Decompose, Then Reconstruct: A Framework of Network Structures for Click-Through Rate Prediction

Jiaming Li[1], Lang Lang[2], Zhenlong Zhu[3], Haozhao Wang[1](✉), Ruixuan Li[1](✉), and Wenchao Xu[4]

[1] School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China
{jiamingli,hz_wang,rxli}@hust.edu.cn
[2] Bytedance Inc, Beijing, China
[3] Microsoft, Beijing, China
zhenlongzhu@microsoft.com
[4] The Hong Kong Polytechnic University, Hong Kong, China
wenchao.xu@polyu.edu.hk

Abstract. Feature interaction networks are crucial for click-through rate (CTR) prediction in many applications. Extensive studies have been conducted to boost CTR accuracy by constructing effective structures of models. However, the performance of feature interaction networks is greatly influenced by the prior assumptions made by the model designer regarding its structure. Furthermore, the structures of models are highly interdependent, and launching models in different scenarios can be arduous and time-consuming. To address these limitations, we introduce a novel framework called DTR, which redefines the CTR feature interaction paradigm from a new perspective, allowing for the decoupling of its structure. Specifically, DTR first decomposes these models into individual structures and then reconstructs them within a unified model structure space, consisting of three stages: Mask, Kernel, and Compression. Each stage of DTR's exploration of a range of structures is guided by the characteristics of the dataset or the scenario. Theoretically, we prove that the structure space of DTR not only incorporates a wide range of state-of-the-art models but also provides potentials to identify better models. Experiments on two public real-world datasets demonstrate the superiority of DTR, which outperforms state-of-the-art models.

Keywords: Recommendation · CTR prediction · Feature interaction

1 Introduction

Click-through rate (CTR) prediction is critical for various applications, including recommender systems, online advertising, and product search. Mainstream CTR prediction models utilize an embedding table to map high-dimensional categorical features (e.g. *user_id* and *item_id*) to low-dimensional dense real-valued

D. Koutra et al. (Eds.): ECML PKDD 2023, LNAI 14169, pp. 422–437, 2023.
https://doi.org/10.1007/978-3-031-43412-9_25

vectors, and a feature interaction network to model interactions and make predictions. Research has focused on optimizing the feature interaction network to capture beneficial interactions and improve accuracy in CTR prediction.

Existing methods for capturing feature interactions can be divided into two categories: inner product and outer product. The inner product or Hadamard product refers to interact on the same elements of pairwise feature embedding vectors [4,6,10,12,14,16,23], relying on a prior assumption that the embedding of different features is in the same vector space. The outer product refer to interact all elements of pairwise feature embedding vectors [14,22] without any prior assumption. AOANet [7] unifies feature interaction operations by designing generalized interaction network (GIN).

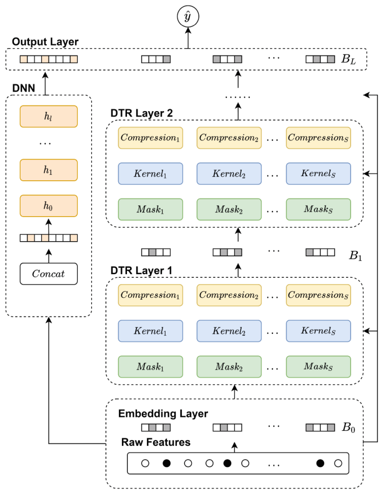
However, the performance of feature interaction networks is heavily impacted by the model designer's prior assumptions, resulting in potential bias in different scenarios. If a scenario arises that contradicts the prior assumptions, the model's performance will deteriorate significantly. For instance, if the embedding vectors of different features are not located in the same vector space, the inner product or Hadamard product will perform poorly. It is recommended to let the scenario or dataset guide the selection of appropriate structures, rather than relying solely on prior assumptions. Furthermore, the highly interdependent nature of existing model structures makes it difficult to identify specific components responsible for observed performance and the process of launching models in different scenarios can be arduous and time-consuming. One example of such challenges is when comparing the performance of models such as DCN [21] and FwFM [12], as it is unclear which aspect of the model contributes to the difference in performance. Additionally, it is typically necessary to implement DCN and FwFM separately for different scenarios due to the high coupling.

To effectively tackle these challenges, it is imperative to redefine the CTR feature interaction paradigm from a new perspective that enables decoupling of its structure. Therefore, we propose DTR, a novel framework that not only accommodates the knowledge of prior model structures but also allows better models to be explored and identified from it. Specifically, DTR first decomposes these models into individual structures and then reconstructs them within a unified model structure space, consisting of three stages: Mask, Kernel, and Compression. The mask stage masks feature interaction information to indicate which parts of the information model pay attention to. The kernel stage extracts masked feature interaction information to determine the model's capacity and degrees of freedom, such as which dimensions should share information. The compression stage aims to compress extracted feature interaction information to balance effectiveness and efficiency. Each stage of DTR's exploration of a range of structures, including existing and additional structures, is guided by the characteristics of the dataset or scenario. Theoretically, we prove that the structure space of DTR not only incorporates a wide range of state-of-the-art models but also provides potentials to identify better models. Furthermore, to inherit benefits from the mixture of experts (MOE) [3,15,17] and Transformer [5,19], we extend DTR from single channel to multiple channels to explore better models in multiple channels. Overall, DTR represents a unified model structure space that enables efficient exploration and identification of superior models. The main contributions are summarized as follows.

