Focusing on the Importance of Features for CTR Prediction

Yuquan Hou, Caimao Li(✉), Hao Li, Hao Lin, and Qiuhong Chen

School of Computer Science and Technology, Hainan University, Haikou 570228, China
lcaim@126.com

Abstract. Traditional CTR recommendation models have concentrated on how to learn low-order and high-order characteristics. The majority of them make many efforts at combining low-order and high-order functions. However, they ignore the importance of the attention mechanism for learning input features. The ECABiNet model is proposed in this article to enhance the performance of CTR. On the one hand, the ECABiNet model can learn the importance of features dynamically via the LayerNorm and ECANET layers. On the other hand, through the use of a bi-interaction layer and a DNN layer, it is capable of effectively learning the feature interactions. According to the experimental results on two public datasets, the ECABiNet model is more effective than the previous CTR model.

Keywords: CTR Model · ECANET · LayerNorm · ECABiNet

1 Introduction

Currently, the online advertising business has gradually become popular, and improving the CTR of advertisements has become a critical issue. Current CTR models mainly include shallow models and deep models. The article [1] proposes a factorization machine (FM) model that is capable of not only solving problems with enormous sparsity in linear time but also working with any real-valued eigenvectors. This is a typical shallow model, and it is incapable of learning the relationship between higher-order features. Paper [2] proposed the FFM model on the basis of the FM model, which focuses on discrete classification features and enhances the factorization machine algorithm. The classic deep models include the wide and deep model, the DeepFM model, and the FiBiNET model, among others. To achieve memory and generalization capabilities, the wide and deep model [3] combines the linear and deep models. The wide part generates memory for feature interaction through feature intersection, whereas the deep part uses low-dimensional dense features as input to generalize cross-features that do not appear in the training samples. The wide part, on the other hand, necessitates artificial feature engineering, resulting in increased workload.

As a result, the DeepFM model [4] is suggested. Its wide section makes use of the FM model to automatically learn feature intersection, while the deep section continues to make use of the deep neural network DNN, which not only reduces the workload associated with manual feature engineering but also improves recommendation efficiency.

Y. Wang et al. (Eds.): ICPCSEE 2022, CCIS 1628, pp. 41–52, 2022.
https://doi.org/10.1007/978-981-19-5194-7_4

While the preceding models emphasize feature intersection, paper [5] proposes the FiBiNET model, which places a premium on feature importance learning. It employs a SENET layer to determine the relative importance of various features and then generates recommendation results by feeding a bilinear layer into a deep neural network; however, the SENET layer requires a dimensionality reduction operation and is quite complex. By combining the aforementioned issues, this article proposes the ECABiNet model.

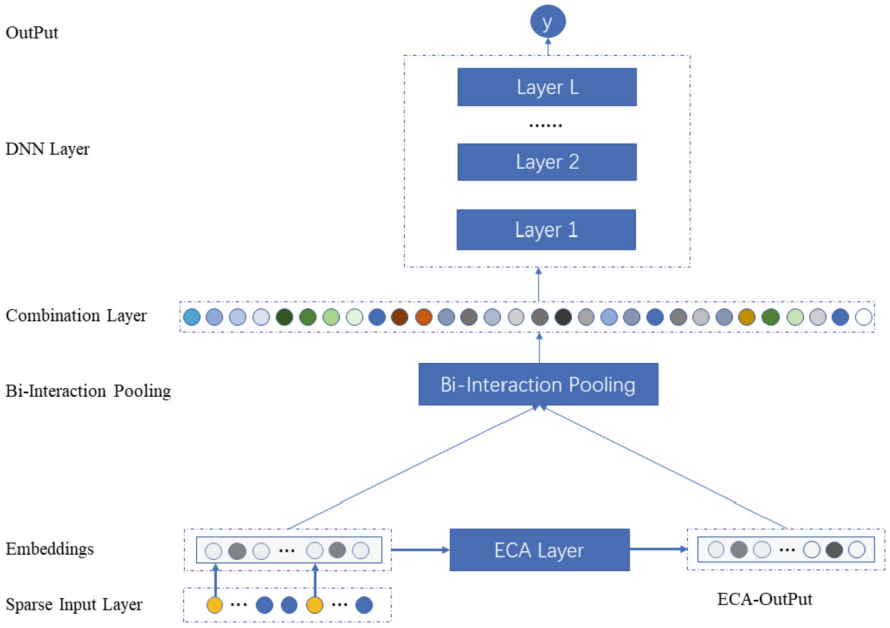

Fig. 1. ECABiNet Model.

