Information Processing and Management xxx (xxxx) xxxx

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

HoAFM: A High-order Attentive Factorization Machine for CTR Prediction

Zhulin Tao[*,a], Xiang Wang[b], Xiangnan He[c], Xianglin Huang[a], Tat-Seng Chua[b]

[a] Communication University of China, Beijing, China
[b] National University of Singapore, Singapore
[c] University of Science and Technology of China, Hefei, China

ARTICLE INFO

Keywords:
Factorization machines
High-order feature interactions
Attention mechanism
Deep neural network

ABSTRACT

Modeling feature interactions is of crucial importance to predict click-through rate (CTR) in industrial recommender systems. However, manually crafting cross features usually requires extensive domain knowledge and labor-intensive feature engineering to obtain the desired cross features. To alleviate this problem, the factorization machine (FM) is proposed to model feature interactions from raw features automatically. In particular, it embeds each feature in a vector representation and discovers second-order interactions as the product of two feature representations. In order to learn nonlinear and complex patterns, recent works, such as NFM, PIN, and DeepFM, exploited deep learning techniques to capture higher-order feature interactions. These approaches lack guarantees about the effectiveness of high-order pattern as they model feature interactions in a rather implicit way. To address this limitation, xDeepFM is recently proposed to generate high-order interactions of features in an explicit fashion, where multiple interaction networks are stacked. Nevertheless, xDeepFM suffers from rather high complexity which easily leads to overfitting.

In this paper, we develop a more expressive but lightweight solution based on FM, named *High-order Attentive Factorization Machine* (HoAFM), by accounting for the higher-order sparse feature interactions in an explicit manner. Beyond the linearity of FM, we devise a cross interaction layer, which updates a feature's representation by aggregating the representations of other co-occurred features. In addition, we perform a bit-wise attention mechanism to determine the different importance of co-occurred features on the granularity of dimensions. By stacking multiple cross interaction layers, we can inject high-order feature interactions into feature representation learning, in order to establish expressive and informative cross features. Extensive experiments are performed on two benchmark datasets, Criteo and Avazu, to demonstrate the rationality and effectiveness of HoAFM. Empirical results suggest that HoAFM achieves significant improvement over other state-of-the-art methods, such as NFM and xDeepFM. We will make the codes public upon acceptance of this paper.

1. Introduction

The prediction of CTR is crucial for many online services, ranging from personalized recommendation (Guo, Tang, Ye, Li, & He, 2017), sponsored search (Kutlwano & Ramaboa, 2018; Richardson, Dominowska, & Ragno, 2007; Xiao & Baia, 2019) to targeted

* Corresponding author.
E-mail addresses: taozhulin@gmail.com (Z. Tao), xiangwang@u.nus.edu (X. Wang), huangxl@cuc.edu.cn (X. Huang), chuats@comp.nus.edu.sg (T.-S. Chua).

https://doi.org/10.1016/j.ipm.2019.102076
Received 15 March 2019; Received in revised form 18 June 2019; Accepted 2 July 2019

advertising (Juan, Zhuang, Chin, & Lin, 2016). Taking sponsored search as an example, when users retrieve some keywords purchased by the advertiser, the search engine would present the corresponding ads; then, the advertiser pays for each ad click of users. Hence, at the core of CTR prediction is the estimation of the probability that a user would click on an ad or item, based on the features of users, items and contexts. Prior efforts have shown the importance of modeling feature interactions in CTR prediction (Guo et al., 2017; He & Chua, 2017; Juan et al., 2016; Qu et al., 2019; Rendle, 2010), aiming at establishing informative signals of user preferences. For example, female teenagers tend to purchase the products with pink color, which suggests a predictive signal as the feature combination of user gender, user age, and product color. Towards this end, prior efforts (Cheng & Cantú-Paz, 2010; Cheng et al., 2016; Wang, He, Feng, Nie, & Chua, 2018) generate *cross features* by manually aggregating multiple features together. Such feature crossing is capable of revealing the relationships and dependencies among individual features. Nevertheless, the manually crafted features usually require extensive domain knowledge and labor-intensive feature engineering. Moreover, it is difficult to generalize these cross features to a new task or domain. Hence, it is crucial to automatically model the feature interactions in order to improve the prediction accuracy of CTR.

A line of research is proposed to learn feature interactions automatically from raw data. Among these methods, FM (Rendle, 2010) has attracted much attentions from both industry and academia and has been viewed as a benchmark solution to CTR prediction. In particular, it embeds each feature into a vector representation, and employs inner product of pairwise feature representations as their interactions. Although FM yields strong performance, one of its shortcomings is that only linear second-order (*i.e.,* pairwise) feature interactions are considered, while the nonlinear and high-order feature interactions are ignored (He & Chua, 2017). This limits the representation capability and prediction performance of FM.

Recent works exploit deep neural networks (DNNs) to model higher-order and non-linear feature interactions, such as NFM (He & Chua, 2017), DeepFM (Guo et al., 2017), and PIN (Qu et al., 2019). For example, NFM (He & Chua, 2017) devises a bilinear interaction layer to obtain the element-wise product of pairwise feature interactions; thereafter, it stacks multiple non-linear layers over the products to capture higher-order patterns. DeepFM is composed of FM and DNN components, where the DNN part is used to mine the underlying relations between features. As such, these DNN-based methods are capable of learning sophisticated and selective feature interactions. Despite their great success, we argue that these methods model feature interactions in a rather implicit manner. Due to the lack of explicit modeling, there lacks a guarantee that the high-order interactions are captured. Moreover, as the DNN component serves as a black-box model, it is hard to figure out what feature interactions are captured and at which order (Lian et al., 2018).

In this paper, we aim to explore the modeling of high-order feature interactions in an explicit manner and demonstrate its impact on the performance of CTR prediction. Although the explicit modeling of feature interactions has been considered in a very recent work — xDeepFM (Lian et al., 2018), it suffers from high complexity. Specifically, the compressed interaction network (CIN) is devised to generate feature interactions as the outer product of their representations, and then compress the feature maps derived from the outer product to update the representation of each feature. By stacking multiple CINs, xDeepFM encodes high-order feature interactions into the feature representations. However, it suffers from generalization and scalability issues, due to the complex operations where the outer product considers all pairwise dimension-wise interactions of two paired feature representations. Moreover, to exhibit the importance of each feature interaction, some prior efforts like AFM (Xiao et al., 2017) introduce the attention mechanism (Chen et al., 2017; Vaswani et al., 2017) into the prediction modeling. However, only the second-order feature interactions are considered, while leaving the higher-order interactions untouched.

