IO-aware Factorization Machine for User Response Prediction

Zhenhao Hu
Software Engineering Institute
East China Normal University
Shanghai, China
hzh@stu.ecnu.edu.cn

Chao Peng
Software Engineering Institute
East China Normal University
Shanghai, China
cpeng@sei.ecnu.edu.cn

Cheng He
Software Engineering Institute
East China Normal University
Shanghai, China
389096133@qq.com

Haibin Cai
Software Engineering Institute
East China Normal University
Shanghai, China
hbcai@sei.ecnu.edu.cn

***Abstract*—As a supervised learning method, Factorization Machine (FM) is famous for its capability of modeling feature interactions. However, FM's performance might be bad if we assign the same weight to all feature interactions, as not all of them are equally useful and productive. Attentional Factorization Machine (AFM) improves FM by discriminating the importance of distinctive feature interactions via a neural attention network. Nevertheless, the neural attention network in AFM is not fine-grained enough and it ignores the information of the fields implied by the features, which limits the performance of the model. In this work, we propose a novel model named IO-aware Factorization Machine (IOFM), which enhances the feature representation ability of attention mechanism in estimating weights via two awareness auxiliary matrices. To make the model more efficient, we further reduce the model parameters using canonical decomposition for the two auxiliary matrices and design a shared matrix to correlate the decomposed matrices. Extensive experiments on two real-world datasets indicate the superiority of our IOFM model over the state-of-the-art methods.**

***Index Terms*—Factorization Machines, Neural Attention Network, Recommender Systems, User Response Prediction**

I. Introduction

Learning and predicting user response makes a crucial contribution to many personalization tasks such as recommender systems, web search, and online advertising. The goal of learning and predicting user response is to estimate a function that maps predictor variables to the corresponding targets. Typically, the targets include real-valued targets for regression and categorical labels for classification. The predictor variables in these tasks are primarily discrete and categorical. A common solution for many machine learning models with these types of datasets is converting them to high-dimensional sparse binary features via one-hot encoding, subsequently embedding them to low-dimensional features. When estimating the function of such discrete and categorical predictor variables, it is essential to model the interactions between features.

As an example, predicting the probability that a user in a particular occupation clicks on an ad is crucial in online advertising. We can cross variable occupation of user = {*driver*, *doctor*} with ad = {*coffee*, *milk*} and get a new occupation_ad = {*driver_coffee*, *driver_milk*, *doctor_coffee*, *doctor_milk*}, in accordance with common sense, {*driver_coffee*} is more likely to be a positive sample. However, manually designing efficient and useful feature interactions is a crucial but laborious task. Therefore, these solutions may be difficult to be generalized to new issues or domains.

To address the difficulty of manual feature engineering, many machine learning models are intended to learn features from raw data automatically. Factorization machines (FMs[1]) were proposed in [1], which can embed feature into a latent space and estimate the weight for a cross feature between these embedding features. Invisible feature interactions can be learned due to the brilliant generalization ability of FM, which has been successfully applied to various applications [2]–[5].

Although effective, FM estimates the weight of each feature interaction equally and may degrade the performance. For this reason, Attention Factorization Machine (AFM) [6] was proposed, which can discriminate the importance of different feature interactions via attention mechanism. Since the attention mechanism has been introduced to deep learning, it has achieved remarkable results, and it has been widely used in various types of deep learning tasks such as machine translation [7], [8], image caption [9] and speech recognition [10]. However, the neural attention network in AFM directly calculates a scalar weight for each feature interaction, which may lose useful information and compromise the powerful representation capability of the attention mechanism. Besides, AFM is directly improved from FM, which ignores the information of the fields implied by the features. Inspired by FFM [11] that the effect of a feature can differ when interacting with features from different fields, the feature interactions also imply information about the interactions between fields.

To solve the above issues, we propose a novel model called IO-aware Factorization Machines (IOFM[2]) to enhance the feature representation of attention mechanism in estimating weight of feature interaction via two awareness auxiliary matrices. The role of the two awareness auxiliary matrices

This research is partially supported by Ministry of Science and Technology of China (NO. 2018AAA0100902), the National Natural Science Foundation of China (NO. 61672240), and the 2030 National Key AI Program of China (NO. 2018AAA0100503).

[1]In this paper, we focus on the second-order FM, which is the most effective and widely used instance of FMs.

[2]Implementation of IOFM is available at: https://github.com/ihaozz/IO-aware-Factorization-Machines

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

is reflected in the input phase and the output phase of the attention network, capturing wealthy interactive information in two steps. Specifically, we use an input-aware auxiliary matrix (as field interactions) to add field information into the pair-wise feature interactions, and we use an output-aware auxiliary matrix to enhance the expression capability of attention networks, thus the attention distribution for each feature interaction will be a vector, instead of a single value. In other words, using a vector instead of a scalar for each feature interaction is to describe the attention distribution from a finer granularity. Additionally, we further reduce the model parameters using canonical decomposition to make the model more efficient, and design a shared matrix correlating the decomposed matrices to improve the performance of the model.

The main contributions of the paper include the following:

- By using two awareness auxiliary matrices we present a novel neural attention network view for AFM, the performance of the model is promoted significantly and no feature engineering is required.
- Using factorization for the two auxiliary matrices, the performance of IOFM is enhanced with fewer parameters.
- By using a shared matrix correlates the two decomposed matrices, performance of IOFM is further improved.
- Experimental results on two well-known datasets demonstrate that the proposed model IOFM is superior to the state-of-the-art methods.

