The Improved XdeepFM Algorithm Based on Attention Mechanism and Factorization Machine

Yingqiao Wang and Zhifeng Wu(✉)

Tianjin University of Technology and Education, TianJin 300222, China
zhifeng.wu@163.com

Abstract. This study proposes an improved recommendation model called MHSA-XdeepFM, which incorporates multi-head self-attention mechanism and an enhanced residual network into XdeepFM to enhance the performance of click-through rate prediction tasks. Click-through rate prediction is one of the key tasks in recommendation systems, aiming to predict the probability of users clicking on candidate items. XDeepFM is a click-through rate prediction model that combines deep neural networks(DNN) with Compressed Interaction Network (CIN) to capture both low-order and high-order feature interactions, while introducing linear layers to emphasize the importance of first-order features. However, in its final output, XDeepFM simply concatenates the outputs of various sub-models without fully considering the connections between them and the importance of features, which may lead to information redundancy and imbalance in feature fusion and representation, resulting in poor accuracy in click-through rate prediction. To address this issue, the study introduces a multi-head self-attention mechanism that allows the model to adaptively focus on the importance of different sub-models. Additionally, the Adaptive Feature Interaction Modeling AFM is incorporated to adaptively model second-order feature interactions. Through these improvements, the model can better capture the correlations and interaction patterns between features, thus improving the accuracy and effectiveness of click-through rate prediction. Experimental results demonstrate that the model outperforms the traditional XdeepFM on the publicly available Criteo dataset, providing important improvements and guidance for the application of recommendation systems in real-world scenarios. The model shows enhancements in performance metrics such as AUC and LogLoss, proving the practical significance of this research in improving the modeling of feature interactions in click-through rate prediction tasks.

Keywords: CTR prediction · Feature Interaction · Multi-Head Self-Attention · Factorization Machines · Neural Networks

1 Introduction

CTR prediction (Click-through Rate Prediction) is one of the key tasks in recommendation systems [1] and online advertising [2]. With the rapid development of the Internet, personalized recommendations and targeted advertising have become essential requirements for major platforms. The goal of CTR prediction is to forecast the probability

Y. Cao and X. Shao (Eds.): DIONE 2023, LNICST 515, pp. 49–59, 2025.
https://doi.org/10.1007/978-3-031-80713-8_4

of users clicking on candidate items based on user information and historical behavior data. Accurate CTR prediction can provide more relevant and personalized recommendations to users, enhancing user satisfaction and advertising effectiveness. Research on ad CTR prediction has a long history, and traditional machine learning algorithms such as Logistic Regression (LR) [3] and Factorization Machines [4] have achieved good results in dealing with raw features and second-order feature interactions and have been applied in industries. However, such CTR prediction models typically rely on manually designed feature combinations to capture feature interactions, which are limited by the manually designed feature combinations and struggle to capture complex higher-order interactions. Additionally, traditional models often require extensive feature engineering and domain knowledge, increasing the complexity of model design and maintenance. In recent years, deep learning techniques have made significant progress in CTR prediction tasks. Deep learning models, such as DNN (Deep Neural Networks) [5], have been widely applied and successful in CTR prediction by learning nonlinear representations of features through multiple layers of neural networks. Deep & Cross [6] is an improvement that uses a method of building cross networks in contrast to DNN. In this network, each layer achieves explicit high-order feature interactions by performing dot products between feature vectors. Wide & Deep [7] introduced the idea of combining traditional models with deep learning models to achieve a balance between breadth and depth in feature representations for CTR prediction. The model captures the breadth of features through linear models and learns the depth representations of features through a DNN. Wide & Deep model can simultaneously utilize shallow features and deep features, thus improving CTR prediction performance. Building upon this, DeepFM [8] combines FM and DNN to more effectively extract breadth information. Empirical evidence shows that this combination can effectively capture both low-order and high-order feature interactions, thereby improving CTR prediction accuracy. To further enhance CTR prediction models, DCN (Deep & Cross Network) [9] and xDeepFM [10] were proposed. The DCN model learns high-order feature interactions through cross-network structures, allowing for complex interactions between features. xDeepFM, building on DCN's Cross Network, introduces the CIN (Compressed Interaction Network) to implement vector-wise high-order explicit feature interactions, showing the best performance among network structures that combine deep and shallow models. However, in xDeepFM, each submodel operates in parallel, and this parallel output scheme isolates the information between sub-modules, limiting their potential correlations. The key to addressing this issue is to introduce a mechanism that promotes information flow and interaction between different sub-modules. In recent years, attention mechanisms have been widely applied in various fields of deep learning, such as machine translation, computer vision, etc. [11, 12]. CFM introduced self-attention to pool the output of the embedding layer and used a CNN to learn the relationships between features. In this paper, the multi-head self-attention mechanism [13] and residual network [14] are introduced into the CTR prediction model to enhance its performance and accuracy. The multi-head self-attention mechanism can adaptively focus on the importance of different features, extracting feature correlations and interaction patterns. The residual network can learn feature representations and transformations at a deeper level, enhancing the model's representational and generalization capabilities. Simultaneously, the Attentional Factorization Machine

