RESEARCH-ARTICLE

CETN: Contrast-enhanced Through Network for Click-Through Rate Prediction

HONGHAO LI, Anhui University, Hefei, Anhui, China

LEI SANG, Anhui University, Hefei, Anhui, China

YI ZHANG, Anhui University, Hefei, Anhui, China

XUYUN ZHANG, Macquarie University, Sydney, NSW, Australia

YIWEN ZHANG, Anhui University, Hefei, Anhui, China

Published: 27 November 2024
Online AM: 12 August 2024
Accepted: 06 August 2024
Revised: 28 June 2024
Received: 21 March 2024

CETN: Contrast-enhanced Through Network for Click-Through Rate Prediction

HONGHAO LI, LEI SANG, and YI ZHANG, Anhui University, Hefei, China
XUYUN ZHANG, Macquarie University, Macquarie Park, Australia
YIWEN ZHANG, Anhui University, Hefei, China

Click-through rate (CTR) prediction is a crucial task in personalized information retrievals, such as industrial recommender systems, online advertising, and web search. Most existing CTR Prediction models utilize explicit feature interactions to overcome the performance bottleneck of implicit feature interactions. Hence, deep CTR models based on parallel structures (e.g., DCN, FinalMLP, xDeepFM) have been proposed to obtain joint information from different semantic spaces. However, these parallel subcomponents lack effective supervision and communication signals, making it challenging to efficiently capture valuable multi-views feature interaction information in different semantic spaces. To address these issues, we propose a simple yet effective novel CTR model: Contrast-enhanced Through Network (CETN). Drawing inspiration from sociology, CETN leverages the complementary nature of diversity and homogeneity to guide the model in acquiring higher-quality feature interaction information. Specifically, CETN employs product-based feature interactions and the augmentation (perturbation) concept from contrastive learning to segment different semantic spaces, each with distinct activation functions. This improves diversity in the feature interaction information captured by the model. Additionally, we introduce self-supervised signals and through connection within each semantic space to ensure the homogeneity of the captured feature interaction information. The experiments conducted on four real datasets demonstrate that our model consistently outperforms twenty baseline models in terms of AUC and Logloss.

CCS Concepts: • **Information systems** → **Information retrieval**; **Recommender systems**;

Additional Key Words and Phrases: Contrastive Learning, Feature Interaction, Neural Network, Recommender Systems, CTR Prediction

ACM Reference format:
Honghao Li, Lei Sang, Yi Zhang, Xuyun Zhang, and Yiwen Zhang. 2024. CETN: Contrast-enhanced Through Network for Click-Through Rate Prediction. *ACM Trans. Inf. Syst.* 43, 1, Article 4 (November 2024), 34 pages. https://doi.org/10.1145/3688571

This work was supported by the National Natural Science Foundation of China (No. 62272001, No. 62206002), the Anhui Provincial Natural Science Foundation (2208085QF195), and the Hefei Key Common Technology Project (GJ2022GX15).
Authors' Contact Information: Honghao Li, Anhui University, Hefei, China; e-mail: salmon1802li@gmail.com; Lei Sang, Anhui University, Hefei, China; e-mail: sanglei@ahu.edu.cn; Yi Zhang, Anhui University, Hefei, China; e-mail: zhangyi.ahu@gmail.com; Xuyun Zhang, Macquarie University, Macquarie Park, Australia; e-mail: xuyun.zhang@mq.edu.au; Yiwen Zhang (corresponding author), Anhui University, Hefei, China; e-mail: zhangyiwen@ahu.edu.cn.

ACM 1558-2868/2024/11-ART4
https://doi.org/10.1145/3688571

1 Introduction

Accurately predicting user responses to items (e.g., products, movies, and advertisements) under certain contexts (e.g., websites and apps) plays a pivotal role in commercial personalized **information retrieval (IR)** scenarios [6, 12, 39, 61, 77] and recommender system [5, 10, 45, 69, 74]. The user's probability of clicking on an item serves as an intuitive evaluation metric, widely applied across various downstream scenarios, such as online advertising [9, 14, 67], recommender systems [20, 49, 63], and web search [1, 13]. Its primary objective is to assess the probability of users clicking on recommended items within a system. On the one hand, the revenue of the vast majority of commercial IR systems is closely tied to user **click-through rates (CTR)**, making the accuracy of CTR prediction directly impact the profitability of these systems. On the other hand, user satisfaction is closely linked to the performance metrics of recommender system. A well-performing CTR prediction model aids in swiftly discerning user interests, thereby enhancing the user experience.

Capturing effective **feature interactions (FI)** is one of the crucial strategies to enhance the performance of CTR models. In the initial period, researchers employ explicit product-based FI to build models [24, 46, 47, 49], aiming to mitigate data sparsity issues. However, due to model complexity constraints, they could only capture low-order FI in practical applications [31, 41, 49, 60]. With the recent proliferation of deep learning, deep learning-based CTR models have emerged, where the **multi-layer perceptron (MLP)** is widely employed [16, 39, 46, 60–62]. MLP implicitly captures high-order FI information. Nevertheless, some existing work has pointed out that the expressive power of a single MLP is inefficient for capturing FI information, and it may even struggle to learn simple inner-product operations [50, 52]. To overcome the performance bottleneck of MLPs in capturing FI information, researchers have attempted to combine explicit product-based FI with the ability of MLPs to implicitly capture high-order FI, leading to significant performance improvements [80]. Depending on the integration approach, this can be divided into two structures, namely the parallel structure and the stacked structure [3, 16, 39, 46]. The stacked structure attempts to first perform explicit product-based FI on the features and then feed them as input to an MLP [46, 61, 62, 72]. In contrast, the parallel structure jointly trains explicit and implicit components of FI in a parallel manner, utilizing a fusion layer to obtain joint information from different semantic spaces [16, 33, 39, 61, 62]. However, many existing CTR models suffer from three main issues, even if they use the two structures mentioned above, still result in a narrow semantic space available for capturing information, and are unable to efficiently capture diversiform FI information in different semantic spaces, as detailed follow.

Lackluster Segmentation in the Semantic Space. Most CTR models with parallel structures share embedding layers among their parallel subcomponents and directly use concatenated embeddings as inputs for the model [16, 33, 60, 61]. This results in the information from different semantic spaces remaining similar, and relying solely on the information capture capacity of a single subcomponent in an attempt to improve diversity in the obtained information is undoubtedly inefficient. For models that employ parallel subcomponents with the same structure [39, 59, 62], the problem of semantic space segmentation becomes even more pronounced. As we observe in Table 1, although TwinDNN shows a slight performance improvement over DNN, it is significantly worse than "TwinDNN + seg." Without capturing FI information in different semantic spaces differently, it often leads to sub-optimal performance and fails to guarantee the diversity of FI information.

Feeble Supervisory Signals in Multi-Semantic Space. CTR models based on parallel structures often require a fusion layer to aggregate the FI information captured by various subcomponents to obtain the final prediction [33, 39, 60, 62], and train only using 1-bit click signals [34]. Therefore, prior to information aggregation, there is a lack of communication and effective supervisory signals

Table 1. Performance of Different Methods on the Avazu, MovieLens, and Frappe Datasets

Method	Avazu		MovieLens		Frappe	
	Logloss↓	AUC(%)↑	Logloss↓	AUC(%)↑	Logloss↓	AUC(%)↑
DNN (base)	0.372142	0.792717	0.208941	0.969276	0.169329	0.980640
TwinDNN	0.372087	0.792863	0.210060	0.969011	0.165159	0.982706
TwinDNN + seg	**0.372029**	**0.793692**	**0.196211**	**0.971422**	**0.160557**	**0.983089**
TwinDNN + div	0.372352	0.792405	0.218850	0.965849	0.168504	0.978737

TwinDNN represents two identical MLPs sharing embeddings. "TwinDNN + seg" indicates TwinDNN with non-shared embeddings. "TwinDNN + div" uses cosine loss to force the model to pursue extreme information diversity. The bold in the table is to emphasize best performance.

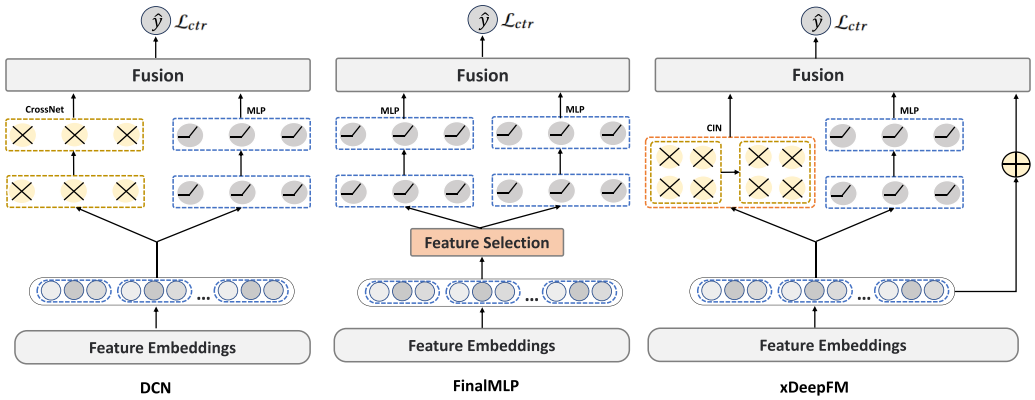


Fig. 1. The architecture of three strong baseline models with parallel structure: DCN, FinalMLP, and xDeepFM.

between the individual subcomponents. This may hinder the model from capturing high-quality, non-redundant information and diminish the diversity of the information captured.

Excessive Noise in a Multi-Semantic Space. While we pursue diversity within semantic spaces, it inevitably leads to the challenge of "being too different." Empirically, as shown by the performance difference between TwinDNN and "TwinDNN + div" in Table 1, we observe that if different parallel subcomponents exclusively capture entirely distinct information, it results in the model aggregating a substantial amount of noise signals in the fusion layer, thereby diminishing model performance. Hence, ensuring homogeneity in the information captured by individual subcomponents is crucial.

To explain the three challenges we proposed in more detail, as illustrated in Figure 1, we present three strong baseline models as examples. In the case of DCN [60], its two subcomponents share an embedding layer. It explicitly captures low-order FI using CrossNet and implicitly captures high-order FI through an MLP. In the fusion layer, a simple summation is used to aggregate information from two semantic spaces. However, this straightforward information capture approach fails to address the three challenges we have identified. FinalMLP [39] begins by segmenting two semantic spaces using a feature selection layer, ensuring diversity in the information contained within these spaces. It then employs two MLPs to implicitly capture FI information within each semantic space. Finally, an explicit product operation is introduced in the fusion layer. This model resolves the issue of semantic space segmentation, ensuring information diversity in these spaces and achieving outstanding performance. However, it lacks effective supervisory signals within each semantic space, making it difficult to ensure that MLP can capture diverse information in semantic space

through a self-supervised manner. It also lacks differentiation between primary and auxiliary semantic spaces, failing to guarantee the homogeneity of captured information across the model. Similarly, xDeepFM [33] divides data into three semantic spaces and models FI in a more complex manner to ensure diversity in the captured FI information. However, it faces challenges similar to DCN, which cannot guarantee the diversity and homogeneity of the captured information.

To address the widespread issues among parallel subcomponents, we introduce a novel CTR prediction model in this paper, named **Contrast-enhanced Through Network (CETN)**. Specifically, to improve diversity in the captured FI information, we employ the semantic space using the product & perturbation paradigm, then utilize multiple Key-Value Blocks with different activation functions as parallel subcomponents of the model. Furthermore, to address issues stemming from diversity, we propose a **through connection (TC)**, which can be viewed as a horizontal variant of residual connection [18], to ensure homogeneity in the information captured by individual subcomponents. Before the fusion layer, we introduce **Denominator-only InfoNCE (Do-InfoNCE)** and cosine loss to self-supervised learning processes within each semantic space. Extensive experiments conduct on four real-world datasets demonstrate that this simple yet efficient model consistently outperforms twenty baseline models in terms of both AUC and Logloss.

In summary, the primary contributions of this work can be summarized as follows:

- —By analyzing the existing CTR models with parallel structures, we summarize three challenges common to these models, i.e., lackluster segmentation, feeble supervisory signals, and excessive noise. To address these three challenges, we introduce the complementary principles of diversity and homogeneity from sociology, leading to the proposal of a new model, CETN.
- —To enhance the diversity of information captured by the model, we further segment the semantic space, utilize Key-Value Blocks with different activation functions as subcomponents, and introduce Do-InfoNCE to provide auxiliary signals for capturing FI.
- —To ensure the homogeneity of the information captured by the subcomponents, we introduced element-wise TC and cosine loss as communication bridges between multiple semantic spaces.
- —We conduct fair and extensive experiments on four real public benchmark datasets to evaluate the performance of our proposed model. Based on the experimental results, we demonstrate that CETN consistently outperforms twenty **state-of-the-art (SOTA)** baseline models.

2 Preliminaries

2.1 Problem Definition

The CTR prediction task aims to enhance the profitability and user satisfaction of commercial recommendation systems by predicting the likelihood of a current user clicking on items recommended by the system. Therefore, about the CTR prediction tasks based on FI, we can define it formally as follows.

Definition 1 (CTR Prediction Based on Feature Interactions). For a given user u and item v, three groups of features are extracted from them:

- —*User profiles* (x_p): age, gender, occupation, etc.
- —*Item attributes* (x_a): brand, price, category, etc.
- —*Context* (x_c): timestamp, device, position, etc.

As shown in Table 2, in general, x_p, x_a, and x_c are multi-field categorical data and are represented using one-hot encoding, and then we utilize an embedding layer to transform them into low-dimensional dense vectors: $\mathbf{e}_i = E_i x_i$, where $E_i \in \mathbb{R}^{d \times s_i}$ and s_i separately indicate the embedding matrix and the vocabulary size for the ith field, d represents the embedding dimension. Analogously,

Table 2. An Example of Multi-Field Categorical Data

Click	User (x_p)			Item (x_a)			Context (x_c)		
	Age	Gender	Occupation	Brand	Price	Category	Timestamp	Device	Position
1	30	Female	engineer	Armani	200	clothing	2022/8/19	Huawei	London
0	20	Male	student	Vuitton	1,000	clothing	2022/11/3	Apple	New York
1	40	Male	engineer	Gucci	600	clothing	2022/3/16	Samsung	Hong Kong
0	20	Female	student	Dior	2,000	cosmetic	2022/5/26	Lenovo	Tokyo
Number	10	2	500	1,000	5,000	1,000	365	1,000	1,000

for numerical data (e.g., age, price), we typically start by discretizing them into categorical data using bucketing method[1] [58, 80] and then transform them into embedding vectors using embedding methods for categorical features. Subsequently, we can obtain the result representation of the embedding layer: $\mathbf{E} = \left[\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_f\right]$, where f denotes the number of fields.

Variable $y \in \{0, 1\}$ is an associated label for user click behavior:

$$y = \begin{cases} 1, & u \text{ has clicked } v, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Let $o^{(1)}$ denotes $\mathbf{E}$ (namely, the first-order FI representation). If we need to obtain higher-order FI we can use the following form:

$$o^{(i+1)} = g(o^{(1)}, o^{(i)}), \tag{2}$$

where $g(\cdot)$ denotes an interaction function and $o^{(i)}$ denotes the ith order FI. Therefore, for a simple CTR model, the common framework is formulated as:

$$\hat{y} = \text{Model}(u, v, \{o^{(1)}, o^{(2)}, \ldots, o^{(n)}\}; \theta), \tag{3}$$

where Model is the CTR model with parameters θ, and o^n represents FI of appropriate order.