II. Related Work

In learning and predicting user response, it is important to capture feature interactions. Recently, many deep learning-based methods have been studied to solve such problems [12], [13]. Many traditional solutions need an army of artificial feature engineering to generate the feature interactions, such as Logistic Regression with FTRL optimizer [14]. Due to the enormous space of combinatorial features, feature engineering becomes a tough work. In order to solve this problem, Factorization Machines [1] have been proposed to learn the feature interactions automatically. FM is a matrix-based machine learning algorithm designed by Steffen Rendle for collaborative recommendation [15], it has good learning ability by embedding high-dimensional space features from raw data into a low-dimensional latent space. Given a real instance with a feature vector $\mathbf{x} \in \mathbb{R}^n$ and a target y, where n denotes the number of features and x_i denotes the i-th feature in feature vector $\mathbf{x}$. FM estimates the target by capturing all interactions between each feature using factorized interaction parameters:

$$\hat{y}_{FM}(\mathbf{x}) = \underbrace{w_0 + \sum_{i=1}^{n} w_i x_i}_{\text{linear regression}} + \underbrace{\sum_{i=1}^{n}\sum_{j=i+1}^{n} \hat{w}_{ij} x_i x_j}_{\text{pair-wise feature interactions}} \tag{1}$$

Where w_0 is the global bias, w_i is the weights of the i-th feature in feature vector, and $\hat{w}_{ij}$ denotes the weights of the pairwise feature interactions $x_i x_j$, which is factorized as $\hat{w}_{ij} = \mathbf{v}_i^T \mathbf{v}_j$, where $\mathbf{v}_i \in \mathbb{R}^k$ denotes the embedding vector of x_i, and k is the size of embedding vector.

FMs use the dot product of two embedding vectors to model the effect of feature interactions, and FFM [11] extended the ideas by additionally leveraging the field information as auxiliary information to improve model performance. In FFM, each feature has separate latent vectors to interact with features from different fields, therefore, a feature may have different effects when interacting with features from diverse fields. Mathematically, pair-wise feature interactions of FFM is:

$$\sum_{i=1}^{n}\sum_{j=i+1}^{n} (w_{i,f_j} w_{j,f_i}) x_i x_j \tag{2}$$

where f_i and f_j are respectively the fields of i and j.

With the great success of deep learning in various research fields, many deep learning-based FM models have also been proposed in recent years. FNN [16] uses a Multilayer Perceptron (MLP) to learn high-order feature dependencies on the hidden vectors of FM, thus avoiding the training of the embedding layer from a random state. The key to PNN [17] is to introduce a product layer between the embedding layer and the first hidden layer. PNN defines the operations of various products such as inner product and outer product to capture different interactive information and enhance the ability of the model to represent different data patterns. The structure of DeepFM [18] is the dual network combination. The improvement is that DeepFM replaces the wide part of the Wide&Deep [19] with FM, enhancing the ability to combine shallow network features. NFM [20] can be seen as an improvement to the deep part of the Wide&Deep, which combines the linearity of FM in modelling second-order feature interactions and the non-linearity of neural network in modelling higher-order feature interactions. FM can be hindered by its modelling of all feature interactions with the same weight, as not all feature interactions are equally useful and predictive. AFM [6] introduces the attention mechanism into FM, utilizes a neural attention network to learn the importance of each feature interaction. IFM [21], based on AFM and FFM models, comprises the feature aspect and the field aspect to learn flexible interactions on two levels.

III. Proposed Approach

A. Architecture of IOFM

Figure 1 illustrates the neural network architecture of our proposed IO-aware Factorization Machines model, we omit linear terms and the global bias in the remaining parts for simplicity. In what follows, we elaborate the details of several layers in the architecture.

1) Embedding layer: The embedding layer projects each feature to a dense latent vector representation, whose essence is a fully connected layer. $\mathbf{v}_i \in \mathbb{R}^k$ denotes the embedding vector for the i-th feature, where k is the size of embedding vector. Since the input layer is composed of sparse features, let the set of non-zero features in the feature vector $\mathbf{x}$ be χ, the size of χ be m, and the set of embedding vectors be $\{\mathbf{v}_i x_i\}_{i \in \chi}$.

