Architecture and Operation Adaptive Network for Online Recommendations

Lang Lang
Didi Chuxing
Beijing, China
langlang@didiglobal.com

Zhenlong Zhu
Didi Chuxing
Beijing, China
zhuzhenlong@didiglobal.com

Xuanye Liu
Didi Chuxing
Beijing, China
liuxuanye@didiglobal.com

Jianxin Zhao
Didi Chuxing
Beijing, China
zhaojianxin@didichuxing.com

Jixing Xu*
Didi Chuxing
Beijing, China
xujixing@didiglobal.com

Minghui Shan
Didi Chuxing
Beijing, China
shanminghui@didiglobal.com

ABSTRACT

Learning feature interactions is crucial for model performance in online recommendations. Extensive studies are devoted to designing effective structures for learning interactive information in an explicit way and tangible progress has been made. However, the core interaction calculations of these models are artificially specified, such as inner product, outer product and self-attention, which results in high dependence on domain knowledge. Hence model effect is bounded by both restriction of human experience and the finiteness of candidate operations. In this paper, we propose a generalized interaction paradigm to lift the limitation, where operations adopted by existing models can be regarded as its special form. Based on this paradigm, we design a novel model to adaptively explore and optimize the operation itself according to data, named generalized interaction network(GIN). We proved that GIN is a generalized form of a wide range of state-of-the-art models, which means GIN can automatically search for the best operation among these models as well as a broader underlying architecture space. Finally, an architecture adaptation method is introduced to further boost the performance of GIN by discriminating important interactions. Thereby, architecture and operation adaptive network(AOANet) is presented. Experiment results on two large scale datasets show the superiority of our model. AOANet has been deployed to industrial production. In a 7-day A/B test, the click-through rate increased by 10.94%, which represents considerable business benefits.

CCS CONCEPTS

• **Information systems** → **Personalization**; • **Computer systems organization** → *Neural networks.*

*Corresponding author

KDD '21, August 14–18, 2021, Virtual Event, Singapore.

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467133

KEYWORDS

neural networks, deep learning, feature interaction, recommendations, learning to rank

ACM Reference Format:
Lang Lang, Zhenlong Zhu, Xuanye Liu, Jianxin Zhao, Jixing Xu, and Minghui Shan. 2021. Architecture and Operation Adaptive Network for Online Recommendations. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3447548.3467133

1 INTRODUCTION

In recent years, deep neural networks have achieved great success in online recommendations. Quite a few studies have demonstrated that learning feature interactions can significantly boost the model performance[3, 6, 12, 16, 18–20], for the sake that combinations of features provides more extensive information than either.

Manually crafting feature interactions requires exhaustive effort and profound experience. In addition it can be difficult to enumerate all combinations and generalize unseen interactions[13]. Thereby many model structures have been proposed aiming to learn feature interactions in an automatic way. High-dimensional raw features are firstly mapped to dense vectors using embedding technique. Then MLPs are adopted to extract implicit interactions, and carefully designed structures with specified formula like CrossNet[19] and CIN[6] are used to extract explicit interactions. The latter is considered to be an enhancement of MLPs, for there is no guarantee that a DNN can converge to expected function[12, 14]. Combining implicit and explicit feature interactions is proven to be effective in researches[6, 18, 19].

However, there are two bottlenecks that limit expressiveness of explicit interaction structures. On the one hand, the representation of interactive information directly relies on certain operation that is artificially specified, such as inner product for FM, outer product for OPNN or customize operations like CrossNet[18]. Too simple operations will lead to loss of information, otherwise it will introduce noise. In practical applications, arduous repetitive work is required to test out the most suitable structure for different scenarios. And due to the limited number of candidate sets, there is no guarantee that the final choice is optimal. On the other hand, as it is pointed out in several literatures, not all interactions are conducive to performance[7, 9, 11, 20]. Redundant parameters introduced by

useless interactions complicate the training process and potentially lower model performance[14].

To address the both downsides mentioned above, we propose Architecture and Operation Adaptive Network, named AOANet. We start with decomposing existing explicit interaction architectures into three distinct components, namely Projection, Interaction and Fusion. In order to relieve the restrictions of manual designed operations in the foremost Interaction phase, we propose a generalized interaction operation paradigm. We demonstrate that extensive existing interaction operations can be regarded as a special form of this paradigm by constraining the weights and set proper structure for Fusion phase. This allows the model to adaptively explore and learn the optimal interaction operation according to data distribution. Based on proposed paradigm we design a generalized interaction network named GIN. Furthermore, we extend GIN to multiple subspaces in order to search duplicate optimal operations in parallel. Inspired by DARTS[8] and AutoFIS[7], we propose a two-stage interaction selection method. Discrete choices of interactions are relaxed to be continues and optimized together with other model parameters. Then unimportant interactions are pruned and model is re-trained with simplified architecture. The main contributions are as follows:

- We first propose a unified framework to understand existing feature interaction models by decomposing structure into Projection, Interction and Fusion phase. Then a generalized interaction paradigm is proposed to overcome the limitations of artificially specified calculations in Interaction phase. Based on the generalized operation, we design specific structures for each phase, called GIN, which can adaptively learn optimal interaction operation w.r.t data distribution.
- Subsequently, we further boost the model performance by architecture adaptation method and thereby introduce AOANet. Gating mechanism and a heuristic selection method are introduced to distinguish importance of interactions and altering model architecture accordingly.
- We conduct offline and online experiments to evaluate performance of AOANet. Results show that our work outperforms state-of-the-arts. In addition, we deploy AOANet to industrial production environment, and a seven-day A/B test shows that our model brings about 10.94% in average uplift in click through rate, which means significant business values.

2 RELATE WORK

Our work is relevant to two lines of work: 1) learning feature interactions, and 2) automated machine learning.

Learning Feature Interactions. A series of improved methods based on FM[13] specify an operation to extract interactive information, such as NFM[4], DeepFM[3] and PNN[12]. AutoInt[16] use multi-head self-attention structure to model interaction. MLP and well-designed cross structure are widely adopted in parallel to model implicit and explicit interactions simultaneously, such as DCN[18] and xDeepFM[6]. Some works like AFM[20] and FwFM[11] focus on distinguish importance of feature interactions.

Automated Machine Learning. AutoML[22] is widely applied to explore suitable model structures, and it has also received attention in the recommendation field in recent years. One of the most popular automl technique is DARTS[8] ,which is a one shot architecture search method(NAS) to select model architecture. This work inspired Autofis[7] to search and select important interactions in order to improve model effect. AutoCross[9] is another work that focuses on discriminating important interactions. It searches for a subset of candidate elements to obtain effective interaction, which also illustrates the necessity of structural adaptation w.r.t interactive information.

3 METHODOLOGY

3.1 Architecture Decomposition

Model architectures that used to explicitly extract feature interactions can be decomposed into three distinct phases, namely **Projection**, **Interaction** and **Fusion** respectively. Generally speaking, in Projection phase, original raw features are projected to singular or multiple low-dimensional vector spaces; In Interaction phase, a matrix operation is specified to obtain interactive information between embedding vectors; In fusion phase, the outputs of interaction operation are fused to get the final prediction. For simplicity, we temporarily ignore MLPs that used for implicit feature interactions.