(AFM) [15] is introduced to replace LR, alleviating storage pressure and overfitting risks caused by high-dimensional sparse features, and emphasizing the importance of second-order feature interactions for prediction results.

2 Related Work

2.1 Xdeepfm

In Deep&Cross [16], Cross Layer operates the inner product of the vectors of all domains, which belongs to the feature interaction at the element level, which causes the model to lose the awareness of the domain, which means that the elements under the same domain may have different weights when interacting with features. x DeepFM [27] is an improvement of the Wide&Deep model in terms of structure, and in terms of function, it is an improvement of the Deep&Cross model in terms of the above defects. The architecture of the XDeepFM algorithm is illustrated in Fig. 1.

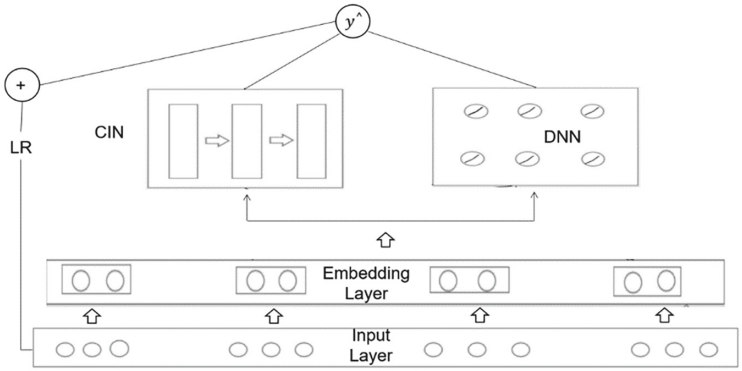

Fig. 1. The structure diagram of XDeepFM.

2.2 Multi-Head Self-Attention

The Multi-Head Self-Attention mechanism is a powerful attention mechanism. Attention mechanism is first proposed in the context of neural machine translation [11] and other fields. Vaswani et al. [13] further proposed multi-head self-attention to model complicated dependencies between words in machine translation. It achieves attention to different aspects and correlations between different features through multiple independent attention heads. In the task of click-through rate prediction, the Multi-Head Self-Attention mechanism can be applied to different components of the model, such as feature cross layers or attention mechanisms, to enhance the modeling capacity of feature interactions and improve the accuracy and effectiveness of click-through rate prediction.

2.3 AFM

Currently, there are many research efforts aimed at improving FM models, including the introduction of Attentional Factorization Machine (AFM).

Traditional FM models learn feature interactions through factorization but fail to consider the non-linear and complex relationships between features. To address this issue, AFM introduces attention mechanisms to adaptively focus on the importance of different feature interactions. With the attention mechanism, AFM dynamically allocates attention based on the importance of each feature pair.

In AFM, attention mechanisms are used to model the importance of feature pairs and are applied to the calculation of feature interactions. By multiplying the cross-term of feature pairs with their corresponding attention weights, AFM can more accurately capture the interaction effects between different feature pairs. This introduction of attention mechanisms allows AFM to better adapt to the characteristics of the data and provide more accurate representations of feature interactions.