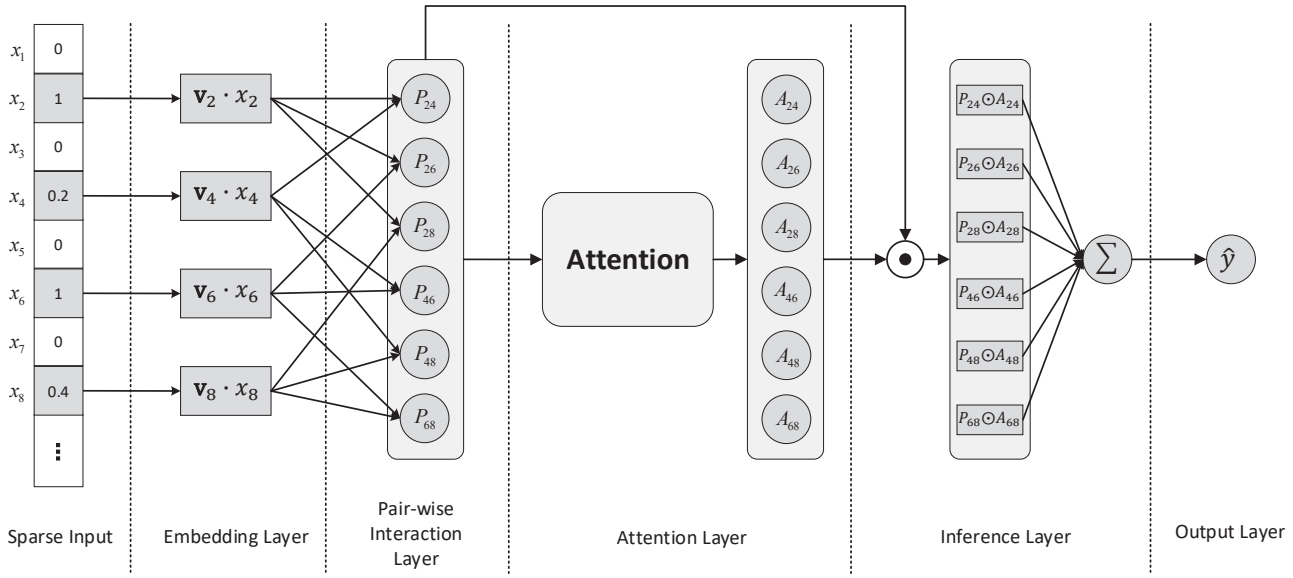


Fig. 1. Architecture of proposed IO-aware Factorization Machines model.

2) Pair-wise interaction layer: The pair-wise interaction layer enumerates interact of every two latent vectors in the embedding layer, whose output consists of the interacted vector of two features, which can be represented as:

$$\mathbf{P}_{ij} = (\mathbf{v}_i \odot \mathbf{v}_j) x_i x_j, \ (i,j) \in \mathcal{R}_\chi \tag{3}$$

Where $\odot$ denotes the Hadamard product, i.e. the element-wise product of two vectors, and $\mathcal{R}_\chi = \{(i,j)\}_{i \in \chi, j \in \chi, j>i}$, since there are m non-zero features in χ, the size of the set $\mathcal{R}_\chi$ is l, where $l = m(m-1)/2$.

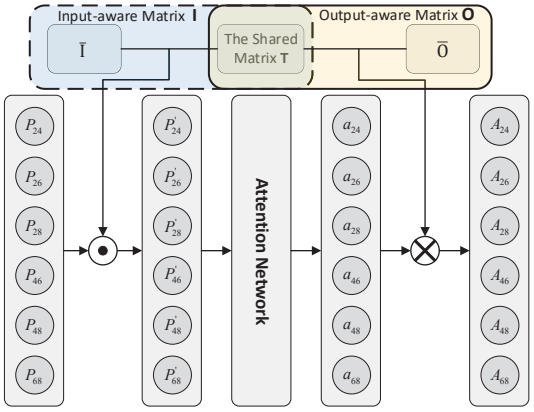


Fig. 2. Architecture of the Attention layer in IOFM.

3) Attention layer: As shown in Figure 2, by adding auxiliary information to the attention mechanism, we design a new architecture to enhance the expression ability of attention mechanisms in two steps. $\mathbf{P}_{ij}$ is the output of pair-wise interaction layer, as the input of attention layer. The output of the attention layer is $\mathbf{A}_{ij}$, which is an attention vector for feature interaction $\hat{w}_{ij}$, can be interpreted as the importance of $\hat{w}_{ij}$ in predicting the target. $\mathbf{I} \in \mathbb{R}^{k \times l}$ and $\mathbf{O} \in \mathbb{R}^{k \times l}$ are two auxiliary matrices. Specifically, the input-aware auxiliary matrix $\mathbf{I}$ plays the role of field interactions information, leveraging the information contained in the feature interactions, and the output-aware auxiliary matrix $\mathbf{O}$ is used to enhance the expression ability of attention networks. Unlike AFM, the attention distribution for each feature interaction is a vector instead of a single value. Formally, the attention network is defined as:

$$\mathbf{P}'_{ij} = \mathbf{P}_{ij} \odot \mathbf{I}_{f_{ij}} \tag{4}$$

$$a'_{ij} = \mathbf{h}^T ReLU(\mathbf{W}\mathbf{P}'_{ij} + \mathbf{b}) \tag{5}$$

$$a_{ij} = \frac{exp(a'_{ij}/\tau)}{\sum_{(i,j)\in\mathcal{R}_\chi} exp(a'_{ij}/\tau)} \tag{6}$$

$$\mathbf{A}_{ij} = a_{ij} \mathbf{O}_{f_{ij}} \tag{7}$$

Where $\odot$ denotes the Hadamard product, $\otimes$ denotes the scalar multiplied by the corresponding vector, i.e. $\mathbf{a} \otimes \mathbf{O} = (a_1, a_2, a_3) \otimes (\mathbf{O}_1, \mathbf{O}_2, \mathbf{O}_3) = (a_1\mathbf{O}_1, a_2\mathbf{O}_2, a_3\mathbf{O}_3)$. $\mathbf{W} \in \mathbb{R}^{t \times k}$, $\mathbf{b} \in \mathbb{R}^t$, $\mathbf{h} \in \mathbb{R}^t$ are parameters of the attention network, t is attention factor, which denotes the hidden layer size of the attention network. τ is a hyperparameter used to control the effective strength of the feature interaction in softmax function [22]. $\mathbf{I}_{f_{ij}}$ is the f_{ij}-th column vector of $\mathbf{I}$, and $\mathbf{O}_{f_{ij}}$ is the f_{ij}-th column vector of $\mathbf{O}$, where f_{ij} denotes the pair-wise interaction of i and j. In the middle part, we refer to the attention network used in AFM, which is a multilayer perceptron. The core standpoint is to increase the generalization ability through MLP.