Projection: For industrial online recommendation scenarios, features are commonly sparse and discrete. High-dimensional sparse features are projected to low dimensional embedding vectors firstly. Let $X = \{x_1, x_2, \ldots, x_n\}$ denotes the set of raw input features, where n is the total number of features, we define the projection process as follows:

$$A = P(X) \tag{1}$$

where $A = \{a_1, a_2, \ldots, a_m\}$ is the set of all projected dense vectors and m is the total number of them. Typically, the mapping between raw feature and embedding is one-to-one. But there are also some models that, based on prior understanding of data, map one original feature to multiple vector spaces. For instance, FFM[5] and similar improvement methods[11, 21] believe that different embedding vectors should be used for interaction between different fields, hence each original feature corresponds to f embedding vectors, where f is the number of fields; Autoint[16] designed based on the self-attention mechanism, maps features into three vectors q, k, v, and the inner product operation between q and k is used to characterize the degree of mutual association, and v is regarded as the representation of corresponding feature itself.

Interaction: Operation that express interactive information is essential to model effect. Researchers make unique designs on which objects to interact and what operations should be applied based on domain knowledge. Interaction phase can be defined as follows:

$$C = \{I(u, v) \mid (u, v) \in \text{pairing}(M, N)\} \tag{2}$$

where M and N are two sets of vectors, pairing$(\cdot)$ function defines the objects involved in interaction operation, $I(\cdot)$ denotes the interaction operation.

There are roughly two types of interaction mode, namely intra-layer and inter-layer. In intra-layer mode, operations between vectors are only performed in the same layer, i.e, $M = N$, such as

Table 1: Relation between generalized interaction operation and others. $u \in \mathbb{R}^d$ and $v \in \mathbb{R}^d$ are two vectors involved in feature interaction. $\odot$ denotes Hadamard product; $\otimes$ denotes outer product. For self attention, q,k and v denotes query,key and value; Z^j is corresponding to interaction between query and the j^{th} key k^j; v^j is the value vector of j^{th} feature.

operation	related models	constraints on W	partial fusion setup
$u^T v$	FM,IPNN	$W = \text{diag}(1, \ldots, 1)$	$\sum_{i=1}^{d} Z_{i,i}$
$u \odot v$	NFM,xDeepFM	$W = \text{diag}(1, \ldots, 1)$	$\left[Z_{1,1}, \ldots, Z_{i,i}, \ldots, Z_{d,d}\right]^T$
$u \otimes v$	OPNN	$\forall i \forall j, W_{i,j} = 1$	–
$\alpha(u^T v)$	FwFM	$W = \text{diag}(\alpha, \ldots, \alpha)$	$\sum_{i=1}^{d} Z_{i,i}$
$\alpha(u \odot v)$	AFM	$W = \text{diag}(\alpha, \ldots, \alpha), \alpha = MLP(Z)$	$\left[Z_{1,1}, \ldots, Z_{i,i}, \ldots, Z_{d,d}\right]^T$
$(u \otimes v)w^D$	DCN	$\forall i \forall j \forall k, W_{i,j} = W_{i,k}$	$\left[\sum_i Z_{i,1}, \ldots, \sum_i Z_{i,j}, \ldots, \sum_i Z_{i,d}\right]^T$
$u \odot (W^D v)$	DCN-M	–	$\left[\sum_j Z_{1,j}, \ldots, \sum_j Z_{i,j}, \ldots, \sum_j Z_{d,j}\right]^T$
self-attention	AutoInt	$Z^j = (q \otimes k^j) \odot W, W = \text{diag}(1, \ldots, 1)$	$(\frac{e^{\sum_i Z^j_{i,i}}}{\sum_j e^{\sum_i Z^j_{i,i}}}) v^j$

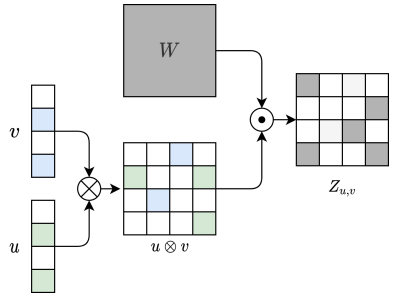


Figure 1: Generalized interaction operation

deepFM and AutoInt. In inter-layer mode, operations are performed between layers, such as DCN and xDeepFM.

Also, interaction operations are of diversity. For example, IPNN[12] and FM[13] use inner product, OPNN[12] uses outer product, NFM[4], AFM[20] and xDeepFM use Hadamard product. AutoInt uses the inner product of query and key to express the correlation between features. DCN defined a formula that iteratively calculate interaction between origin input x_0 and current layer input x_l; DCN-M improves the model expressiveness while framework stays same.

Fusion: Subsequently, the output tensors of Interaction phase are melted and fused to predict final objective or further interact. Let B be the output of fusion phase, we have:

$$B = F(C) \tag{3}$$

For instance, FM adds up the result of inner product; AFM and NFM feed interacted vector in to a sum-pooling layer; IPNN and OPNN feeds the tensor into a MLP. The special one is xDeepFM, which stitches each layer after sum-pooling into a long vector for fitting high-order interaction.

3.2 Generalized Interaction

Manually designed operation is bounded by domain knowledge, therefore it can be quit challenging to pick the optimal one. A natural idea is to let the model learn appropriate interaction operation by itself. A generalized interaction operation between two vectors u and v is proposed as follows:

$$I(u, v) = (u \otimes v) \odot W \tag{4}$$

where $u \in \mathbb{R}^d$, $v \in \mathbb{R}^d$, $W \in \mathbb{R}^{d \times d}$, d is the dimension of vectors, $\otimes$ and $\odot$ denotes outer product and Hadamard product respectively. The calculation process is illustrated in Figure 1. Denote interaction map $Z_{u,v}$ as result of generalized interaction between u and v, then we can rewrite Formula 2:

$$C = \{Z_{u,v} \mid \in \text{pairing}(M, N)\} \tag{5}$$

where $Z_{u,v} \in \mathbb{R}^{d \times d}$, $Z_{u,v} = I(u, v)$.

We argue that interaction operations of existing models are special forms of Formula 4. We list interaction operations of each model and how Formula 4 simulate each of them by setting fusion structure and constraints on W in Table 1. Detail proofs can be found in appendix. By optimizing the weights of the W in model training, suitable operation w.r.t the data distribution can be learned adaptively. In this way, we naturally remove the limitations of artificial setup for explicit feature interaction.