– We propose a novel framework called DTR, which redefines the CTR feature interaction paradigm from a new perspective, consisting of three stages: Mask, Kernel, and Compression. DTR can accommodate knowledge from existing approaches and provide potential for discovering better models.
– Theoretically, we prove that the structure space of DTR not only incorporates a wide range of state-of-the-art models but also provides potentials to identify better models.
– Experiments on two public real-world datasets for CTR prediction tasks demonstrate the superiority of DTR over state-of-the-art algorithms in terms of CTR prediction performance. In addition, ablation studies provide a deeper insight into the workings of different stages of the model and their impact on performance and other stages.

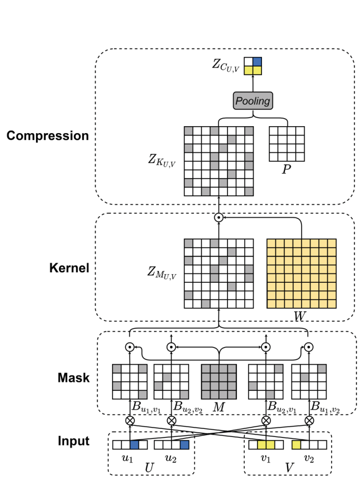
2 Related Work

In this section, we provide an overview of the related work in the literature. Existing methods for capturing feature interactions can be divided into two categories: inner product and outer product. The inner product or Hadamard product refers to the interaction of pairwise feature embedding vectors on the same elements. FM [16] is an early work in the field of recommendation, which introduces second-order feature interaction to solve the problem that logistic regression [13] cannot automatically extract the feature interaction information. Since FM only considers the second-order feature interaction, a series of improved methods based on FM specify operations have been proposed to extract the feature interaction information, such as AFM [23], FwFM [12], NFM [6], DeepFM [4] and IPNN [14]. Some of these works, such as AFM and FwFM, focus on the importance of distinguishing feature interactions. On the other hand, the outer product refers to the interaction of all elements of pairwise feature embedding vectors without any prior assumption, as in OPNN [14], DCN-V2 [22]. DCN-V2 represents a remarkable improvement over the DCN, as it eschews prior assumptions regarding both feature interaction and weight learning, resulting in a significant performance boost. AOANet [7] proposed a generalized interaction network (GIN) to overcome the limitations of artificially specified operations in feature interaction. However, as discussed in Sect. 1, existing models are designed by model designers based on prior assumptions. For instance, DCN-V2 and AOANet remove certain hypotheses from the model structure based on prior assumptions, rather than specific scenarios or datasets. This means that models may introduce biases in different scenarios and adversely impact performance.



$\hat{y}$
Output Layer
B_L
DNN
h_l
h_1
h_0
Concat
DTR Layer 2
Compression$_1$ Compression$_2$ … Compression$_S$
Kernel$_1$ Kernel$_2$ … Kernel$_S$
Mask$_1$ Mask$_2$ … Mask$_S$
B_1
DTR Layer 1
Embedding Layer
B_0
Raw Features

(a) Pipeline of DTR.



Compression
$Z_{C_{U,V}}$
Pooling
$Z_{K_{U,V}}$
P
Kernel
$Z_{M_{U,V}}$
W
Mask
B_{u_1,v_1} B_{u_2,v_2} M B_{u_2,v_1} B_{u_1,v_2}
Input
u_1 u_2 U v_1 v_2 V

(b) Structure of DTR.

Fig. 1. Framework of DTR architecture. The left figure (a) shows the pipeline of DTR. Raw features are first fed into an embedding layer to compress them into low-dimensional feature vectors. Then, they are input in parallel to DNN and DTR to extract feature interaction information, which is finally used for prediction. The right figure (b) shows the structure of a single layer of DTR, consisting of the **Mask**, **Kernel**, and **Compression** stages. In the mask stage, the feature interaction matrix is first masked to retain only the relevant information. Next, the kernel stage interacts the masked feature interaction matrix with the weight matrix to extract feature interaction information. This is followed by the Compression stage which compresses the extracted feature information using pooling, for use as input to the next layer or for prediction.

3 Methodology

3.1 Framework of DTR Architecture

Overview. To establish a framework of existing model structures, we decompose the model architecture into three distinct stages, i.e., **Mask**, **Kernel**, and **Compression**, with an integrated DNN in parallel for extracting feature interaction information, as illustrated in Fig. 1(a). Specifically, as shown in Fig. 1(b), the mask stage first masks the matrix of the feature interaction information, and then the kernel stage extracts information from the masked feature interaction matrix, which is followed by the compression stage compressing the extracted feature information.

Embedding. Features of candidate items are usually sparse, discrete, and highly dimensional in industrial online recommendation scenarios, causing that the feature interaction information is hard to extract. To handle this challenge, existing model architectures usually adopt an embedding function $E(\cdot)$ to transform the input features x_i into continuous and low-dimensional vectors $a_i \in \mathbb{R}^d$:

$$a_i = E(x_i) = V_i \cdot x_i, \quad \forall i = 1, 2, \ldots, n, \tag{1}$$

where V_i denotes the embedding matrix. For clarity, we denote $X = \{x_1, x_2, ..., x_n\}$ by the integrated input features, and $A = \{a_1, a_2, ..., a_n\}$ by the matrix of embedding vectors of all input features. For simplicity, we use $A = E(X)$ to denote the embedding operation (1) in the following.

Mask. After obtaining the embedding vectors $A = \{a_1, a_2, ..., a_n\}$, the outer product is applied to any two vectors $a_i, a_j \in A$ to extract the feature interaction information $Z \in \mathbb{R}^{nd \times nd}$, as $Z_{(i-1)d+1:id,(j-1)d+1:jd} = a_i \otimes a_j$, where $\otimes$ denotes the outer product and $Z_{(i-1)d+1:id,(j-1)d+1:jd} \in \mathbb{R}^{d \times d}$ denotes the block in the i-th row and j-th column of Z. For clarity, we denote $Z_{(i-1)d+1:id,(j-1)d+1:jd}$ by B_{ij}. Considering that inappropriate interaction may even bring interventions between features, each block B of feature interaction matrix Z is further masked as

$$B_M = B \odot M, \tag{2}$$

where B_M denotes the masked result, and $\odot$ denotes the element-wise product of two matrixes. After the mask stage, Z is transformed into masked feature interaction matrix $Z_M \in \mathbb{R}^{nd \times nd}$.