In order to make the model more efficient, we present a more generalized architecture. The shape of the auxiliary matrix both are $k \times l$, excessive parameter quantities bring difficulties to the model training. We decompose the two auxiliary matrices using canonical decomposition [23] to reduce the model parameters. In addition, considering the strong

correlation between the two auxiliary matrices and the feature interactions, we design a shared matrix in the factorization to improve the correlation of the decomposed matrices:

$$\mathbf{I} = \bar{\mathbf{I}}^T\mathbf{T}, \tag{8}$$
$$\mathbf{O} = \bar{\mathbf{O}}^T\mathbf{T} \tag{9}$$

where $\bar{\mathbf{I}} \in \mathbb{R}^{d\times k}$, $\bar{\mathbf{O}} \in \mathbb{R}^{d\times k}$, and $\mathbf{T} \in \mathbb{R}^{d\times l}$ is a shared factor matrix, with column vector $\mathbf{T}_{f_{ij}}$ representing the f_{ij}-th shared vector of auxiliary matrices, d is the number of latent factorization factors. The shared matrix $\mathbf{T}$ correlates the decomposed matrices and further reduces the model parameters. Correspondingly, the calculation method of attention becomes:

$$a'_{ij} = \mathbf{h}^T ReLU(\mathbf{W}(\mathbf{P}_{ij} \odot (\bar{\mathbf{I}}^T\mathbf{T}_{f_{ij}})) + \mathbf{b}) \tag{10}$$
$$a_{ij} = \frac{\exp(a'_{ij}/\tau)}{\sum_{(i,j)\in\mathcal{R}_\chi}\exp(a'_{ij}/\tau)} \tag{11}$$
$$\mathbf{A}_{ij} = a_{ij}\bar{\mathbf{O}}^T\mathbf{T}_{f_{ij}} \tag{12}$$

4) Inference layer: The inference layer multiplies the attention vector obtained in the attention layer with pair-wise feature interactions, used to calculate the final output:

$$\hat{y} = \boldsymbol{\alpha}^T\sum_{i=1}^{n}\sum_{j=i+1}^{n}\mathbf{A}_{ij}\odot\mathbf{P}_{ij} \tag{13}$$

Where $\boldsymbol{\alpha} \in \mathbb{R}^k$ is $\mathbf{1}$, $\odot$ denotes the Hadamard product, which means that the attention mechanism is used internally from vectors that represent intersecting features, rather than using scalars as attention weights to globally scale feature vectors as in AFM.

To summary, the overall formulation of IOFM model is:

$$\hat{y}_{IOFM}(\mathbf{x}) = w_0 + \sum_{i=1}^{n} w_i x_i + \boldsymbol{\alpha}^T\sum_{i=1}^{n}\sum_{j=i+1}^{n}\mathbf{A}_{ij}\odot(\mathbf{v}_i\odot\mathbf{v}_j)x_i x_j \tag{14}$$

where $\mathbf{A}_{ij}$ is defined in Equation (12).

B. Space Complexity Analysis

IOFM has two additional auxiliary matrices compared to AFM. After using canonical decomposition with a shared matrix, the awareness matrices require $l\times d + 2\times d\times k$ parameters totally. Besides, the parameters of embedding vectors are $n\times k$, and the parameters of attention network is $k\times t + 2t$. Thus, the overall space complexity is $O(ld + (2d + n + t)k + 2t)$. Owing to sparse representation of $\mathbf{x}$, the values of l, d, k, and t are ussually much smaller compared to n, so the space complexity is similar to that of AFM, which is $O(nk)$.

C. Relation to AFM and IFM

The key difference between IOFM and AFM is the input and output of the attention network, the two auxiliary matrices are trained to improve the performance of the IOFM from two phases. Specifcally, AFM can be seen as a special case of IOFM when the elements in the two auxiliary matrices are all 1.

Compared to IFM, which is also inspired by FFM, IOFM adds the field interactions information into feature interactions directly, while in IFM the field informations come from every feature. Direct use of field interactions information not only reduces the computational complexity of the model but also reduces the dependence of the field interactions information on the embedding features, making the control to field informations more flexible. Besides, IFM treats the feature aspect and the field aspect as two independent parts with no information overlap, which makes the attention mechanism unable to capture the attentional distribution of field interactions. In IOFM, the field interactions information is added to the feature interactions as the input to the attention network. More importantly, IOFM uses an output-aware matrix to describes the attention distribution at a more fine-grained level.