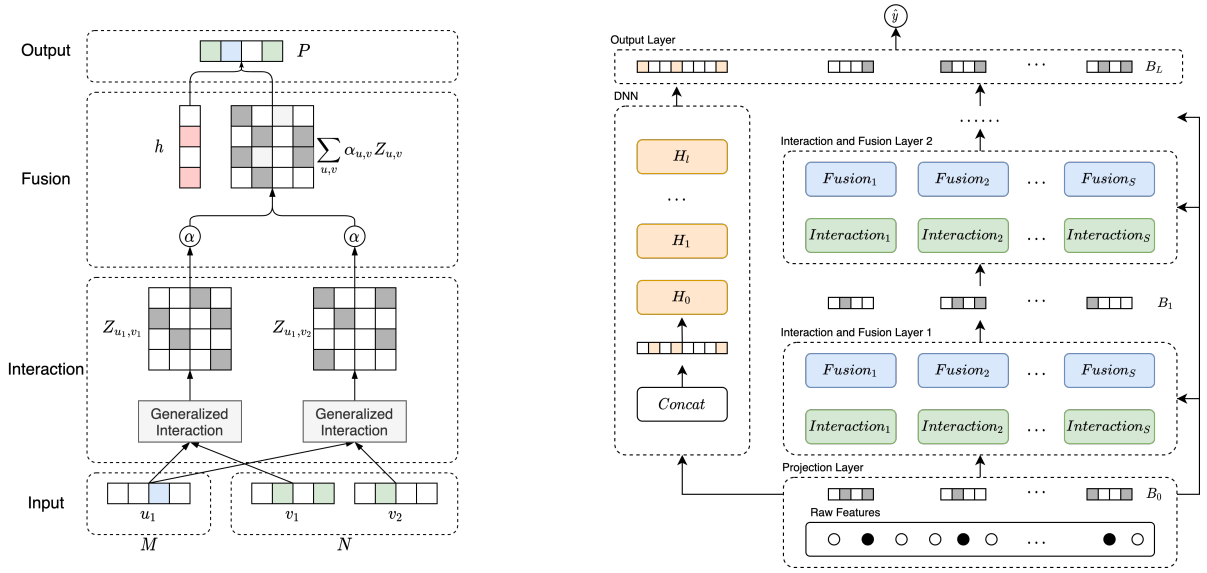
3.3 Generalized Interaction Network

In this section, we detail a model designed based on Formula 5. The specific structure of Projection, Interaction and Fusion are introduced, and then we parallelize interaction optimization into multiple subspaces to further improve the effect. The proposed structure can be approximate to extensive models with proper setting, named generalized interaction network(GIN), and relation among them are discussed in depth. Finally we detail an architecture adaptation method based on GIN, thereby propose architecture and operation adaptive network, called AOANet.

3.3.1 ***Projection Layer***. Embedding is a commonly used technique that projects high-dimensional feature into a low-dimensional vector space. We adopt one-to-one fashioned embedding as projection layer. Specifically, Formula 1 is rewritten as:

$$A = E(X) \tag{6}$$

where $X = \{x_0, x_1, \ldots, x_n\}$, $A = \{a_0, a_1, \ldots, a_n\}$, $a_i \in \mathbb{R}^d$ is the embedding vector of x_i, n is the total number of input features. For



(a) Structure of interaction and fusion layer.

(b) Generalized Interaction Network(GIN)

Figure 2: Framework of architecture and operation adaptive network.

each individual element, $a_i = V_i \cdot x_i$ where V_i denotes the embedding matrix.

3.3.2 ***Interaction Layer.*** We adopt interaction layers under inter-layer mode. Let us denote output of l^{th} interaction layer and fusion layer mentioned later as C_l and B_l respectively, The formula of l^{th} interaction layer is given as:

$$Z_{u,v} = (u \otimes v) \odot W_l \tag{7}$$

$$C_l = \{Z_{u,v} \mid (u,v) \in \text{pairing}(B_0, B_{l-1})\} \tag{8}$$

where $Z_{u,v} \in \mathbb{R}^{d \times d}$ represents the interaction map of vector u and v, $\text{pairing}(M,N) = \{(u,v) \mid u \in M, v \in N\}$, C_l is a set of interaction maps produced between vectors in B_0 and B_{l-1}, $B_0 = A$ is the output of projection layer.

3.3.3 ***Fusion Layer.*** There are quit a few recent researches focus on modeling the importance of different feature interaction[7, 11, 20]. We introduce a parameter α for each feature map Z involved to explicitly model their importance, specifically:

$$P_l = h_l^T \sum_{u,v} (\alpha_{u,v} Z_{u,v}) \tag{9}$$

where $\alpha_{u,v} \in \mathbb{R}$ represents the importance of interaction map $Z_{u,v}$ in predicting target, $h \in \mathbb{R}^d$ is a projection matrix. Weighted sum is performed on interaction maps and then projected to a vector $P_l \in \mathbb{R}^d$. Then the output of l^{th} fusion layer is constructed as:

$$B_l = \{P_l\} \tag{10}$$

where B_l is a set of fused vectors. Fusion layer along with interaction layer forms a complete structure that models interactive information between two groups of vectors. Figure 2(a) illustrates the architecture.

3.3.4 ***Multiple Subspace Interaction.*** In our practical production, we found that ensemble of information derived by different feature structures boosted overall performance. This implies that exploring interaction operations in multiple subspaces may has positive effect on model expressiveness. Additionally, similar ideas are reported in mixture of experts(MOE)[2, 10, 15] and Transformer[17]. MOE-based methods fuse information extracted by multiple expert layers with gate mechanism to improve accuracy of each individual task. Transformer propose mutli-head attention that projects origin query, key and value to parallel subspaces and found it beneficial. We have interaction and fusion layer under scope of multiple subspaces as follows:

$$Z^s_{u,v} = (u \otimes v) \odot W^s \tag{11}$$

$$P^s = h^{sT} \sum_{u,v} (\alpha^s_{u,v} Z^s_{u,v}) \tag{12}$$

$$B = \{P^1, P^2, \ldots, P^S\} \tag{13}$$

where $s = 1, 2, ..., S$ is the index of subspace, S is the total number of subspaces, $Z^s_{u,v}$ denotes interaction of u and v in the s^{th} subspace, P^s is the fused interaction map in s^{th} subspace, and the output of fusion layer B is a set consist of P from all subspaces.

3.3.5 ***Deep Network.*** We adopt fully connected network to extract implicit interaction information:

$$H_l = \sigma(W_l H_{l-1} + b_l) \tag{14}$$

where H_l is output of l^{th} layer, $\sigma(\cdot)$ is activation function which is RELU in our model, W_l and b_l are weights and bias of l^{th} respectively.

3.3.6 ***Output Layer***. Finally, we concatenate all the output vectors in B and output of last layer of deep network, then apply a non-linear projection for final prediction:

$$Q = \text{concat}(P^1, P^2, \ldots, P^S) \tag{15}$$

$$\hat{y} = \sigma(w^T \text{concat}(Q, H) + b) \tag{16}$$

where $w \in \mathbb{R}^{dS+|H|}$, b is bias, $\sigma(\cdot)$ is sigmoid function that $\sigma(x) = 1/(1+e^{-x})$. For binary classifications, the loss function is the log loss:

$$\mathcal{L} = -\frac{1}{n}\sum_{i=1}^{n} y_i log(\hat{y_i}) + (1-y_i) log(1-\hat{y_i}) \tag{17}$$

where n is the total number of input samples, y_i and $\hat{y_i}$ are label and prediction of i^{th} sample respectively. The overall structure of GIN is illustrated in Figure 2(b)

3.4 Model Analysis

In this section, we dive into connections between GIN and related models. We demonstrate that GIN can be approximately equal to extensive known feature interaction models under specific hyper-parameters and constraints, which means that GIN naturally explore the function space modeled by state-of-the-art models as well as a potentially larger hypothesis space.