2 ECABiNet Model

At first, attention models were applied to machine translation [6] and then to neural network models. The paper [7] proposes the AFM model, which uses an attention mechanism, allowing the model to learn the weights of second-order feature items effectively.

The majority of these established models of attention place a premium on the component of feature intersection. As illustrated in Fig. 1, the ECABiNet model emphasizes the feature embedding portion. The sparse vector input is converted to a feature embedding vector and then normalized before being passed to the ECANET layer via the LayerNorm layer. After pooling the output of the ECANET layer and the feature embedding vector, it is passed to the DNN layer for recommendation.

2.1 Sparse Input and Embedding Layer

The ECABiNet model uses an embedding layer to map the features of the original input into a low-dimensional space and converts discrete variables into continuous vectors. By using an embedding layer, the model not only has the ability to reduce the spatial dimension of discrete variables but also meaningfully represents the features of these original inputs. The output of the embedding layer is $E = [e_1, \ldots, e_n]$.

2.2 Layer Norm

Normalization [8] plays a crucial role in neural networks and can normalize the distribution of data. If the data were not normalized, their distribution would be different, and the distribution of data in each network layer would be constantly changing, which is likely to result in the neural network failing to converge. After training, the eigenvalues of each dimension of the sample are found to be unequal after the traditional machine learning algorithm SVM [9] performs uneven scaling (for example, the eigenvalues of each dimension are multiplied by different coefficients). At the scaling level, we assert that such an algorithm is not immutable. Unless the distribution range of each dimension feature is relatively close, it must be normalized for this type of algorithm.

As illustrated in the following equation, the ECABiNet model performs layer normalization on the feature embedding vector $[e_1, \ldots, e_n]$:

$$y = \frac{x - E[x]}{\sqrt{\text{Var}[x] + \epsilon}} * \gamma + \beta \tag{1}$$

where E and Var denote the mean and variance of all samples in this batch on the k-th feature, respectively. β and γ are used to control the ability of the network to express direct mappings that restore the features previously learned by the LN. The final result is the new feature embedding vector:

$$LN(E) = \text{concat}\big(LN(e_1), \ldots, LN\big(e_f\big)\big) \tag{2}$$

where LN denotes the LayerNorm layer and E, e_f denotes the embedding vector.

2.3 ECANET Layer

Since the introduction of SENet [10], the channel attention mechanism has demonstrated considerable promise, and this approach has been shown to be a viable method for improving the overall efficiency of deep convolutional neural networks. SENet is dedicated to developing more complex attention modules to improve performance. For instance, the SENet model makes use of two fully connected layers to enhance its nonlinear capability and fit capability. However, it introduces additional parameters and increases the complexity of the model. To resolve the conflict between complexity and performance, a more efficient channel attention (ECANET) module is proposed in [11], as shown in Fig. 2, which not only reduces the complexity of the model but also maintains the performance by avoiding the dimensionality reduction operation of SENet and proper cross-channel interaction.

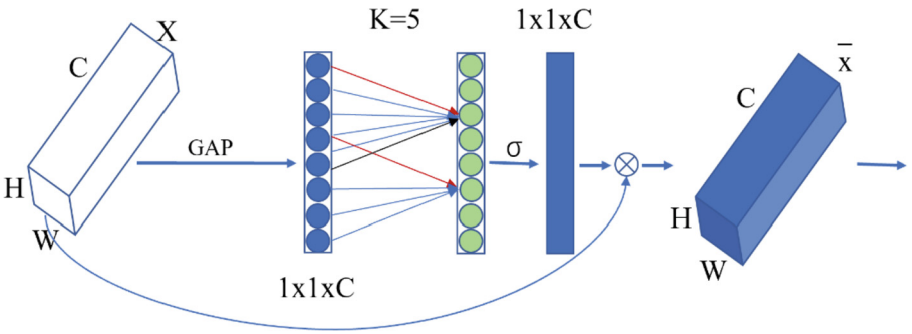

Fig. 2. ECANET model.