D. Learning

As IOFM directly enhances AFM from the perspective of data modelling, it can also be applied to a variety of prediction tasks, including regression, classification and ranking. Similar to AFM, in this paper we also give priority to the regression task and optimize the squared loss. For a regression task, the target $y(\mathbf{x})$ is a real value, a common objective function is squared loss:

$$L = \sum_{\mathbf{x}\in\varphi}(\hat{y}_{IOFM}(\mathbf{x}) - y(\mathbf{x}))^2 \tag{15}$$

where φ denotes the set of training instances. We also employ dropout and L_2 regularization to prevent overfitting. Since there are two auxiliary matrices $\mathbf{I}$ and $\mathbf{O}$ in the attention network except the weight matrix $\mathbf{W}$, we additionally apply a regularization for them. That is, the actual objective function we optimize is:

$$L = \sum_{\mathbf{x}\in\varphi}(\hat{y}_{IOFM}(\mathbf{x}) - y(\mathbf{x}))^2 + \lambda_1\|\mathbf{W}\|^2 + \lambda_2(\|\mathbf{I}\|^2 + \|\mathbf{O}\|^2) \tag{16}$$

where λ_1 controls the regularization strength for the weight matrix $\mathbf{W}$, λ_2 controls the regularization strength for two auxiliary matrices $\mathbf{I}$ and $\mathbf{O}$, respectively.

IV. Experiments

In this section, we evaluate the performance of the proposed IOFM model on two real-world datasets and conduct extensive experiments to answer the following research questions:

QR1 How does IOFM perform compared to the state-of-art-methods?

QR2 How do the key hyperparameters of IOFM impact its performance?

QR3 How do the two auxiliary matrices of IOFM impact the prediction accuracy?

QR4 How do the factorization of auxiliary matrix and the shared matrix impact the performance of IOFM?

A. Experiment Setting

1) Datasets and Evaluation Metric: We evaluate our models on two real-world datasets, Frappe [24] and MovieLens [25], for context-aware recommendation and personalized tag recommendation, respectively. Both datasets contain categorical variables and positive records merely, we use the datasets preprocessed in the previous work [20] directly, generate negative samples by randomly pairing two negative samples with each log and converting each log into a feature vector via one-hot encoding. The datasets are divided into a training set (70%), a validation set (20%), and a test set (10%) like the experimental settings in the previous work. Table I shows the description of the processed datasets.

TABLE I
DATASET DESCRIPTION.

Dataset	instance	Feature	User	Item	Field
Frappe	288,609	5,382	957	4,082	10
MovieLens	2,006,859	90,445	17,045	23,743	3

We use the best parameter settings reported by the authors [6], [20] for fair comparison. All models are trained on the training set, and the optimal parameters are obtained on the validation set. Performance is evaluated by Root Mean Square Error (RMSE) on the test set with the best parameters, with lower scores indicating better performance. Note that RMSE has been widely used to evaluate regression tasks, such as recommendation with explicit ratings [15], [26] and click-through rate prediction [27].

2) Baselines: We compare our model with the following competitive embedding-based methods:

- FM [1]. As described in Equation (1). FM is recognized as the most effective linear embedding method for sparse data prediction, which can subsume many specific factorization models [28], such as Matrix Factorization (MF) [29], parallel factor analysis, and SVD++ [30]. It has shown strong performance for personalized tag recommendation and the context-aware prediction [15].
- DeepFM [18]. DeepFM uses FM to enhance the ability of shallow network features, and shares the feature embedding between the FM and the deep component. It combines the output of FM and DNN to obtain the final probability prediction, which can learn both the lower-order and higher-order feature interactions.
- NFM[3] [20]. NFM seamlessly combines the linearity of FM in modelling second-order feature interactions and performs a non-linear transformation called Bi-interaction on the latent space of the second-order feature interactions.
- AFM[4] [6]. AFM introduces the attention mechanism to recommender systems, which learns one coefficient for every feature interaction via a neural attention network to enable feature interactions that contribute differently to the prediction.
- IFM[5] [21]. IFM combines AFM and FFM with the introduction of the feature aspect and the field aspect control, learning feature interaction importance in a stratified manner.

For the models that give the official implementation, we use the code and optimal parameters they provide directly. We also try to tune the relevant hyperparameters for better performance.

3) Hyperparameters: The model-independent hyperparameters are set to the optimal values reported by the previous work [6], [21] for fairness. Specifically, the embedding size of features is set to 256 for all models, τ in attention net is set to 10 for IFM and IOFM. The attention factor is set to 256 and 8, and the batch size is set to 128 and 4096 for Frappe and MovieLens dataset, respectively. An early stopping criterion is used to prevent overfitting in the process of training models. We also pretrain the feature embeddings with FM to get better results and tune the other hyperparameters on the validation set.

B. Performance Comparison (QR1)

TABLE II
TEST ERROR AND NUMBER OF PARAMETERS OF DIFFERENT METHODS ON EMBEDDING SIZE 256. M DENOTES "MILLION".

	Frappe		MovieLens	
Method	**Param#**	**RMSE**	**Param#**	**RMSE**
FM	1.38M	0.3340	23.24M	0.4666
DeepFM	1.64M	0.3311	23.32M	0.4642
NFM	1.45M	0.3125	23.31M	0.4536
AFM	1.45M	0.3117	23.25M	0.4314
IFM	1.46M	0.3108	23.25M	0.4210
IOFM	1.46M	**0.3075**	23.25M	**0.4133**

The model-independent hyperparameters in all models are set to the same as mentioned in the previous section. We carefully adjusted other hyperparameters to get the best performance of each model on the Frappe dataset and the MovieLens dataset, respectively. The best performances as shown in Table II. IOFM achieves the best performance among all methods, which demonstrates the effectiveness of IOFM. Specifically, IOFM betters FM with a 7.9% and 11.4% relative improvement by using less than 0.1M additional parameters on the Frappe and MovieLens dataset, respectively. In the case of using almost the same number of parameters, IOFM improves by 1.3% and 4.2% compared to AFM and outperforms the second best method IFM with 1.1% and 1.8% on the Frappe and MovieLens dataset, respectively.