Notations. First, we define and rephrase common notations for convenience:

- L: Total number of layers.
- S: Total number of subspaces.
- W: Weight matrix of generalized interaction operation in Formula 4
- Z: Interaction map generated by generalized operation defined in Formula 5.
- h: Projection matrix of fusion layer in Formula 9.
- α: Importance coefficient of interaction map in Formula 9.
- others: Some unique notations will be explained when mentioned.

FM-based Models. The connection between GIN and FM-based models is somewhat clear and proved in Section 3.2. The core operations of deepFM and IPNN are both inner product, which can be simulated by setting $W = \text{diag}(1, \ldots, 1)$ and then adding up the diagonal elements of Z. FwFM improved deepFM by explicitly modeling importance for each interaction. From the perspective of GIN settings, FwFM removes the limitation of parameter α on the basis of deepFM and IPNN.

The core operation of NFM is Hadamard product, where the constraint on W is same as inner product and then project diagonal elements of Z to a vector. Thereby, comparing to deepFM and IPNN, the restriction on h is removed for NFM. $\sum Z$ plays the role of output vector of Bi-Interaction Layer in NFM, and h and following output layer models non-linear feature interaction. AFM improves NFM by discriminating the importance of different feature interactions. Specifically, α is modeled by attention mechanism that $\alpha_{u,v} = MLP(Z_{u,v})$, where the MLP is Attention Net of AFM.

To conclude from perspective of GIN, the improvements of each model based on FM are: FwFM removes the constraints on α, NFM removes the constraints on h, and AFM learns α through the attention mechanism on the basis of NFM.

OPNN The interaction of OPNN is outer product. Approximation setting is similar as IPNN, except that all elements of Z are taken into account.

DCN and DCN-M DCN and DCN-M design there own interaction operations respectively. The input of these two models is the concatenation of embedding vectors, and output of each layer is a vector of same dimension. Let us denote output of l^{th} layer as x_l, where x_0 represents the input of first layer. The operation of l^{th} layer is given by:

$$x_{l+1} = x_0 x_l^T w_l + x_l + b_l \tag{18}$$

where w_l and b_l are weights and bias of l^{th} layer respectively. Denote B_l as the output of l^{th} layer of GIN. The residual term x_l and bias term b_l can be naively approximated by append B_{l-1} and a constant vector b_l to B_l. The key interaction operation lies in the first term of Formula 18, which is a special form of GIN with single subspace.

Lemma 3.1. *Let $h = [1, \ldots, 1]^T$, $\alpha = [1]^T$, $\forall j \forall k, w_{i,j} = w_{i,k}$, the interaction operation defined by GIN is equivalent to the first term of DCN.*

Proof. Let us denote $x_l = u$, $x_0 = v$, $x_{l+1} = B$, then we can rewrite Formula 18 of DCN as follows:

$$B = vu^T w = (\sum_i u_i w_i) v \tag{19}$$

Given $\forall j \forall k, W_{i,j} = W_{i,k}$, without loss of generality, we can make $W = \left[w^T, \ldots, w^T\right]$, thus:

$$W_{i,j} = w_i \tag{20}$$

For GIN, we have $B = h^T \sum_{u,v} \alpha Z_{u,v}$, $Z = (u \otimes v) \odot W$, where $Z_{i,j} = u_i v_j W_{i,j}$. Given $\alpha = [1]^T$, $h = [1, \ldots, 1]^T$, $W_{i,j} = w_i$, we have:

$$B_i = \sum_i Z_{i,j} = v_j \sum_i u_i w_i \tag{21}$$

Thereby we have:

$$B = \left[v_1 \sum_i u_i w_i, v_2 \sum_i u_i w_i, \ldots, \right]^T = (\sum_i u_i w_i) v \tag{22}$$

Thus proved. □

DCN-M improves the expressiveness of DCN by changing the interaction operation. l^{th} layer is given as follows:

$$x_{l+1} = x_0 \otimes (W^D x_l + b_l) + x_l \tag{23}$$

Note that the basic structure of DCN-M is same as DCN, hence the residual term x_l and bias term b_l can be approximated in the similar way. We focus on the first term, specifically $x_0 \otimes (W_d x_l)$. This is also a special form of GIN.

Lemma 3.2. *Let $h = [1, \ldots, 1]^T$, $\alpha = [1]^T$, $W^T = W^D$. The interaction operation defined by GIN is equivalent to the first term of DCN-M.*

Proof. Let us denote $x_l = u$, $x_0 = v$, $x_{l+1} = B$, then we can rewrite Formula 23 of DCN-M as follows:

$$B = v \otimes (W^D u) \tag{24}$$

Table 2: Connection between GIN and related models. B_0 denotes the input of first layer; Q is the concatenated vector in Formula 15; S_l is the number of subspaces of l^{th} layer in GIN; H_l is the number of feature vectors of l^{th} layer in xDeepFM; n denotes the total number of input features.

model	approximation settings					
	L	S	W	h	α	others
deepFM,IPNN	1	1	$\text{diag}(1,\ldots,1)$	$[1,\ldots,1]^T$	$[1,\ldots,1]^T$	–
FwFM	1	1	$\text{diag}(1,\ldots,1)$	$[1,\ldots,1]^T$	-	–
AFM	1	1	$\text{diag}(1,\ldots,1)$	$[1,\ldots,1]^T$	$MLP(Z_{u,v})$	–
OPNN	1	1	$\forall i \forall j, w_{i,j}=1$	–	$[1,\ldots,1]^T$	–
DCN	–	1	$\forall i \forall j \forall k, w_{i,j}=w_{i,k}$	$[1,\ldots,1]^T$	$[1]^T$	$B_0=\{\text{concat}(A)\}$
DCN-M	–	1	–	$[1,\ldots,1]^T$	$[1]^T$	$B_0=\{\text{concat}(A)\}$
xDeepFM	–	$S_l=H_L$	$\text{diag}(1,\ldots,1)$	$[1,\ldots,1]^T$	–	$B=\left[\sum_i P_i^1,\ldots,\sum_i P_i^s\right], O=\text{concat}(B_1,\ldots,B_L)$
AutoINT	–	n	$W_{i,j}=\frac{k_i}{k_j}v_j$	$[1,\ldots,1]^T$	$\alpha_i=\frac{e^{q_i^T k_j}}{(\sum_j e^{q_i^T k_j})q_i^T k_j}$	partly pairing, inter-layer mode

Where we have $B_i = v_i(\sum_j W_{i,j}^D u_j)$. Given that $W^D = W^T$, thus $W_{i,j}^D = W_{j,i}$, thereby:

$$B_i = v_i(\sum_j W_{j,i} u_j) \tag{25}$$

For GIN, we have $B = h^T \sum_{u,v} \alpha Z_{u,v}$, $Z = (u \otimes v) \odot W$, where $Z_{i,j} = u_i v_j W_{i,j}$. Provided that $h = [1,\ldots,1]^T$, $\alpha = [1]^T$, we have:

$$B_j = \sum_i u_i v_j W_{i,j} = v_j(\sum_i W_{i,j} u_i) \tag{26}$$

Let us switch the notation of subscript i and j for Formula 26, then it is same as Formula 25. Thus proved. □

The above derivations can lead to an interesting interpretation. From the perspective of GIN, the key improvement of DCN-M to DCN is mainly to remove the constraint on W. Therefore, the model can explore the suitable interaction operation in a larger solution space. This may shed light on why DCM-M outperforms previous works more clearly.

xDeepFM. The l^{th} layer of xDeepFM model is given by:

$$X_l^{s,*} = \sum_{i,j} W_{i,j}^s (X_0^{j,*} \odot X_{l-1}^{i,*}) \tag{27}$$

where X_l denotes the output, s is a hyper-parameter represents the the number of feature vectors in l^{th} layer. W^s is weight matrix for s^{th} feature vector. The main connection lies in the the number of feature vector s, where one feature vector corresponds to one subspace of GIN.