Definition 2 (Semantic Space in CTR). Consider an enhanced embedding as an additional auxiliary semantic space, and the original embedding as the original semantic space. For any semantic space, we can define it as follows:

$$space_i = \text{segment}(\mathbf{E}), \tag{4}$$

where the segment represents various enhancement or no operations, such as gating mechanisms, masking mechanisms, noise, and so on.

Definition 3 (Capturing Feature Interactions from Multi-Semantic Spaces). Learning FI in only one semantic space can result in a limited range of available FI information. Therefore, we need to expand the semantic space further, as shown in Figure 2. We further refine the basic CTR model, enabling its formalization as follows:

$$sub_i = \text{subcomponent}_i(u, v, \{o^{(1)}, o^{(2)}, \ldots, o^{(s)}\}; \theta_i), \tag{5}$$

where sub_i denotes the FI information implicit in the ith semantic space, and o^s, θ_i denote the appropriate FI order and parameters in the current semantic space, respectively. After that, we utilize the fusion layer to further aggregate information from each semantic space, yielding the ultimate prediction.

$$\hat{y} = \text{fusion}(sub_1, sub_2, \ldots, sub_m), \tag{6}$$

[1] https://www.csie.ntu.edu.tw/r01922136/kaggle-2014-criteo.pdf

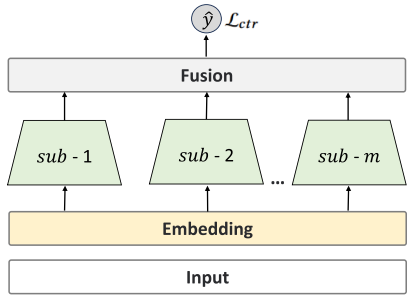
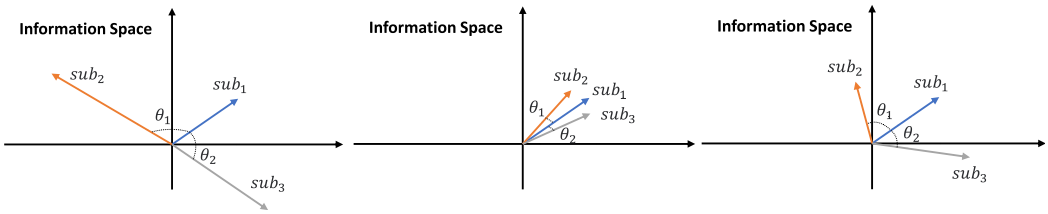


Fig. 2. The primary backbone structures of common CTR prediction models.



(a) Excessive diversity (b) Excessive homogeneity (c) Achieving a certain balance.

Fig. 3. An illustration of the diversity and homogeneity in $\mathbb{R}^2$.

where m represents the number of suitable semantic spaces, and fusion denotes the aggregate function.

Definition 4 (Diversity and Homogeneity). The concepts of diversity and homogeneity have garnered attention in sociology [2, 36]. Sociologists have delved deeply into these two concepts, seeking to comprehend different aspects and dimensions within social structures. Diversity underscores differences among individuals or groups, emphasizing the uniqueness of each member. Conversely, homogeneity emphasizes commonality and similarity, highlighting shared features and commonalities among social members. The dynamic balance between these complementary attributes constitutes diverse and interconnected social systems, aiding sociologists in gaining a better understanding of the complexity of society [28, 44].

Motivated by the insights from the aforementioned studies, we introduce these two concepts to assist the model in capturing better FI information. Specifically, we can incorporate additional self-supervisory signals in the fusion layer to balance both the diversity and homogeneity of information across various semantic spaces. To illustrate the benefits of this approach more intuitively, we can illustrate these two concepts in $\mathbb{R}^2$. In Figure 3, if the model captures FI information that is too dissimilar across different semantic spaces, it results in the scenario depicted in Figure 3(a). Distinct subspaces acquire significantly different information, leading to excessively large angles between θ_1 and θ_2. Conversely, as shown in Figure 3(b), if the model captures overly similar information, redundant information is generated, resulting in suboptimal performance. Figure 3(c) represents a balanced scenario. Assuming sub_1 is the semantic space with the highest information quality, we can use self-supervisory signals to compel sub_2 and sub_3 to converge towards sub_1 (i.e., reducing the angles θ_1 and θ_2), while preserving their individual information to some extent.

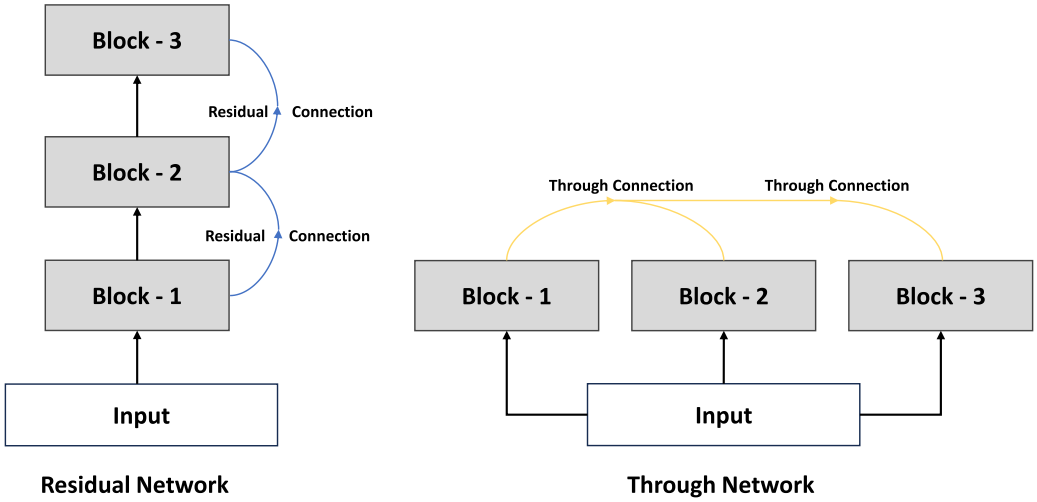


Fig. 4. The relationship between the TC and Residual Connection.

2.2 Relationship between the TC and Residual Connection

In previous research represented by residual networks, they use vertical, element-wise addition skip connections to pass information from earlier layers to later layers in the neural network, preventing the model from experiencing the degradation phenomenon. This residual connection, as shown in Figure 4, can be simply defined as:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}) + \mathbf{x}, \tag{7}$$

in this formula, $\mathbf{x}$ represents the input in the network, $\mathcal{F}$ is the transformation applied to $\mathbf{x}$ by the layer, and $\mathbf{y}$ is the output. The key idea here is that the output $\mathbf{y}$ is the sum of the input $\mathbf{x}$ and its residual (i.e., the difference between $\mathbf{x}$ and $\mathbf{y}$). Next, by horizontally generalizing it, we arrive at the fundamental definition of TC:

$$\begin{aligned} \mathbf{y}_{\text{Block}_1} &= \mathcal{F}_{\text{Block}_1}(\mathbf{x}), \\ \mathbf{y}_{\text{Block}_i} &= \mathcal{F}_{\text{Block}_i}(\mathbf{x}) + \mathbf{y}_{\text{Block}_1}, \end{aligned} \tag{8}$$

where we first obtain the output $\mathbf{y}_{\text{Block}_1}$ through the $\mathcal{F}_{\text{Block}_1}$, then add the output $\mathbf{y}_{\text{Block}_1}$ to the output of the other block element-wise to get $\mathbf{y}_{\text{Block}_i}$. In fact, the TC can be regarded as a horizontal variant of the residual connections. We can further redefine the residual connection to the following form:

$$\begin{aligned} \mathbf{y}_{\text{Block}_1} &= \mathcal{F}_{\text{Block}_1}(\mathbf{x}), \\ \mathbf{y}_{\text{Block}_i} &= \mathcal{F}_{\text{Block}_i}(\mathbf{y}_{\text{Block}_{i-1}}) + \mathbf{y}_{\text{Block}_{i-1}}, \end{aligned} \tag{9}$$

comparing Equations (8) and (9), we observe that the difference between the TC and the residual connection lies in the following: the TC learns the difference between the output of a fixed block and those of other blocks, thereby horizontally expanding the neural network; whereas the residual connection learns the difference between the outputs of consecutive blocks, enabling the neural network to deepen.

2.3 Contrastive Learning

Contrastive learning (CL) is one of the mainstream methods in self-supervised learning [64, 68, 70, 71], which first *augments* the input samples to obtain representations from multiple perspectives, then tries to encourage consistency between pairs of positive samples (*alignment*) and minimize

the agreement between pairs of negative samples (*uniformity*), ultimately enhancing the model's performance. InfoNCE [40] loss plays a pivotal role in CL, in which the CL loss is:

$$\mathcal{L}_{cl} = \sum_{i \in \mathcal{B}} -\log \frac{\exp\left(\text{sim}(\mathbf{x}'_i, \mathbf{x}''_i)/\tau\right)}{\sum_{j \in \mathcal{B}} \exp\left(\text{sim}(\mathbf{x}'_i, \mathbf{x}''_j)/\tau\right)}, \tag{10}$$

where $\mathcal{B}$ is the batch size, x', x'' are input instance representations learned from two different augmentations, τ represents the temperature coefficient, $\text{sim}(\cdot, \cdot)$ is employed to evaluate the mutual information score between them. However, as pointed out in some previous works [17, 70, 71], the augmentation and alignment operations on positive samples are not always effective. Therefore, we redefine a contrastive loss more suitable for multi-semantic space learning, which will be presented in Section 3.2.

3 CETN

In this section, we will introduce the CETN model. We will approach it from three perspectives and describe the architecture of the CETN model in detail.

3.1 How to Simply and Efficiently Capture Feature Interactions from Multi-Semantic Spaces

Most existing parallel-structured CTR models attempt to capture FI information from multi-semantic spaces using *subcomponents*, such as DCN [60], DCNv2 [61], AutoInt [54], xDeepFM [33], and EDCN [3]. From their model architectures, we can learn that if we want to efficiently capture FI information from multi-semantic spaces, we need multiple subcomponents to model FI in parallel. However, as we pointed out in Section 1, relying only on the ability of subcomponents to capture information is undoubtedly inefficient. Therefore, in order to better capture the FI information in a multi-semantic space, we need to further *segment* the information implicit in the semantic space. What can be further considered is that we also need a suitable fusion layer to aggregate information from multi-semantic spaces. Some existing work performs simple summing [6, 9, 16] or more complex operations [3, 31, 39] (concat, product, linear transform) as a fusion layer for the outputs of each subcomponent, but does not take into account the information weights of the semantic spaces. Therefore, we need a spatial-level attention mechanism to learn the appropriate *weights* for each semantic space.

In summary, we identify three key elements for extracting FI information from multiple semantic spaces: subcomponent, segmentation, and weight. So, how do we construct a simple and effective CTR model from these elements?

Firstly, considering the efficiency and flexibility of MLP, which performs well in parallel, we utilize it as the subcomponent of the model in each semantic space, and capture information in each semantic space:

$$\begin{aligned} \mathbf{v} &= \mathbf{MLP}_v(\mathbf{E}), \\ \mathbf{v}' &= \mathbf{MLP}'_v(\mathbf{S}_{EP}), \\ \mathbf{v}'' &= \mathbf{MLP}''_v(\mathbf{S}_{IP}), \end{aligned} \tag{11}$$

where $\mathbf{MLP}_v$ mentioned here is a customizable MLP that uses LeakyReLU as an activation function for its hidden layer and no activation function for its output layer, $\mathbf{S}$ denotes the augmented embeddings, $\mathbf{v}$ is a scalar and represents the final real-values of the FI information in the current semantic space.

Table 3. Performance Comparison of simMHN with Three Strong Baseline Models

Models	Avazu		Criteo		Movielens	
	Logloss↓	AUC(%)↑	Logloss↓	AUC(%)↑	Logloss↓	AUC(%)↑
DNN (base)	0.372142	79.2717	0.438233	81.3728	0.208941	96.9276
DCN	0.372353	79.3142	0.438091	81.4103	0.204643	97.0140
FinalMLP	**0.370886**	79.4707	**0.437631**	**81.4472**	**0.196641**	**97.2373**
xDeepFM	0.371944	79.3121	0.437820	81.4291	0.206501	96.9769
simMHN	0.371091	**79.4810**	0.437681	81.4355	0.199238	97.0165

Bold indicates best performance, underline indicates second-best performance.

Secondly, the product operation is highly effective for modeling FI [26, 47, 78]. Different product operations often yield distinct FI information. Hence, we conduct pairwise product operations on the feature embeddings, represented as segmented **E**, to yield **S**.

$$\begin{aligned} \mathbf{S}_{EP} &= \|_{i=1}^{n} \|_{j=i}^{n} \mathcal{F}_{EP}(\mathbf{e_i}, \mathbf{e_j}), \\ \mathbf{S}_{IP} &= \|_{i=1}^{n} \|_{j=i}^{n} \mathcal{F}_{IP}(\mathbf{e_i}, \mathbf{e_j}), \end{aligned} \tag{12}$$

where $\|$ denotes the concat operation, $\mathcal{F}_{(\cdot)}$ indicates a product operation, and $\forall \mathbf{e_i}, \mathbf{e_j} \in \mathbf{E}$. *EP* denotes element-wise product, while *IP* represents vector inner product.

Thirdly, attention mechanisms have been widely employed in CTR [12, 31, 54, 65, 77], but these attention mechanisms are only feature-level and do not work well to weight the information in the semantic space. Therefore, we introduce a spatial-level attention mechanism to better aggregate information from various semantic spaces within the fusion layer, in which the spatial-level attention mechanism is:

$$\mathbf{K}_i = \mathbf{MLP}_k^i \left(space_i \right), \tag{13}$$

where $\mathbf{MLP}_k$ and $\mathbf{MLP}_v$ are nearly identical, with the difference being the use of LeakyReLU activation in the output layer instead of no activation function. **K** is a scalar that represents the weight of information implied by the current semantic *space* (e.g., **E** and **S**). To facilitate the discussion, we collectively refer to the subcomponent consisting of $\mathbf{MLP}_k$ and $\mathbf{MLP}v$ as the *Key-Value Block*.

In the end, we obtain the final prediction result by performing a straightforward weighted summation pooling as the fusion layer.

$$\hat{y} = \sum_{i=1}^{H} \mathbf{K}_i \mathbf{v}_i, \tag{14}$$

where H represents the number of semantic spaces. We call this model, which simply and efficiently captures FI information from multi-semantic spaces, the **Simple Multi-Head Network (simMHN)**. This model is the predecessor of CETN, whose performance and architecture are shown in Table 3 and Figure 6.

It can be observed that this simple model achieves SOTA performance, which obtains sub-optimal results and even outperforms the strongest baseline model FinalMLP [39].

Next, we attempt to enhance the model's performance by focusing on the diversity and homogeneity of information, without significantly altering the complexity of the simMHN model.