In this work, we proposed a new model, termed *HoAFM*, to explore high-order feature interactions in an explicit and efficient manner. At its core is our carefully designed *cross interaction layer*, which updates a features representation by aggregating the representations of other co-occurred features. By stacking multiple such layers, HoAFM is capable of capturing the high-order feature interactions. Moreover, a *bit-wise sparse attention mechanism* is adopted to learn the weight of each interaction at the granularity of feature dimensions, such that the sparse attentive weights can not only suggest the importance of a feature interactions, but also indicate the varying representation ability of each feature. Beyond feature representations, the model parameters largely come from the attention mechanism, which is lightweight as compared to xDeepFM.

Owing to these two designs, HoAFM is capable of capturing the complicated impacts of high-order feature interactions in an explicit and efficient manner. We conduct extensive experiments on two benchmark datasets from Criteo and Auazu, verifying the effectiveness of HoAFM. This work opens up research opportunities of CTR prediction, as well as cross feature modeling in generic prediction tasks. Furthermore, our proposed HoAFM can be directly used in real-world scenarios and helps increase the revenue of sponsored search.

The key contributions of this work are:

- We proposed a new model named HoAFM to encode high-order feature interactions into feature representations in an explicit and efficient manner.
- We highlight the varying importance of interactions via two bit-wise attention mechanism.
- We perform extensive experiments on two datasets to verify the effectiveness of HoAFM for CTR prediction. We have released the code to facilitate further development on high-order feature interactions modeling[1]

[1] https://github.com/zltao/HoAFM.

2. Related works

In the last years, many models are devised to improve the performance of prediction, while the models based on feature interactions aim to incorporate the relationships between features into the learning process of instance representation, further boost the prediction performance. The implicit fashion leverages neural networks (e.g., MLP used in NFM) to approximate the underlying interactions between features. While achieving great success, these approaches serve as black-box and cannot explicitly present the learned interactions. In contrast, the explicit fashion designs the function such as inner product to capture the second-order interactions, making the learning process transparent and explainable. Typically, we can divide them into two types: Classic Algorithm and Algorithm with Deep learning.

2.1. Classic algorithm

The models based on logistical regression are the most classic models for CTR prediction, such as FTRL (McMahan et al., 2013) by Google; these models usually rely on experts to generate feature interactions manually by their domain knowledge, which is a high labor cost work. To address this problem, the tree-based models were adopted to learn feature interactions automatically, such as GBDT (He et al., 2014; Ling et al., 2017), which was used in many online services successfully. However, they are only suitable for features appeared in the training process, which limits the capability of the model much.

FM is another classic algorithm used for CTR prediction, which attracted much attention from both the industry and academia. FM is different from matrix factorization (MF) (Koren, Bell, & Volinsky, 2009) used in traditional recommendation system, which is extended as BPR (He et al., 2017; Rendle, Freudenthaler, Gantner, & Schmidt-Thieme, 2009; Rendle & Schmidt-Thieme, 2010), NCF (He et al., 2017); it is a general predictor working with any real value feature vector. It embeds each feature into a vector representation and employs inner-product of pairwise feature representations as to the feature interactions. The feature interactions can be formulated as $v_i^T \mathbf{v}_j$, where $v_i \in R^k$, and k denotes the embedding vector for the ith feature. Because of its generality, FM has been recognized as one of the most effective embedding methods for sparse data prediction. Following the success of FM, several variants of FM has been proposed, such as FFM (Juan et al., 2016), which assumes each field has a different embedding vector for each other fields. However, they only adopt the second order feature interactions, which dramatically limits its presentation capability and performance. Besides, HOFM (Blondel, Fujino, Ueda, & Ishihata, 2016) is devised for high-order factorization machines, but it still only models the linear features.

2.2. Algorithm with deep learning

With the success of Deep Neural networks(DNN) in image and speech recognition (Graves, Mohamed, & Hinton, 2013; He, Zhang, Ren, & Sun, 2016; Krizhevsky, Sutskever, & Hinton, 2012), the DNN is also adopted many other areas, such as social network, e-commerce, medical database (Hui, Yates, Berberich, & de Melo, 2017a; 2017b; Li, He, Sun, & Sun, 2018; Ran, He, Hui, Xu, & Sun, 2018; Yang & He, 2016). A few of DNN-based models for CTR prediction (Liu, Yu, Wu, & Wang, 2015; Zhang et al., 2014) are also proposed, which are utilize DNN to model the high order feature interactions. FNN (Zhang, Du, & Wang, 2016) adopts DNN to learn useful patterns from categorical feature interactions automatically. PNN (Qu et al., 2016) adopts a product layer to capture the feature interactions between inter-field categories and further employs fully connected layers to explore the high-order feature interactions. DCN (Wang, Fu, Fu, & Wang, 2017) introduces a novel cross network based on residual network (He et al., 2016) that is more efficient in learning specific bounded-degree feature interactions. NFM (He & Chua, 2017) devises a bilinear interaction layer to obtain the element-wise product of pairwise feature interactions, then stacks multiple non-linear layers over the bilinear interaction layer to capture the higher-order non-linear feature interactions, which is different from models concatenate or average vectors. Wide& Deep (Cheng et al., 2016) comprises the wide and deep components, where the wide component models the low-dimension linear feature interactions, and the deep component is used to learn the nonlinear high-order feature interactions. DeepFM (Guo et al., 2017) uses FM to replace the wide component in Wide&Deep, where FM is used to mine the low-order feature interactions without any feature engineering, and the DNN part is used to learn the underlying relations between features. The DNN-based models improve the models' expressiveness significantly, but they model the feature interactions in an implicit manner-black box. Recently, xDeepFM (Lian et al., 2018) utilizes both the implicit and explicit high-order feature interactions for prediction. In this model, the compressed interaction network (CIN) component models the high order feature interactions explicitly; the DNN component models the high order feature interactions in an implicit manner. However, it suffers from complex computing and time-consuming.