3 Our Proposed Model

3.1 General Description

In response to the drawbacks of information isolation between modules and insufficient deep feature representation capacity in the xDeepFM model, this section introduces a new model called MHSA-XdeepFM. It consists of three sub-models arranged in parallel layers: the AFM layer for second-order interaction, the CIN layer for high-order explicit feature interaction, and the DNN layer. Additionally, a Multi-Head Self-Attention Network layer is incorporated to explore the connections among the sub-models. The following sections will provide a detailed introduction to each component of the model. The structure of the MHSA-XdeepFM model is illustrated in Fig. 2.

3.2 Problem Formulation

Assume the training dataset comprises m categorical feature fields (where m represents the number of feature fields) along with corresponding labels $y \in \{0, 1\}$ denoting user click behaviors. The primary aim of CTR prediction is to estimate $\hat{y}$ for the given m feature fields, indicating the probability of a user clicking.

3.3 Input and Embedding Layer

In the input layer, the original features undergo one-hot encoding to generate high-dimensional sparse features, which are then processed through an embedding layer to compress and map the features into a dense D-dimensional vector. The specific embedding operation involves matrix multiplication between an $n \times m$ matrix formed by concatenating the one-hot vectors and an $m \times d$ embedding matrix, resulting in: $e = [e^1, e^2, \ldots, e^n]$, where n represents the number of features, $e^i \in R^{n \times d}$ denotes the embedding vector of a feature, m is the dimension of the original features, and d is the dimension after embedding. Typically, $d << m$, meaning the dimension of embedded features is

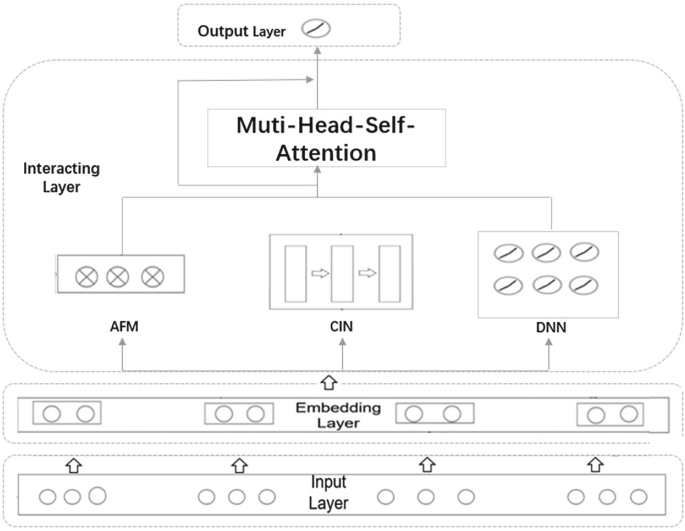

Fig. 2. The structure diagram of MHSA-XDeepFM.

much smaller than that of the original features, effectively addressing the storage space waste caused by sparse data. Finally, these low-dimensional embedding vectors are fed into both shallow and deep models as input values.

3.4 AFM Layer

In our research, we found that introducing second-order feature interactions can significantly improve the model's performance compared to first-order feature interactions. AFM is an attention-based factorization machine model used for tasks such as recommendation systems and click-through rate prediction. The AFM model utilizes attention weights to model second-order feature interactions, and its specific formula is as follows:

$$Y_{AFM} = Relu(w_0 + \sum_{i=1}^{n} w_i x_i + \frac{1}{2} \sum_{i=1}^{n} \sum_{j=i+1}^{n} AFM_{ij}) \tag{1}$$

Y_{AFM} represents the model's predicted output, w_0 is the bias term, w_i is the linear weight of the i-th feature, and x_i is the input of the i-th feature, Relu denotes the activation function AFM_{ij} represents the interaction weight between feature i and feature j, indicating the attention between them.

The calculation of the attention weight AFM_{ij} is as follows:

$$AFM_{ij} = \text{softmax}(a_{ij}) \tag{2}$$

a_{ij} is the attention score between feature i and feature j, obtained by element-wise multiplication and aggregation of their embedding vectors. The softmax function is then applied to normalize the attention scores and obtain attention weights.