3.2 How to Improve the Diversity of Information Captured

Perturbation operations in CL can further differentiate information within semantic spaces [63, 71]. Therefore, we employ product & perturbation as the better segmentation approach for semantic

spaces and distinguish between primary and auxiliary semantic spaces (the semantic space processed by segmentation is called the auxiliary space, and vice versa). In this way we can get an augmented $\mathbf{E}'$:

$$\begin{aligned} \mathbf{E}' &= \mathbf{E} + \Delta', \\ \Delta' &= \bar{\Delta} \odot \operatorname{sign}(\mathbf{E}), \\ \bar{\Delta} &\in \mathbb{R}^d \sim U(0,1), \end{aligned} \tag{15}$$

where Δ' represents a noise signal, $\bar{\Delta}$ is a sample drawn from a uniform distribution between 0 and 1. Afterwards, we further differentiate information within the semantic space using the product operation:

$$\begin{aligned} \mathbf{S}'_{EP} &= \|_{i=1}^{n} \|_{j=i}^{n} \mathcal{F}_{EP}(\mathbf{e}'_{\mathbf{i}}, \mathbf{e}'_{\mathbf{j}}), \\ \mathbf{S}'_{IP} &= \|_{i=1}^{n} \|_{j=i}^{n} \mathcal{F}_{IP}(\mathbf{e}'_{\mathbf{i}}, \mathbf{e}'_{\mathbf{j}}), \end{aligned} \tag{16}$$

where $\forall \mathbf{e}'_{\mathbf{i}}, \mathbf{e}'_{\mathbf{j}} \in \mathbf{E}'$. After segmentation, we improve the diversity of information within the semantic space. However, it becomes apparent that there is no inherent supervisory signal between multiple subcomponents to ensure their ability to capture distinct FI. Relying solely on a self-adaptive joint learning strategy among subcomponents often leads to suboptimal performance. Therefore, we introduce auxiliary supervisory signals for the real-valued semantic space.

Given that we imitate the augmentation concept from CL during segmentation, we employ a contrastive loss to supervise the information-capturing behavior of subcomponents within the auxiliary semantic space. However, it's worth noting that the alignment strategy in CL is not always effective, as indicated in previous studies [70, 71]. A similar observation holds for CTR tasks based on user historical behavior sequences [17]. As a result, we modify the contrastive loss, abandoning alignment while retaining uniformity. More specifically, we consider the information in both auxiliary spaces to be negative samples even if they are obtained from the same $\mathbf{S}$, and the modified loss is formulated to minimize the agreement with these samples, as follows:

$$\mathcal{L}_{cl} = \sum_{i \in \mathcal{B}} -\log \frac{\exp(1/\tau)}{\sum_{j \in \mathcal{B}} \exp\left(\operatorname{sim}(\mathbf{V}'_i, \mathbf{V}''_j)/\tau\right)}, \tag{17}$$

where $\mathbf{V}', \mathbf{V}'' \in \mathbb{R}^{d_v}$ represent the real-values (obtained from different *Key-Value Blocks*) within the auxiliary semantic space after segmentation, d_v represents the real-values vector dimension. We refer to this modified loss as Do-InfoNCE.

For subcomponents, we seek to further enhance their inherent ability to capture information from different semantic spaces. Some studies suggest that choosing appropriate activation functions can significantly impact the performance of neural networks in various application scenarios [11, 25, 66]. For instance, when there is excessive noise in the current semantic space, we can consider using ReLU as the activation function to filter out irrelevant information. Therefore, utilizing suitable activation functions in different semantic spaces can aid the model in effectively capturing diverse FI. We employ different activation functions within $\mathbf{MLP}_v$ belonging to different semantic spaces.

3.3 How to Ensure Homogeneity of Information

After segmenting the semantic space and enhancing the ability of individual subcomponents to capture diverse information, while the model can better capture a variety of information, the increase in the number of feature spaces can lead to an issue of excessive noise. Therefore, we further introduce the concept of homogeneity, which complements diversity. Specifically, we aim for the information captured by various semantic spaces to be as distinct as possible (diversity).

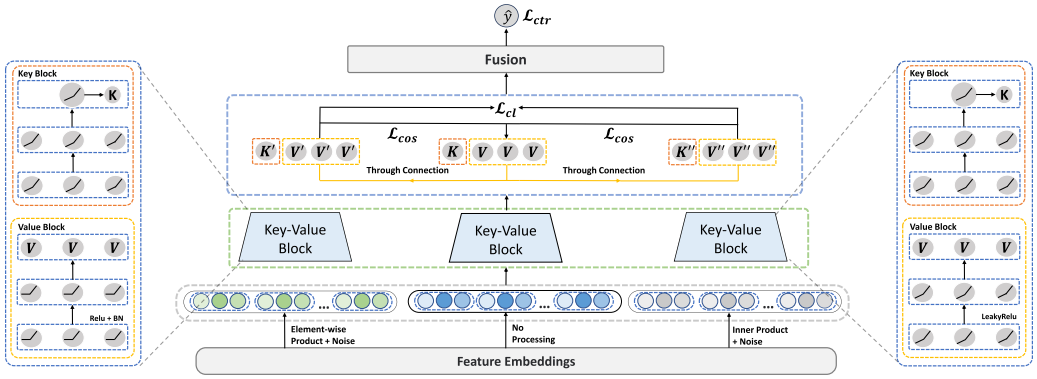


Fig. 5. The architecture of CETN.

However, we also seek this information to be fundamentally similar (homogeneity), without being entirely dissimilar. To achieve this goal, we draw inspiration from residual networks [18] and introduce a *Through Network* to ensure the homogeneity of the captured information across various semantic spaces. Formally, we define this TC as:

$$\begin{aligned} \mathbf{V} &= \mathbf{MLP}_v(\mathbf{E}), \\ \mathbf{V}' &= \mathbf{MLP}'_v(\mathbf{S}'_{EP}) + \mathbf{V}, \\ \mathbf{V}'' &= \mathbf{MLP}''_v(\mathbf{S}'_{IP}) + \mathbf{V}, \end{aligned} \tag{18}$$

where $\mathbf{V} \in \mathbb{R}^{d_v}$ denotes the real-values information in the main semantic space, $\mathbf{S}'$ represents the augmented embeddings in the auxiliary semantic space. By constructing the model in this manner, we establish a framework that is able to connect subcomponents in multiple semantic spaces, ensuring their homogeneity in capturing information while mitigating overfitting. Additionally, we also simply introduce the cosine similarity as an auxiliary loss, which takes the following form:

$$\begin{aligned} \mathcal{L}'_{cos} &= \sum_{i \in \mathcal{B}} 1 - \mathrm{sim}(\mathbf{V_i}, \mathbf{V}'_\mathbf{i}), \\ \mathrm{sim}(\mathbf{V}, \mathbf{V}') &= \frac{\mathbf{V}^\top \mathbf{V}'}{\|\mathbf{V}\| \, \|\mathbf{V}'\|}. \end{aligned} \tag{19}$$

The $\mathcal{L}'_{cos}$ encourages homogeneity between $\mathbf{V}$ and $\mathbf{V}'$, other similarity functions can also be utilized here.

3.4 Fusion Layer and Multi-Task Training

After capturing information concurrently from H semantic spaces, the real-values ($\mathbf{V}_i$) and weights ($\mathbf{K}_i$) from each semantic space are aggregated by the fusion layer:

$$\hat{y} = \sum_{i=1}^{H} \mathbf{K}_i(\mathbf{W}^\top \mathbf{V}_i + \mathbf{b}), \tag{20}$$

where $\mathbf{W} \in \mathbb{R}^{d_v}$ and $\mathbf{b}$ are the weight and bias parameters. At this point, we have obtained a new CTR model, CETN, whose architecture is illustrated in Figure 5.

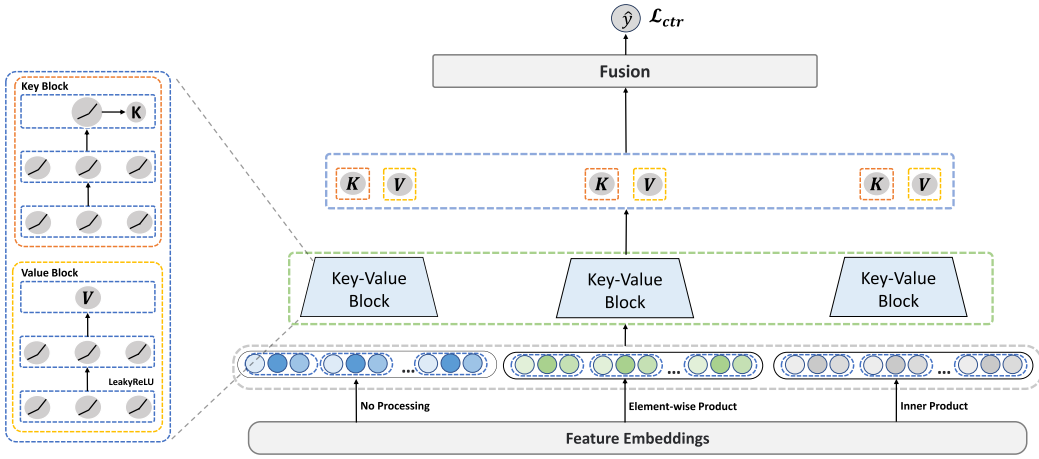


Fig. 6. The architecture of simMHN.

As we are targeting the CTR, which is a binary classification task, we have employed the widely used logloss [31, 33, 54, 80] as the loss function for our model:

$$\mathcal{L}_{ctr} = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log\left(\hat{y}_i\right) + \left(1-y_i\right)\log\left(1-\hat{y}_i\right)\right), \tag{21}$$

where N is the total number of training samples, i represents the sample index, y and $\hat{y}$ represent the true label and the predicted result of CETN. Next, we can obtain the total loss, denoted as $\mathcal{L}_{total}$:

$$\mathcal{L}_{total} = \mathcal{L}_{ctr} + \alpha \cdot \mathcal{L}_{cl} + (\beta' \cdot \mathcal{L}'_{cos} + \beta'' \cdot \mathcal{L}''_{cos}), \tag{22}$$

where α, β', β'' represent hyperparameters that control the balance between the loss functions.

3.5 Training Procedure

In the above sections, we engaged in a detailed discussion on how to simply yet effectively capture FI across various semantic spaces, as well as how to balance the diversity and homogeneity of the captured information. To facilitate a deeper understanding of our proposed CETN model, we provide a comprehensive depiction of its training process in Algorithm 1.

Initially, we decide whether to perturb the embedding $\mathbf{E}$ for augmentation (lines 2–6), followed by conducting product-based augmentation (line 7) to distinguish the FI information implied in different semantic spaces. At this point, we have made the necessary segmentation of the semantic spaces. Subsequently, we employ the *Key-Value Block* to capture FI information within each semantic space (lines 9-10) and calculate the self-supervised signal (lines 11–12). Finally, we aggregate the information from all semantic spaces in the fusion layer and update the parameters (lines 13–14).

3.6 Model Analysis

3.6.1 Model Size. To effectively capture FI information across various semantic spaces, CETN employs the *Key-Value Block* and product & perturbation. For ease of discussion, we regard W_Ψ as the set of weights in the corresponding MLP and ignore the embedding parameters. In the *Key-Value Block*, it can be simply viewed as six parallel MLPs, so its corresponding space complexity is $O(6|W_\Psi|)$. For the product & perturbation operation, we further divide additional auxiliary semantic spaces, hence its corresponding space complexity is $O(df + \frac{df(f+1)}{2} + \frac{f(f+1)}{2})$. As the

Algorithm 1: The Overall Training Process of CETN

Require: feature embeddings $\mathbf{E}$;

Initialize all parameters.
if use perturbing **then**
  Get $\mathbf{E}'$ according to Equation (15);
else
  Get $\mathbf{E}' = \mathbf{E}$;
end
Construct $\mathbf{S}'_{EP}$ and $\mathbf{S}'_{IP}$ according to Equation (16);
while CETN not converge **do**
  Calculate spatial information weights $\mathbf{K}$, $\mathbf{K}'$, and $\mathbf{K}''$ according to Equation (13);
  Calculate real-values of each semantic space $\mathbf{V}$, $\mathbf{V}'$, and $\mathbf{V}''$ according to Equation (18);
  Calculate $\mathcal{L}_{cl}$ according to Equation (17);
  Calculate $\mathcal{L}_{cos}$ according to Equation (19);
  Fusion information from various semantic spaces according to Equation (20);
  Update the model parameters using Equation (22);
end while

Table 4. Comparison of Analytical Time Complexity
$N \gg s > |W_\Psi| \approx |W_{gate}| > d_1 \approx d_2 \approx |h_1| > f \approx d \approx f_s \approx M \approx U$

Model	Embedding	Implicit interaction	Explicit interaction	Objective Function
DNN	$O(2dfs)$	$O(\|W_\Psi\|)$	-	$O(N)$
DCN	$O(2dfs)$	$O(\|W_\Psi\|)$	$O(4dfL)$	$O(N)$
FinalMLP	$O(2dfs + 2dsf_s)$	$O(2\|W_\Psi\|)$	$O(2d_1d_2 + 2\|W_{gate}\| + 2df_s)$	$O(N)$
xDeepFM	$O(2dfs)$	$O(\|W_\Psi\|)$	$O(dfM(1 + ML)))$	$O(N)$
AFN+	$O(4dfs)$	$O(\|W_\Psi\|)$	$O(2df(1 + U))$	$O(N)$
CL4CTR	$O(2dfs)$	$O(3\|W_\Psi\|)$	-	$O(N + N\|h_1\| + \frac{dfN(1+N)}{2} + \frac{dfN^2(f-1)}{2})$
CETN	$O(2dfs)$	$O(6\|W_\Psi\|)$	$O(df(1 + f))$	$O(N + 2Nd_v)$

space complexity represented by the inner product $\frac{f(f+1)}{2}$ is of constant level, therefore, the space complexity of CETN is $O(6|W_\Psi| + df + \frac{df(f+1)}{2})$. For more detailed information on the parameter size, refer to Tables 6 and 7.

3.6.2 Time Complexity. In a manner similar to the calculation of space complexity, the inference time of CETN is primarily due to the *Key-Value Block* and product & perturbation. Therefore, the time complexity during inference is $O(6|W_\Psi| + 2dfs + \frac{df(f+1)}{2} + df(f + 1))$. It's worth mentioning that due to the relationship between the Hadamard product and the inner product (the latter being the sum of the former), in practical operations, we can optimize it as follows: $O(6|W_\Psi| + 2dfs + df(f + 1))$.

Furthermore, for a more detailed comparison, we present the time complexities in the training of DNN, DCN, FinalMLP, xDeepFM, **Adaptive Feature Network (AFN)+**, and our proposed CETN model in Table 4. We let L, U, and M represent the number of explicit interaction layers, logarithmic neurons, and feature maps, respectively. d_1, d_2, W_{gate}, and f_s represent the output dimension of the two MLPs in FinalMLP, the number of parameters in the gate unit, and the feature fields to be filtered, respectively. (h_1, h_2) represent the outputs of the two FI encoders in CL4CTR.