Kernel. The kernel stage extracts information from the masked feature interaction matrix Z_M, indicating the capacity and degrees of freedom of the model. Inspired by the mixture of experts (MOE) [3,15,17] and Transformer [5,19], we build the framework of DTR with multiple channels to learn the extracted information in parallel. Specifically, we consider that there is C channels with kernel parameters matrix $W^c \in \mathbb{R}^{nd \times nd}$ to learn from the masked feature interaction matrix Z_M as

$$Z_k^c = Z_M \odot W^c, \quad \forall c = 1, 2, \cdots, C. \tag{3}$$

where $Z_k^c \in \mathbb{R}^{nd \times nd}$ denotes the result obtained from the c-th channel. Similarly, we use $Z_K = \{Z_k^1, Z_k^2, \ldots, Z_k^C\}$ to denote the result set consisting of Z_k^c from all channels.

Compression. Considering that the feature information obtained from the multi-channels kernel are highly-dimensional, we adopt the compression technology to reduce the size of the features for improving the efficiency of the model. Specifically, we leverage the pooling operation $\text{Pool}(\cdot)$ with the matrix P to make compression, as:

$$Z_C = \text{Pool}(Z_K, P), \tag{4}$$

where Z_C denotes the obtained matrix after the compression stage P denotes the shape of the submatrix of Z_K which is aggregated to one scalar element in Z_C. It is worthwhile to note that the size of P is adaptively determined in the learning process. Finally, the feature interaction compression matrix Z_C is flattened and delivered to the fully connected layer for CTR prediction.

Deep Network. Like previous studies, we adopt fully connected network to extract implicit interaction information:

$$H_l = \sigma(W_l H_{l-1} + b_l), \tag{5}$$

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.

Table 1. Connection between DTR and related models. n denotes the number of features, d denotes the size of the embedding vector. Mask matrix $M \in \mathbb{R}^{d \times d}$ is a zero-one matrix, where $m_{i,j}$ denotes the element in i-th row and j-th column, $\forall i \forall j \in [d]$. All models using $\sum$ (sum compression) in CM. **Note:** Kernel weight sharing description can be seen in Appendix.

Model	Mask matrix	Kernel weight sharing	Compression matrix
IPNN	$M = \{m_{ij} = 1 \mid \forall i \forall j, i = j\}$	Intra-block Sharing	$P \in \mathbb{R}^{d \times d}, Z_{C_{i,j}} = \sum_{p=i}^{i+d} \sum_{q=j}^{j+d} Z_{K_{p,q}}, \forall i \forall j \in [n]$
OPNN	$M = \{m_{ij} = 1 \mid \forall i \forall j\}$	Intra-block Sharing	$P \in \mathbb{R}^{d \times d}, Z_{C_{i,j}} = \sum_{p=i}^{i+d} \sum_{q=j}^{j+d} Z_{K_{p,q}}, \forall i \forall j \in [n]$
FwFM	$M = \{m_{ij} = 1 \mid \forall i \forall j, i = j\}$	Intra-block Sharing	$P \in \mathbb{R}^{d \times d}, Z_{C_{i,j}} = \sum_{p=i}^{i+d} \sum_{q=j}^{j+d} Z_{K_{p,q}}, \forall i \forall j \in [n]$
AFM	$M = \{m_{ij} = 1 \mid \forall i \forall j, i = j\}$	Non-linear Intra-block Sharing	$P \in \mathbb{R}^{d \times d}, Z_{C_{i,j}} = \sum_{p=i}^{i+d} \sum_{q=j}^{j+d} Z_{K_{p,q}}, \forall i \forall j \in [n]$
DCN	$M = \{m_{ij} = 1 \mid \forall i \forall j, i = j\}$	Row Sharing	$P \in \mathbb{R}^{1 \times nd}, Z_{C_i} = \sum_{j=1}^{nd} Z_{K_{i,j}}, \forall i \in [nd]$
DCN-V2	$M = \{m_{ij} = 1 \mid \forall i \forall j\}$	No Sharing	$P \in \mathbb{R}^{1 \times nd}, Z_{C_i} = \sum_{j=1}^{nd} Z_{K_{i,j}}, \forall i \in [nd]$
xDeepFM	$M = \{m_{ij} = 1 \mid \forall i \forall j\}$	Intra-block Sharing	$P \in \mathbb{R}^{n \times n}, Z_{C_{i,j}} = \sum_{p=1}^{n} \sum_{q=1}^{n} Z_{K_{i+pd,j+pd}}, \forall i \forall j \in [d]$
AOANet	$M = \{m_{ij} = 1 \mid \forall i \forall j\}$	Intra-block Sharing & Block Element Sharing	$P \in \mathbb{R}^{nn \times d}, Z_{C_i} = \sum_{p=1}^{n} \sum_{q=1}^{nd} Z_{K_{i+pd,q}}, \forall i \in [d]$

3.2 Model Analysis

We dive into connections between DTR and related models, as shown in Table 1. Theoretically, we show that DTR can be equal to extensive known CTR feature interaction networks. Due to space constraints, we analyze xDeepFM, IPNN&OPNN here, and defer other model analyses to the Appendix[1]. For Pool$(\cdot)$, all of the CTR feature interaction networks use $\sum$ (sum compression). Since multi-layer CTR feature interaction networks are constructed recursively with the same structure, for simplicity we propose analysis on the first layer.