Both NFM and DeepFM contain captures of higher-order feature interactions, and their performances are better than FM, which proves that modelling higher-order feature interactions can improve performance. However, their performances are

[3]https://github.com/hexiangnan/neural_factorization_machine

[4]https://github.com/hexiangnan/attentional_factorization_machine

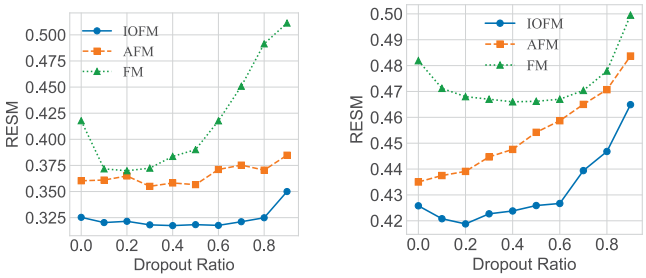
[5]https://github.com/cstur4/interaction-aware-factorization-machines

weaker than the rest of the model, which shows that the improvement of the second-order feature interaction is still worth studying.

Although the architectures of IOFM and IFM are different, it can be seen from the comparison with AFM that modelling the fields of features is effective.

C. Hyperparameter Investigation (QR2)

In this subsection, we mainly explore the effect of dropout on the pair-wise interaction layer, L_2 regularization, and the number of latent factorization factors d.



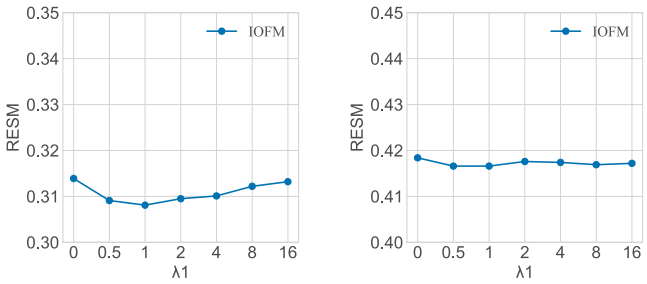
(a) Frappe. (b) MovieLens.

Fig. 3. Validation error of IOFM, AFM, and FM *w.r.t.* different dropout ratios on the pair-wise interaction layer.

1) Dropout: Dropout can effectively alleviate the occurrence of overfitting and achieve regularization to a certain extent [31]. For exploring the effect of dropout on the pair-wise interaction layer, no L_2 regularization is used in models. We apply dropout to FM and AFM on feature interaction vectors and obtain better performances as a benchmark. Note that we have tried the learning rate provided in official AFM implementation, but the performance is weaker, so we set the learning rate to 0.05 for the three models. In this setting, the AFM model achieves a good performance. We set the dropout ratio from 0 to 0.9 with increments of 0.1, as shown in Figure 3, by setting the dropout ratio to a proper value, both IOFM and FM can be improved, but when the dropout ratio tends to 1, all models suffer from underfitting issue and have poor performances.

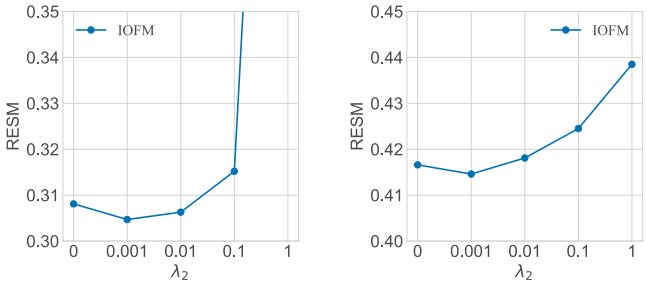
From the Figure 3, we can see that the performances of IOFM and FAM fluctuated less with the change in the dropout rate on the Frappe dataset, while it fluctuated significantly on the MovieLens dataset. We believe that the effect is caused by the dataset itself. Because each sample of the MovieLens dataset has fewer non-zero features than the Frappe dataset, it is more sensitive to changes in dropout rate. For nearly all dropout rates, IOFM performs the best.

2) L_2 regularization: We set up two sets of experiments to verify the effects of λ_1(which is control the regularization strength for the weight matrix $\mathbf{W}$) and λ_2(which is control the regularization strength for two auxiliary matrices $\mathbf{I}$ and $\mathbf{O}$), respectively. Figure 4 shows how IOFM performs when the L_2 regularization hyperparameter λ_1 varies while keeping λ_2 to 0 and the dropout ratio to the optimal value from the validation set. Correspondingly, Figure 5 shows how IOFM performs only when the L_2 regularization hyperparameter λ_2 varies while keeping λ_1 and the dropout ratio to the optimal value from the validation set. As can be seen from experimental results, the performance of IOFM will improve when λ_1 is set to a value larger than 0, but the impact on performance is not significant. Tuning λ_2 to an applicable value can further improve the generalization of IOFM, but a too large value will result in worse performance.



(a) Frappe. (b) MovieLens.