3.5 DNN Layer

In the DNN layer, the model captures complex relationships between features by learning implicit feature interactions. The feature interactions in the DNN layer are achieved through a fully connected feedforward neural network. The formula for its forward propagation is as follows:

$$X^l = \text{Relu}\Big(W_l X^{l-1} + b_l\Big) \tag{3}$$

In Eq. (3), $X^0 \in R_m \times D$ represents the output of the embedding layer as the input to the DNN layer; W_l、b_l represent the weight matrix and bias vector, respectively. The final output of the DNN layer is given by:

$$Y_{DNN} = Relu(X^l + b) \tag{4}$$

3.6 CIN Layer

The CIN layer achieves explicit high-order feature interactions at the vector level, and its complexity does not significantly increase with the degree of interactions. The CIN layer utilizes cross-product and pooling operations for feature interactions. Specifically, for each layer, the calculation formula is as follows:

$$X_{h,*}^k = \sum_{i=1}^{H_{k-1}} \sum_{j=1}^{m} W_{ij}^{k,h} (X_{I,*}^{k-1} \circ X_{j,*}^0) \tag{5}$$

Where $X^0 \in R^{m*D}$ represents the input matrix, where m denotes the number of vectors, and D denotes the dimension. $X_{j,*}^0$ represents the j-th feature vector of the 0-th layer, similarly, $X_{I,*}^{k-1}$ and $X_{h,*}^k$ can be obtained. $W_{ij}^{k,h} \in H_{k-1} \times m$ represents the weight matrix during the computation of the h-th cross-feature in the k-th layer, where $1 \leq h \leq H_k$, and $\circ$ denotes element-wise Hadamard product between vectors. The i-th feature interaction vector of the (k-1)-th layer is element-wise multiplied with the m feature vectors in the input matrix, resulting in $H_{k-1} \times m$ D-dimensional vectors. Then, a pooling operation is applied to obtain a D-dimensional vector, which represents the value of the h-th feature vector in the k-th layer. Ultimately, the k-th layer will have an $H_k \times D$ feature interaction matrix. CIN's computation is done at the vector level, which involves element-wise Hadamard product operations between vectors. After obtaining the output of the final layer, each layer's vectors in $X^0 \sim X^k$ are separately pooled along the D dimension, resulting in k D-dimensional pooling vectors X^i, where i $\in 1 \sim k$. These pooling vectors are then pooled again to obtain:

$$P^+ = [P^1, P^2, \ \ldots, P^T] \in R \sum_{i=1}^{T} H_i \tag{6}$$

The final result of the CIN network is:

$$Y_{CIN} = Relu(P^+ + b) \tag{7}$$

3.7 Multi-head Self-Attention Layer

In Sects. 3.4, 3.5, and 3.6, we obtained three n-dimensional vectors, Y_{AFM}, Y_{CIN}, Y_{DNN}, which are the output results of AFM, CIN, and DNN, respectively. Next, we combine these three feature interaction vectors into a matrix and feed it into the multi-head self-attention network for computation. By calculating the correlation between features in multiple sub-space networks, the network can adaptively capture the complex relationships between features and generate corresponding attention weights.

Taking a specific sub-space h as an example, for feature m, we perform a linear transformation through matrix multiplication to obtain its vector representation in the attention space. For each feature in the specific attention sub-space h, we have three representation vectors: $Q^{(h)} = W_q^{(h)} * \mathrm{M}$ (8), $K^{(h)} = W_k^{(h)} * \mathrm{M}$ (9), and $V^{(h)} = W_v^{(h)} * \mathrm{M}$ (10), where M is the matrix obtained by stacking all features. $W_q^{(h)}$, $W_k^{(h)}$, and $W_v^{(h)}$, are matrices used to map the original matrix space to the new space for attention calculation. The mapped matrices are then used for attention computation. Next, we update the representation of feature m in sub-space h by combining all relevant features guided by the coefficients $\alpha_{m,k}^{(h)}$.

$$e_m^{(h)} = \sum\nolimits_{k=1}^{M} \alpha_{m,k}^{(h)} \left(W_v^{(h)} e_k\right) \tag{8}$$

$\mathrm{e}_m^{(h)}$, The representation of the feature vector after being weighted in the h space.