It can be concluded that CETN, compared to other models that can also capture FI information in multiple semantic spaces, has a comparable time complexity. It is worth noting that, due to the

use of contrastive loss in CETN, this leads to a longer training time. However, it does not affect the corresponding inference speed in actual applications. In the case of implicit interactions, although CETN requires six MLPs, the practical time complexity does not increase proportionally. This is because MLPs are parallel-friendly and simple yet effective, which ensures the time complexity remains manageable.

3.6.3 Comparison with AFN+. The AFN [7] is a solid mathematical model that cleverly applies logarithmic operation rules for adaptive order explicit FI. Its enhanced version, AFN+, employs DNN and uses a **Logarithmic Transformation (LT)** Layer and MLP as parallel components to capture FI information in two semantic spaces. Unlike previous works [16, 60], it neither shares an embedding layer nor uses gating units to segment the semantic space, instead simply maintaining separate embedding matrices for the two parallel components. However, maintaining two sets of embedding matrices is inefficient due to the majority of parameters and computational costs being taken up by the embedding operation (the performance of AFN+ will be demonstrated in Section 4). The LT layer in AFN+ uses logarithmic operations to simulate the Hadamard product, which can be formulated as follows:

$$\exp\left(\sum_{i=1}^{m} w_{ij} \ln \mathbf{e}_i\right) = \mathbf{e}_1^{w_{1j}} \odot \mathbf{e}_2^{w_{2j}} \odot \ldots \odot \mathbf{e}_m^{w_{mj}}, \tag{23}$$

where w_{ij} is the coefficient of the jth neuron in the ith feature field and $\odot$ denotes Hadamard product. It is not difficult to observe that when the model learns the correct value for w_{ij}, it can adaptively learn FI of any order. However, since w_{ij} is usually non-zero, the interaction learned by AFN is essentially an exponentially weighted, fixed full-order FI. Interestingly, it is pointed out in some existing works that high-order FI do not bring the expected performance improvement [37, 78], leading to subpar performance of AFN+. In CETN, we only use the Hadamard product to model second-order FI, serving as an enhanced auxiliary semantic space, thus achieving better performance in both model complexity and performance metrics.

3.6.4 Comparison with CL4CTR. CL4CTR [59] is the first to introduce the CL paradigm into FI-based CTR prediction models. It uses Euclidean distance loss for feature alignment to make feature representations in the same feature field as similar as possible, and employs cosine loss for field uniformity to maximize the difference between feature representations in different feature fields. Additionally, it uses a perturbation operation to divide into two new FI auxiliary spaces and compares the encoded FI information through an FI encoder. Moreover, the CL4CTR still uses Euclidean distance loss as a contrastive loss to minimize the encoding differences between the two FI encoders, as shown in Equation (24), without introducing the widely acclaimed InfoNCE loss.

$$\mathcal{L}_{CL4CTR} = \frac{1}{B} \sum_{i=1}^{B} \left\| h_{i,1} - h_{i,2} \right\|_2^2 . \tag{24}$$

The fundamental difference between CETN and CL4CTR lies in their approach to handling information captured in the auxiliary semantic spaces. Where CL4CTR aims to make the captured information increasingly similar, CETN uses Do-InfoNCE to increase the diversity (i.e., dissimilarity) of captured information, while ensuring homogeneity (i.e., similarity) with the information in the original semantic space. Furthermore, as shown in Table 4, the time complexity of the contrastive loss in CL4CTR is very high, making it challenging to realistically apply in real-world production scenarios.

Table 5. Dataset Statistics

Dataset	#Instances	#Fields	#Features	#Split
Avazu	40M	24	8.37M	8:1:1
Criteo	46M	39	5.55M	8:1:1
MovieLens	2,006,859	3	90,445	7:2:1
Frappe	288,609	10	5,382	7:2:1

4 Experiments

In this section, we provide a detailed account of our experimental setup and substantiate the superiority of CETN over other SOTA models through a fair and extensive series of experiments. Subsequently, we conduct ablation experiments to investigate the impact of our configured hyperparameters on model performance and assess the rationale behind the presence of various modules.

4.1 Experimental Settings

To ensure a fair comparison, we closely followed the settings of the [39, 80] work and selected the same CTR benchmark datasets originating from real production environments. Table 5 below provides detailed information about these datasets.

- —Avazu[2]: This dataset contains 10 days of data on user clicks to ads while using mobile devices, as well as 15 explicit and 9 anonymous feature fields.
- —Criteo[3]: It is the well-known CTR benchmark dataset, which contains a 7-day stream of real data from Criteo, covering 39 anonymous feature fields.
- —MovieLens[4]: It consists of users' tagging records on movies. The datasets have been widely used in various research on recommender systems.
- —Frappe[5]: It contains app usage logs from users under different contexts (e.g., daytime, location). The target value indicates whether the user has used the app under the context.

Data preprocessing: We follow the approach outlined in [80]. For the Avazu dataset, we transform the timestamp field it contains into three new feature fields: hour, weekday, and weekend. For the Criteo dataset, we discretize the numerical feature fields by rounding down each numeric value x to $\lfloor \log^2(x) \rfloor$ if $x > 2$, and $x = 1$ otherwise. For all datasets' categorical features, infrequent features (min_count=2) are uniformly replaced with a default "OOV" token.

4.1.1 Evaluation Metrics. In order to compare the performance, we utilize two commonly used metrics in CTR models: AUC and logloss [16, 31, 46, 54].

- —AUC: AUC stands for Area Under the ROC Curve. It measures the probability that a positive instance will be ranked higher than a randomly chosen negative one. A higher AUC indicates a better performance.
- —Logloss: logloss is the calculation result of Equation (21). A lower Logloss suggests a better capacity for fitting the data.

[2]https://www.kaggle.com/c/avazu-ctr-prediction
[3]https://www.kaggle.com/c/criteo-display-ad-challenge
[4]https://grouplens.org/datasets/movielens/
[5]http://baltrunas.info/research-menu/frappe

It's worth noting that even a slight improvement in AUC is meaningful in the context of CTR prediction tasks. Typically, CTR researchers consider improvements at the *0.001–level (0.1%)* to be statistically significant, as often highlighted in existing literature [3, 6, 16, 59, 60, 80]. Following previous work [53, 59, 62, 77], we further use RelaImpr to measure the relative improvement of AUC and Logloss, as defined:

$$\begin{aligned} \text{RelaImpr}_{AUC} &= \left(\frac{\text{AUC (target model)} - 0.5}{\text{AUC (base model)} - 0.5} - 1 \right) \times 100\%, \\ \text{RelaImpr}_{Logloss} &= \left(\frac{\text{Logloss (base model)} - \text{Logloss (target model)}}{\text{Logloss (base model)}} \right) \times 100\%, \end{aligned} \tag{25}$$

4.1.2 Baselines. We compare CETN with some classical SOTA models. Given that deep CTR models often perform better, for models that have both non-DNN and DNN versions, we tend to choose the latter. The list of models we have chosen in chronological order of publication is as follows:

- *LR* [51]: **Logistic regression (LR)** is a simple baseline model for CTR prediction.
- *FM* [49]: This model employs factorization techniques to address the challenge of learning on sparse datasets.
- *DNN* [8]: This approach utilizes a feedforward neural network that takes a straightforward concatenation of feature embeddings as input.
- *IPNN* [46]: The model is an inner product-based feedforward neural network.
- *Wide & Deep* [6]: It encompasses LR (wide network) and the integration of a feedforward neural network (deep network).
- *DeepFM* [16]: This model combines FM and feedforward neural networks in parallel by sharing embedding layers.
- *NFM* [19]: This method vertically combines FM and feedforward neural networks through the Bi-interaction layer.
- *AFM* [65]: The model incorporates an attention mechanism to discern the importance of different FI.
- *DCN* [60]: The model introduces the CrossNet that can explicitly model FI, and integrate feedforward neural networks in parallel.
- *xDeepFM* [33]: Similar to DCN, this model introduces the Compressed Interaction Network, enhancing FI from bit-wise to vector-wise.
- *FiGNN* [31]: This model pioneers the use of graph neural networks to model FI.
- *AutoInt+* [54]: This model is the first to learn higher-order FI using a multi-headed attention mechanism.
- *AFN+* [7]: The model utilizes an LT layer to learn adaptive-order FI.
- *DCNv2* [61]: Expanding upon DCN, this model enhances the projection matrix's dimensionality and introduces a mixture of low-rank expert systems to optimize model inference speed.
- *EDCN* [3]: This model introduces both a bridge module and a regulation module, thereby enhancing the performance of the DCN model.
- *MaskNet* [62]: This model utilizes the MaskBlock as its foundational structure. The introduced Instance-Guided Mask in the MaskBlock assists the model in acquiring high-quality information more effectively.
- *GraphFM* [32]: This model employs graph structure learning to address FM's limitations in selecting and learning appropriate higher-order FI.
- *FinalMLP* [39]: The model demonstrates the efficacy of the two-stream MLP model for implicit FI learning.

Table 6. Performance Comparison of Different Models for CTR Prediction

Year	Models	Avazu			Criteo		
		Logloss↓	AUC↑	#Params	Logloss↓	AUC↑	#Params
2007	LR [51]	0.381727	0.777286	3.7M	0.456573	0.793558	5.5M
2010	FM [49]	0.376240	0.786085	78.7M	0.444295	0.807856	116.5M
2016	DNN [8]	0.372142	0.792717	78.5M	0.438233	0.813728	115.7M
2016	IPNN [46]	**0.371156(3)**	**0.794330(3)**	75.6M	0.438217	0.813945	114.6M
2016	Wide & Deep [6]	0.372015	0.792982	82.2M	0.438110	0.813814	120.4M
2017	DeepFM [16]	0.371890	0.793116	82.4M	**0.437676(3)**	**0.814200(5)**	117.6M
2017	NFM [19]	0.373843	0.789997	79.2M	0.446362	0.805424	118.6M
2017	AFM [65]	0.378901	0.782119	78.7M	0.444468	0.807100	116.5M
2017	DCN [60]	0.372353	0.793142	76.5M	0.438091	0.814103	132.5M
2018	xDeepFM [33]	0.371944	0.793121	79.4M	0.437820	**0.814291(4)**	120.5M
2019	FiGNN [31]	0.374346	0.789236	75.0M	0.438860	0.813340	111.1M
2019	AutoInt+ [54]	0.372085	0.792639	77.5M	0.438919	0.812950	170.5M
2020	AFN+ [7]	0.372007	0.793513	177.9M	0.439244	0.813098	226.3M
2021	DCNv2 [61]	0.372111	0.793050	77.2M	0.438392	0.813951	114.9M
2021	EDCN [3]	**0.371627(5)**	**0.793874(4)**	75.7M	**0.437742(4)**	**0.814292(3)**	112.8M
2021	MaskNet [62]	**0.371623(4)**	**0.793677(5)**	77.8M	0.439455	0.812424	116.8M
2022	GraphFM [32]	0.372646	0.791912	75.0M	0.441261	0.810482	111.0M
2023	FinalMLP [39]	**0.370886(2)**	**0.794707(2)**	152.7M	**0.437631(2)**	**0.814472(2)**	116.2M
2023	CL4CTR [59]	0.373158	0.791745	76.2M	**0.437794(5)**	0.814159	112.5M
2023	EulerNet [56]	0.376032	0.792391	75.6M	0.442632	0.811003	113.4M
ours	CETN	**0.370402(1)**	**0.796238(1)**	85.1M	**0.437319(1)**	**0.814804(1)**	140.1M

We highlight in bold the top-5 best results in each dataset. "+": Integrating the original model with DNN networks.

—*CL4CTR* [59]: This model pioneers the integration of CL into CTR prediction based on FI by introducing the concepts of feature alignment and field uniformity.
—*EulerNet* [56]: This model employs Euler's formula to explicitly model FI, integrating it with linear layers to adaptively learn FI of the arbitrary order.

4.1.3 Implementation Details. We implement all models using Pytorch [43] and refer to existing works [22, 80]. We employ the Adam optimizer [27] to optimize all models, with a default learning rate set to 0.001. For the sake of fair comparison, we set the embedding dimension to 20, and the batch size to 10,000 for all models. The hyperparameters of the baseline model were configured based on the *optimal values* provided in [22, 80] and their original paper. For the hyperparameters α, β', and β'', we use grid search to select the optimal loss balance weights. Specifically, we first fix β' and β'' at 0.5 and search for the optimal α in the range [0.1, 0.5] with a step size of 0.05. Subsequently, we fix α and search for the optimal β' and β'' in the range [0.1, 0.9] with a step size of 0.1. To prevent overfitting, we employ early stopping with a patience value of 2. To facilitate reproducible research, we have open-sourced the code and running logs of CETN.[6]

4.2 Overall Comparison

The performance of CETN and the baseline model is shown in Tables 6 and 7. it can be observed that deep CTR models, with DeepFM [16] as their representative, consistently outperform shallow models, typified by FM [49]. This underscores the effectiveness of combining implicit

[6]https://github.com/salmon1802/CETN

Table 7. Performance Comparison of Different Models for CTR Prediction

Year	Models	MovieLens			Frappe		
		Logloss↓	AUC↑	#Params	Logloss↓	AUC↑	#Params
2007	LR [51]	0.345537	0.931499	88597	0.364208	0.931528	5384
2010	FM [49]	0.276611	0.942978	1.8M	0.193952	0.969701	0.1M
2016	DNN [8]	0.208941	0.969276	2.1M	0.169329	0.980640	0.5M
2016	IPNN [46]	0.206157	**0.970354(3)**	2.1M	**0.151626(5)**	**0.984333(4)**	0.5M
2016	Wide & Deep [6]	0.207965	0.969820	2.2M	0.152807	0.984018	0.5M
2017	DeepFM [16]	**0.205239(4)**	0.969749	2.2M	0.153563	0.983501	0.5M
2017	NFM [19]	0.302908	0.939400	2.1M	0.160586	0.980831	0.4M
2017	AFM [65]	0.277981	0.942794	1.8M	0.242329	0.955657	0.1M
2017	DCN [60]	0.204643	**0.970140(5)**	2.1M	0.155340	0.983995	0.5M
2018	xDeepFM [33]	0.206501	0.969769	2.3M	0.155676	0.983409	0.5M
2019	FiGNN [31]	0.256296	0.952781	1.8M	0.226627	0.964828	0.2M
2019	AutoInt+ [54]	**0.203297(3)**	**0.970191(4)**	2.2M	**0.150433(2)**	**0.984076(5)**	0.7M
2020	AFN+ [7]	**0.205358(5)**	0.969492	8.0M	0.156007	0.980949	3.8M
2021	DCNv2 [61]	0.206726	0.969712	2.1M	**0.150948(4)**	**0.984368(3)**	0.6M
2021	EDCN [3]	0.228215	0.968716	1.7M	0.161859	0.983283	0.2M
2021	MaskNet [62]	0.242475	0.968499	2.8M	0.189036	0.982834	1.6M
2022	GraphFM [32]	0.222897	0.964445	1.7M	0.288119	0.939295	0.1M
2023	FinalMLP [39]	**0.196641(2)**	**0.972373(2)**	2.2M	**0.150553(3)**	**0.984854(2)**	0.6M
2023	CL4CTR [59]	0.206971	0.969982	2.4M	0.152105	0.983712	1.0M
2023	EulerNet [56]	0.213534	0.965486	1.7M	0.153704	0.980542	0.2M
ours	CETN	**0.185652(1)**	**0.973957(1)**	1.9M	**0.150283(1)**	**0.985710(1)**	1.6M

We highlight in bold the top-5 best results in each dataset. "+": Integrating the original model with DNN networks.

high-order FI with explicit shallow FI, resulting in improved CTR predictions. Concurrently, it emphasizes the necessity of leveraging explicit FI to overcome the performance bottlenecks associated with MLP.