[1] https://github.com/GeekRaw/Decompose-Then-Reconstruct-A-Framework-of-Network-Structures-for-Click-Through-Rate-Prediction.

xDeepFM. The first layer of xDeepFM is given by:

$$X^1 = \sum_{i,j} w^s_{i,j}(X_0^{j,*} \odot X_0^{i,*}) = \sum_{i,j} w^s_{i,j}(e^i \odot e^j), \tag{6}$$

where X^1 denotes the first layer output, s is a hyper-parameter that represents the number of feature vectors. w^s is the weight matrix for s-th feature vector. The main connection lies in the number of feature vectors, where one feature vector corresponds to one channel of DTR.

Theorem 1. *The structure of DTR is equivalent to xDeepFM when its mask matrix* $M = \{m_{ij} = 1 | \forall i \forall j, i = j\}$*, kernel weight matrix* $W = \{w_{ij} | \forall i \forall j, w_{i,j} = w_{\lfloor i/d \rfloor, \lfloor j/d \rfloor}\}$*, and compression matrix* $P \in \mathbb{R}^{n \times n}, Z_{C_{i,j}} = \sum_{p=1}^{n} \sum_{q=1}^{n} Z_{K_{i+pd,j+pd}}, \forall i, j \in [d]$.

Proof. Interaction of p-th and q-th feature vector in the first layer of DTR is given by $B^{p,q} = e^p \otimes e^q$. Give mask matrix $M = \{m_{ij} = 1 | \forall i \forall j, i = j\}$, we have

$$B^{p,q}_M = B^{p,q} \odot M = diag(e^p_1 e^q_1, e^p_2 e^q_2, \ldots, e^p_d e^q_d). \tag{7}$$

Denote $W[p,q] \in \mathbb{R}^{d \times d}$ by a submatrix of W which equals to $B^{p,q}_M$. Considering that $\forall i$, $\forall j, w_{i,j} = w_{\lfloor i/d \rfloor, \lfloor j/d \rfloor}$, we conclude that each element in $W[p,q]$ shares the same value denoted as $w_{p,q}$. Thereby, we have

$$Z_K = W[p,q] \odot B^{p,q}_M = [W[p,q]]_{d \times d} \odot B^{p,q}_M = w_{p,q} B^{p,q}_M. \tag{8}$$

Given $P \in \mathbb{R}^{n \times n}, Z_{C_{i,j}} = \sum_{p=1}^{n} \sum_{q=1}^{n} Z_{K_{i+pd,j+pd}}, \forall i, j \in [d]$, combining with (7) and (8), we have

$$Z^1_C = \sum_{p,q} w_{p,q} diag(e^p_1 e^q_1, e^p_2 e^q_2, \ldots, e^p_d e^q_d). \tag{9}$$

Noting that

$$e^p \odot e^q = [e^p_1 e^q_1, e^p_2 e^q_2, \ldots, e^p_d e^q_d], \tag{10}$$

we can derive that

$$X^1 = Z^1_C [1, \ldots, 1]^T, \tag{11}$$

which completes the proof.

IPNN & OPNN. The first layer of IPNN is given by:

$$X^1 = \boldsymbol{W}^n_p \odot \boldsymbol{p} = \sum_{i=1}^{n} \sum_{j=1}^{n} \langle \boldsymbol{\theta}^i_n, \boldsymbol{\theta}^j_n \rangle \langle \boldsymbol{f}_i, \boldsymbol{f}_j \rangle = \sum_{i=1}^{n} \sum_{j=1}^{n} (\theta^n_i \odot \theta^n_j)(e_i \odot e_j), \tag{12}$$

where n denotes the number of field feature, $\theta^i_n, \theta^j_n \in \mathbb{R}^n$, $f_i, f_j \in \mathbb{R}^d$.

Theorem 2. *The structure of DTR is equivalent to IPNN when its mask matrix* $M = \{m_{ij} = 1|\forall i \forall j, i = j\}$, *kernel weight matrix* $W = \{w_{ij}|\forall i \forall j, w_{i,j} = w_{\lfloor i/d \rfloor, \lfloor j/d \rfloor}\}$, *and compression matrix* $P \in \mathbb{R}^{d\times d}, Z_{C_{i,j}} = \sum_{p=i}^{i+d}\sum_{q=j}^{j+d} Z_{K_{p,q}}, \forall i \forall j \in [n]$.

Proof. For the first layer of IPNN, denote $\theta_n^i, \theta_n^j \in \mathbb{R}^n$ as $w_ij \in \mathbb{R}^{n\times n}$, we can derive that

$$X^1 = \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(e_i \odot e_j), \tag{13}$$

Interaction of p-th and q-th feature vector in the first layer of DTR is given by $B^{p,q} = e^p \otimes e^q$. Give mask matrix $M = \{m_{ij} = 1|\forall i \forall j, i = j\}$, we have

$$B_M^{p,q} = B^{p,q} \odot M = diag(e_1^p e_1^q, e_2^p e_2^q, \dots, e_d^p e_d^q). \tag{14}$$

Denote $W[p,q] \in \mathbb{R}^{d\times d}$ by a submatrix of W which equals to $B_M^{p,q}$. Considering that $\forall i,\ \forall j, w_{i,j} = w_{\lfloor i/d \rfloor, \lfloor j/d \rfloor}$, we conclude that each element in $W[p,q]$ shares the same value denoted as $w_{p,q}$. Thereby, we have

$$Z_K = W[p,q] \odot B_M^{p,q} = [W[p,q]]_{d\times d} \odot B_M^{p,q} = w_{p,q} B_M^{p,q}. \tag{15}$$

Given $P \in \mathbb{R}^{d\times d}, Z_{C_{i,j}} = \sum_{p=i}^{i+d}\sum_{q=j}^{j+d} Z_{K_{p,q}}, \forall i \forall j \in [n]$, combining with (14) and (15), we have

$$Z_C^1 = \sum_{p,q} w_{p,q} diag(e_1^p e_1^q, e_2^p e_2^q, \dots, e_d^p e_d^q). \tag{16}$$

Noting that

$$e^p \odot e^q = [e_1^p e_1^q, e_2^p e_2^q, \dots, e_d^p e_d^q], \tag{17}$$

we can derive that

$$X^1 = Z_C^1 [1, \dots, 1]^T, \tag{18}$$

which completes the proof.