The ECANET layer first compresses the embedding vector by the global average pooling method. It compresses the corresponding spatial information on each channel into one value in the corresponding channel; at this time, a pixel represents a channel, and the final dimension becomes $1 \times 1 \times C$. The global average pooling method is shown in the following equation:

$$e_i^{'} = \mathrm{F}_{GAP}(e_i) = \frac{1}{\mathrm{k}} \sum_{d=1}^{\mathrm{k}} e_{id} \tag{3}$$

where k denotes the dimension of the embedding vector and e_i denotes the embedding vector.

After that, the ECANET layer inputs the e_i vector obtained in the previous step into the equation:

$$w_e = \sigma\left(conv1d_k\left(e'\right)\right) \tag{4}$$

The difference of this method is that the size of the convolution kernel used is k, and then to obtain better results, one-dimensional convolution is used to realize the information interaction between channels. conv$1d_k$ denotes one-dimensional convolution, which requires only k parameters, and σ denotes the activation function. The ECANET layer can adaptively select an appropriate value of k, which is proportional to the channel dimension.

The final step is to update the weights. The new embedding vector is obtained by calculating the previously calculated weight factor and the initial embedding vector. The precise formula is depicted by the equation:

$$\begin{aligned} Z &= F_{\text{update}}\left(W_e, E\right) \\ &= \left[w_{e1} \cdot e_1, w_{e2} \cdot e_2, \ldots, w_{ef} \cdot e_f\right] \\ &= \left[v_1, v_2, \ldots, v_f\right] \end{aligned} \tag{5}$$

where E denotes the embedding vector processed by the LayerNorm layer, w_f denotes the corresponding weight embedding vector, and v_f denotes the new embedding vector.

In summary, the ECANET layer has a thorough understanding of the SENet model. The SENet model can learn significant features without adding a large number of parameters and is improved on the basis of the SENet model, which not only reduces complexity but also improves the effect.

2.4 Feature Cross Layer

The ECANET layer learns the new feature embedding vector computed by layerNorm and then computes the weighted feature embedding vector, which is then passed to the bilinear-interaction layer.

Rather than using the conventional inner product (Eq. 6) or Hadamard product (Eq. 7), the bilinear-interaction layer employs a novel bilinear interaction method that combines the two and introduces a new parameter matrix W to learn the feature intersection (Eq. 8).

$$\begin{aligned} &[a_1, a_2, \cdots, a_n] \cdot [b_1, b_2, \cdots, b_n] \\ &= \sum_{i=1}^{n} a_i b_i \end{aligned} \tag{6}$$

$$\begin{aligned} &[a_1, a_2, \cdots, a_n] \odot [b_1, b_2, \cdots, b_n] \\ &= [a_1 b_1, a_2 b_2, \cdots, a_n b_n] \end{aligned} \tag{7}$$

$$p_{i,j} = v_i \cdot W \odot v_j, p_{i,j} \in R^k \tag{8}$$

The intersection vector $p_{i,j}$ can be obtained in three ways:

- All feature groups share a parameter matrix when they are crossed two by two, and the number of extra parameters is $k \times k$, as shown in Eq. 8.
- Each feature group i maintains a parameter matrix W_i with an additional number of parameters $f \times k \times k$, as shown in the following equation:

$$p_{i,j} = v_i \cdot W_i \odot v_j, p_{i,j} \in R^k \tag{9}$$

- Each pair of interaction features $p_{i,j}$ has a parameter matrix W_{ij} with the number of extra parameters $\frac{f \times (f-1)}{2} \times k \times k$, as shown in the following equation:

$$p_{i,j} = v_i \cdot W_{ij} \odot v_j, p_{i,j} \in R^k \tag{10}$$

2.5 DNN Layer

The DNN layer is composed of multiple fully connected layers that are capable of capturing higher-order combinatorial features. Unlike the traditional DeepFM recommendation model, DNN utilizes the output of the combination layer as an embedding vector, which efficiently captures features. As shown in the following equation, the combination layer is primarily responsible for stitching the original embedding vector p with the weight embedding vector q obtained after the ECANET layer:

$$\begin{aligned} c &= F_{\text{concat}}(p, q) = \left[p_1, \cdots, p_n, q_1, \cdots, q_n\right] \\ &= [c_1, \cdots, c_{2n}] \end{aligned} \tag{11}$$

2.6 Output

The overall formulation of ECABiNet is:

$$\hat{y} = \sigma\left(w_0 + \sum_{i=0}^{k} w_i x_i + y_d\right) \tag{12}$$

where σ represents the sigmoid function, $\hat{y}$ is the predicted result of the ECABiNet, k is the feature size, x is an input and w_i is the *i-th* weight of the linear part.

3 Experiment

3.1 Experimental Setup

Datasets This experiment uses two publicly available datasets, and we randomly divided the two datasets into two parts: 90% for training and 10% for testing.

- Criteo. Criteo [12] is one of the most commonly used datasets in the CTR field, and it has 13 continuous characteristics and 26 categorical characteristics. It also has data of more than 40 million user clicks on ads.
- Avazu. Like Criteo, the Avazu [13] dataset is also the most common dataset in the CTR field. It contains more than 40 million user clicks on ads over several days, and it is sorted by time.

Evaluation Criteria AUC and LOGLOSS Are used in this paper to evaluate the model. The AUC value is between 0.5 and 1, with a value closer to 1.0 indicating that the detection method is more authentic. Logloss is the most important probability-based classification metric; the lower the loss score is, the better.

Model Comparison In the experiments, a total of six models are selected for comparison in this paper, namely, FM, AFM, Wide&Deep, DeepFM, FiBiNET and ECABiNet.

Parameter Setting To conduct experiments reasonably, as shown in Table 1, some default hyperparameters are set in this paper.

Table 1. Experimental Hyperparameters.

Parameter name	Parameter value
Dropout	0.5
Optimizer	Adam
Hidden units	(256,128)
Activation	Relu
l2_reg_linear	0.00001
l2_reg_embedding	0.00001

3.2 LayerNorm Effect Comparison

To demonstrate the performance improvement brought by normalizing the embedded feature vector and then performing the feature crossover operation, we will conduct a comparative experiment with the ECABiNet model using LayerNorm and without LayerNorm, where the hyperparameters are as shown in Table 1. The LayerNorm layer is set to the same hidden layer dimension as the ECABiNet model, and the dimension is set to 6 in this experiment. The specific comparison is shown in Fig. 3.

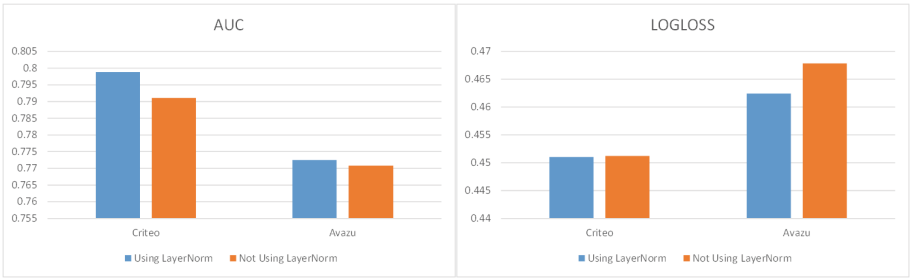

Fig. 3. Comparison of whether to use LayerNorm on the Criteo and Avazu datasets.

From the comparison of AUC and LOGLOSS scores given in Fig. 3, it can be seen that using the LayerNorm layer to perform layer normalization on the feature vector can bring better experimental results.