$$\alpha_{m,k}^{(h)} = exp(\psi^{(h)}(m,k)) / \sum\nolimits_{l=1}^{M} exp(\psi^{(h)}(m,k)) \tag{9}$$

$$\psi^{(h)}(m,k) = \left\langle W_q^{(h)} m, W_k^{(h)} k \right\rangle \tag{10}$$

Where $\psi^{(h)}(\cdot,\cdot)$ is the attention function, which defines the similarity between feature m and k. In this paper, a simple and effective inner product method is used. In order to preserve the previously learned feature interaction results, standard residual connections are added to the network.$e_m^{Res} = \mathrm{ReLU}(\mathrm{e_m}+\mathrm{m})$ (14) represents the non-linear activation function. Assuming there are H attention spaces, the results from each attention space are concatenated to obtain the final representation of features. By repeating this process for each feature interaction vector, we obtain the feature interaction vectors with attention scores, namely e_{AFM}^{Res}, e_{CIN}^{Res}, e_{DNN}^{Res}.

3.8 Output Layer

The overall prediction result is the sum of the outputs from each module:

$$y = \sigma\left(w_{AFM}^T e_{AFM}^{Res} + w_{CIN}^T e_{CIN}^{Res} + w_{DNN}^T e_{DNN}^{Res} + b\right) \tag{11}$$

CTR prediction tasks typically use the logarithmic loss function as the objective function, aiming to minimize it during training. The formula is as follows:

$$L = -\frac{1}{N} \sum\nolimits_{i=1}^{N} (y_i)\, log(y_i^{\wedge}) + (1 - y_i)\, log(1 - y_i^{\wedge}) \tag{12}$$

Where N is the total number of training samples, i is the index of the training sample, $\hat{y_i}$ represents the model's predicted value, and y_i is the true value. Adam optimizer is used to update the parameters, and to avoid overfitting, Dropout modules and L2 regularization are also added.

4 Experiments and Conclusions

In this section, experiments will be conducted to evaluate the performance of the proposed model MASA-XdeepFM. Firstly, the datasets and parameter settings used in the experiments are introduced, and some comparative models are presented. Subsequently, MASA-XdeepFM is compared with other models on the datasets to assess its effectiveness. Finally, experimental conclusions are drawn to validate the rationality and effectiveness of the proposed model.

4.1 Dataset and Data Preprocessing

To evaluate the model's predictive performance, this study conducted experiments using the Criteo dataset and the Avazu dataset. These datasets were provided by leading advertising companies and contain millions of ad impressions and click feedback, making them suitable benchmarks for click-through rate estimation. Each ad has descriptive features, and the data were randomly split into training, testing, and validation sets in a ratio of 4:1:1. Preprocessing was performed on both datasets, including mapping categorical features to numerical values, normalizing numerical features, checking for and removing any outliers (e.g., duplicates and missing values). The specific parameters of the datasets are shown in Table 1.

Table 1. Dataset parameter

Dataset	Instance	Fields	Features
Criteo	45M	39	998,960
Avazu	40M	24	1,544,488

4.2 Evaluation Metrics

In the click-through rate (CTR) prediction problem, common evaluation metrics are AUC (Area Under the Curve) and LogLoss (Logarithmic Loss). AUC and LogLoss are two commonly used evaluation metrics to assess the performance of classifiers on a given dataset. AUC utilizes the ROC (Receiver Operating Characteristic) curve to visualize the performance of the classifier, where a larger area indicates better classifier performance. Additionally, since the datasets used here are predominantly composed of negative samples, AUC takes into account the classifier's ability to classify both positive and negative examples, making it less sensitive to class imbalance compared

Table 2. The performance of popular CTR prediction models on the Criteo dataset is evaluated using the optimal parameter values obtained after tuning.

Models	AUC	Logloss
AFM	0.7965	0.4541
DeepFM	0.8085	0.4445
DCN	0.8067	0.4419
XdeepFM	0.8091	0.4418
MHSA-XdeepFM	0.8134	0.4387

Table 3. The performance of popular CTR prediction models on the Avazu dataset is evaluated using the optimal parameter values obtained after tuning.

Models	AUC	Logloss
AFM	0.7733	0.4742
DeepFM	0.7797	0.4698
DCN	0.7785	0.4727
XdeepFM	0.7826	0.4691
MHSA-XdeepFM	0.7872	0.4631

to other classification metrics.On the other hand, LogLoss measures the probability of classifier prediction errors, where a smaller value indicates better classifier performance. Therefore, in this study, AUC and LogLoss are used as evaluation metrics to better assess the model's performance.