OPNN has the same kernel and compression structure as IPNN except that the mask structure is different from IPNN.

4 Evaluations

In this section, we conduct experiments on two public datasets to verify the effectiveness of the model in a real-world application environment. We aim to answer the following questions:

- RQ1: How does our proposed model perform as compared to the state-of-the-art methods?
- RQ2: How do different structures and settings influence the performance?

4.1 Evaluation Setup

Datasets. We conduct offline experiments on two real-world dataset: Criteo[2] and Avazu[3].

- **Criteo** dataset consists of user click data for displayed ads over a period of 7 d, which contains 13 numeric fields and 26 categorical fields. To process the numeric features, we apply a discretization function, which maps each value x to $\lfloor log^2(x) \rfloor$ if $x > 2$, and $x = 1$ otherwise. For the categorical features, we replace any feature that appears less than 10 times with a default "OOV" token.
- **Avazu** dataset contains mobile advertising data that spans a period of 10 d, and includes 22 feature fields that describe both user characteristics and advertisement attributes. We extract three new fields from the timestamp field: hour, weekday, and is_weekend. In addition, we handle categorical features that appear less than twice by replacing them with a default "OOV" token.

We randomly split each dataset into 80/10/10% train-test-validation splits. Criteo_x4 and Avazu_x4 dataset are used in experiments, and data preprocessing refers to FuxiCTR [25].

Table 2. Mask structure w.r.t. mask matrix. d denotes the size of the embedding vector. Mask $M \in \mathbb{R}^{d \times d}$ is a zero-one matrix, where $m_{i,j}$ denotes the element in i-th row and j-th column, $\forall i \forall j \in [d]$. M_5 denotes random mask according to r, which indicates the random mask ratio (RMR). $r = 0$ indicates all-one matrix, $r = 1$ indicates all-zero matrix.

Mask Structure	Mask Matrix	Mask Structure	Mask Matrix
M_0	$\{m_{ij} = 1\}$	M_1	$\{m_{ij} = 1 \mid i \leq j\}$
M_2	$\{m_{ij} = 1 \mid i \geq j\}$	M_3	$\{m_{ij} = 1 \mid i \neq j\}$
M_4	$\{m_{ij} = 1 \mid i + j - 1 \neq d\}$	M_6	$\{m_{ij} = 1 \mid i = j\}$

Baselines. We compared our model with eight feature interaction networks commonly used in the industry, including IPNN [14], OPNN [14], FwFM [12], AFM [23], DCN [21], DCN-V2 [22], xDeepFM [10], and AOANet [7]. Results of some models such as FmFM [18], AFN+ [2] and InterHAT [9] are not presented in this paper, because more recent models like xDeepFM [10] and DCN-V2 [22] have outperformed these methods significantly as experiments in BARS [24] shows.

Metrics. We use two widely-used metrics, Logloss and AUC, to evaluate the performance of all models. Notably, for CTR prediction task, a **0.001-level**

[2] https://www.kaggle.com/competitions/criteo-display-ad-challenge.
[3] https://www.kaggle.com/c/avazu-ctr-prediction.

Table 3. Kernel structure w.r.t. kernel weight sharing. Their constraints can be found in Appendix.

Kernel Struc	Sharing Type of Kernel Weight	Kernel Struc	Sharing Type of Kernel Weight
K_0	Full	K_1	No
K_2	Row	K_3	Intra-block
K_4	Block Row	K_5	Intra-block Element
K_6	Intra-block Dimension	K_7	Non-linear
K_8	Intra-block & Intra-block Element		

Table 4. Compression structure w.r.t. matrix. n denotes the number of features. d denotes the size of embedding vector.

Compression Struc	Compression Matrix
P_0	$P \in \mathbb{R}^{1\times 1}, Z_C = \sum_{p=1}^{nd}\sum_{p=1}^{nd} Z_{K_{p,q}}$
P_1	$P \in \mathbb{R}^{nd\times nd}, Z_{C_{i,j}} = Z_{K_{i,j}}, \forall i \forall j \in [nd]$
P_2	$P \in \mathbb{R}^{1\times nd}, Z_{C_i} = \sum_{j=1}^{nd} Z_{K_{i,j}}, \forall i \in [nd]$
P_3	$P \in \mathbb{R}^{d\times d}, Z_{C_{i,j}} = \sum_{p=i}^{i+d}\sum_{q=j}^{j+d} Z_{K_{p,q}}, \forall i \forall j \in [n]$
P_4	$P \in \mathbb{R}^{nd\times d}, Z_{C_i} = \sum_{p=1}^{nd}\sum_{q=1}^{d} Z_{K_{p,q}}, \forall i \in [n]$
P_5	$P \in \mathbb{R}^{1\times d}, Z_{C_i} = \sum_{j=1}^{d} Z_{K_{i,j}}, \forall i \in [nd], \forall j \in [n]$
P_6	$P \in \mathbb{R}^{n\times n}, Z_{C_{i,j}} = \sum_{p=1}^{n}\sum_{q=1}^{n} Z_{K_{i+pd,j+pd}}, \forall i \forall j \in [d]$
P_7	$P \in \mathbb{R}^{nn\times d}, Z_{C_i} = \sum_{p=1}^{n}\sum_{q=1}^{nd} Z_{K_{i+pd,q}}, \forall i \in [d]$

improvement is considered significant, as has been pointed out in existing literature [4,21,22].

