010) Factorization machines. In: 2010 IEEE International conference on data mining, IEEE, pp 995–1000
Song W, Shi C, Xiao Z, Duan Z, Xu Y, Zhang M, Tang J (2019) Autojoint: Automatic feature interaction learning via self-attentive neural networks. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp 1161–1170
Wang R, Fu B, Fu G, Wang M (2017) Deep & cross network for ad click predictions. In: Proceedings of the ADKDD’17, pp 1–7
Xiao J, Ye H, He X, Zhang H, Wu F, Chua TS (2017) Attentional factorization machines: Learning the weight of feature interactions via attention networks. arXiv preprint arXiv:1708.04617
Xu W, He H, Tan M, Li Y, Lang J, Guo D (2020) Deep interest with hierarchical attention network for click-through rate prediction. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp 1905–1908
Yang Y, Xu B, Shen S, Shen F, Zhao J (2020) Operation-aware neural networks for user response prediction. Neural Netw 121:161–168
Zhang W, Du T, Wang J (2016) Deep learning over multi-field categorical data. In: European conference on information retrieval, Springer, pp 45–57
Zhou C, Bai J, Song J, Liu X, Zhao Z, Chen X, Gao J (2018a) Attrank: An attention-based user behavior modeling framework for recommendation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol 32
Zhou G, Zhu X, Song C, Fan Y, Zhu H, Ma X, Yan Y, Jin J, Li H, Gai K (2018b) Deep interest network for click-through rate prediction. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp 1059–1068
Zhou G, Mou N, Fan Y, Pi Q, Bian W, Zhou C, Zhu X, Gai K (2019) Deep interest evolution network for click-through rate prediction. Proc AAAI Conf Artif Intell 33:5941–5948

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.