Focusing on the models that achieved top-5 performance in the experiments, it can be observed that all of these models are based on parallel or stacked structures. Interestingly, over time, when all models are configured with their respective optimal parameters, the performance improvements of these SOTA models are relatively modest. Notably, the FinalMLP [39], consistently delivers strong performance across multiple datasets, further highlighting the importance of subcomponents and segments.

Taking an overall view, the CETN consistently maintains the highest AUC and Logloss performance, even when all models are configured with their respective optimal parameters. As is shown in Table 8, in comparison to the best baseline model, CETN exhibits improvements of 0.52%, 0.11%, 0.34%, and 0.18% compared to FinalMLP in AUC on the four datasets separately, while achieving corresponding reductions in Logloss by 0.13% (compared to FinalMLP), 0.07% (compared to FinalMLP), 5.59% (compared to FinalMLP), and 0.1% (compared to AutoInt+). Compared to the xDeepFM model, which also utilizes three semantic spaces, CETN achieves AUC average improvements of 0.75% in AUC and 3.52% in Logloss. This performance enhancement can be attributed to its ability to capture diversity within the feature space while preserving information homogeneity. It is noteworthy that, in contrast to intricate explicit FI networks, the proposed CETN model simply integrates CL into simMHN, effectively improving its ability to capture FI information.

Compared to the CL4CTR model, which also employs the CL paradigm, CETN demonstrates an average relative improvement of 3.09% in Logloss and 0.86% in AUC across the four datasets.

Table 8. Relative Improvement of AUC and Logloss with CETN

Model	$\textbf{Avazu}_{RelaImpr}$		$\textbf{Criteo}_{RelaImpr}$		$\textbf{MovieLens}_{RelaImpr}$		$\textbf{Frappe}_{RelaImpr}$		△Logloss ↓	△AUC ↑
	Logloss↓	AUC(%)↑	Logloss↓	AUC(%)↑	Logloss↓	AUC(%)↑	Logloss↓	AUC(%)↑		
LR [51]	2.97%	6.83%	4.22%	7.24%	46.27%	9.84%	58.74%	12.56%	28.05%	8.44%
FM [49]	1.55%	3.55%	1.57%	2.26%	32.88%	6.99%	22.52%	3.41%	14.63%	4.95%
DNN [8]	0.47%	1.20%	0.21%	0.34%	11.15%	1.00%	11.25%	1.05%	5.77%	0.89%
IPNN [46]	0.20%	0.65%	0.20%	0.27%	9.95%	0.77%	0.89%	0.28%	2.81%	0.61%
Wide & Deep [6]	0.43%	1.11%	0.18%	0.32%	10.73%	0.88%	1.65%	0.35%	3.25%	0.80%
DeepFM [16]	0.40%	1.07%	0.08%	0.19%	9.54%	0.90%	2.14%	0.46%	3.04%	0.76%
NFM [19]	0.92%	2.15%	2.03%	3.07%	38.71%	7.86%	6.42%	1.01%	12.02%	5.24%
AFM [65]	2.24%	5.00%	1.61%	2.51%	33.21%	37.04%	37.98%	6.60%	18.76%	5.40%
DCN [60]	0.52%	1.06%	0.18%	0.22%	9.28%	0.81%	3.26%	0.35%	3.31%	0.73%
xDeepFM [33]	0.41%	1.06%	0.11%	0.16%	10.10%	0.89%	3.46%	0.48%	3.52%	0.75%
FiGNN [31]	1.05%	2.42%	0.35%	0.47%	27.56%	4.68%	33.69%	4.49%	15.66%	3.06%
AutoInt+ [54]	0.45%	1.23%	0.36%	0.59%	8.68%	0.80%	0.10%	0.34%	2.40%	0.86%
AFN+ [7]	0.43%	0.93%	0.44%	0.54%	9.60%	0.95%	3.67%	0.99%	3.53%	0.84%
DCNv2 [61]	0.46%	1.09%	0.24%	0.27%	10.19%	0.90%	0.44%	0.28%	2.83%	0.79%
EDCN [3]	0.33%	0.80%	0.10%	0.16%	18.65%	1.12%	7.15%	0.50%	6.56%	0.80%
MaskNet [62]	0.33%	0.87%	0.49%	0.76%	23.43%	1.16%	20.50%	0.60%	11.19%	0.99%
GraphFM [32]	0.60%	1.48%	0.89%	1.39%	16.71%	2.05%	47.84%	10.57%	16.51%	1.74%
FinalMLP [39]	0.13%	0.52%	0.07%	0.11%	5.59%	0.34%	0.18%	0.18%	1.49%	0.29%
CL4CTR [59]	0.74%	1.54%	0.11%	0.21%	10.30%	0.85%	1.20%	0.41%	3.09%	0.86%
EulerNet [56]	1.50%	1.32%	1.20%	1.22%	13.06%	1.82%	2.23%	1.08%	4.50%	1.54%

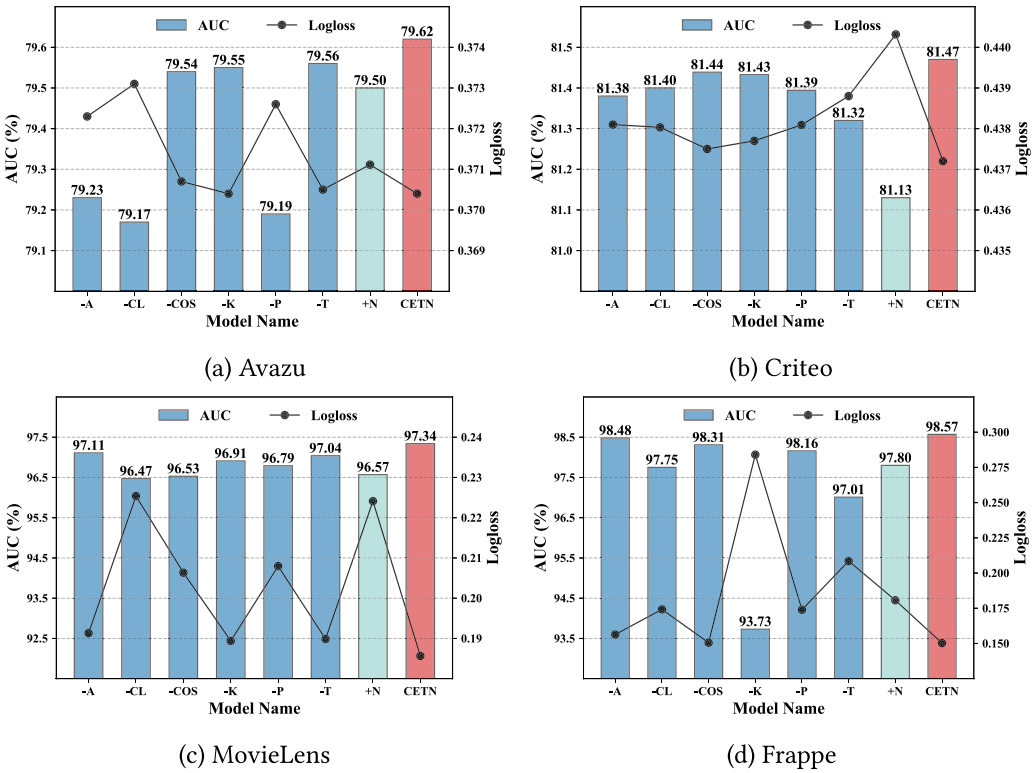
△Logloss and △AUC denote the average performance improvement.

Interestingly, in CL4CTR, the contrastive loss does not adopt the popular InfoNCE paradigm used in graph neural network-based recommender. Instead, it utilizes Euclidean distance loss as the contrastive loss, resulting in the maximization of similarity in the enhanced auxiliary semantic space embeddings. This stands in contrast to our Do-InfoNCE approach and is similar to $\mathcal{L}_{cos}$. This suggests that CL4CTR overlooks the connection between information in the auxiliary semantic space and the primary semantic space, emphasizing the necessity of homogeneity and diversity as unsupervised guiding signals for model learning.

4.3 Ablation Study

In this section, we conduct extensive ablation studies to assess the contributions of individual modules of CETN to its overall performance. Several variants were designed to validate the effectiveness of the various CETN modules:

— *CETN (−A)*: To further enhance the diversity of information captured by the model, we employ different activation functions in multiple semantic spaces. To assess the necessity of this approach, we only specify the use of ReLU activation functions in the MLP for modeling FI within multiple semantic spaces. This will reduce the diversity of information captured by the model.
— *CETN (−CL)*: Contrastive loss serves to self-supervise and augment the diversity of information captured by the model. To determine whether it aids in improving model performance, we experiment by removing it from the model.
— *CETN (−COS)*: The cosine loss primarily reinforces the homogeneity of the model from a supervised signal perspective. To ascertain its necessity, we exclude it from the model.
— *CETN (−K)*: It serves to fine-tune the contribution weights of various subcomponents to the final prediction, further refining the model's predictive results. To establish the necessity of the $\mathbf{MLP}_k$ (spatial-level attention) within the *Key-Value Block*, we eliminate it from the model.
— *CETN (−P)*: To verify the effectiveness of our proposed product & perturbation method, we replace it with the original embeddings $\mathbf{E}$ without any additional processing.



(a) Avazu (b) Criteo

(c) MovieLens (d) Frappe

Fig. 7. Ablation study of CETN on Avazu (a), Criteo (b), MovieLens (c), and Frappe (d) datasets.

— *CETN (−T)*: The TC ensures the model maintains homogeneity in the information captured across multiple semantic spaces. To evaluate its usefulness, we remove it from the model.
— *simMHN (+N)*: Noise is introduced to perturb the original **E** and enhance the diversity of the original embedding representation. To verify the impact of noise on model performance, we add perturbation augmentation to simMHN.

Figure 7 illustrates the performance of CETN and its six variants across four datasets: Avazu, Criteo, MovieLens, and Frappe. We can observe that CETN outperforms all the ablation models, providing compelling evidence for the necessity of each component in CETN. To delve deeper into this, let's break down the performance across different datasets.

Specifically, on the Avazu dataset, we notice that the performance drop is most pronounced for CETN (−A, −CL, −P). This indicates that the Avazu dataset benefits significantly from diverse FI information to boost model performance. The substantial drop in performance for CETN (−A) suggests that the different activation functions are crucial. The removal of $\mathcal{L}_{cl}$ in CETN (−CL) leads to a notable decline, emphasizing the importance of Do-InfoNCE in maintaining diverse and meaningful representations. Similarly, the performance hit in CETN (−P) underscores the role of the product & perturbation method in segmenting the semantic space. As described in Section 3.2, the three variants enhance the model's ability to capture diverse information. This proves that the Avazu dataset needs a greater diversity of real-valued information to improve the model's generalization ability and performance. On the other hand, the small performance loss of CETN (−COS, −T) suggests that the Avazu dataset needs to strike a balance between homogeneity and

diversity in favor of the latter to achieve optimal performance. The simCEN (+N) achieves a slight performance improvement compared to Table 3, further validating our conclusion from CETN (−CL) that the Avazu dataset requires more diverse representation information.

On the Criteo dataset with the biggest number of feature fields, we observe the most significant performance drop in CETN (−T), which means the TC is particularly important for the Criteo dataset, as it helps in maintaining a consistent flow of information and mitigating noise. This highlights that an increase in the number of feature fields often introduces more noise signals. Therefore, it is crucial to ensure information homogeneity while enhancing information diversity on the Criteo dataset. The simCEN (+N) also supports this view. When noise is introduced to simCEN, there is a significant performance drop compared to Table 3, further proving that Criteo requires maintaining homogeneity of representations to reduce the impact of noise. The variants CETN (−A) and CETN (−P) also cause significant performance degradation, indicating that despite the large number of features in the Criteo dataset, certain data augmentation is still needed to enhance the diversity of information in the semantic space. The variants CETN (−CL) and CETN (−COS) experience performance drops, though less severe, indicating that both CL and cosine similarity loss play supportive roles in balancing FI information.

On the Movielens dataset, We empirically think that due to its limited number of feature fields and relatively fewer instances, it's susceptible to overfitting. Consequently, CETN (−CL, −COS), both complementing each other, play a significant role in constraining the scope of information captured by the model, allowing it to capture high-quality interaction information, thereby preventing overfitting to some extent. The correctness of this hypothesis can be intuitively observed in Figure 7(c), where the model's performance experiences a noticeable decline when CETN removes $\mathcal{L}_{cl}$, $\mathcal{L}_{cos}$. The variant simCEN (+N) also shows similar performance. Due to the absence of $\mathcal{L}_{cl}$ and $\mathcal{L}_{cos}$, merely introducing noise to enhance representation diversity leads to significant performance degradation. The variants CETN (−A), CETN (−P), and CETN (−T) exhibit certain performance degradation, further indicating the role of balancing diversity and homogeneity in improving performance on MovieLens.

On the Frappe dataset, the performance of CETN (−K) exhibits a cliff-like decline, providing evidence for the effectiveness of the spatial-level attention mechanism. It is noteworthy that the Frappe dataset has only 0.7% and 0.6% of the data volume compared to the Avazu and Criteo datasets, respectively. This effectively underscores the challenge of models to adaptively assess the importance of information in various semantic spaces when dealing with relatively fewer data. Consequently, this leads to a poorer predictive performance of the model. In the other three datasets, CETN (−K) does not cause significant performance degradation, indicating that smaller datasets require more accurate attention scores to assist model learning. CETN (−CL, −COS, −P, −T) also results in noticeable performance losses, reaffirming the value of balancing diversity and homogeneity, and CETN (−T) causes the most performance loss, this indicates that ensuring information homogeneity is more crucial for the Frappe dataset. Notably, the variant CETN (−A) causes minimal performance loss, which we attribute to the smaller data size and lower sample complexity of the Frappe dataset. The model can capture sufficient information from existing features without much need for the activation function's assistance. Compared to CETN (−T), simCEN (+N) results in less performance loss, which is consistent with CETN (−CL). This demonstrates that the Frappe dataset has a more important need to maintain homogeneity of representations.

4.4 Complementary Nature of Diversity and Homogeneity

As we mention in Definition 4, we can understand this idea through a simple sociological philosophy: *Everyone has a self that is similar to others and a self that is unique, only the proportions differ.* The aspect *similar to others* represents human homogeneity, while the *unique* aspect represents human

diversity. The reason a person is both different and similar to others is that diversity and homogeneity find a balance.

From the perspective of representation learning, we want the model's final output representations to also conform to these characteristics. As shown in Figure 2 of this paper, for each sub_i captured by different subcomponents, we aim for a balance between diversity and homogeneity. If there is too much diversity, the information contained in sub_i will be completely different, leading to inconsistency and difficulty in integrating the representations, and even causing the model only to capture varying noise information. Conversely, if there is too much homogeneity, sub_i will be nearly identical, making it difficult for the model to capture different features, thereby reducing generalization ability. Therefore, by finding an appropriate balance between diversity and homogeneity, we can ensure that the learned representations are both diverse to enhance the model's generalization ability and homogeneous to reduce the impact of noise in the representations.