Model Settings. The structures of DTR are shown in Table 2, Table 3, Table 4. In addition, the compression method includes three types: S_0, S_1, and S_2. S_0 corresponds to the operation of maximum compression, S_1 corresponds to average compression, and S_2 adopts accumulation compression. It is worth noting that, in order to better explore the space of model architectures, we have developed novel structures that have not been previously identified by existing models. This approach enables us to more thoroughly investigate the design space and potentially discover more effective models that were previously undiscovered.

Parameter Setting. For fair comparison, we set the same embedding size, batch size, and optimizer for all models, which are 8, 8192, and Adam Optimizer respectively. Specifically, for each dataset of different scenarios, such as Criteo and Avazu, we utilize the Block Coordinate Descent (BCD) [1,20] method in

Table 5. Model structure settings. r denotes the random mask ratio. C denotes the number of channels.

Model	Mask structure	Kernel structure	Compression structure
IPNN	M_6	K_3	P_3, S_2
OPNN	M_0	K_3	P_3, S_2
FwFM	M_6	K_3	P_3, S_2
AFM	M_6	K_7	P_3, S_2
DCN	M_6	K_2	P_2, S_2
DCN-V2	M_0	K_1	P_2, S_2
xDeepFM	M_6	K_3	P_6, S_2
AOANet	M_0	K_8	P_7, S_2
DTR$_{Criteo}$	$M_5, r = 0.1$	$K_1, C = 1$	P_3, S_2
DTR$_{Avazu}$	$M_5, r = 0.4$	$K_1, C = 2$	P_3, S_2

combination with Beam Search [8,11] to search for the optimal structure of DTR in the scenario. Moreover, we use the DTR framework to reproduce each model, as Table 5 shows. The deep component also keeps the same for all models. The numbers of hidden units for each layer are $[400, 400, 400]$ from bottom to top respectively. For other models, we take the optimal settings from original papers. We conducted three rounds of repeated experiments for each model, and then recorded the average of metrics as the final results.

4.2 Performance Comparison (RQ1)

Table 6. Overall performance on Criteo and Avazu dataset. (Logloss $\times 10^{-2}$)

Dataset	Metrics	Model									Improv.
		IPNN	OPNN	FwFM	AFM	DCN	DCN-V2	xDeepFM	AOANet	DTR	
Criteo	LogLoss	44.59	45.09	44.60	44,59	44.69	<u>44.31</u>	44.51	44.45	**44.13**	−0.18
	AUC(%)	80.57	80.02	80.56	80.56	80.46	<u>80.83</u>	80.65	80.73	**81.07**	+0.24
Avazu	LogLoss	37.79	38.14	37.76	37.79	37.80	<u>37.68</u>	37.77	37.73	**37.54**	−0.14
	AUC(%)	78.33	77.73	78.38	78.32	78.30	<u>78.51</u>	78.35	78.41	**78.75**	+0.24

In this section, we discuss the experimental results demonstrated in Table 6. The results demonstrate that DTR outperforms all baselines on Criteo and Avazu two datasets. In particular, compared to the current SOTA model DCN-V2, DTR achieves a Logloss reduction of 0.18×10^{-2} and AUC improvement of 0.24% on the Criteo dataset, as well as a Logloss reduction of 0.14×10^{-2} and AUC improvement of 0.24% on the Avazu dataset. This demonstrates that DTR can not only implement extensive known CTR feature interaction networks, but also discover the optimal structure for different stages, which identifies some novel models that have not been proposed yet.

In addition, there are also some other interesting observations. We find that the mask, kernel, and compression structures are not independent, but rather exhibit a degree of coupling, and that a local optimal structure at any stage may not be optimal for the entire framework. OPNN has the worst performance among all models, and the mask structure uses M_0, but DCN-V2 and AOANet have good performance, which also use M_0. Another observation is that the performance of AFM ties with FwFM in both datasets. From the perspective of the DTR framework, the difference between FwFM and AFM lies in the difference in the kernel structure. FwFM adopts an intra-block weight sharing structure, while AFM learns the block importance through a shared MLP called attention network on this basis. It is not critical to establish the connection kennel between feature interaction terms and their significant coefficients. Furthermore, The model performance is closely related to the setting of the kernel structure. The sharing type of kernel weight greatly affects the performance of the model, such as DTR ($r = 0$) and OPNN. The biggest difference lies in the kernel structure. There is a huge gap in its performance, with an improvement of 1.05% on Criteo dataset and 1.02% on the Avazu dataset in terms of AUC.

4.3 Ablation Study (RQ2)

We conducted several ablation experiments to investigate the effectiveness of each stage in the DTR framework. During the ablation experiments, we set other parameters as initial settings to reduce interference and better explore the performance of each structure of DTR.