3.3 Comparison of the Effects of Different Attention Modules

In this part, we compare the effectiveness of the model that uses the ECANET layer as the model to calculate the importance of features and the model that uses the SENET layer to calculate the importance of features through experiments. The results are shown in Fig. 4.

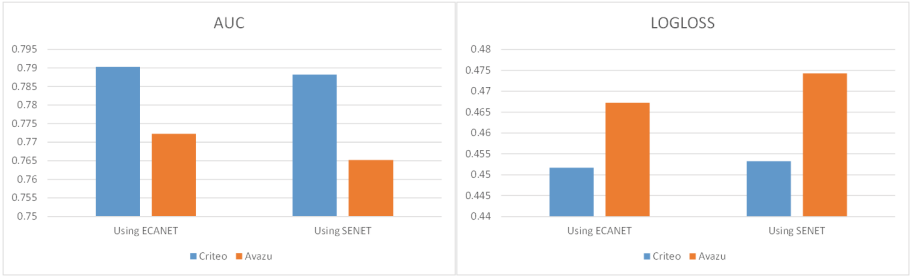

Fig. 4. Comparison of the ECANET and SENET effects on the Criteo and Avazu datasets.

It can be seen from Fig. 4 that the CTR model using the ECANET layer outperforms the SENET layer in both AUC and LOGLOSS.

3.4 Comparison of the Classic Model

To demonstrate the effectiveness of the ECABiNet model, this article compares multiple groups of models using the publicly available Criteo and Avazu datasets. Table 2 illustrates the comparison results:

Table 2. Comparison of forecast results.

	Criteo		Avazu	
	AUC	LOGLOSS	AUC	LOGLOSS
FM	0.7681	0.47831	0.7432	0.41012
AFM	0.7722	0.46217	0.7511	0.39551
Wide&Deep	0.7796	0.46011	0.7545	0.39425
FiBiNet	0.7802	0.45628	0.7598	0.39334
DeepFM	0.7891	0.45243	0.7621	0.39021
ECABiNet	**0.7988**	**0.45102**	**0.7753**	**0.38432**

Being good at using attention modules can sometimes bring good results for CTR models. Comparing the AFM model and the FM model in Table 2, it can be found that the effect of the model can be improved by adding an attention module during feature intersection.

Effectively combining shallow and deep models, that is, learning both high-order and low-order features concurrently, can significantly improve the CTR model. As demonstrated in Table 2, the performance of the deep DeepFM and FiBiNET models is superior to that of the shallow AFM and FM models.

Traditional attention models add attention modules when features intersect, and sometimes adding attention mechanisms to embedded feature vectors works surprisingly well. As shown in Table 2, both ECABiNet and FiBiNET outperform AFM and DeepFM.

3.5 Study HyperParameter

To find suitable hyperparameters for the ECABiNet model, we conduct comparative experiments from different hyperparameters (activation function, dropout, and the number of hidden layers).

– Activation Function
  To select the appropriate activation function more accurately, this paper compares the prediction results of the ECABiNet model using different activation functions on the two datasets. As illustrated in Fig. 5, the relu function is more suitable for deep models.

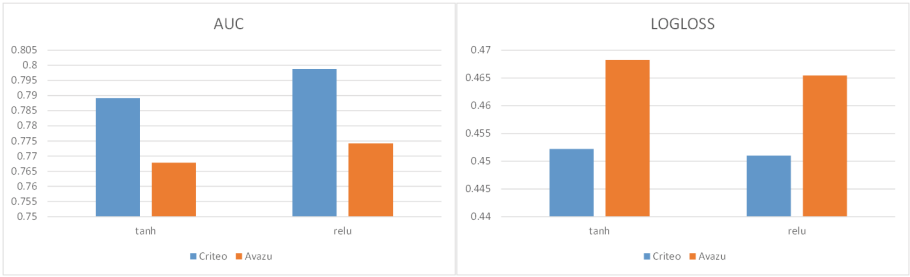

Fig. 5. Comparison of different activation functions on the Criteo and Avazu datasets.