In the three semantic spaces constructed in this paper, $\mathbf{E}$ can be seen as an anchor, while $\mathbf{S}_{EP}$ and $\mathbf{S}_{IP}$ are augmented semantic spaces obtained by element-wise production or inner product with $\mathbf{E}$. Therefore, the information in $\mathbf{S}_{EP}$ and $\mathbf{S}_{IP}$ originates from $\mathbf{E}$, resulting in a certain level of homogeneity in the corresponding real-valued information $\mathbf{V}$, $\mathbf{V}'$, and $\mathbf{V}''$. On the other hand, we want the $\mathbf{V}'$ and $\mathbf{V}''$ captured by the model to exhibit diversity to enhance the model's generalization ability. These seemingly contradictory yet complementary properties form the core idea of this paper. We ensure diversity through $\mathcal{L}_{cl}$, maintain homogeneity using $\mathcal{L}_{cos}$ and TC, and attempt to find a balance between them by adjusting the loss weights.

In the ablation study shown in Figure 7, we can further observe the performance loss caused by Figure 3(a) and (b). CETN (−CL) is a variant of the model with $\mathcal{L}_{cl}$ removed, leading the model to overly pursue homogeneity, as seen in Figure 3(b). This variant results in performance loss across all four datasets in the ablation study, with the most significant loss on the Avazu dataset. CETN (−COS) and CETN (−T) correspond to Figure 3(a), where the model focuses too much on diversity, also causing a certain degree of performance loss. In contrast, CETN balances these two properties and demonstrates superior performance across all four datasets, clearly outperforming its variants.

4.5 Which Semantic Space Is More Useful

For the three semantic Spaces we define and their subcomponents, we conduct experiments to explore their individual contributions to the model's final predictions. The performance of each separate subcomponent on the four datasets is presented in Table 9. On the Avazu dataset, $\mathbf{S}_{EP}$ achieved outstanding performance, even surpassing IPNN (the best baseline model). However, its performance declined under the simple fusion of simMHN, demonstrating the drawbacks of this simple fusion approach. On the Criteo and MovieLens datasets, the original embeddings $\mathbf{E}$ showed better results, and simMHN further improved performance through simple fusion. On the Frappe dataset, both $\mathbf{S}_{EP}$ and $\mathbf{S}_{IP}$, respectively, achieved the best AUC and Logloss, but in simMHN, although the AUC performance improved further, Logloss increased, indicating that simMHN might struggle to effectively predict true click probabilities.

To validate the contribution of the combined semantic spaces to the final prediction results, we pair the three semantic spaces in pairs. As observed in Table 9, $\mathbf{S}_{EP}$ & $\mathbf{S}_{IP}$ achieves better performance in terms of Logloss and AUC on the Avazu, Criteo, and MovieLens datasets, demonstrating the effectiveness of combining $\mathbf{S}_{EP}$ & $\mathbf{S}_{IP}$ information. Meanwhile, by comparing the combined performance with the performance of the semantic space alone, we found that simply combining information from two semantic spaces often leads to performance degradation. For instance, $\mathbf{S}_{EP}$ compared to $\mathbf{E}$ & $\mathbf{S}_{EP}$ on Avazu, $\mathbf{E}$ compared to $\mathbf{E}$ & $\mathbf{S}_{IP}$ on Criteo, and $\mathbf{E}$ compared to $\mathbf{E}$ & $\mathbf{S}_{EP}$ on MovieLens. Even the strongest performing $\mathbf{S}_{EP}$ & $\mathbf{S}_{IP}$ combination still falls short of simMHN, further proving the effectiveness of simMHN.

Table 9. Single Performance of Each Semantic Space

Model	Avazu		Criteo		MovieLens		Frappe	
	Logloss	AUC	Logloss	AUC	Logloss	AUC	Logloss	AUC
S_{EP}	0.370488	0.795332	0.441558	0.811001	0.236711	0.964678	0.166236	0.982632
S_{IP}	0.373471	0.790807	0.439198	0.813027	0.227524	0.960124	0.161219	0.977373
E	0.372142	0.792717	0.438233	0.813728	0.208941	0.969276	0.169329	0.980640
E & S_{EP}	0.371928	0.793311	0.440322	0.812303	0.243368	0.960318	0.330557	0.967380
E & S_{IP}	0.370938	0.794757	0.441664	0.811307	0.294099	0.944847	0.152292	0.980229
S_{EP} & S_{IP}	0.370653	0.795164	0.439900	0.813206	0.235955	0.961429	0.162203	0.982697
simMHN	0.371091	0.794810	0.437681	0.814355	0.199238	0.970165	0.180469	0.983528
CETN	**0.370402**	**0.796238**	**0.437319**	**0.814804**	**0.185652**	**0.973957**	**0.150283**	**0.985710**
RelaImpr	0.18%	0.48%	0.08%	0.14%	6.82%	0.81%	16.72%	0.45%

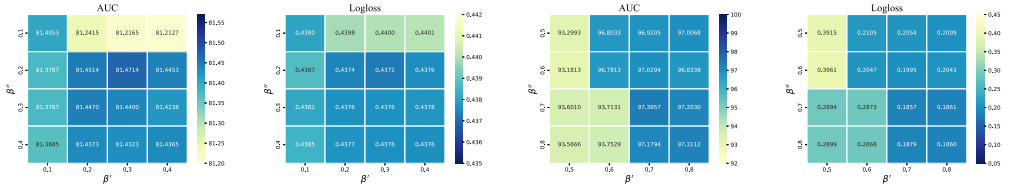
RelaImpr denotes the relative improvements compared with the simMHN. Bold indicates best performance, underline indicates second-best performance.

To further enhance the performance of the simMHN model without significantly increasing its complexity, we introduced diversity- and homogeneity-guided self-supervised signals. Additionally, we incorporated skip connections and various activation functions to balance both homogeneity and diversity. Consequently, CETN achieved improvements in Logloss by 0.18%, 0.08%, 6.82%, and 16.72% on the four datasets, and in AUC by 0.48%, 0.14%, 0.81%, and 0.45%, respectively. This demonstrates the effectiveness of ensuring both homogeneity and diversity in the information captured by the model.

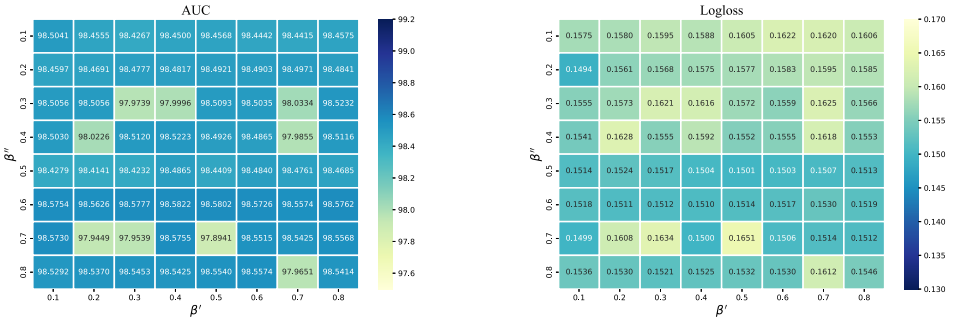
4.6 Hyper-Parameter Analysis

4.6.1 Impact of the Weights in the $\mathcal{L}_{cos}$. We conduct a further investigation into the impact of different weight parameter combinations on the model's performance. For the sake of discussion, we confine the ranges of β' and β'' to $[0.1 \sim 0.4, 0.5 \sim 0.8]$ while keeping other settings fixed. The results are presented as a heat map in Figure 8, and it is evident that CETN achieves its optimal performance when $\beta' = 0.3$ and $\beta'' = 0.2$ on the Criteo dataset. In the heatmap of MovieLens, multiple chunked regions in the model's performance are observed. The performance continues to decrease when β' and β'' are set to 0.5 or 0.6, but reaches optimal performance at 0.7 and 0.8. This phenomenon precisely validates the results presented in Table 9. Due to the relatively lower effective information content in the auxiliary semantic space, setting a larger $\mathcal{L}_{cos}$ becomes necessary to prevent the model from capturing noise and ensure homogeneity of information. On the Frappe dataset, it is evident that the model performs well when β'' is $0.6 \sim 0.8$ and achieves optimal results when β' equals 0.1, aligning with our previous reporting in Table 9. Notably, the model attains the lowest Logloss when $\beta' = 0.1$ and $\beta'' = 0.2$. This is reasonable, as on the Frappe dataset, even though S_{IP} exhibits poor AUC performance, which achieves the lowest Logloss in multiple semantic spaces.

4.6.2 Impact of α in the $\mathcal{L}_{cl}$. We make modifications to the contrastive loss weight on the Avazu and Criteo datasets with a step size of 0.1, and in the smaller datasets MovieLens and Frappe, the modification step size is set to 0.05. The results are depicted in Figure 9. On the Criteo dataset, it's observed that the model achieves better performance when the contrastive loss weight is set to 0.2 or 0.3. Subsequently, as the weight increases, the model's performance starts to decline. Notably, On the Movielens dataset, due to its limited three feature fields, the model is more sensitive to hyperparameter variations. Therefore, there is a substantial performance change when the weight is increased from 0.15 to 0.2. Compared to the MovieLens dataset, the performance of different

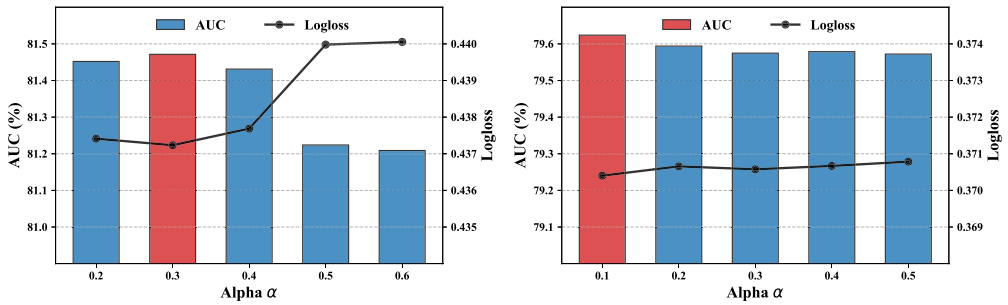


(a) AUC on Criteo (b) Logloss on Criteo (c) AUC on MovieLens (d) Logloss on MovieLens

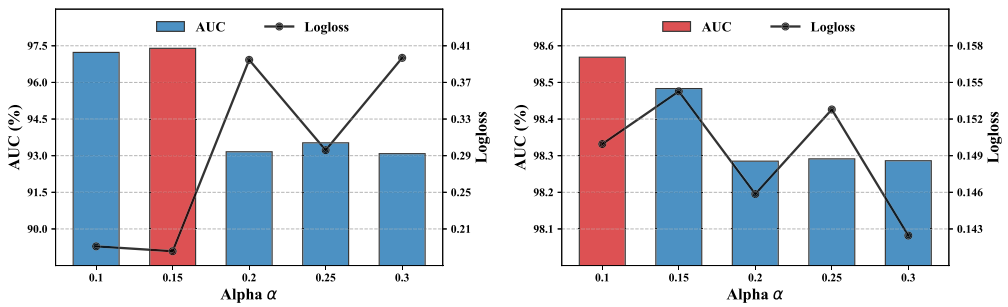


(e) AUC on Frappe (f) Logloss on Frappe

Fig. 8. Performance comparison of different weights of cosine loss on Criteo (a and b), MovieLens (c and d), and Frappe (e and f) datasets.



(a) Criteo (b) Avazu



(c) MovieLens (d) Frappe

Fig. 9. Performance comparison of different weights of contrast loss on Criteo (a), Avazu (b), MovieLens (c), and Frappe (d) datasets.

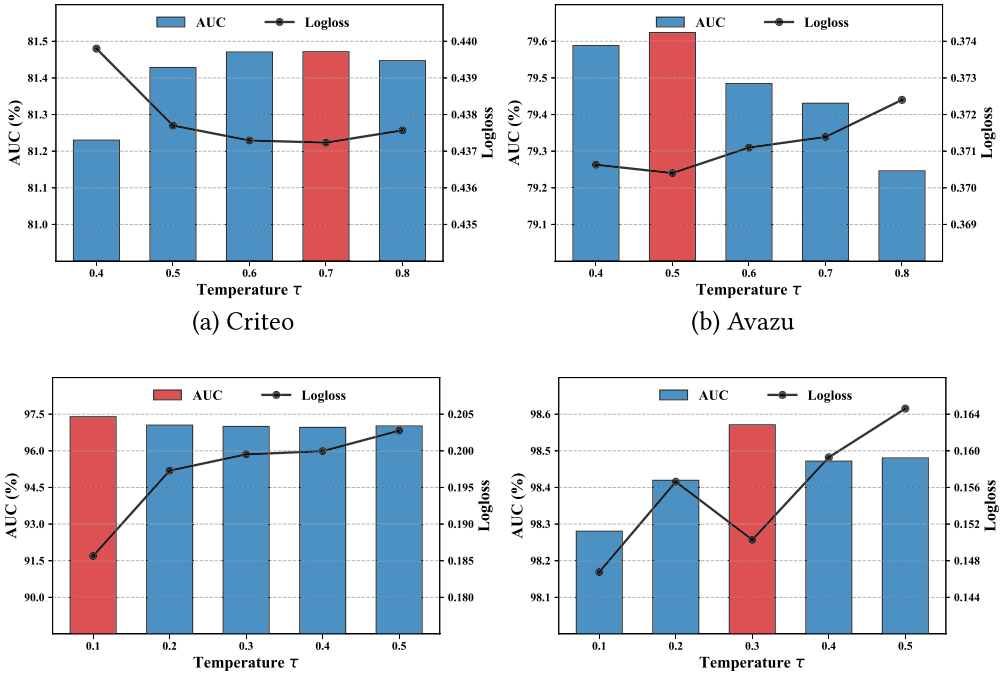
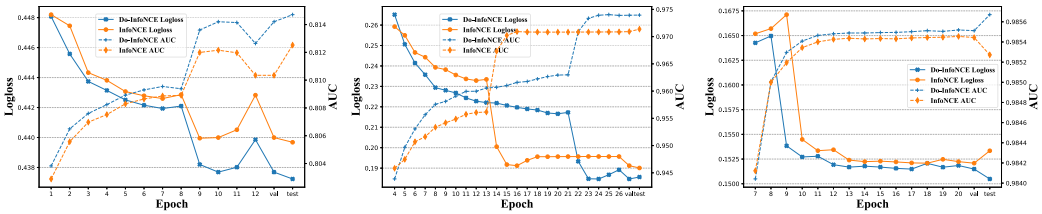


Fig. 10. Performance comparison of different temperature coefficients of CETN on Criteo (a), Avazu (b), MovieLens (c), and Frappe (d) datasets.

values for the parameter α on the Avazu dataset is more stable. However, there is still a trend of declining performance with the increase in weight values. It's worth mentioning that compared to the simMHN model, when $\alpha = 0.1$, the model's AUC performance improves by 0.48%. This demonstrates the double-edged nature of diversity. On one hand, it can help the model capture additional information, and on the other hand, it can easily introduce noise signals. On the Frappe dataset, the model's performance peaks when $\alpha = 0.1$, after which the model's performance shows a precipitous decline at 0.2, consistent with the trend observed on the MovieLens dataset. Looking at the overall Figure 9, we find that when the CETN model achieves optimal performance in various datasets, the value of α is between 0.15 and 0.3. This suggests that although diversity of information is necessary, it still needs to be managed moderately. Otherwise, it will bring unnecessary noise signals to the model in various semantic spaces.