Lemma 3.3. *Given $h = [1,\ldots,1]^T$, $W = \text{diag}(1,\ldots,1)$, the number of subspaces for GIN in l^{th} layer is equal to xDeepFM, specifically $S_l = H_l$, then the interaction operation of GIN is equivalent to xDeepFM.*

Proof. For xDeepFM, let B_l=X_l, we can rewrite Formula as:

$$B_l^{s,*} = \sum_{i,j} W_{i,j}^s (B_0^{j,*} \odot B_{l-1}^{i,*}) \tag{28}$$

For the l^{th} layer and s^{th} channel of GIN, we have:

$$P_l^s = h^{sT} \sum_{u,v} \alpha_{u,v}^s Z_{u,v}^s \tag{29}$$

Given that $h = [1,\ldots,1]^T$, $W = \text{diag}(1,\ldots,1)$,

$$\begin{aligned} P_l^s &= \sum_{u,v} \alpha_{u,v}^s (h^T (u \otimes v) W) \\ &= \sum_{u,v} \alpha_{u,v}^s (u \odot v) \end{aligned} \tag{30}$$

Without loss of generality, we define:

$$B = \left[p^{0T}, p^{1T}, \ldots, p^{ST}\right]^T \tag{31}$$

where S denotes the total number of subspaces. Then we have the i^{th} row of B denote as:

$$B_{i,*} = P_i^T \tag{32}$$

Let $u = B_0^{i,*}$, $v = B_l^{i,*}$, $\alpha_{u,v}^s = W_{i,j}^s$, then we have:

$$P_l^s = B_l^{s,*} = \sum_{i,j} W_{i,j}^s (B_0^{j,*} \odot B_{l-1}^{i,*}) \tag{33}$$

Thus proved. □

AutoInt. AutoInt adopt self-attention mechanism to model feature interaction. The i^{th} feature x_i is projected to three vetors q_i,k_i and v_i respectively. The output $\widetilde{x}_i$ is as follows:

$$\widetilde{x}_i = \sum_j (\frac{e^{q_i^T k_j}}{\sum_j e^{q_i^T k_j}} v_j) \tag{34}$$

Interaction layer of AutoInt can be regarded as GIN with n subspaces, where n is the total number of input features. There are two alters of GIN structure to approximate AutoInt. First, standard GIN project one feature to a single embedding vector while AutoInt to three. Second, the latter is of inter-layer interaction mode.

Lemma 3.4. *Given the projection phase $Pr(X) = \{Q, K, V\}$, where $q_i \in Q$, $k_i \in K$, $v_i \in V$, $i = 1, 2, \ldots, n$ correspond to the i^{th} input. For specified interaction between i^{th} an j^{th} feature, the W of generalized interaction is defined by $W_{i,j} = \frac{k_i}{k_j} v_j$; For specified channel s, paring function is defined by* $\text{pairing}(Pr(X), Pr(X)) = \{(q_s, k_i) | q_s \in$

$Q, k_i \in K\}$, *importance coefficient is defined by* $\alpha_i = \frac{e^{q_i^T k_j}}{(\sum_j e^{q_i^T k_j}) q_i^T k_j}$; *Then interaction layer of GIN is equivalent to AutoInt.*

Proof. Let $P_s = \widetilde{x}_s$, we can rewrite Formula 34 as:

$$\widetilde{P}_s = \sum_j (\frac{e^{q_s^T k_j}}{\sum_j e^{q_s^T k_j}} v_j) \tag{35}$$

For GIN operation between q and k is given by:

$$Z_{i,j} = q_i k_j W_{i,j} \tag{36}$$

Since $W_{i,j} = \frac{k_i}{k_j} v_j$, then:

$$Z_{i,j} = q_i k_j (\frac{k_i}{k_j} v_j) = q_i k_i v_j \tag{37}$$

For the s^{th} channel of GIN, we have:

$$P_s = h^T \sum_{u,t} \alpha_{u,t} Z_{u,t} = \sum_{u,t} \alpha_{u,t} (h^T Z_{u,t}) \tag{38}$$

From the pairing function we know that $u = q_s$, $t \in K$, then:

$$P_s = \sum_j \alpha_j (h^T Z) \tag{39}$$

Since $h = [1, \ldots, 1]^T$, $Z_{i,j} = q_i k_i v_j$, we know:

$$\begin{aligned} h^T Z &= \left[\sum_i Z_{i,1}, \ldots, \sum_i Z_{i,j}, \ldots \right]^T \\ &= \left[\sum_i q_i k_i v_1, \ldots, \sum_i q_i k_i v_j, \ldots \right]^T \\ &= \sum_i q_i k_i \left[v_1, \ldots, v_j, \ldots \right]^T \\ &= (q^T k) v \end{aligned} \tag{40}$$

Recall that this is in the s^{th} subspace, we have $\alpha_j^s = \frac{e^{q_s^T k_j}}{(\sum_j e^{q_s^T k_j}) q_s^T k_j}$, then we have Formula 39 as:

$$\begin{aligned} P_s &= \sum_j (\frac{e^{q_s^T k_j}}{(\sum_j e^{q_s^T k_j}) q_s^T k_j}) (q_s^T k_j) v_j \\ &= \sum_j (\frac{e^{q_s^T k_j}}{\sum_j e^{q_s^T k_j}} v_j) \end{aligned} \tag{41}$$

Note that $P_s = \widetilde{P}_s$, thus proved. □

3.5 Architecture Adaptation

Quit a few studies implies the effectiveness of distinguishing importance of feature interactions. Inspired by DARTS and AutoFIS, we adopt a two-stage method to select key interactions by relaxing the choice of interactions to be continues. Then interactions are selected according to α and then the model is re-trained under new architecture.

Selection Stage We first train the model proposed in Section 3.3 until converge. Recall the Formula 9 of fusion layer:

$$P = h^T \sum_{u,v} \alpha_{u,v} Z_{u,v} \tag{42}$$

The fusion result is given by wighted-sum over all interactions, where $\alpha_{u,v}$ acts like a gate w.r.t $Z_{u,v}$, controlling how much information of each interaction map can go through. Thereby, the coefficients naturally play the role indicating the amount of contribution . In this stage, α is trainable.

Re-train Stage. Then we prune those unimportant interactions based on α. Note that the complexity of brute force searching over all combinations of $Z_{u,v}$ is incredibly up-to $2^{C^2_{|paring(M,N)|}}$ where $u \in M, v \in N$. Therefore we adopt a heuristic algorithm. We assume that the importance of interaction map Z is positively correlated with the absolute value of α and those unimportant interactions will be permanently pruned in one-shot. A gating function $G(\alpha)$ is used to determine whether Z is pruned or kept. $G(\alpha) = 0$ means that corresponding interaction is pruned. Finally the model is re-trained under new architecture that those unimportant interactions an abandoned. Specifically, fusion process in this stage is given by:

$$P = h^T \sum_{u,v} G(\alpha) \alpha_{u,v} Z_{u,v} \tag{43}$$

By applying architecture adaptation method to GIN, we propose Architecture and Operation Adaptive Network, namely AOANet.