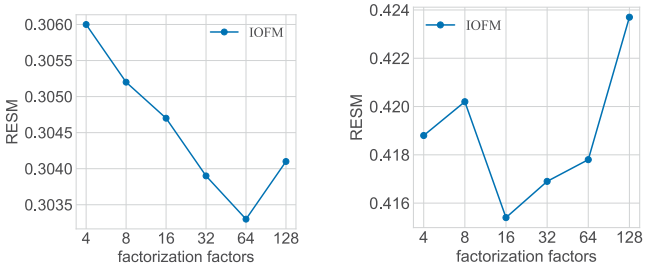
Fig. 4. Validation error of IOFM *w.r.t.* different λ_1 of L_2 regularization.



(a) Frappe. (b) MovieLens.

Fig. 5. Validation error of IOFM *w.r.t.* different λ_2 of L_2 regularization.

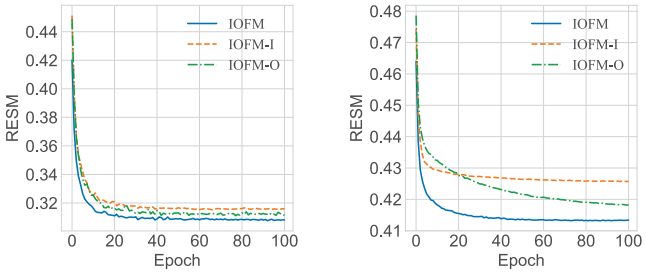
We can conclude that using dropouts only on the pair-wise interaction layer pairs simply is not enough to prevent overfitting, carefully tuning the regularization strength of the attention layer can further improve the generalization of IOFM.



(a) Frappe. (b) MovieLens.

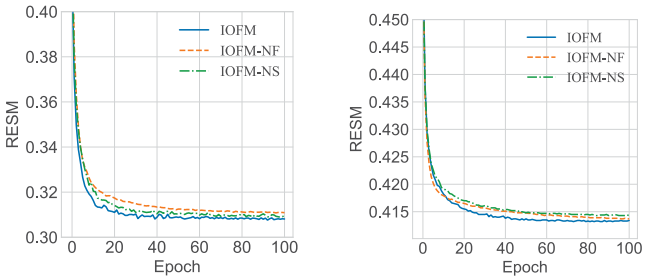
Fig. 6. Validation error of IOFM *w.r.t.* different factorization factors.

3) The number of factorization factors d : Figure 6 shows how IOFM performs when the number of factorization factors d varies. Dropout rate and L_2 regularization are set to the optimal value. Specifically, the optimal factorization factors d on Frappe and MovieLens is 64 and 16, respectively. We explain this phenomenon by studying the datasets. The size of the set $\mathcal{R}_\chi$ for the Frappe dataset is much bigger compared to the MovieLens dataset, so the optimal factorization factor d is also bigger, while a too-large factorization factor d can also cause performance degradation since too many parameters can increase the difficulty of model training. Besides, the performance of IOFM fluctuates significantly on the MovieLens dataset with the change in the factorization factors d. This phenomenon is the same as the case on the impact of dropout.



(a) Frappe. (b) MovieLens.

Fig. 7. Performance comparison on the test set of IOFM, IOFM-I, and IOFM-O.



(a) Frappe. (b) MovieLens.

Fig. 8. Performance comparison on the test set of IOFM, IOFM-NF, and IOFM-NS.

D. Impact of IO-aware Auxiliary Matrix (QR3)

IOFM enhances the representation of attention mechanism by using two awareness matrices. To investigate how each auxiliary matrix affects predictive performance, we only preserve an auxiliary matrix and explore how IOFM performs. We named Input-aware-auxiliary-matrix-only-version IOFM-I, named Output-aware-auxiliary-matrix-only-version IOFM-O. As shown in Figure 7, although on different datasets, the two auxiliary matrices have different effects on the performance of the model, IOFM performs best for both datasets, which illustrates the validity of both auxiliary matrices. Actually, as can be seen from Figure 7 that the output-aware matrix has a more significant effect on the performance than the input-aware matrix, which means that the improvement of the attention distribution plays a greater role.

TABLE III
NUMBER OF PARAMETERS OF THE AUXILIARY MATRICES AND TEST ERROR OF IOFM, IOFM-NF AND IOFM-NS.

	Frappe		MovieLens	
Method	**Param#**	**RMSE**	**Param#**	**RMSE**
IOFM	8,912	0.3076	8,240	0.4133
IOFM-NF	23,040	0.3110	1,536	0.4137
IOFM-NS	9,632	0.3091	8,288	0.4144

E. Impact of factorization and the shared matrix (QR4)

We now focus on studying the impact of the factorization and the shared matrix on IOFM. We conduct experiments with the non-factorized version (indicated as IOFM-NF), the non-shared-matrix version (indicated as IOFM-NS), and IOFM to determine performance affects. Note that early stopping is disabled in this experiment, factorization factor d is set to 16 for IOFM and IOFM-NS. As shown in Figure 8, performance is further improved by using the shared matrix for both datasets, since the shared matrix correlates the two decomposed matrices and reduces the parameter amount of the model. Factorization can speed up the convergence for Frappe dataset, while the performance for the Movielens dataset is different. We explain this phenomenon by studying the relationship between the datasets and the number of model parameters. As shown in Table III, since the difference in model parameter quantities is only reflected in the two auxiliary matrices, we only show the parameter quantities of the auxiliary matrices for the there models. For the Frappe dataset, the number of parameters is reduced by 58.2% using factorization, further reduced by 3.1% using the shared matrix, and IOFM betters IOFM-NF with a 1.1% relative improvement. However, the situation is reversed on the MovieLens dataset that the amount of parameters becomes more after the factorization. Since the MovieLens dataset contains only 3 interaction features while we use a bigger factorization factor d. Therefore, the use of factorization and shared matrix on the MovieLens dataset does not significantly improve the performance of the IOFM.

