Alignment-Uniformity Aware Feature Representation Learning for CTR Prediction

Fangye Wang[1,2] and Xiaoran Yan[1,3(✉)]

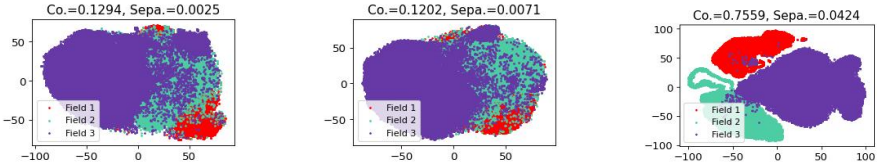
[1] Zhejiang Lab, Hangzhou, China
[2] fywang@zhejianglab.org
[3] yanxr@zhejianglab.com

Abstract. Click-through rate (CTR) prediction has been widely used in online recommendation systems. Most CTR prediction methods enhance performance by modeling complex feature interactions using well-designed network architectures. However, these methods still encounter challenges related to feature distribution imbalance, parameter distribution imbalance, and parameter optimization imbalance. Such imbalances result in suboptimal feature representation, ultimately degrading prediction performance. To address these issues, we propose a novel framework based on contrastive learning, termed Alignment-Uniformity Aware Representation Learning for CTR Prediction (AU4CTR). AU4CTR is compatible with existing CTR prediction models as a model-agnostic representation learning framework. It directly improves feature representation quality by optimizing two core properties, alignment and uniformity, with three auxiliary contrastive losses: the intra-field alignment loss for assigning similar representations to the features in the same field, the inter-field uniformity loss for maximizing the representation distance across different fields, and the interaction alignment loss for aligning the original interaction representation with its perturbed counterpart. Finally, extensive qualitative and quantitative analyses on six public datasets demonstrate the proposed AU4CTR can enhance the feature representation and show remarkable effectiveness and compatibility when integrated with various representative CTR methods.

Keywords: Representation Learning, Contrastive Learning, CTR Prediction

1 Introduction

Recently, many successful deep CTR models have been proposed by designing complex feature interaction (FI) encoders to ensure superior generalization in case of data sparsity for better prediction performance. Representative FI encoders include attention structure [5], cross-network [11,14,15], etc. Despite the remarkable success, those CTR models still face problems with feature distribution imbalance, parameter distribution imbalance, and parameter optimization imbalance. Specifically, the feature frequencies show a long-tailed distribution [13], i.e. low-frequency features make up most of the proportion. Thus,



(a) DCNv2 **(b)** $CL4CTR_{DCNv2}$ **(c)** $AU4CTR_{DCNv2}$

Fig. 1. The visualization of feature representation for different fields learned by base DCNv2, $CL4CTR_{DCNv2}$ and $AU4CTR_{DCNv2}$ over ML-tag dataset. "Co." and "Sepa." are two proposed quantitative metrics: Cohesion and Separation.

low-frequency feature representations are optimized at a few frequencies, leading to sub-optimal representation learning. Then, for most CTR models, the representation parameters take up the vast majority of all model parameters. Non-embedding parameters make up only a tiny portion, causing the embedding parameters to be under-optimized and the non-embedding parameters to be over-optimized. These problems cannot be solved by only designing FI structures and training with supervision signal, leading to sub-optimal feature representation.

This paper seeks to utilize contrastive learning (CL) for CTR models to solve the mentioned issues. CL has shown its superior ability to learn meaningful representations from unlabeled data. [7] have shown that the quality of feature representation is closely related to two properties: alignment and uniformity. For CTR prediction, CL4CTR [13] introduced three CL losses based on these two properties to optimize feature representation, helping base models achieve superior performance. However, its vast computational complexity prevents it from being used in realistic systems. Most importantly, the overly complex loss design leads to its inability to truly optimize the alignment and uniformity properties of the feature representation. As shown in Fig.1, we visualize the 2D plots for all feature representations of the ML-tag dataset. Unfortunately, CL4CTR fails to distinguish learned features from the different fields (represented by various colors) and exhibits similarly to the features learned by base DCNv2. In contrast, our AU4CTR achieves more robust feature representation learning for clustering features from different fields with better prediction performance.