The attention mechanism is also introduced for CTR prediction, which has been used successfully in computer vision, recommend (Chen et al., 2017; Li et al., 2017; Zhai, Chang, Zhang, & Zhang, 2016; Zhou et al., 2018), NLP (Bahdanau, Cho, and Bengio, 2015; Ren, Chen et al., 2017(@). AFM (Xiao et al., 2017) adopts the attention mechanism to model the importance of all feature interactions; AutoInt (Song et al., 2018) combines the multi-head attention mechanism and the residual network for prediction. The attention mechanism does not only improve the performance of the model but also makes the model be more explainable (Ren, Liang, Li, Wang, & de Rijke, 2017; Wang et al., 2018; Zhao et al., 2016).

3. Methods

In this section, we present our proposed model as showing Fig. 1. HoAFM is composed of three components: 1) embedding layer, which embeds the field-aware raw features as the vector embeddings; 2) cross interaction layers, which update a feature's

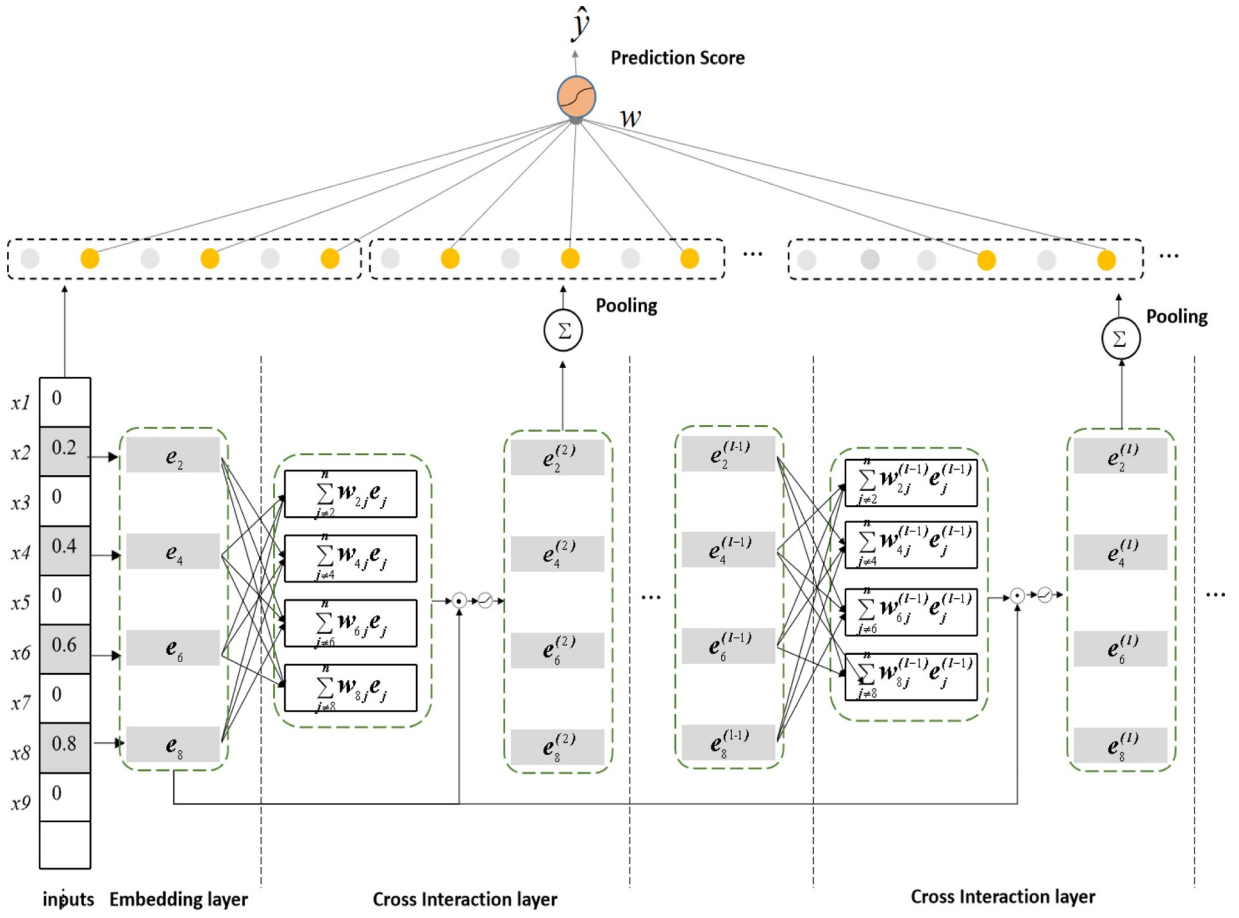


Fig. 1. High-order attention factorization machine.

representation by aggregating its interactions from other co-occurred features, and leverage the bit-wise attention mechanism to learn the sparse attentive weights for each interaction; and 3) prediction layer, which integrates all feature interactions with the linearity of features to perform CTR prediction.

3.1. Embedding layer

The features for CTR prediction are mostly organized in a multi-field form (He & Chua, 2017; Lian et al., 2018; Qu et al., 2016; 2019; Shan et al., 2016). Typically, each field is a categorical variable, such as the user gender, user age, or item attributes. Meanwhile, one input instance usually consists of field-aware values, such as *[user gender = female, user age = 18-24, item attribute = pink]*. As such, an input instance is usually converted into a multi-hot encoding over the fields, as follows:

$$\mathbf{x} = [\underbrace{0, 1, 0, \cdots, 0}_{\text{field } 1}, \underbrace{0, 1}_{\text{field } 2}, \cdots, \underbrace{1, 0, 0, \cdots, 0}_{\text{field } I}], \tag{1}$$

where I is the number of fields. For a feature of one field i, an embedding layer parameterizes it as a vector representation $\mathbf{e}_i \in \mathbb{R}^d$, where d is the embedding size. Following the prior efforts (Lian et al., 2018; Qu et al., 2019), for the field that has multiple features, we sum embeddings of these features up as the fields representation — $\mathbf{e}_i$; for the missing features, we feed all-zero embeddings into the cross interaction layers. As such, we ensure each instance to have the fixed number of fields. Formally, an input instance is represented as:

$$\mathbf{e} = [\mathbf{e}_1, \cdots, \mathbf{e}_I]. \tag{2}$$

3.2. Cross interaction layers

Second-order interaction modeling: Following the bi-linear interaction layer of NFM (He & Chua, 2017), for a pair of field i and j, we model their interactions as the element-wise product of embeddings $\mathbf{e}_i \odot \mathbf{e}_j$. Element-wise product is known as Hadamard product. Distinct from inner product that outputs scalar, it produces another vector as the operands where each element is the