– Dropout
To improve model performance, dropout randomly drops neural units. To improve the performance of the ECABiNet model, this paper studies the performance of dropout from 0-1, as shown in Fig. 6. The model works best when dropout is 0.9.

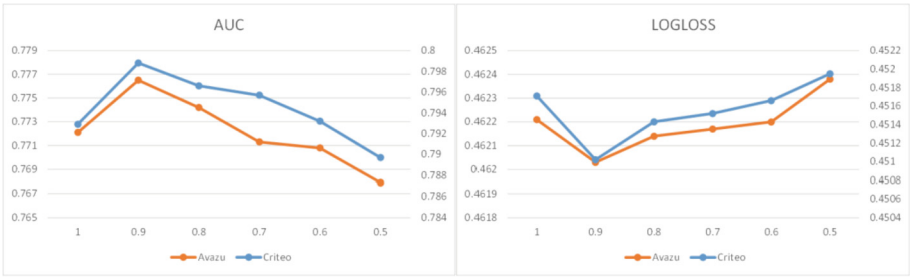

Fig. 6. Comparison of different dropout effects on the Criteo and Avazu datasets.

– Number of hidden layer
In the DNN layer, the number of different hidden layers also affects the performance of the model. This paper compares the effect of the number of hidden layers from layers 1–7 on two public datasets. As shown in Fig. 7, the results show that ECABiNet works best when the number of hidden layers is 5 or 6.

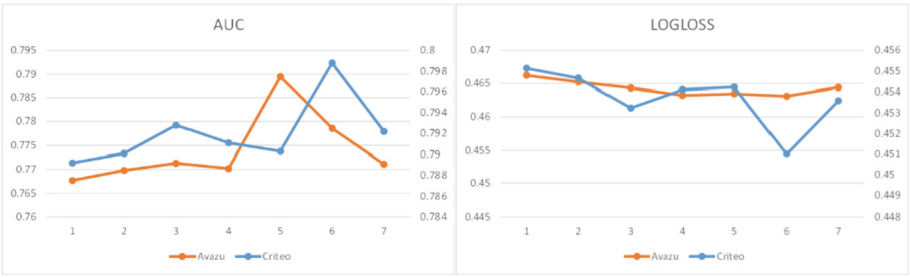

Fig. 7. Comparison of different numbers of hidden layers on the Criteo and Avazu datasets.

4 Related Work

Currently, many CTR models have been proposed in the field of recommender sys-tems. Thanks to the proposal of ResNet [14], paper [15] proposed the deep crossing model, which converts sparse features into low-dimensional dense features by adding an embedding layer and uses a stacking layer, or concat layer, to connect the segment-ed feature vectors. Then, the combination and transformation of features are complet-ed through the multilayer neural network, and finally, the calculation of CTR is com-pleted with the scoring layer. On the basis of the Deep Crossing model, the FNN model [16] uses the hidden layer vector of FM as the embedding of user and item, which avoids training the embedding from random state completely, thereby improving the recommendation effect. The traditional DNN directly completes the intersection and combination of features through multilayer fully connected layers, but this method lacks a certain 'target', so the paper in [17] proposed the PNN model. To balance the memory ability and generalization ability, Google proposed the Wide&Deep model. However, the wide part requires artificial feature engineering, which leads to increased workload. Therefore, the DeepFM model is proposed. To address the issue of the insufficient expression capability of the wide part, Google published the DCN model [18] in the following year. The main idea is to use the cross network to replace the original wide part, which increases the interaction between features. Considering the possibility of improvement in the DNN part, paper [19] proposes the NFM model. From the perspective of modifying the second-order part of the FM, the NFM model replaces the feature intersection part of the FM with a DNN with a Bi-interaction Pooling layer, forming the unique Wide&Deep architecture improves the recommendation effect. User characteristics are important, but introducing the interests of users into the model often brings unexpected effects. Paper [20] proposed the DIEN model, which is not only a further 'evolution' of the DIN model [21], but more importantly, DIEN simulates the process of user interest evolution by introducing the sequence model AUGRU.