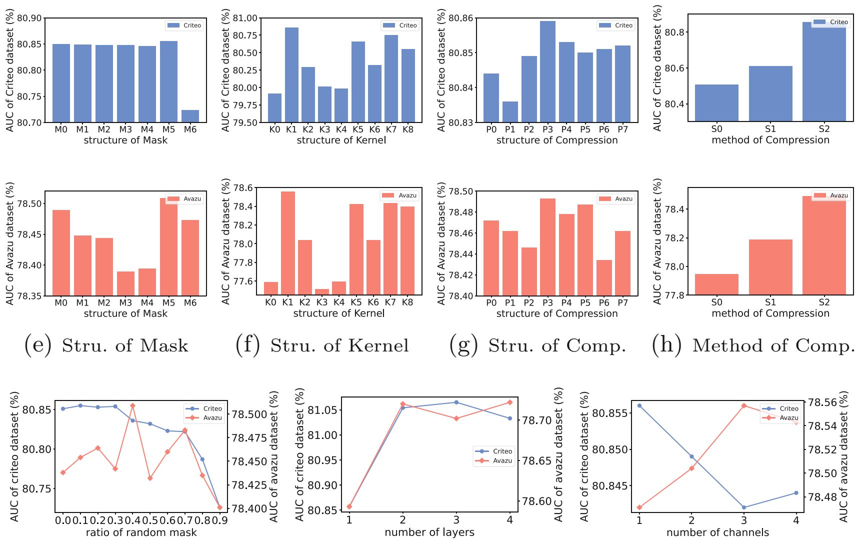


Fig. 2. Ablation study of model setting on the performance of AUC.

Structure of Mask. Table 2 presents the mask structures used in our experiments, and the results are shown in Fig. 2(e). For the random mask structure M_5, we experiment with different ratios and selected the final optimal result for M_5. The experimental results for various random mask ratios are shown in Fig. 2(i). The results indicate that, except M_0 and M_5 without any assumptions, have relatively good and stable performance on both datasets, while other mask structures vary significantly. Specifically, M_6 performs poorly in the Criteo dataset but performs well in Avazu. This reinforces our previous statement that prior assumptions regarding the model structure can significantly impact performance in different scenarios. Additionally, this further highlights the importance of the DTR framework, which can optimize the mask structure for each CTR prediction task to achieve optimal performance. Among all mask structures, M_5 performs the optimal in both datasets. Moreover, Fig. 2(i) indicates that the optimal random mask ratios are 10% and 40% for the Criteo and Avazu datasets, respectively. These results suggest that there exists some redundancy in the feature interaction information and that appropriate random mask can improve the performance. However, excessive masking can significantly affect the expression of feature interaction information, resulting in performance degradation.

Structure of Kernel. The descriptions of kernel structures are shown in Table 3, and the experimental results are presented in Fig. 2(f). Among all kernel structures, the optimal kernel structure is K_1, indicating that the interaction between any dimension of any feature embedding vector in the feature interaction matrix is different, and that the kernel structure achieves optimal performance without any assumptions. Furthermore, the size of the kernel weight matrix may also affect the performance. For K_0, the entire feature interaction matrix shares the same weight, resulting in the worst performance. On the other hand, K_1 does not share the weight of the feature interaction matrix, which leads to the most weight learned and optimal performance achieved. It is worth noting that different kernel weight sharing structures have a significant impact on performance. For instance, even though K_3 learns much more weight than K_5 and K_6, its performance is far worse than K_5 and K_6 due to the different weight sharing of structures. Specifically, K_3 uses the same weight for each block, while K_5 and K_6 adopt the same weight for all blocks of the same element and dimension, respectively. Consequently, different feature interaction information between different dimensions of feature embedding vectors and small interaction difference between different feature embedding vectors result. Overall, this experiment highlights the importance of selecting the appropriate kernel weight sharing structure for achieving optimal performance.

Structure of Compression. Table 4 describes the compression structures used in the experiment, while the experimental results are presented in Fig. 2(g).

The results demonstrate that P_3, which uses intra-block compression, achieves the optimal performance in both datasets. This suggests that information of the same feature interaction block can more effectively express the entire feature interaction matrix information. Furthermore, P_1 adopts full compression and has the worst performance, followed by P_0 which does not compress to any level and retains the entire matrix. Therefore, it is crucial to use an appropriate compression structure when compressing the feature interaction matrix.

Moreover, we explored a novel question that previous research has not addressed: which compression method works better. Figure 2(h) presents the experimental results. Interestingly, the sum compression method, which is commonly used, achieves the optimal performance. It is noteworthy that the sum compression method is a scalar multiple of the average compression method, with the scalar value equal to the length of the pairwise feature embedding vectors product. However, the average compression method performs much worse than the sum compression method, which may be due to the fact that it reduces the amount of feature interaction term information and the difference between different feature interaction terms, thus affecting the expression of the original feature interaction term information.

Number of Channels. Figure 2(k) shows that the optimal setting for C is 1 and 3 for Criteo and Avazu dataset, respectively. We can know that the large dataset contains more feature information, so the number of channels is relatively less important and does not necessarily boost performance. When working with a small dataset, increasing parallelism can lead to performance improvements.

Number of Layers. Figure 2(j) shows that the model performance promotes when L increases from 1 to 2. However, as L continues to increase, the performance improves slightly and even starts to decay. In addition, Table 5 shows the optimal setting of L does not exceed 3, which implies that feature interactions above third order may provide very little information for the sake that they are extremely sparse.

5 Conclusions

In this paper, we propose a unified framework called DTR that explores and optimizes the model structure for CTR prediction tasks. DTR decomposes these models into individual structures and then reconstructs them within a unified model structure space, consisting of three stages: Mask, Kernel, and Compression. Theoretically, we have demonstrated that the structure space of DTR not only incorporates a wide range of state-of-the-art models but also provides potentials to identify better models. Experimental results on two public real-world datasets confirm the superiority of DTR over state-of-the-art algorithms.

Acknowledgements. This work is supported by National Natural Science Foundation of China under grants 62206102, U1836204, U1936108, and Science and Technology Support Program of Hubei Province under grant 2022BAA046.