product of corresponding elements of input vectors. Element-wise product is widely used in the research field of data mining, computer vision, and information retrieval, and adopted in NFM and xDeepFM. Distinct from NFM which considers the sum of all interactions between each pair of features as the new representation for the instance, we update the representation of each field by aggregating its interactions with all the other fields. The updated representations of i is formulated as:

$$\mathbf{e}_i^{(2)} = \mu(\mathbf{e}_i \odot \sum_{j \neq i} \mathbf{e}_j), \tag{3}$$

where $\mu(\cdot)$ is a nonlinear activation function set as LeakyReLU (Vaswani et al., 2017) here, capturing the nonlinear patterns inherent in the raw features. As a result, we establish the new representations of an input instance as:

$$\mathbf{e}^{(2)} = [\mathbf{e}_1^{(2)}, \cdots, \mathbf{e}_I^{(2)}]. \tag{4}$$

Bit-wise attention mechanism: Prior efforts (Xiao et al., 2017) have shown that different feature contributes differently to the final prediction. The self-attention mechanism is introduced to assign lower weights to less useful features. While discriminating the importance of each feature, it is applied at the vector-wise level, where all dimensions of feature interactions are associated with the same weights, rather than at the bit-wise level. Hence we argue that the vector-wise attentions result in suboptimal representation ability. Considering this limitation, we design a bit-wise attention mechanism is designed to specify the importance of each interaction at the granularity of feature dimensions:

$$\mathbf{e}_i^{(2)} = \mu(\mathbf{e}_i \odot \sum_{j \neq i} (\mathbf{w}_{ij} \odot \mathbf{e}_j)), \tag{5}$$

where $\mathbf{w}_{ij} \in \mathbb{R}^d$ is the attentive weight vectors for the interaction $\mathbf{e}_i \odot \mathbf{e}_j$, where each entry indicates the dimension importance for the interaction. Here we exploit 3 ways of implementing of $\mathbf{w}_{ij}$:

- **Soft attention:** we employ a multi-layer perceptron (MLP) as the attention network, which takes the interacted vector of two features as input and outputs:

$$\mathbf{a}_{ij} = \mathbf{H}^\top \delta(\mathbf{W}(\mathbf{e}_i \odot \mathbf{e}_j)), \tag{6}$$

  where $\mathbf{W} \in \mathbb{R}^{d' \times d}$ and $\mathbf{H} \in \mathbb{R}^{d \times d'}$ are the transformation matrices; d' is the attention size; $\delta(\cdot)$ is an activation function set as tanh (Vaswani et al., 2017) here. Hereafter, we formulate the output for all fields as a matrix $\mathbf{A}_i \in \mathbb{R}^{d' \times I}$, where the j-th column is the attentive vector of j-th field and k-th row is denoted as $\mathbf{a}_{i*k}$. Hereafter, we normalize the coefficients across all fields by adopting the softmax function:

$$w_{ijk} = \frac{\exp(a_{ijk})}{\sum_{j' \neq i} \exp(a_{ij'k})}. \tag{7}$$

  As such, the attention scores can suggest which feature interaction should be more important and get more attention.
- **Sparse attention:** We adopt sparsemax (Martins & Astudillo, 2016) to introduce sparse probabilities into attentive weights so as to make parts of coefficients involved into feature interactions and avoid overfitting. Such attention function is defined as:

$$\mathbf{w}_{i*k} = \arg \min_{\mathbf{w}_{i*k} \in \Delta^{d'-1}} ||\mathbf{w}_{i*k} - \mathbf{a}_{i*k}||_2^2, \tag{8}$$

  where the original attention scores $\mathbf{a}_{i*k}$ are projected onto a $d' - 1$ simplex $\Delta^{d'-1}$. Compared to softmax used in Equation (7), this has in addition the ability of producing sparse distributions.**High-order interaction modeling:**Having the updated field-aware representations that encode the second-order feature interactions, we can stack more cross interaction layers to model the higher-order interactions. After l steps, the updated representations of feature i is formulated as:

$$\mathbf{e}_i^{(l)} = \mu\left(\mathbf{e}_i^{(l-1)} \odot \sum_{j \neq i} (\mathbf{w}_{ij}^{(l-1)} \odot \mathbf{e}_j^{(l-1)})\right), \tag{9}$$

where $\mathbf{e}_i^{(l-1)}$ is the representation of i encoded $(l-1)$-th order feature interactions, and $\mathbf{e}_i^{(1)}$ is set as $\mathbf{e}_i$. Hereafter, we obtain the representation of the input instance as:

$$\mathbf{e}^{(l)} = [\mathbf{e}_0^{(l-1)}, \cdots, \mathbf{e}_I^{(l-1)}]. \tag{10}$$

Which encodes the high-order interactions into the feature representations.

3.3. Prediction layer

After stacking L cross interaction layers, we have multiple representations for one instance as $\{\mathbf{e}^{(0)}, \cdots, \mathbf{e}^{(L)}\}$. Base on these representations, we conduct a prediction layer to obtain the prediction score as:

$$\hat{y}(\mathbf{x}) = \sigma(w_0 + \sum_{l=1}^{L} \mathbf{h}^{(l)\top} \mathbf{e}^{(l)}), \tag{11}$$

where $\mathbf{h}^{(l)}$ is the trainable vector for the l-order output; and $\sigma(\cdot)$ is the sigmoid function. Note that we conduct only weighted sum of these representations for simplicity, and leave the exploration of the prediction layer (*e.g.,* neural networks) as future work.

Following prior efforts on CTR prediction, we cast the prediction task as a classification problem. We opt for Logloss as the loss function, and hence the objective function to optimize is as follows:

$$\mathcal{L} = \sum_{\mathbf{x} \in O} -y(\mathbf{x}) \ln \hat{y}(\mathbf{x}) - (1 - y(\mathbf{x})) \ln(1 - \hat{y}(\mathbf{x})) + \lambda \|\Theta\|_2^2, \tag{12}$$

where O denotes the set of training instances; and we impose L_2 regularization controlled by hyper-parameter λ on the model parameter set Θ to avoid overfitting.