5 Conclusions

In the CTR field, new models are constantly being introduced, but few have considered the importance of embedding vector layers. The ECANET network layer is used in this paper to teach the CTR model about the importance of embedding vectors, which

improves the performance of the model. Additionally, this paper introduces LayerNorm in the embedded feature vector layer to normalize the features and improve accuracy. Our experimental results also show that the ECABiNet model outperforms existing models on both datasets.

Acknowledgment. This work is supported by Hainan Province Science and Technology Special Fund, which is Research and Application of Intelligent Recommendation Technology Based on Knowledge Graph and User Portrait (No.ZDYF2020039). Thanks to Professor CaiMao Li, the correspondent of this paper.

References

1. Rendle, S.: Factorization machines. In: ICDM 2010, The 10th IEEE International Conference on Data Mining, Sydney, Australia, 14–17 December 2010 (2010)
2. Juan, Y., Zhuang, Y., Chin, W.S., Lin, C.J.: Field-aware factorization machines for ctr prediction. In: Proceedings of the 10th ACM conference on recommender systems, pp. 43–50 (2016)
3. Cheng, H.T., et al.: Wide & deep learning for recommender systems. In: Proceedings of the 1st workshop on deep learning for recommender systems. pp. 7–10 (2016)
4. Guo, H., Tang, R., Ye, Y., Li, Z., He, X.: Deepfm: a factorization-machine based neural network for ctr prediction. In: Twenty-Sixth International Joint Conference on Artificial Intelligence (2017)
5. Huang, T., Zhang, Z., Zhang, J.: Fibinet: 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 (2019)
6. Rivera-Trigueros, I.: Machine translation systems and quality assessment: a systematic review. Language Resources and Evaluation 1–27 (2021)
7. Xiao, J., Ye, H., He, X., Zhang, H., Wu, F., Chua, T.S.: Attentional factorization machines: Learning the weight of feature interactions via attention networks. arXiv preprint arXiv:1708.04617 (2017)
8. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016)
9. Chauhan, V.K., Dahiya, K., Sharma, A.: Problem formulations and solvers in linear svm: a review. Artif. Intell. Rev. **52**(2), 803–855 (2019)
10. Hu J, Shen L, Sun G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. Pp. 7132–7141 (2018)
11. Wang, Q., Wu, B., Zhu, P., Li, P., Hu, Q.: Eca-net: Efficient channel attention for deep convolutional neural networks. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
12. Kaggle Science Community: Display advertising challenge: Predict click-through rates on display ads. https://www.kaggle.com/c/criteo-display-ad-challeng (2014)
13. Kaggle Science Community: Click-Through Rate Prediction: predict whether a mobile ad will be clicked. https://www.kaggle.com/c/avazu-ctr-prediction (2015)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778 (2016)
15. Shan, Y., Hoens, T.R., Jiao, J., Wang, H., Yu, D., Mao, J.: Deep crossing: webscale modeling without manually crafted combinatorial features. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp. 255–262 (2016)

16. Zhang, W., Du, T., Wang, J.: Deep learning over multi-field categorical data. In: European conference on information retrieval, pp. 45–57. Springer (2016)
17. Qu, Y., Cai, H., Ren, K., Zhang, W., Yu, Y., Wen, Y., Wang, J.: Product-based neural networks for user response prediction. In: 2016 IEEE 16th International Conference on Data Mining (ICDM), pp. 1149–1154. IEEE (2016)
18. Wang, R., Fu, B., Fu, G., Wang, M.: Deep & cross network for ad click predictions. In: Proceedings of the ADKDD'17, pp. 1–7 (2017)
19. He, X., Chua, T.S.: 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 (2017)
20. Zhou, G., et al.: Deep interest evolution network for click-through rate prediction. In: Proceedings of the AAAI conference on artificial intelligence, vol. 33, pp. 5941–5948 (2019)
21. Zhou, G., et al.: 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 (2018)