4.3 Implementation Details

All experiments in this paper were implemented in the deep learning platform PyTorch. The programming language used was Python 3.6, and the PyTorch version was 1.7.1. The batch size for each round of experiments was set to 4096, and the learning rate for all models was set to 0.001. For all deep learning models, the dropout rate was set to 0.5, and L2 regularization was set to 0.1667,and early stopping was applied during training. The experiment conducted a longitudinal comparison by selecting AFM, DeepFM, DCN and XdeepFM models to evaluate the relative improvement of the MHSAXdeepFM model over the XdeepFM model and other classical models. The experimental results are presented in Tables 2 and 3.

From the experimental results, it can be observed that single models like AFM, which rely on embedding +MLP, perform noticeably worse than models combining explicit and implicit feature interactions. In addition, experimental results on Criteo and Avazu datasets show that MHSAXdeepFM outperforms XdeepFM and other models using the Wide&Deep This indicates that the feature interaction model, which combines attention-enhanced high-order and low-order interactions, has certain advantages.

5 Conclusions

This paper proposes a novel click-through rate prediction model that combines attention mechanism with second-order feature interaction. The model replaces the original LR model with a second-order attention model called AFM, which achieves weighted second-order feature interactions. Additionally, it applies a multi-head self-attention network with residual structure to adaptively distinguish the three sub-modules of the model in multiple subspaces. This enables the model to capture the correlations and interaction patterns among features, thereby enhancing the overall performance and representational capacity of the model.

References

1. Ren, K., Zhang, W., Rong, Y., Zhang, H., Yu, Y., Wang, J.: User response learning for directly optimizing campaign performance in display advertising. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, pp. 679–688 (2016)
2. McMahan, H.B., et al.: Ad click prediction: a view from the trenches. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1222–1230 (2013)
3. Dafang, Z., Zidong, W., Leimin, Z., et al.: Deep field relation neural network for click-through rate prediction. Inform. Sci. **577**, 128–139 (2021)
4. Rendle, S.: Factorization machines. In: Proceedings of the Tenth IEEE International Conference on Data Mining, Sydney, pp. 995–1000 (2010)
5. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708 (2017)
6. Ruo-xi, W., Gang, F., et al.: Deep&cross network for ad click predictions. In: Proceedings of the ADKDD17, NewYork, NY, United States, pp. 1–7 (2017)
7. Heng-Tze, C., Levent, K., Jeremiah, H.. et al.: Wide&deep learning for recommender system. In: Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, pp. 7–10 (2016)
8. Hui-feng, G., Rui-ming, T., Yun-ming, Y., et al.: DeepFM: a factorization-machine based neural network for CTR prediction. arXiv preprint arXiv: 1703.04247 (2017)
9. Ruoxi, W., Bin, F., Gang, F., et al.: Deep&Cross network for Ad click predictions. In: Proceedings of the Twenty-third ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1–7 (2017)
10. Lian, J.-x., Zhou, X.-h., Zhang, F.-z., et al.: xdeepfm: combing explicit and implicit feature interactions for recommender systems. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1754–1763 (2018)
11. Devlin, J., Chang, M.-w., Lee, K., et al.: Bert: pre-training of deep bidirectional transformers for language understanding. arXiv preprint, arXiv: 1810.04805 (2018)
12. Jinhyuk, L., Wonjin, Y., Sungdong, K., et al.: BioBERT: a pre-trained biomedical language representation model for biomedical text mining. Bioinformatics **36**(4), 1234–1240 (2020)
13. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 6000–6010 (2017)
14. Xiao, J., Ye, H., He, X.-n., et al.: Attentional factorization Machines: learning the weight of feature interactions via attention networks. arXiv preprint, arXiv:1708.04617 (2017)

15. 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)
16. Shan, Y., Ryan Hoens, T., Jiao, J., Yu, H.W.D., Mao, J.C.: Deep crossing: web-scale modeling withing manually crafted combinatorial features. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 255–262. ACM (2016)