3.4. Discussion

Space complexity: As mentioned above, HoAFM's parameters are mainly used for attention mechanism. In cross interaction layer, the number of parameters used for the *i-th* vector's interaction is $(I - 1) \times d$, so there are $I \times (I - 1) \times d$ parameters used for the *l-th* cross interaction layer. Considering the parameters used for the prediction layers(d parameters for each order), the total number of parameters used for HoAFM is $\sum_{l=1}^{L} d \times (1 + I \times (I - 1))$. In contrast, xDeepFM contains $\sum_{l=1}^{L} H_l \times (1 + H_{l-1} \times I)$ parameters. Hence, the space complexity of xDeepFM is $O(IH^2L)$, while the space complexity of HoAFM is $O(I^2dL)$. However, H is usually greater than d for xDeepFM's high time complexity, so the space complexity of HoAFM is much better than that of xDeepFM relatively.

Time complexity: In the cross interaction layer, its time complexity is $O(I^2d)$, so the time complexity of HoAFM is $O(I^2dL)$. Relatively, the time complexity of CIN-layers is $O(IH^2d)$, so the time complexity of xDeepFM is $O(IH^2dL)$. Hence the time complexity of xDeepFM is much higher. Moreover, when the value of d increases, the time complexity of xDeepFM will be increased exponentially, which severely limits the capability of xDeepFM. Due to its CIN component that includes the outer product and successive feature map operations, xDeepFM has rather high model complexity. Although it is efficient to do matrix multiplication on GPU, the feature map operation works similarly as CNNs, which could have rather high time cost. In contrast, the time complexity of HoAFM is acceptable.

4. Experiments

In this section, we present our experiments in detail, including the experimental settings, performance comparison, and hyper-parameter study.

4.1. Experiment settings

4.1.1. Datasets

We evaluate our proposed models on two benchmark datasets:

- **Criteo dataset:** This dataset is widely used for CTR prediction and is publicly accessible on Kaggle.[2] There are 39 feature fields in each instance, including 13 numerical fields and 26 categorical fields. Following the prior efforts (Lian et al., 2018), we discretize these numerical features for simplicity.
- **Avazu dataset:** This is also published on Kaggle for developing models to perform CTR prediction.[3] Wherein, all the fields are categorical.

For each dataset, we randomly divide the instances into three parts — training (80%), validation (10%), and test (10%) sets. The validation set is used to tune the hyper-parameters, and we report the final performance on the test set. The characteristics of the two datasets are presented in Table 1.

4.1.2. Evaluation metrics

To evaluate the predictive performance of CTR, we adopt two widely-used metrics, AUC and Logloss (He & Chua, 2017; Lian et al., 2018; Qu et al., 2019; Wang et al., 2018).

- **AUC:** This protocol considers the relative ranking order between positive and negative instances. In particular, it measures the probability that a positive instance is ranked higher than the negative one. It indicates that performance is better when it is higher.
- **Logloss:** This metric measures the probability that one prediction diverges from the ground truth. A lower Logloss suggests a better capacity for fitting the data.

4.1.3. Baselines

We compare our proposed HoAFM with the state-of-the-art methods for CTR prediction: Here, we consider the classic algorithms

[2] http://labs.criteo.com/2014/02/kaggle-display-advertising-challenge-dataset/.
[3] https://www.kaggle.com/c/avazu-ctr-prediction/data.

Table 1
Feature statistic of Criteo and Avazu.

Datasest	#instances	#fields	#features
Criteo	45.0M	39	2.09M
Avazu	40.5M	22	1.54M

(GBDT + LR and FM), and the DNN-based approaches (Wide& Deep, NFM, and xDeepFM).

- **GBDT + LR** (He et al., 2014) is a benchmark model used in industry, which utilizes GBDT to generate cross features automatically, and then feeds them into LR to predict CTR.
- **FM** (Rendle, 2010) is a benchmark factorization model, which considers the second-order interactions between paired features as the inner product of their representations.
- **Wide&Deep** (Cheng et al., 2016) is composed of the wide and deep components. Wherein, the wide component takes manually crafted cross features as inputs, while the deep component uses deep neural networks to capture the nonlinear and higher-order feature interactions.
- **NFM** (He & Chua, 2017) adopts a bi-interaction layer upon the element-wise product of feature representations to capture the higher-order interactions in an implicit fashion.
- **CIN** (Lian et al., 2018) generates feature interactions by the outer product of their representations and then compresses the feature maps derived from the outer product to update the representation of each feature.
- **xDeepFM** (Lian et al., 2018) considers the implicit and explicit modeling of high-order feature interactions simultaneously, which are implemented with the DNN and CIN components.

4.1.4. Parameter settings

For fair comparison, we set the objective function of all the methods as Equation (12). We optimize these methods with the Adagrad optimizer while using the Ftrl optimizer for GBDT + LR, where the batch size is 1024. The embedding size for Criteo dataset is fixed as 64, while that for Avazu dataset is 10. We apply a grid search to determine hyper-parameters: where the learning rate is searched in [0.005, 0.01, 0.02, 0.05], the coefficient of L_2 regularization is tuned in $[10^{-5}, 10^{-4}, \cdots, 10^{-1}]$, and the dropout ratio for FM, NFM, Wide&Deep, xDeepFM, and HoAFM is searched in [0.1, 0.2, ⋯, 0.9]. To model high-order feature interactions, we search the depth of NFM, xDeepFM, and HoAFM in [2,3]. Without specification, the depth of HoAFM is set as three. For other settings, we follow the settings suggested in the original algorithms. The early stopping strategy is performed, where the training is stopped if the Logloss on the validation set increased for two successive epochs.

4.2. Performance comparison

We report the performance comparison of all the methods and discuss how the modeling of high-order feature interactions influences the predictive results. Table 2 presents the performance comparison. We have the following observations:

- CIN performs poorly on the two datasets. One possible reason is that it establishes a rather heavy model complexity and leads to overfitting easily.
- GBDT + LR performs poorly on the two datasets. It suggests that it fails to capture the nonlinear relations between features, and the cross features derived from GBDT are suboptimal to capture feature interactions.
- FM achieves better predictive results as compared to GBDT + LR, this indicate that the second-order feature interactions have a positive effect on prediction. In particular, all pairwise interactions can be modeled via the inner product of feature representations, endowing the model better representation ability.
- Wide&Deep outperforms GBDT + LR and FM on Criteo dataset while yielding poorer performance on Avazu dataset. It is reasonable since the deep component of Wide&Deep is capable of capturing complex relations between features, but easily leads to

Table 2
Performances comparison.