V. CONCLUSION AND FURURE WORK

In this paper, we propose a novel model IOFM that enhanced the feature representation ability of attention mechanism from two aspects. Expressly, the input-aware auxiliary matrix is inspired by FFM, used to add field interactions information into feature interactions, and the output-aware auxiliary matrix describes the attention distribution at a more fine-grained level. To make the model more efficient, we further reduce the model parameters via canonical decomposition with a shared matrix. Experimental results on two well-known datasets illustrate that our proposed model is superior to the state-of-the-art methods.

In the future, we will explore how to increase the ability of the model to capture higher-order features and try to use more advanced methods to extract more valuable information from datasets [32] to see whether it can further improve the performance. Besides, we will further generalize IOFM to a more flexible structure and explore the application of IOFM to other different types of data.

REFERENCES

[1] S. Rendle, "Factorization machines," in *2010 IEEE International Conference on Data Mining*. IEEE, 2010, pp. 995–1000.

[2] T. Chen, X. He, and M.-Y. Kan, "Context-aware image tweet modelling and recommendation," in *Proceedings of the 24th ACM international conference on Multimedia*. ACM, 2016, pp. 1018–1027.

[3] F. Petroni, L. d. Corro, and R. Gemulla, "Core: Context-aware open relation extraction with factorization machines." Assoc. for Computational Linguistics, 2015.

[4] H. Liu, X. He, F. Feng, L. Nie, R. Liu, and H. Zhang, "Discrete factorization machines for fast feature-based recommendation," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. AAAI Press, 2018, pp. 3449–3455.

[5] L. Luo, W. Zhang, Z. Zhang, W. Zhu, T. Zhang, and J. Pei, "Sketched follow-the-regularized-leader for online factorization machine," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 1900–1909.

[6] J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, and T.-S. Chua, "Attentional factorization machines: learning the weight of feature interactions via attention networks," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 2017, pp. 3119–3125.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[8] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

[9] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International conference on machine learning*, 2015, pp. 2048–2057.

[10] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in neural information processing systems*, 2015, pp. 577–585.

[11] W. Zhang, T. Du, and J. Wang, "Deep learning over multi-field categorical data," in *European conference on information retrieval*. Springer, 2016, pp. 45–57.

[12] M. Mahmud, M. S. Kaiser, A. Hussain, and S. Vassanelli, "Applications of deep learning and reinforcement learning to biological data," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 6, pp. 2063–2079, 2018.

[13] M. Mahmud, M. S. Kaiser, and A. Hussain, "Deep learning in mining biological data," 2020.

[14] H. B. McMahan, G. Holt, D. Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin *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*. ACM, 2013, pp. 1222–1230.

[15] S. Rendle, Z. Gantner, C. Freudenthaler, and L. Schmidt-Thieme, "Fast context-aware recommendations with factorization machines," in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 2011, pp. 635–644.

[16] Y. Juan, Y. Zhuang, W.-S. Chin, and C.-J. Lin, "Field-aware factorization machines for ctr prediction," in *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 2016, pp. 43–50.

[17] Y. Qu, H. Cai, K. Ren, W. Zhang, Y. Yu, Y. Wen, and J. Wang, "Product-based neural networks for user response prediction," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 2016, pp. 1149–1154.

[18] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "Deepfm: a factorization-machine based neural network for ctr prediction," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. AAAI Press, 2017, pp. 1725–1731.

[19] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu, and H. Shah, "Wide & deep learning for recommender systems," 2016.

[20] X. He and T.-S. Chua, "Neural factorization machines for sparse predictive analytics," in *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 2017, pp. 355–364.

[21] F. Hong, D. Huang, and G. Chen, "Interaction-aware factorization machines for recommender systems," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 3804–3811.

[22] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *stat*, vol. 1050, p. 9, 2015.

[23] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[24] L. Baltrunas, K. Church, A. Karatzoglou, and N. Oliver, "Frappe: Understanding the usage and perception of mobile app recommendations in-the-wild," *arXiv preprint arXiv:1505.03014*, 2015.

[25] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *Acm transactions on interactive intelligent systems (tiis)*, vol. 5, no. 4, p. 19, 2016.

[26] D. Cao, X. He, L. Nie, X. Wei, X. Hu, S. Wu, and T.-S. Chua, "Cross-platform app recommendation by jointly modeling ratings and texts," *ACM Transactions on Information Systems (TOIS)*, vol. 35, no. 4, p. 37, 2017.

[27] R. J. Oentaryo, E.-P. Lim, J.-W. Low, D. Lo, and M. Finegold, "Predicting response in mobile advertising with hierarchical importance-aware factorization machine," in *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 2014, pp. 123–132.

[28] S. Rendle, "Factorization machines with libfm," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 3, no. 3, p. 57, 2012.

[29] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, no. 8, pp. 30–37, 2009.

[30] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2008, pp. 426–434.

[31] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[32] G. Rabby, S. Azad, M. Mahmud, K. Z. Zamli, and M. M. Rahman, "Teket: a tree-based unsupervised keyphrase extraction technique," *Cognitive Computation*, pp. 1–23, 2020.