4 EXPERIMENTS

In this section, we conduct extensive offline experiments on one public dataset and one private dataset. Then an online A/B test is conducted to investigate the model effect in real industrial environment. We aim to answer the following questions:

- **RQ1**: How does our proposed model perform offline comparing to other related models?
- **RQ2**: How do different components and settings influence the performance?
- **RQ3**: How does the AOANet performs in online A/B test?

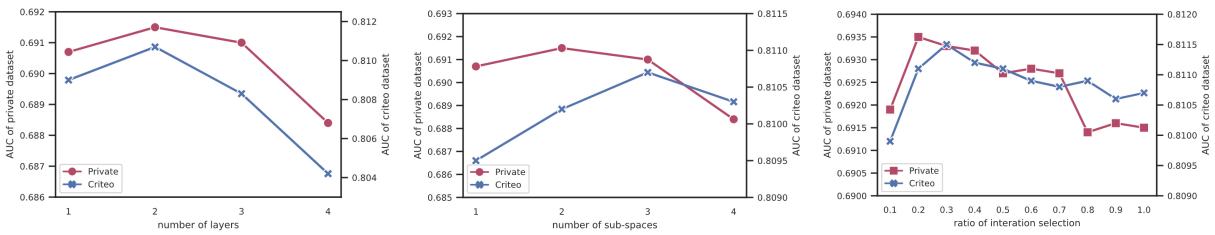
4.1 Experiment Setup

Datasets. We conduct offline experiments on one public dataset and one private dataset. **Criteo** is a famous benchmark for ctr prediction tasks, which includes user clicking records on displayed ads in 7 days. The total amount of samples is about 45M and there are 13 numeric fields and 26 categorical fields in the dataset. We randomly divide the dataset into two parts: 90% for training and the remaining 10% for testing. **Private** is a real industrial dataset of DiDiChuXing recommendations with 58M samples. We collected 14 days of log data as training set and following day is used as testing set. The dataset contains 54 fields, including user attributes, item attributes and contextual features that describe the environment at the time.

Baselines. We compare our model with 9 related feature interaction models including DeepFM, IPNN, OPNN, FwFM, AFM, DCN, DCN-M, xDeepFM and AutoInt. As we discussed in section 3.4, these model are strongly related to our work and can be seen as a special form of GIN.

Model Settings. We implemented AOANet using TensorFlow[1]. For fair comparison, we set same embedding size, batch size and optimizer for all models, which are 16, 1024 and Adam Optimizer respectively. Deep component also stays same for all models. The

[1]https://github.com/mpt-algorithm/aoanet



(a) Number of Layers (b) Number of Subspaces (c) Number of Interaction Selection Ratio

Figure 3: Ablation study of model setting on performance of AUC.

Table 3: Overall performance on Criteo and Private dataset.

model	Criteo		Private	
	AUC	log loss	AUC	logloss
IPNN	0.8019	0.4531	0.6819	0.4746
OPNN	0.8032	0.4425	0.6835	0.4655
DeepFM	0.8023	0.4533	0.6821	0.4752
FwFM	0.8037	0.4517	0.6852	0.4652
AFM	0.8034	0.4519	0.6859	0.4650
DCN	0.8048	0.4497	0.6845	0.4672
DCN-M	0.8101	0.4479	0.6910	0.4603
xDeepFM	0.8077	0.4491	0.6882	0.4623
AutoINT	0.8079	0.4498	0.6889	0.4639
GIN	0.8107	0.4477	0.6915	0.4595
AOANet	**0.8115**	**0.4470**	**0.6935**	**0.4588**

Table 4: Model settings. F denotes attention factor; L denotes number of layers; S denotes number of subspaces; n denotes number of feature vectors for xDeepFM; h denotes the head number of multi-head attention; d denotes the dimension of query,key and value for AutoInt; r denotes interaction selection ratio of AOANet.

model	architecture setting	
	Criteo	Private
AFM	$F = 3$	$F = 5$
DCN	$L = 3$	$L = 3$
DCN-M	$L = 2$	$L = 3$
xDeepFM	$L = 2, n = 100$	$L = 2, n = 100$
AutoINT	$L = 2, h = 3, d = 16$	$L = 2, h = 4, d = 32$
GIN	$L = 2, S = 3$	$L = 2, S = 2$
AOANet	$L = 2, S = 3, r = 0.3$	$L = 2, S = 2, r = 0.2$

numbers of hidden units for each layer are [512, 256, 64] from bottom to top respectively. Other Hyper parameters is tuned and selected via grid search. We conducted five rounds of repeated experiments for each model and each data set, and then the average of metrics are recorded as final results.

Metrics. We take log-loss and AUC(Area under ROC curve) as offline metric and uplift of click throgh rate as online metric.

4.2 Performance Comparison (RQ1)

In this section we discuss the experiment results demonstrated in Table 3. The best architecture settings for each model are listed in Table 4.

The result shows that GIN outperforms all baselines on both datasets and interaction pruning further boost the model performance.

In addition, there are also some other interesting observations. OPNN performs better than IPNN on both datasets. From perspective of GIN, the main difference of them is that IPNN only concern about diagonal elements of Z while OPNN take all elements into account. DCN-M outperforms all other baselines on both datasets including AutoInt and xDeepFM. The number of subspaces of the latter two models is much larger than DCM-M, which are h and 100 respectively. Note that DCN-M is the only model that apply no constraint on W. Both observations emphasize the necessity of exploring and optimizing feature interaction operations.

Another observation worth noting is that the performance of AFM ties with FwFM in both datasets. The main difference between the two models is the way they learn importance coefficient, i.e attention weights for each interactions. From the perspective of GIN, FwFM set no constraint on α while AFM relate it to corresponding feature map by a shared MLP called attention net. The result implies that, or at least in these two datasets, it may not be crucial to explicit establish connection between interactions and their importance coefficient. Parameter α can naturally serve as an indicator of the interaction importance under a fusion structure like weighted-sum. This is also consistent with our understanding of α that serve as gate to each interaction.

4.3 Ablation Study (RQ2)

We conduct several ablation experiments to investigate the effect of each components based on AOANet.

Number of layers. First the number of layers L is tested. Figure 3(a) shows that the model performance promotes when L increase from 1 to 2, however as L continues to increase, the model effect begins to decay. Also, we can see that the best setting of L is no more that 3 for all models from Table 4. This implies that feature interactions above third order may provide very few information for the sake that they are extremely sparse. We empirically suggest that appropriate number of interaction layers is 2 to 3.

Number of subspaces. Second we test the influence of number of subspaces S. From Figure 3(b) we can see the best setting

Table 5: Improvement of click through rate in online A/B test.

Day 1	Day 2	Day 3	Day 4
+14.55%	+8.70%	+9.24%	+13.71%
Day 5	**Day 6**	**Day 7**	**Average**
+9.78%	+8.92%	+11.68%	*+10.94%*

for S are 3 and 2 for Criteo and Private dataset respectively. The number of subspaces for AutoInt and xDeepFM far exceeds that of AoANET while performs relatively weaker. This once again verified the effectiveness generalized interaction operation. The number of parallelism is relatively less important when we choose suitable interaction operation.