	Criteo			Avazu		
Model	AUC	Logloss	time	AUC	Logloss	time
GBDT + LR	0.7954	0.4548	8320s	0.7608	0.3906	7004s
FM	0.8011	0.4512	933s	0.7793	0.3813	748s
Wide&Deep	0.8017	0.4492	1054s	0.7602	0.3913	627s
NeuralFM	0.8058	0.4459	1044s	0.7789	0.3812	548s
CIN	0.7779	0.4701	10024s	0.7318	0.4066	9287s
xDeepFM	0.8051	0.4466	16039s	0.7473	0.3972	15039s
HoAFM	**0.8090**	**0.4427**	3263s	**0.7828**	**0.3787**	1373s

Table 3
Significance test (*i.e., p*-value) *w.r.t.* AUC and logloss.

	Criteo		Avazu	
Model	AUC	Logloss	AUC	Logloss
GBDT + LR	5e − 9	3e − 10	1e − 07	5e − 9
FM	7e − 9	2e − 3	2e − 4	1e − 4
Wide&Deep	1e − 8	2e − 8	1e − 8	2e − 7
NeuralFM	2e − 7	1e − 6	6e − 5	2e − 5
CIN	7e − 12	5e − 11	1e − 10	1e − 10
xDeepFM	1e − 6	1e − 7	6e − 10	9e − 10
HoAFM	–	–	–	–

overfitting when the data distribution is skewed (*e.g.,* the features of Avazu datasets come mainly from three fields).

- NFM and xDeepFM achieve comparable performance to that of Wide&Deep on Criteo dataset, while NFM is the strongest baseline on Avazu detaset. It indicates that modeling higher-order interactions between features are of importance. Meanwhile, xDeepFM suffers heavily on the high complexity and large-scale parameters, leading to overfitting on the Avazu dataset. Moreover, such observation is consistent with (Lian et al., 2018).
- HoAFM consistently outperforms all baselines on the two datasets. By stacking multiple cross interaction layers, HoAFM is capable of exploiting the high-order interactions in an explicit manner, so as to model cross feature more effectively. Compared with xDeepFM, it alleviates the overfitting issue due to the lightweight parameters and outperforms xDeepFM by a large margin on the Avazu dataset. Moreover, HoAFM is also much more efficient than xDeepFM, with a much lower training time per epoch as shown in Table 2. It verifies the rationality and effectiveness of the cross interaction layer in HoAFM. We further conduct one-sample t-tests to verify that all improvements are statistically significant with p-value < 0.05 as shown in Table 3.

4.3. Study of HoAFM

At the core of HoAFM is the cross interaction layer. Here we investigate its impact on the performance, considering several key factors: the number of cross interaction layers, the design of attention mechanism, and the embedding size.

4.3.1. Effect of the order of feature interactions

To study how HoAFM benefits from the high-order feature interactions, we vary the number of cross interaction layers (*i.e.,* model depth). In particular, the model depth is searched in {1, 2, 3, 4}; HoAFM-L indicates the model with L cross interaction layers. Table 4 presents the experimental results in terms of model depth, from which we derive the following findings:

- Stacking more cross interaction layers substantially improve the accuracy of predictions. HoAFM-2 and HoAFM-3 consistently outperform HoAFM-1, indicating the significance of second- and third-order feature interactions.
- When further stacking such a layer on HoAFM-3, we find that HoAFM-4 leads to overfitting on Criteo dataset; meanwhile, HoAFM-3 has caused the overfitting on Avazu dataset. This might be caused by considering too complicated feature crossing, which might serve as noise. This also verifies that the third-order feature interactions could be sufficient to exhibit the underlying patterns.

4.3.2. Effect of attention mechanisms

To investigate how the attention mechanism affects the prediction results, we consider variants of HoAFM-2 that use different attention networks. In particular, we remove the attention mechanism, termed HoAFM_{no}. Variants that use the soft and sparse attentions as shown in Equations (6) and (8), are termed as $\text{HoAFM}_{\text{soft}}$ and $\text{HoAFM}_{\text{sparse}}$, respectively. Moreover, we employ the L_2 norm on $\text{HoAFM}_{\text{soft}}$ to enforce the attention coefficients in a simplex, termed as $\text{HoAFM}_{\text{norm}}$. We show the experimental results in Table 5 with the following findings:

- Without modeling the importance of feature interactions, HoAFM_{no} performs poorly on two datasets. It verifies the varying

Table 4
Effect of cross interaction layer numbers (L).

	Criteo		Avazu	
Order	AUC	Logloss	AUC	Logloss
HoAFM-1	0.7865	0.4694	0.7419	0.4013
HoAFM-2	0.8083	0.4435	**0.7828**	**0.3787**
HoAFM-3	**0.8090**	**0.4427**	0.7810	0.3798
HoAFM-4	0.8089	0.4428	0.7792	0.3811

Table 5
Effect of different attention mechanisms.

	Criteo		Avazu	
Attention	AUC	Logloss	AUC	Logloss
$HoAFM_{no}$	0.8011	0.4512	0.7793	0.3813
$HoAFM_{soft}$	0.8069	0.4450	0.7813	0.3795
$HoAFM_{norm}$	**0.8083**	0.4435	0.7827	0.3789
$HoAFM_{sparse}$	0.8082	**0.4434**	**0.7828**	**0.3787**

importance of feature interactions which play a pivotal role in capturing these interactions.

- Compared with $HoAFM_{soft}$ that does not control the coefficients, $HoAFM_{norm}$ and $HoAFM_{sparse}$ achieve better improvements on two datasets. We attribute these improvements to the constraints employed on the bit-wise attentions. In particular, the sparse attention scores of $HoAFM_{sparse}$ makes the important interactions contribute more to the final prediction.

4.3.3. Effect of different embedding sizes

As the embedding size is of crucial importance to determine the representation ability of model and the computational cost, we also explore how the embedding size influences the prediction performance. Fig. 2 presents the experimental results *w.r.t.* different embedding sizes on two datasets. As we can see, setting the embedding size as 64 and 10 leads to the best performance on Criteo and Avazu datasets, respectively. When the size exceeds the optimal settings, the prediction results of HoAFM decrease, which might be caused by insufficient representation ability. We use different search spaces for Criteo and Avazu datasets, since Criteo dataset has richer information (e.g., more fields and features) than Avazu dataset. Intuitively, the model trained on Criteo should assign bigger embedding size, so as to have stronger representation ability to fit the data. We hence tune the embedding size in {8, 16, 32, 64} and {5, 10, 15, 20} for Criteo and Avazu datasets, respectively.