4.6.3 Impact of τ. In many models based on InfoNCE loss [63, 70, 71], the temperature coefficient is commonly set to 0.2 by default. However, due to the sensitivity of CTR prediction tasks to performance metrics, we further explored its impact on the model's performance. We keep other parameters fixed and change the values for τ with a step size of 0.1 in the four datasets, as shown in Figure 10. As can be observed, with an increase in the temperature coefficient, the model's performance gradually improves and then gradually decreases. However, MovieLens is an exception. To achieve good performance, the model requires an extreme temperature coefficient. This suggests that at this point, the model needs to rely on a stronger force to differentiate the semantic information between different input instances. In summary, we recommend fine-tuning the temperature coefficient of the contrastive loss within the range of 0.1 to 1.0 for optimal performance.



(a) Criteo (b) MovieLens (c) Frappe

Fig. 11. Optimization Process of CETN on Criteo (a), MovieLens (b), and Frappe (c) datasets.

4.7 Do-InfoNCE vs InfoNCE

4.7.1 Role of Do-InfoNCE. To make InfoNCE more suitable for FI-based CTR tasks, we make modifications to it. By simplifying and deriving the formula for InfoNCE (Equation 10), we can transform it into the following form:

$$\sum_{i\in\mathcal{B}} -\log \frac{\exp\left(\text{sim}(\mathbf{V}'_i, \mathbf{V}''_i)/\tau\right)}{\sum_{j\in\mathcal{B}} \exp\left(\text{sim}(\mathbf{V}'_i, \mathbf{V}''_j)/\tau\right)},$$

$$\Rightarrow \sum_{i\in\mathcal{B}} -\log \frac{\exp\left(1/\tau\right)}{\sum_{j\in\mathcal{B}} \exp\left(\text{sim}(\mathbf{V}'_i, \mathbf{V}''_j)/\tau\right)}, \tag{26}$$

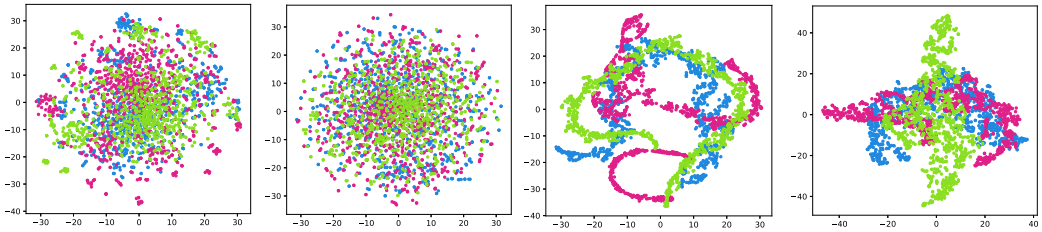
$$\Rightarrow \sum_{i\in\mathcal{B}} \log\left(\exp(\text{sim}(\mathbf{V}'_i, \mathbf{V}''_i)/\tau) + \sum_{j\in\mathcal{B}/\{i\}} \exp\left((\text{sim}(\mathbf{V}'_i, \mathbf{V}''_j)/\tau)\right)\right), \tag{27}$$

in Equation (26), we discarded alignment, retaining only uniformity, thereby obtaining Do-InfoNCE. In Equation (27), we disregarded $1/\tau$, it becomes apparent that when the model optimizes this loss, it ensures that the model acquires dissimilar information across different semantic spaces, even if $\mathbf{V}_i$ and $\mathbf{V}_j$ originate from the same input.

In some studies [63, 70, 71], this uniformity is interpreted as minimizing similarity. This perspective is especially prevalent in the context of graph neural networks, where the aim is to disperse nodes away from dense clusters in the representation space, leading to a more uniformly distributed representation. In fact, looking at this from the perspective of information capture, the reason for the improvement in model performance due to this uniform distribution can be simply attributed to the model capturing more diverse information. In other words, we ensure the diversity of the captured information through the InfoNCE loss function.

4.7.2 Optimization Process. To further investigate the impact of Do-InfoNCE and InfoNCE on the model optimization process, we fix the model's hyperparameters and only change the contrastive loss function and early stopping patience to force the models to train for the same number of epochs. We visualize the model's training process, and the results are presented in Figure 11.

On the Criteo dataset, InfoNCE and Do-InfoNCE perform almost identically in Logloss optimization during the first epoch, followed by a consistent performance gap, though the overall optimization trends are similar. In terms of AUC optimization, Do-InfoNCE consistently outperforms InfoNCE, with the performance gap widening as the number of epochs increases. On the MovieLens dataset, InfoNCE converges faster than Do-InfoNCE. The Logloss for InfoNCE drops rapidly and stabilizes in the early training stages (around the 15th epoch), whereas Do-InfoNCE requires a longer training period (approximately until the 23rd epoch) to reach optimal performance. Additionally, the AUC optimization process shows similar trends, with InfoNCE stabilizing



(a) CETN on Frappe (b) CETN on MovieLens (c) TriDNN on Frappe (d) TriDNN on MovieLens

Fig. 12. The visualization of information within semantic spaces on Frappe and MovieLens datasets. The three colors in the plot represent three semantic spaces. TriDNN denotes the use of three independent MLPs as subcomponents without employing self-supervisory signals.

around the 13th epoch and Do-InfoNCE around the 21st epoch. However, in both AUC and Logloss optimization, Do-InfoNCE consistently performs better on the test set than InfoNCE. On the Frappe dataset, InfoNCE and Do-InfoNCE exhibit very similar optimization trends. Notably, InfoNCE shows poorer generalization on the test set compared to the validation set, while Do-InfoNCE performs better on the test set than on the validation set, demonstrating Do-InfoNCE's advantage in enhancing model generalization. Looking at the overall picture, our proposed Do-InfoNCE always achieves better performance, further proving the effectiveness of Do-InfoNCE.

4.7.3 Visualization of Information within Semantic Spaces. To further explore the impact of self-supervisory signals on capturing FI information within various semantic spaces, we randomly sampled 1,000 instances and visualized the information captured by the model, as shown in Figure 12. In Figure 12(a) and (b), the representation distribution is more dispersed due to the influence of $\mathcal{L}_{cl}$, indicating that the model has captured richer and more diverse information. On the other hand, in the case of TriDNN, the FI information captured by the model across the three semantic spaces tends to be more similar. Consequently, in Figure 12(c) and (d), the representation distribution is more concentrated, suggesting that the model's acquired information is narrower, thus reducing its effectiveness.

4.8 Through Network vs Residual Network

4.8.1 The Widespread Phenomenon of Shallow Networks in Recommender Systems. In the field of **computer vision (CV)**, neural networks often tend to develop in a deeper direction [18, 55], but in the field of recommender systems, shallow networks are often sufficient to achieve the expected task objectives. For example, in graph neural network-based recommender systems [20, 71, 75], the number of layers in the graph neural network is often set to 3. Similarly, in the CTR prediction tasks based on FI [39, 80], the number of layers in the MLP is often also set to 3. The primary reason researchers set it this way is due to the issues of over-smoothing and degradation. This leads to a rapid descent of the neural network as the number of layers increases, eventually causing the model to collapse. Moreover, this severe data sparsity issue can even lead to a unique phenomenon in the field of recommender systems known as the one-epoch phenomenon [76]. Therefore, in CTR prediction tasks, neural networks often tend to widen rather than deepen.

4.8.2 Reproduction of the Collapse Phenomenon in CTR Prediction. To verify the degradation phenomenon of deep neural networks in the field of recommender systems, we conduct experiments on the MovieLens dataset using MLP with different numbers of layers. We also visualize the training process of the model, as shown in Figure 13. In order to intuitively observe the impact of the number of neural network layers on model training, we stopped using methods to prevent model overfitting,

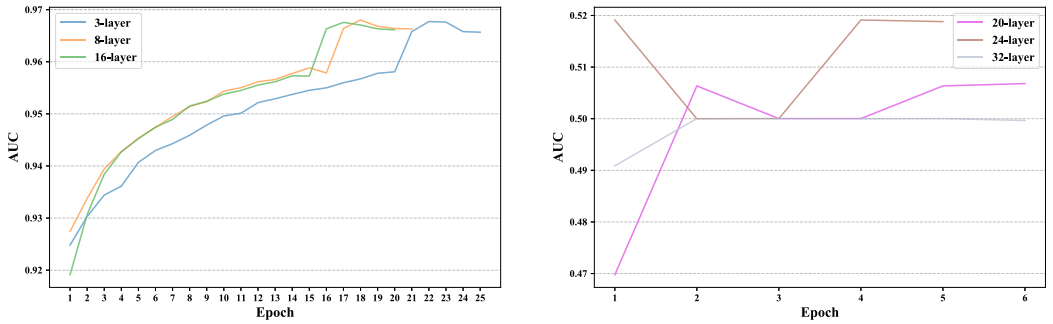


Fig. 13. Optimization process of plain DNN with different number of layers on MovieLens datasets. The last two points of each line represent the validation set performance and the test set performance, respectively.

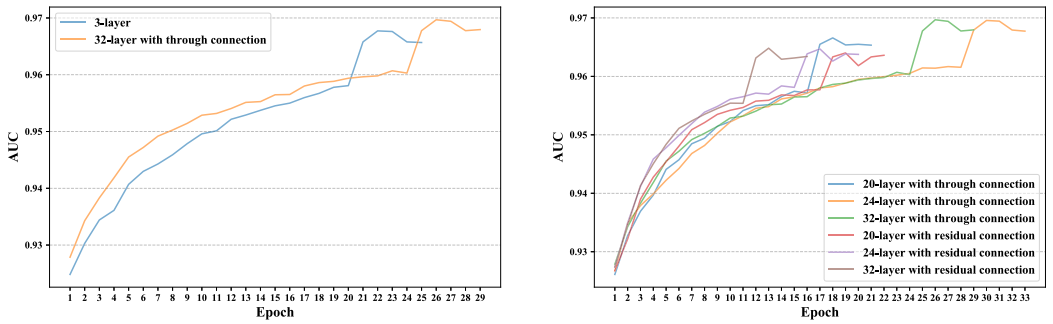


Fig. 14. Optimization Process on MovieLens datasets. Left: comparison between a 3-layer plain DNN and a 32-layer DNN using TC. Right: comparison between the residual network and the through network. The last two points of each line represent the validation set performance and the test set performance, respectively.

such as dropout and batch norm, and only retained L_2 normalization to prevent the model from experiencing the one-epoch phenomenon. From the experimental results, it can be observed that as the number of MLP layers gradually increased from 3 to 8 and then to 16, the performance of the model does not show a significant improvement, but it does consume more computational resources. Therefore, further deepening the depth of the neural network is not a good choice. Next, when we increased the number of layers in the MLP to 20, the model experienced a collapse, and the AUC performance was close to 0.5. Even if we further deepened the depth of the MLP, the situation did not improve.

4.8.3 Validation of Effectiveness. By changing the skip connections in the residual network from vertical to horizontal, we can generalize to the Through Network, which can help the model overcome the model collapse phenomenon brought about by the deepening of network layers. To demonstrate this hypothesis, we combined a 32-layer DNN with a 3-layer DNN using TC. The experimental results are shown in Figure 14 (left). What we can observe is that after using TC, the 32-layer DNN not only avoids the model collapse phenomenon but also further improves the performance of the 3-layer DNN.

To compare the performance differences between through networks and residual networks in CTR prediction tasks, we conduct experiments using the layer settings where model collapse occurred. The results are shown in Figure 14. One observation is that the performance of the model using TC exceeds that of residual connections. The residual connections perform best at 24 layers,

while TC perform best at 32 layers. This implies that TC can better utilize deep neural networks to fit the target function on sparse datasets. Another point worth mentioning is that in our subsequent experiments, as the depth of the network continued to increase, the performance of the DNN model using the TC continued to improve. This suggests that the TC not only prevent the model from collapsing in deeper settings but also contribute to better performance as depth increases.

5 Related Work

5.1 CL for Recommendation

To the best of our knowledge, until now, there has been very limited work combining CL with CTR prediction tasks based on FI. This can occur because users often have a propensity for multiple interests in items, and it becomes challenging to definitively distinguish between positive and negative samples based solely on the user's click behavior. Therefore, traditional CL based on alignment and uniformity principles cannot be directly applied to CTR. With the rapid advancement of self-supervised learning in the fields of **Natural Language Processing (NLP)** [30, 38] and CV [4, 15, 21], due to the similarities between CTR prediction models based on user behavior sequences and NLP, they initially incorporate CL [23, 57, 73]. MISS [17] analyzes user behavior sequences and employs contrastive loss to enhance user interest representations at the feature level, thereby improving model performance. AQCL [42] attempts to alleviate the problem of learning difficulties in representing the user's click history feature sequences in cold-start scenarios by introducing AQCL loss. CL4CTR [59] introduced CL for the first time into FI-based CTR prediction tasks. It proposes feature alignment and field uniformity for the feature field concept of CTR to enhance the quality of feature representation. However, it does not incorporate InfoNCE [40] loss and fails to address the issues of diversity and homogeneity from an architectural perspective.

For Top-K recommendation, there is a amount of related work on CL [35, 64]. SGL [63] uses masking to randomly disrupt the interaction graph structure to create different views. simGCL [71] and xSimGCL [70] use random noise to augment data and reveal deeper mechanisms of CL. DENS [29] introduces the hierarchical gating mechanism and factor-aware sampling strategy. The former plays a crucial role in disentangling the relevant and irrelevant factors of both positive and negative samples for CL, while the latter is designed to select the best negative samples by contrasting the relevant factors while keeping the irrelevant factors similar. DCCF [48] introduces an adaptive approach to data augmentation, using parameterized interaction mask generators. Subsequently, DCCF performs CL on the disentangled multi-view embeddings to enhance the performance of Top-K recommendation models. From the perspective of disentangled learning, our model can be seen as a representation CL method that disentangles diversity and homogeneity.

5.2 CTR Models Based on Feature Interactions

Most existing CTR models based on FI predominantly follow the Embedding & Cross paradigm. They begin by encoding categorical and continuous features through embedding techniques. Subsequently, they employ a variety of complex cross-operations to augment the first-order feature data, achieving the goal of FI, and ultimately enhancing model performance. MLP has played an indelible role in implicit FI (Cross), greatly improving the benchmark performance of deep CTR models. However, many researchers have highlighted the inefficiency of MLP in learning product-based FI (inner product, outer product, or Hadamard product) [50, 52]. Therefore, researchers have endeavored to employ additional feature data augmentation techniques to overcome the performance bottleneck of MLPs. Enhanced MLP models can primarily be categorized into two types: those enhanced based on stack structures and those enhanced based on parallel structures.