Interaction selection. Finally we investigate the ratio of interaction selection, denoted as r. The best setting for r are 30% and 20% for Criteo and Private dataset respectively. This result shows that most of the feature interactions may bring negative effects. Similar point of view is raised in other works[1, 7], where they improved the effect by pruning some weights of models. An intuitive explanation is that, most features are sparse and even more sparse after interaction, this can easily introduce noise as well as too much parameters that can increase the difficulty of convergence. For example, interaction between birthday and age can be redundant for the equivalent meaning of the two fields, however both are commonly used in user profile.

4.4 Online Evaluation (RQ3)

AOANet is evaluated online in DiDiChuXing real production environment. It can be seen from the offline experiments that DCN-M is the best baseline. Therefore, we use DCN-M as the baseline in the online test. We deploy the model to a recommendation senario where the major business metric is click through rate(CTR). The main traffic is divided into 50/50 for experiment and control group. As shown in Table 5, the average of CTR is uplifted by 10.94% during the 7-day experiment period. The result proves the value of AOANet in practical and our proposed model now serves the main traffic of DiDiChuXing.

5 CONCLUSION

In this paper, we first provides a unified perspective for understanding the existing interaction models, that is, the model is decomposed into three phases of Projection, Interaction and Fusion. In order to solve the problem that interaction operations are restricted by domain knowledge, we propose generalized interaction operation and design GIN based on this. We further boost the performance of GIN by distinguishing importance of interactions and pruning unimportant ones. Results shows that AOANet outperforms state-of-the-arts and bring benifit in real industrial production.

REFERENCES

[1] Wei Deng, Junwei Pan, Tian Zhou, Deguang Kong, Aaron Flores, and Guang Lin. 2020. DeepLight: Deep Lightweight Feature Interactions for Accelerating CTR Predictions in Ad Serving. *arXiv e-prints* (2020), arXiv–2002.

[2] David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. 2013. Learning factored representations in a deep mixture of experts. *arXiv preprint arXiv:1312.4314* (2013).

[3] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *arXiv preprint arXiv:1703.04247* (2017).

[4] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. 355–364.

[5] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware factorization machines for CTR prediction. In *Proceedings of the 10th ACM conference on recommender systems*. 43–50.

[6] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1754–1763.

[7] Bin Liu, Chenxu Zhu, Guilin Li, Weinan Zhang, Jincai Lai, Ruiming Tang, Xiuqiang He, Zhenguo Li, and Yong Yu. 2020. Autofis: Automatic feature interaction selection in factorization models for click-through rate prediction. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2636–2645.

[8] Hanxiao Liu, Karen Simonyan, and Yiming Yang. 2018. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055* (2018).

[9] Yuanfei Luo, Mengshuo Wang, Hao Zhou, Quanming Yao, Wei-Wei Tu, Yuqiang Chen, Wenyuan Dai, and Qiang Yang. 2019. Autocross: Automatic feature crossing for tabular data in real-world applications. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1936–1945.

[10] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1930–1939.

[11] Junwei Pan, Jian Xu, Alfonso Lobos Ruiz, Wenliang Zhao, Shengjun Pan, Yu Sun, and Quan Lu. 2018. Field-weighted factorization machines for click-through rate prediction in display advertising. In *Proceedings of the 2018 World Wide Web Conference*. 1349–1357.

[12] Yanru Qu, Han Cai, Kan Ren, Weinan Zhang, Yong Yu, Ying Wen, and Jun Wang. 2016. Product-based neural networks for user response prediction. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 1149–1154.

[13] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International Conference on Data Mining*. IEEE, 995–1000.

[14] Shai Shalev-Shwartz, Ohad Shamir, and Shaked Shammah. 2017. Failures of gradient-based deep learning. In *International Conference on Machine Learning*. PMLR, 3067–3075.

[15] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538* (2017).

[16] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. Autoint: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1161–1170.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762* (2017).

[18] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*. 1–7.

[19] Ruoxi Wang, Rakesh Shivanna, Derek Z Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed H Chi. 2020. DCN-M: Improved Deep & Cross Network for Feature Cross Learning in Web-scale Learning to Rank Systems. *arXiv preprint arXiv:2008.13535* (2020).

[20] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional factorization machines: Learning the weight of feature interactions via attention networks. *arXiv preprint arXiv:1708.04617* (2017).

[21] Yi Yang, Baile Xu, Shaofeng Shen, Furao Shen, and Jian Zhao. 2020. Operation-aware Neural Networks for user response prediction. *Neural Networks* 121 (2020), 161–168.

[22] Quanming Yao, Mengshuo Wang, Yuqiang Chen, Wenyuan Dai, Yu-Feng Li, Wei-Wei Tu, Qiang Yang, and Yang Yu. 2018. Taking human out of learning applications: A survey on automated machine learning. *arXiv preprint arXiv:1810.13306* (2018).

A PROOFS

In this section, we detail the proofs of relations listed in Table 1. First we clarify some notations. The operation of generalized interaction is given by:

$$Z = (u \otimes v) \odot W \tag{44}$$

where $u \in \mathbb{R}^d$ and $u \in \mathbb{R}^d$ are interacted vectors, $W \in \mathbb{R}^{d\times d}$ and $Z \in \mathbb{R}^{d\times d}$ are weight matrix and result matrix of generalized interaction respectively, where d is the dimension of u and v. Let $\widetilde{O}$ and O represents the output of target operation and generalized interaction operation after specified fusion respectively. Proofs are as follows.

Lemma A.1. *Given $W = \text{diag}(1, \ldots, 1)$, $O = \sum_i Z_{i,i}$, then generalized interaction is equivalent to inner product.*

Proof. For inner product, we have:

$$\widetilde{O} = u^T v = \sum_i u_i v_i \tag{45}$$

Given that $W = \text{diag}(1, \ldots, 1)$, for generalized interaction operation, we have

$$\begin{aligned} Z &= (u \otimes v) \odot W \\ &= \text{diag}(u_1 v_1, \ldots, u_d v_d) \end{aligned} \tag{46}$$

Give $O = \sum_i Z_{i,i}$, then:

$$O = \sum_i u_i v_i \tag{47}$$

Note that $\widetilde{O} = O$, thus proved. □

Lemma A.2. *Given $W = \text{diag}(1, \ldots, 1)$, $O = \left[Z_{1,1}, \ldots, Z_{d,d}\right]^T$, then generalized interaction is equivalent to Hadamard product.*

Proof. For Hadamard product, we have:

$$\widetilde{O} = u \odot v = [u1v1, \ldots, u_d, v_d]^T \tag{48}$$

Given that $W = \text{diag}(1, \ldots, 1)$, for generalized interaction operation, we have:

$$\begin{aligned} Z &= (u \otimes v) \odot W \\ &= \text{diag}(u_1 v_1, \ldots, u_d v_d) \end{aligned} \tag{49}$$

Given $O = \left[Z_{1,1}, \ldots, Z_{d,d}\right]^T$, then:

$$O = [u1v1, \ldots, u_d, v_d]^T \tag{50}$$

Note that $\widetilde{O} = O$, thus proved. □

Lemma A.3. *Given $\forall i \forall j, W_{i,j} = 1$, then generalized interaction is equivalent to outer product.*