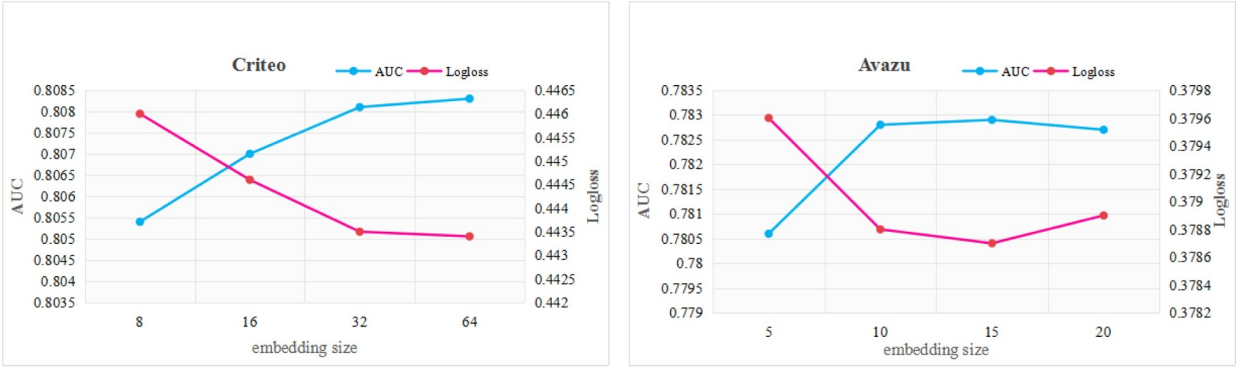
4.4. Case study

As introduced in Section 3, our model adopts the bit-wise attention mechanism, which assigns different weights with the various feature to discriminates the importance of each feature. In Fig. 3, we visualize an attention coefficient matrix of the 21-the filed on Avazu dataset, displaying varying importance of interactions with the other fields. Wherein, each row denotes the ID of co-occurred fields, and each column denotes the dimension of representation. The sum of all coefficients in a column is 1, so as to reflect the importance of interactions.

As we can see, the coefficient between the 21-th and 3-rd fields are assigned with the highest attention score 0.38 at the 9-th dimension, indicating the correlation between these two fields. Furthermore, such sparse coefficients make parts of representations involved in the interaction modeling, so as to help avoid overfitting.

5. Conclusions

In this paper, we proposed a new effective model named HoAFM to encode high-order feature interactions into feature representations explicitly and efficiently. Bit-wise attention mechanism was used to highlight different importance of feature interactions. Extensive experiments on two datasets were conducted to verify the effectiveness of HoAFM for CTR prediction. The results



(a) Criteo (b) Avazu

Fig. 2. Effect of different embedding sizes.

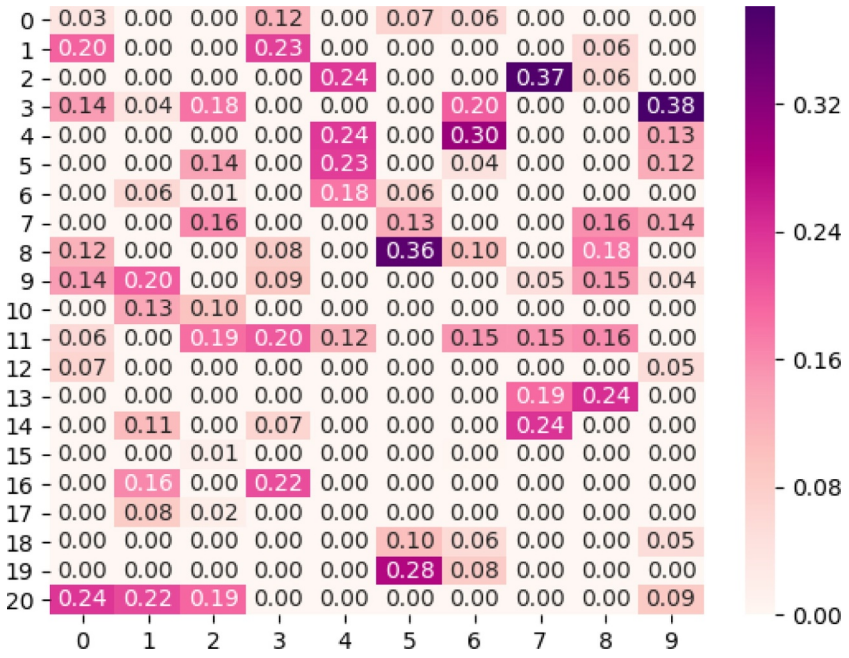


Fig. 3. Visualization of attention scores for a real example from Avazu dataset.

demonstrate that our model could learn high order feature interactions successfully, and it outperforms GBDT + LR, FM and the other state-of-the-art deep models, including Wide&Deep, NFM, and xDeepFM.

There are three directions to explore in the future. First, the scalability of the dimension should be improved in our model. The current structure makes all the dimensions the same with embedding size on all orders, which limits the capability of the model. Secondly, we need to simplify the attention mechanism, since our model becomes overfitting at the 4th order; hence, simplifying the attention mechanism should be useful to improve the performance of the model. Lastly, we would like to incorporate external data (*e.g.,* knowledge graph, multimedia content) into CTR prediction, to enrich features and capture users' intents behind their behaviors.

Acknowledgments

This research is part of NExT + + research and also supported by the National Research Foundation Singapore under its AI Singapore Programme, Linksure Network Holding Pte Ltd and the Asia Big Data Association (Award No.: AISG-100E-2018-002). NExT + + is supported by the National Research Foundation, Prime Minister's Office, Singapore under its IRC@SG Funding Initiative.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.ipm.2019.102076