NFM [19], PNN [46], MaskNet [62], and xCrossNet [72] employ a stack structure to enhance MLP-based CTR models. They aim to introduce explicit product-based FI operations before using embeddings as inputs to the MLP, thereby breaking through the performance bottleneck of the MLP. From a semantic space segmentation perspective, this explicit product operation further enriches the FI information within the current semantic space, resulting in improved performance. Wide & Deep [6], DeepFM [16], DeepLight [9], FinalMLP [39], FINAL [79], xDeepFM [33], and DCN [60] employ a parallel structure. These models aim to introduce explicit FI in a parallel manner to the simple MLP model. They achieve this by incorporating a fusion layer to capture both explicit and implicit FI information simultaneously. While this parallel strategy of capturing FI information in different semantic spaces has yielded promising results, it still fails to address the three issues we have identified, ultimately resulting in sub-optimal performance.

6 Conclusion and Future Work

In this paper, we revisit the problem of effectively capturing FI information from multiple semantic spaces, and compose the simMHN with multiple Key-Value Blocks as parallel subcomponents. We then further enhance simMHN around the two complementary principles of diversity and homogeneity, thereby proposing a new simple and effective CTR model, called the CETN. CETN builds upon simMHN by leveraging CL and TC to further capture high-quality FI information. Experimental results on four benchmark datasets validate the effectiveness of our proposed model. As we look toward future work, we are interested in refining the architecture of this model, seeking opportunities to make it even more simple and efficient.

References

[1] Alexey Borisov, Ilya Markov, Maarten De Rijke, and Pavel Serdyukov. 2016. A neural click model for web search. In *Proceedings of the 25th International Conference on World Wide Web*, 531–541.

[2] Erika Bourguignon and Lenora Greenbaum. 1973. *Diversity and Homogeneity in World Societies*. HRAF Press, New Haven, CT.

[3] Bo Chen, Yichao Wang, Zhirong Liu, Ruiming Tang, Wei Guo, Hongkun Zheng, Weiwei Yao, Muyu Zhang, and Xiuqiang He. 2021. Enhancing explicit and implicit feature interactions via information sharing for parallel deep CTR models. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 3757–3766.

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*. PMLR, 1597–1607.

[5] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*, 108–116.

[6] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, 7–10.

[7] Weiyu Cheng, Yanyan Shen, and Linpeng Huang. 2020. Adaptive factorization network: Learning adaptive-order feature interactions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34, 3609–3616.

[8] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, 191–198.

[9] Wei Deng, Junwei Pan, Tian Zhou, Deguang Kong, Aaron Flores, and Guang Lin. 2021. Deeplight: Deep lightweight feature interactions for accelerating CTR predictions in ad serving. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 922–930.

[10] Zhenhua Dong, Zhe Wang, Jun Xu, Ruiming Tang, and Jirong Wen. 2022. A brief history of recommender systems. arXiv:2209.01860. Retrieved from https://arxiv.org/abs/2209.01860

[11] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. 2018. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks* 107 (2018), 3–11.

[12] Yufei Feng, Fuyu Lv, Weichen Shen, Menghan Wang, Fei Sun, Yu Zhu, and Keping Yang. 2019. Deep session interest network for click-through rate prediction. Retrieved from https://dl.acm.org/doi/abs/10.5555/3367243.3367359

[13] Jianfeng Gao, Wei Yuan, Xiao Li, Kefeng Deng, and Jian-Yun Nie. 2009. Smoothing clickthrough data for web search ranking. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 355–362.

[14] Zhabiz Gharibshah and Xingquan Zhu. 2021. User response prediction in online advertising. *ACM Computing Surveys (CSUR)* 54, 3 (2021), 1–43.

[15] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. 2018. Unsupervised representation learning by predicting image rotations. Retrieved from https://openreview.net/forum?id=S1v4N2l0-

[16] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine Based Neural Network for CTR Prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI '17)*. AAAI Press, 1725–1731.

[17] Wei Guo, Can Zhang, Zhicheng He, Jiarui Qin, Huifeng Guo, Bo Chen, Ruiming Tang, Xiuqiang He, and Rui Zhang. 2022. Miss: Multi-interest self-supervised learning framework for click-through rate prediction. In *Proceedings of the IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 727–740.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

[19] 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.

[20] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, 639–648.

[21] Olivier J Hénaff, Skanda Koppula, Jean-Baptiste Alayrac, Aaron Van den Oord, Oriol Vinyals, and Joao Carreira. 2021. Efficient visual pretraining with contrastive detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10086–10096.

[22] Huawei. 2021. An Open-Source CTR Prediction Library. Retrieved from https://fuxictr.github.io

[23] Mengyuan Jing, Yanmin Zhu, Tianzi Zang, and Ke Wang. 2023. Contrastive self-supervised learning in recommender systems: A survey. *ACM Transactions on Information Systems* 42, 2 (Nov 2023), Article 59, 39 pages. DOI: https://doi.org/10.1145/3627158

[24] 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.

[25] Martin Kaloev and Georgi Krastev. 2021. Comparative analysis of activation functions used in the hidden layers of deep neural networks. In *Proceedings of the 3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*. IEEE, 1–5.

[26] Farhan Khawar, Xu Hang, Ruiming Tang, Bin Liu, Zhenguo Li, and Xiuqiang He. 2020. Autofeature: Searching for feature interactions and their architectures for click-through rate prediction. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 625–634.

[27] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. arXiv:1412.6980. Retrieved from https://arxiv.org/abs/1412.6980

[28] Annelies Knoppers, Inge Claringbould, and Marianne Dortants. 2015. Discursive managerial practices of diversity and homogeneity. *Journal of Gender Studies* 24, 3 (2015), 259–274.

[29] Riwei Lai, Li Chen, Yuhan Zhao, Rui Chen, and Qilong Han. 2023. Disentangled negative sampling for collaborative filtering. In *Proceedings of the 16th ACM International Conference on Web Search and Data Mining*, 96–104.

[30] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. arXiv:1909.11942. Retrieved from https://arxiv.org/abs/1909.11942

[31] Zekun Li, Zeyu Cui, Shu Wu, Xiaoyu Zhang, and Liang Wang. 2019. FiGNN: Modeling feature interactions via graph neural networks for ctr prediction. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 539–548.

[32] Zekun Li, Shu Wu, Zeyu Cui, and Xiaoyu Zhang. 2022. GraphFM: Graph factorization machines for feature interaction modeling. arXiv:2105.11866. Retrieved from https://arxiv.org/abs/2105.11866

[33] 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.

[34] Jianghao Lin, Yanru Qu, Wei Guo, Xinyi Dai, Ruiming Tang, Yong Yu, and Weinan Zhang. 2023. MAP: A model-agnostic pretraining framework for click-through rate prediction. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1384–1395.

[35] Zihan Lin, Changxin Tian, Yupeng Hou, and Wayne Xin Zhao. 2022. Improving graph collaborative filtering with neighborhood-enriched contrastive learning. In *Proceedings of the ACM Web Conference 2022*, 2320–2329.

[36] Patricia W. Linville. 1998. The heterogeneity of homogeneity. In *Attribution and Social Interaction: The Legacy of Edward E. Jones*. J. M. Darley and J. Cooper (Eds.), American Psychological Association, 423–487.

[37] 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.

[38] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. arXiv:1907.11692. Retrieved from https://arxiv.org/abs/1907.11692

[39] Kelong Mao, Jieming Zhu, Liangcai Su, Guohao Cai, Yuru Li, and Zhenhua Dong. 2023. FinalMLP: An enhanced two-stream MLP model for CTR prediction. *Proceedings of the AAAI Conference on Artificial Intelligence* 37, 4 (2023), 4552–4560.

[40] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. arXiv:1807.03748. Retrieved from https://arxiv.org/abs/1807.03748

[41] 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.

[42] Yujie Pan, Jiangchao Yao, Bo Han, Kunyang Jia, Ya Zhang, and Hongxia Yang. 2021. Click-through rate prediction with auto-quantized contrastive learning. arXiv:2109.13921. Retrieved from https://arxiv.org/abs/2109.13921

[43] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 8026–803.

[44] Katherine W. Phillips and Robert B. Lount. 2007. The affective consequences of diversity and homogeneity in groups. In *Affect and Groups*, Vol. 10, Emerald Group Publishing Limited, 1–20.

[45] Ruihong Qiu, Jingjing Li, Zi Huang, and Hongzhi Yin. 2019. Rethinking the item order in session-based recommendation with graph neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 579–588.

[46] 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 *Proceedings of the IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 1149–1154.

[47] Yanru Qu, Bohui Fang, Weinan Zhang, Ruiming Tang, Minzhe Niu, Huifeng Guo, Yong Yu, and Xiuqiang He. 2018. Product-based neural networks for user response prediction over multi-field categorical data. *ACM Transactions on Information Systems* 37, 1 (2018), 1–35.

[48] Xubin Ren, Lianghao Xia, Jiashu Zhao, Dawei Yin, and Chao Huang. 2023. Disentangled contrastive collaborative filtering. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1137–1146.

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

[50] Steffen Rendle, Walid Krichene, Li Zhang, and John Anderson. 2020. Neural collaborative filtering vs. matrix factorization revisited. In *Proceedings of the 14th ACM Conference on Recommender Systems*, 240–248.

[51] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting clicks: Estimating the click-through rate for new ads. In *Proceedings of the 16th International Conference on World Wide Web*, 521–530.

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

[53] Yanyan Shen, Lifan Zhao, Weiyu Cheng, Zibin Zhang, Wenwen Zhou, and Lin Kangyi. 2023. RESUS: Warm-up cold users via meta-learning residual user preferences in CTR prediction. *ACM Transactions on Information Systems* 41, 3 (2023), 1–26.

[54] 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.

[55] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1–9.

[56] Zhen Tian, Ting Bai, Wayne Xin Zhao, Ji-Rong Wen, and Zhao Cao. 2023. EulerNet: Adaptive feature interaction learning via Euler's formula for CTR prediction. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1376–1385.

[57] Chenyang Wang, Weizhi Ma, Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2023. Sequential recommendation with multiple contrast signals. *ACM Transactions on Information Systems* 41, 1 (2023), 1–27.

[58] Fangye Wang, Hansu Gu, Dongsheng Li, Tun Lu, Peng Zhang, and Ning Gu. 2023. Towards deeper, lighter and interpretable cross network for CTR prediction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 2523–2533.

[59] Fangye Wang, Yingxu Wang, Dongsheng Li, Hansu Gu, Tun Lu, Peng Zhang, and Ning Gu. 2023. CL4CTR: A contrastive learning framework for CTR prediction. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, 805–813.

[60] 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.

[61] Ruoxi Wang, Rakesh Shivanna, Derek Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed Chi. 2021. DCNv2: Improved deep & cross network and practical lessons for web-scale learning to rank systems. In *Proceedings of the Web Conference 2021*, 1785–1797.

[62] Zhiqiang Wang, Qingyun She, and Junlin Zhang. 2021. MaskNet: Introducing feature-wise multiplication to CTR ranking models by instance-guided mask. arXiv:2102.07619. Retrieved from https://arxiv.org/abs/2102.07619

[63] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised graph learning for recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 726–735.

[64] Lianghao Xia, Chao Huang, Yong Xu, Jiashu Zhao, Dawei Yin, and Jimmy Huang. 2022. Hypergraph contrastive collaborative filtering. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 70–79.

[65] 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. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 3119–3125.

[66] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. 2015. Empirical evaluation of rectified activations in convolutional network. arXiv:1505.00853. Retrieved from https://arxiv.org/abs/1505.00853

[67] Yanwu Yang and Panyu Zhai. 2022. Click-through rate prediction in online advertising: A literature review. *Information Processing & Management* 59, 2 (2022), Article 102853.

[68] Tiansheng Yao, Xinyang Yi, Derek Zhiyuan Cheng, Felix Yu, Ting Chen, Aditya Menon, Lichan Hong, Ed H. Chi, Steve Tjoa, Jieqi Kang, and Evan Ettinger. 2021. Self-supervised learning for large-scale item recommendations. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 4321–4330.

[69] Hongzhi Yin, Bin Cui, Xiaofang Zhou, Weiqing Wang, Zi Huang, and Shazia Sadiq. 2016. Joint modeling of user check-in behaviors for real-time point-of-interest recommendation. *ACM Transactions on Information Systems* 35, 2 (2016), 1–44.

[70] Junliang Yu, Xin Xia, Tong Chen, Lizhen Cui, Nguyen Quoc Viet Hung, and Hongzhi Yin. 2023. XSimGCL: Towards extremely simple graph contrastive learning for recommendation. *IEEE Transactions on Knowledge and Data Engineering* 36 (2023), 913–926. DOI: https://doi.org/10.1109/TKDE.2023.3288135

[71] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Quoc Viet Hung Nguyen. 2022. Are graph augmentations necessary? Simple graph contrastive learning for recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1294–1303.

[72] Runlong Yu, Yuyang Ye, Qi Liu, Zihan Wang, Chunfeng Yang, Yucheng Hu, and Enhong Chen. 2021. Xcrossnet: Feature structure-oriented learning for click-through rate prediction. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 436–447.

[73] Tianzi Zang, Yanmin Zhu, Ruohan Zhang, Chunyang Wang, Ke Wang, and Jiadi Yu. 2023. Contrastive multi-view interest learning for cross-domain sequential recommendation. *ACM Transactions on Information Systems* 42, 3 (Nov 2023), 1–30. DOI: https://doi.org/10.1145/3632402

[74] Yongfeng Zhang and Xu Chen. 2020. Explainable recommendation: A survey and new perspectives. *Foundations and Trends® in Information Retrieval* 14, 1 (2020), 1–101.

[75] Yi Zhang, Yiwen Zhang, Dengcheng Yan, Shuiguang Deng, and Yun Yang. 2023. Revisiting graph-based recommender systems from the perspective of variational auto-encoder. *ACM Transactions on Information Systems* 41, 3 (2023), 1–28.

[76] Zhao-Yu Zhang, Xiang-Rong Sheng, Yujing Zhang, Biye Jiang, Shuguang Han, Hongbo Deng, and Bo Zheng. 2022. Towards understanding the overfitting phenomenon of deep click-through rate models. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2671–2680.

[77] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1059–1068.

[78] Chenxu Zhu, Bo Chen, Weinan Zhang, Jincai Lai, Ruiming Tang, Xiuqiang He, Zhenguo Li, and Yong Yu. 2023a. AIM: automatic interaction machine for click-through rate prediction. *IEEE Transactions on Knowledge and Data Engineering* 35, 4 (2023), 3389–3403. DOI: https://doi.org/10.1109/TKDE.2021.3134985

[79] Jieming Zhu, Qinglin Jia, Guohao Cai, Quanyu Dai, Jingjie Li, Zhenhua Dong, Ruiming Tang, and Rui Zhang. 2023b. FINAL: Factorized interaction layer for ctr prediction. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2006–2010.

[80] Jieming Zhu, Jinyang Liu, Shuai Yang, Qi Zhang, and Xiuqiang He. 2021. Open benchmarking for click-through rate prediction. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2759–2769.

Received 21 March 2024; revised 28 June 2024; accepted 6 August 2024