Proof. Outer product is given by:

$$\begin{aligned} \widetilde{O} &= u \otimes v \\ \widetilde{O}_{i,j} &= u_i v_j \end{aligned} \tag{51}$$

For generalized interaction operation, we have $\forall i \forall j, W_{i,j} = 1$, then

$$\begin{aligned} O_{i,j} &= u_i v_j W_{i,j} \\ &= u_i v_j \end{aligned} \tag{52}$$

Note that $\widetilde{O} = O$, thus proved. □

Lemma A.4. *Given $W = \text{diag}(\alpha, \ldots, \alpha)$, $O = \sum_{i=1}^d Z_{i,i}$, then generalized interaction is equivalent to operation defined by FwFM.*

Proof. The operation of FwFM is as follows:

$$\widetilde{O} = \alpha u^T v \tag{53}$$

For generalized interaction operation, given $W = \text{diag}(\alpha, \ldots, \alpha)$, we have:

$$\begin{aligned} Z_{i,i} &= u_i v_i W_{i,i} \\ &= \alpha u_i v_i \end{aligned} \tag{54}$$

Since $O = \sum_{i=1}^d Z_{i,i}$, we have:

$$\begin{aligned} O &= \alpha \sum_i Z_{i,i} \\ &= \alpha u^T v \end{aligned} \tag{55}$$

Note that $\widetilde{O} = O$, thus proved. □

Lemma A.5. *Given $W = \text{diag}(\alpha, \ldots, \alpha)$, $\alpha = MLP(u \odot v)$, $O = \left[Z_{1,1}, \ldots, Z_{d,d}\right]^T$, then generalized interaction is equivalent to operation defined by AFM.*

Proof. The operation of AFM is as follows:

$$\widetilde{O} = MLP(u \odot v) u \odot v \tag{56}$$

For generalized interaction operation, from Lemma A.2 and $O = \left[Z_{1,1}, \ldots, Z_{d,d}\right]^T$ we know that:

$$\begin{aligned} Z &= (u \otimes v) \odot (\alpha \, \text{diag}(1, \ldots, 1)) \\ &= \alpha((u \otimes v) \odot W) \\ O &= \alpha u \odot v \end{aligned} \tag{57}$$

Given $\alpha = MLP(u \odot v)$ we know that $\widetilde{O} = O$, thus proved. □

Lemma A.6. *Given $\forall i \forall j \forall k, W_{i,j} = W_{i,k}, O = \left[\sum_i Z_{i,1}, \ldots, \sum_i Z_{i,d}\right]^T$, then generalized interaction is equivalent to operation defined by DCN.*

Proof. The cacultion of DCN is as follows:

$$\begin{aligned} \widetilde{O} &= (u \otimes v) w^D \\ &= (u^T w) v \\ &= (\sum_i u_i w_i^D) v \end{aligned} \tag{58}$$

Given $\forall i \forall j \forall k, W_{i,j} = W_{i,k}$, without loss of generality we can make $W = [w^D, \ldots, w^D]$, then we have:

$$Z_{i,j} = u_i v_j W_{i,j} = u_i v_j w_i^D \tag{59}$$

Given $O = \left[\sum_i Z_{i,1}, \ldots, \sum_i Z_{i,d}\right]^T$, we have:

$$\begin{aligned} O &= \left[\sum_i u_i v_1 W_{i,1}, \ldots, \sum_i u_i v_d W_{i,d}\right]^T \\ &= \left[v_1 \sum_i u_i w_i^D, \ldots, v_d \sum_i u_i w_i^D\right]^T \\ &= (\sum_i u_i w_i^D) v \end{aligned} \tag{60}$$

Note that $\widetilde{O} = O$, thus proved. □

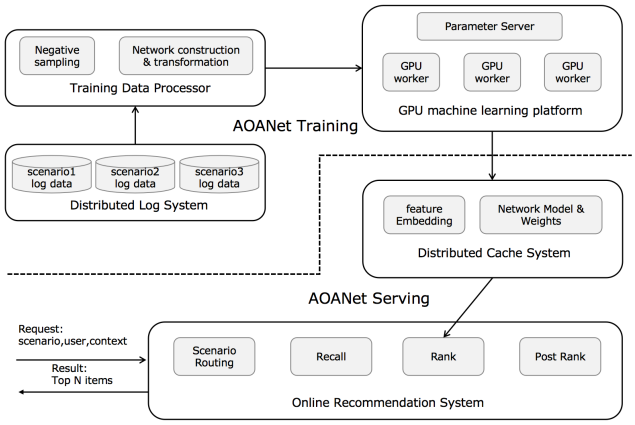


Figure 4: Launched System Architecture

Lemma A.7. *Given* $O = \left[\sum_j Z_{1,j}, \ldots, \sum_j Z_{d,j}\right]^T$, *then generalized interaction is equivalent to operation defined by DCN-M.*

Proof. The cacultion of DCN-M is as follows:

$$\begin{aligned} \widetilde{O} &= u \odot (W^D v) \\ \widetilde{O}_i &= u_i \sum_j W^D_{i,j} v_j \end{aligned} \tag{61}$$

For generalized interaction we have $O = \left[\sum_j Z_{1,j}, \ldots, \sum_j Z_{d,j}\right]^T$, then:

$$O_i = \sum_j Z_{i,j} \tag{62}$$

Since $Z_{i,j} = u_i v_j W_{i,j}$, we have:

$$O_i = \sum_j u_i v_j W_{i,j} = u_i \sum_j W_{i,j} v_j \tag{63}$$

Without loss of generality, let $W = W^D$, then we have $\widetilde{O}_i = O_i$, thus proved. □

Lemma A.8. *Suppose each origin vector is projected to three vectors named q, k and v, generalized interaction between q and* j^{th} *k is defined as* $Z^j = (q \otimes k^j) \odot W$, *where* $W = \mathrm{diag}(1, \ldots, 1)$. *Given* $O^j = (\frac{e^{\sum_i Z^j_{i,i}}}{\sum_j e^{\sum_i Z^j_{i,i}}}) v^j$, *then generalized interaction is equivalent to operation defined by AutoInt*

Proof. For AutoInt, the interaction of q w.r.t j^{th} key is given by:

$$\widetilde{O}^j = \frac{e^{q^T k_j}}{\sum_j e^{q^T k_j}} v_j \tag{64}$$

For generalized operation, $Z^j = (q \otimes k^j) \odot W$. Given $W = \mathrm{diag}(1, \ldots, 1)$ we can know from Lemma A.1 that:

$$q^T k^j = \sum_i Z^j_{i,i} \tag{65}$$

Then we have the output of generalized interaction that:

$$\begin{aligned} O^j &= (\frac{e^{\sum_i Z^j_{i,i}}}{\sum_j e^{\sum_i Z^j_{i,i}}}) v^j \\ &= \frac{e^{q^T k_j}}{\sum_j e^{q^T k_j}} v_j \end{aligned} \tag{66}$$

Thus proved. □

B ONLINE DEPLOYMENT

AOANet has been launched to production environment of DiDiChuxing. The system architecture is depicted in Figure 4.