References

Bahdanau, D., Cho, K., & Bengio, Y. (2015). *Neural machine translation by jointly learning to align and translate. ICLR*.
Blondel, M., Fujino, A., Ueda, N., & Ishihata, M. (2016). *Higher-order factorization machines. NIPS*3351–3359.
Chen, J., Zhang, H., He, X., Nie, L., Liu, W., & Chua, T. (2017). *Attentive collaborative filtering: Multimedia recommendation with item- and component-level attention. SIGIR*335–344.
Cheng, H., & Cantú-Paz, E. (2010). *Personalized click prediction in sponsored search. WSDM*351–360.
Cheng, H., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., et al. (2016). *Wide & deep learning for recommender systems. DLRS*7–10.
Graves, A., Mohamed, A., & Hinton, G. E. (2013). *Speech recognition with deep recurrent neural networks. ICASSP*6645–6649.
Guo, H., Tang, R., Ye, Y., Li, Z., & He, X. (2017). *Deepfm: A factorization-machine based neural network for CTR prediction. IJCIA*1725–1731.
He, K., Zhang, X., Ren, S., & Sun, J. (2016). *Deep residual learning for image recognition. CVPR*770–778.
He, X., & Chua, T. (2017). *Neural factorization machines for sparse predictive analytics. SIGIR*355–364.
He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. (2017). *Neural collaborative filtering. WWW*173–182.
He, X., Pan, J., Jin, O., Xu, T., Liu, B., Xu, T., et al. (2014). *Practical lessons from predicting clicks on ads at facebook. ADKDD*5:1–5:9.
Hui, K., Yates, A., Berberich, K., & de Melo, G. (Yates, Berberich, de Melo, 2017a). *PACRR: A position-aware neural IR model for relevance matching. EMNLP*1049–1058.
Hui, K., Yates, A., Berberich, K., & de Melo, G. (Yates, Berberich, de Melo, 2017b). *Position-aware representations for relevance matching in neural information retrieval. WWW*799–800.
Juan, Y., Zhuang, Y., Chin, W., & Lin, C. (2016). *Field-aware factorization machines for CTR prediction. RECSYS*43–50.
Koren, Y., Bell, R. M., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *IEEE Computer,* 30–37.
Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). *Imagenet classification with deep convolutional neural networks. NIPS*1106–1114.
Kutlwano, P. F., & Ramaboa, K. K. M. (2018). Keyword length and matching options as indicators of search intent in sponsored search. *IPM,* 175–183.

Li, C., He, B., Sun, L., & Sun, Y. (2018). *Neural precision medicine by mining implicit treatment concepts. BIBM*893–898.
Li, J., Ren, P., Chen, Z., Ren, Z., Lian, T., & Ma, J. (2017). *Neural attentive session-based recommendation. CIKM*1419–1428.
Lian, J., Zhou, X., Zhang, F., Chen, Z., Xie, X., & Sun, G. (2018). *xdeepfm: Combining explicit and implicit feature interactions for recommender systems. KDD*1754–1763.
Ling, X., Deng, W., Gu, C., Zhou, H., Li, C., & Sun, F. (2017). *Model ensemble for click prediction in bing search ads. WWW*689–698.
Liu, Q., Yu, F., Wu, S., & Wang, L. (2015). *A convolutional click prediction model. Cikm*1743–1746.
Martins, A. F. T., & Astudillo, R. F. (2016). *From softmax to sparsemax: A sparse model of attention and multi-label classification. ICML*1614–1623.
McMahan, H. B., Holt, G., Sculley, D., Young, M., Ebner, D., Grady, J., et al. (2013). *Ad click prediction: A view from the trenches. KDD*1222–1230.
Qu, Y., Cai, H., Ren, K., Zhang, W., Yu, Y., Wen, Y., et al. (2016). *Product-based neural networks for user response prediction. ICDM*1149–1154.
Qu, Y., Fang, B., Zhang, W., Tang, R., Niu, M., Guo, H., et al. (2019). Product-based neural networks for user response prediction over multi-field categorical data. *TOIS,* 5:1–5:35.
Ran, Y., He, B., Hui, K., Xu, J., & Sun, L. (2018). Neural relevance model using similarities with elite documents for effective clinical decision support. *IJDMB,* 91–108.
Ren, P., Chen, Z., Ren, Z., Wei, F., Ma, J., & de Rijke, M. (2017). *Leveraging contextual sentence relations for extractive summarization using a neural attention model. SIGIR*95–104.
Ren, Z., Liang, S., Li, P., Wang, S., & de Rijke, M. (2017). *Social collaborative viewpoint regression with explainable recommendations. WSDM*485–494.
Rendle, S. (2010). *Factorization machines. ICDM*995–1000.
Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2009). *BPR: Bayesian personalized ranking from implicit feedback. UAI*452–461.
Rendle, S., & Schmidt-Thieme, L. (2010). *Pairwise interaction tensor factorization for personalized tag recommendation. WSDM*81–90.
Richardson, M., Dominowska, E., & Ragno, R. (2007). *Predicting clicks: Estimating the click-through rate for new ads. WWW*521–530.
Shan, Y., Hoens, T. R., Jiao, J., Wang, H., Yu, D., & Mao, J. C. (2016). *Deep crossing: Web-scale modeling without manually crafted combinatorial features. KDD*255–262.
Song, W., Shi, C., Xiao, Z., Duan, Z., Xu, Y., Zhang, M., et al. (2018). Autoint: Automatic feature interaction learning via self-attentive neural networks. *CoRR*.
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). *Attention is all you need. NIPS*6000–6010.
Wang, R., Fu, B., Fu, G., & Wang, M. (2017). *Deep & cross network for ad click predictions. ADKDD*12:1–12:7.
Wang, X., He, X., Feng, F., Nie, L., & Chua, T. (2018). *TEM: Tree-enhanced embedding model for explainable recommendation. WWW*1543–1552.
Xiao, J., Ye, H., He, X., Zhang, H., Wu, F., & Chua, T. (2017). *Attentional factorization machines: Learning the weight of feature interactions via attention networks. IJCAI*3119–3125.
Xiao, B., & Baia, B. (2019). Impact of response latency on sponsored search. *IPM,* 110–129.
Yang, C., & He, B. (2016). *A novel semantics-based approach to medical literature search. BIBM*1616–1623.
Zhai, S., Chang, K., Zhang, R., & Zhang, Z. M. (2016). *Deepintent: Learning attentions for online advertising with recurrent neural networks. KDD*1295–1304.
Zhang, W., Du, T., & Wang, J. (2016). *Deep learning over multi-field categorical data – A case study on user response prediction. ECIR*45–57.
Zhang, Y., Dai, H., Xu, C., Feng, J., Wang, T., Bian, J., et al. (2014). *Sequential click prediction for sponsored search with recurrent neural networks. AAAI*1369–1375.
Zhao, Y., Liang, S., Ren, Z., Ma, J., Yilmaz, E., & de Rijke, M. (2016). *Explainable user clustering in short text streams. SIGIR*155–164.
Zhou, G., Zhu, X., Song, C., Fan, Y., Zhu, H., Ma, X., et al. (2018). *Deep interest network for click-through rate prediction. KDD*1059–1068.