Ethical Statement. This research work on feature interaction networks and click-through rate (CTR) prediction was conducted with a focus on developing a novel framework for improving the accuracy of CTR prediction models. The research work was conducted with adherence to ethical principles and standards of research integrity. The research does not involve any human subjects or any sensitive data, and all the data are evaluated from the most mainstream public datasets in CTR prediction task, so no ethical approval is required. The research work was conducted with the aim of advancing the state-of-the-art in CTR prediction models, and the results of this study can have potential implications for businesses and industries that rely on CTR prediction models. The authors acknowledge the contributions of prior research in this area and have given appropriate credit to previous works. The authors have also disclosed any potential conflicts of interest related to this research work. The research work was conducted with transparency and openness, and has been peer-reviewed and vetted.

References

1. Birgin, E., Martínez, J.: Block coordinate descent for smooth nonconvex constrained minimization. Comput. Optim. Appl. **83**(1), 1–27 (2022)
2. Cheng, W., Shen, Y., Huang, L.: Adaptive factorization network: learning adaptive-order feature interactions. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 3609–3616 (2020)
3. Du, N., et al.: GLaM: efficient scaling of language models with mixture-of-experts. In: International Conference on Machine Learning, pp. 5547–5569. PMLR (2022)
4. Guo, H., TANG, R., Ye, Y., Li, Z., He, X.: DeepFM: a factorization-machine based neural network for CTR prediction. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17, pp. 1725–1731 (2017). https://doi.org/10.24963/ijcai.2017/239
5. Han, K., Xiao, A., Wu, E., Guo, J., Xu, C., Wang, Y.: Transformer in transformer. Adv. Neural. Inf. Process. Syst. **34**, 15908–15919 (2021)
6. He, X., Chua, T.S.: Neural factorization machines for sparse predictive analytics. In: Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval, pp. 355–364 (2017)
7. Lang, L., Zhu, Z., Liu, X., Zhao, J., Xu, J., Shan, M.: Architecture and operation adaptive network for online recommendations. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 3139–3149 (2021)
8. Lemons, S., López, C.L., Holte, R.C., Ruml, W.: Beam search: faster and monotonic. In: Proceedings of the International Conference on Automated Planning and Scheduling, vol. 32, pp. 222–230 (2022)
9. Li, Z., Cheng, W., Chen, Y., Chen, H., Wang, W.: Interpretable click-through rate prediction through hierarchical attention. In: Proceedings of the 13th International Conference on Web Search and Data Mining, pp. 313–321 (2020)
10. Lian, J., Zhou, X., Zhang, F., Chen, Z., Xie, X., Sun, G.: xDeepFM: combining explicit and implicit feature interactions for recommender systems. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1754–1763 (2018)

11. Libralesso, L., Focke, P.A., Secardin, A., Jost, V.: Iterative beam search algorithms for the permutation flowshop. Eur. J. Oper. Res. **301**(1), 217–234 (2022)
12. Pan, J., Xu, J., Ruiz, A.L., Zhao, W., Pan, S., Sun, Y., Lu, Q.: Field-weighted factorization machines for click-through rate prediction in display advertising. In: Proceedings of the 2018 World Wide Web Conference, pp. 1349–1357 (2018)
13. Peng, C.Y.J., Lee, K.L., Ingersoll, G.M.: An introduction to logistic regression analysis and reporting. J. Educ. Res. **96**(1), 3–14 (2002)
14. Qu, Y., Cai, H., Ren, K., Zhang, W., Yu, Y., Wen, Y., Wang, J.: Product-based neural networks for user response prediction. In: 2016 IEEE 16th International Conference on Data Mining (ICDM), pp. 1149–1154. IEEE (2016)
15. Rajbhandari, S., et al.: DeepSpeed-MoE: advancing mixture-of-experts inference and training to power next-generation AI scale. In: International Conference on Machine Learning, pp. 18332–18346. PMLR (2022)
16. Rendle, S., Gantner, Z., Freudenthaler, C., Schmidt-Thieme, L.: Fast context-aware recommendations with factorization machines. In: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 635–644 (2011)
17. Riquelme, C., et al.: Scaling vision with sparse mixture of experts. Adv. Neural. Inf. Process. Syst. **34**, 8583–8595 (2021)
18. Sun, Y., Pan, J., Zhang, A., Flores, A.: FM2: field-matrixed factorization machines for recommender systems. In: Proceedings of the Web Conference 2021, pp. 2828–2837 (2021)
19. Tay, Y., Dehghani, M., Bahri, D., Metzler, D.: Efficient transformers: a survey. ACM Comput. Surv. **55**(6), 1–28 (2022)
20. Tseng, P.: Convergence of a block coordinate descent method for nondifferentiable minimization. J. Optim. Theory Appl. **109**(3), 475 (2001)
21. Wang, R., Fu, B., Fu, G., Wang, M.: Deep & cross network for ad click predictions. In: Proceedings of the ADKDD'17, pp. 1–7 (2017)
22. Wang, R., Shivanna, R., Cheng, D., Jain, S., Lin, D., Hong, L., Chi, E.: DCN V2: improved deep & cross network and practical lessons for web-scale learning to rank systems. In: Proceedings of the web conference 2021, pp. 1785–1797 (2021)
23. Xiao, J., Ye, H., He, X., Zhang, H., Wu, F., Chua, T.S.: Attentional factorization machines: learning the weight of feature interactions via attention networks. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17, pp. 3119–3125 (2017). https://doi.org/10.24963/ijcai.2017/435
24. Zhu, J., et al.: BARS: towards open benchmarking for recommender systems. In: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'22) (2022)
25. Zhu, J., Liu, J., Yang, S., Zhang, Q., He, X.: Open benchmarking for click-through rate prediction. In: The 30th ACM International Conference on Information and Knowledge Management (CIKM'21), pp. 2759–2769 (2021)