To this end, we propose a feature representation learning framework for CTR models, i.e., **AU4CTR**, which enhances the underlying supervision signal by constructing three CL losses to optimize alignment and uniformity for feature representation directly. As a model-agnostic representation learning framework, AU4CTR is compatible with existing CTR models. Besides the basic CTR model, AU4CTR contains two critical modules: the alignment-uniformity module and the interaction alignment module. Considering the multi-field categorical data in the CTR prediction task, the alignment-uniformity module constructs the alignment and uniformity losses. The alignment loss assigns similar representations to the features in the same field, as they contain similar characteristics, e.g., features "Nike" and "Adidas" in the field "BRAND". In contrast, the uniformity loss encourages the learned feature representations of different fields to be

distributed as far away as possible, as they contain discriminate characteristics, e.g., "Nike" in "BRAND" versus "NYC" in "CITY". Furthermore, the interaction alignment module produces the interaction alignment loss, which aligns the augmented and the original FI representation to boost the generalization ability of the feature representation. The above three CL losses will directly optimize the feature representations in the training process.

In conclusion, the major contributions are summarized as follows: (1) We propose a model-agnostic representation learning framework with CL, namely AU4CTR. It explicitly enhances feature representation during the training process by optimizing alignment and uniformity properties. (2) In AU4CTR, we construct three contrastive losses: the alignment loss for assigning similar representations to the features in the same field, the uniformity loss for maximizing the representation distance across different fields, and the interaction alignment loss for aligning the original interaction representation with its perturbed counterpart. (3) Extensive experiments conducted on six benchmark datasets demonstrate the exceptional effectiveness and compatibility of AU4CTR.

2 Related Work

Deep CTR Prediction Models. Most deep CTR methods follow the same architecture: feature embedding layer, FI layer and prediction layer [11,12]. Modeling complex feature interactions with novel structures has been the core challenge for most research. The factorization machine (FM) [9] is one of the most famous methods for capturing the second-order interaction via the inner product. Then, more advanced FI structures are proposed to capture higher or arbitrarily order interaction info and achieve considerable success. FNN [1] integrates the DNN to capture high-order information. DeepFM [2] and xDeepFM [3] combine the DNN with FM and CIN separately to capture high-order information explicitly. Google proposes DCN [14] and DCN-v2 [15] with cross-vector/matrix networks to extract bounded-degree feature interactions automatically. The attention mechanism is widely adopted as an FI encoder [5, 10, 12]. Furthermore, many CTR models have realized the importance of feature representation and made efforts to adaptively refine feature representations before capturing complex FI information [4, 12, 16]. Although those methods achieve great success, they only optimize models with the supervision signal and fail to solve those imbalance problems, which hinders their performance.

3 Our Proposed Method: AU4CTR

As shown in Fig.2, AU4CTR contains three key components: (a) The basic CTR model, (b) the Alignment-Uniformity module, and (c) the Interaction Alignment module. Notably, AU4CTR operates independently of the underlying CTR models, enhancing their prediction accuracy by refining the feature representation without altering the structure of these base models.

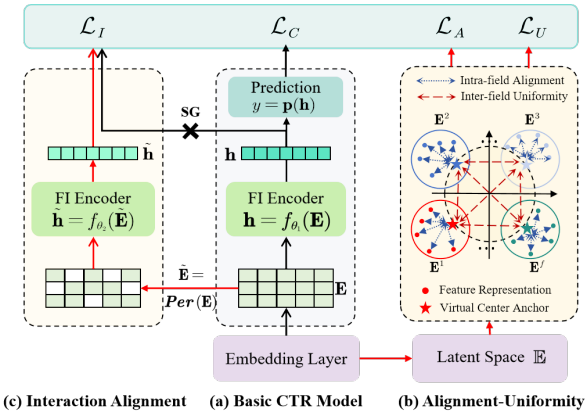


Fig. 2. The architecture of AU4CTR framework. **SG** indicates the stop-gradient.

3.1 The Architecture of CTR Prediction

CTR prediction is a binary classification task based on multi-field categorical data. Each instance $\mathbf{x}_i$ contains F fields. It is represented by $(\mathbf{x}_i, y_i)$, where $y_i \in \{0, 1\}$ is the true label. $\mathbf{x}_i$ is the multi-categorical data. Each dataset contains N instances with M features. The CTR models aim to predict the click probability $P(y_i = 1|\mathbf{x}_i)$. Most CTR models consist of three layers:

Embedding layer transforms the sparse features $\mathbf{x}_i$ into a dense embedding matrix $\mathbf{E} = [\mathbf{v}^1; \mathbf{v}^2; ...; \mathbf{v}^F] \in \mathbb{R}^{F \times d}$, where d is the dimension size. We represent the total feature representations map with $\mathbb{E} = [\mathbf{E}^1, \mathbf{E}^2, ..., \mathbf{E}^F] \in \mathbb{R}^{M \times d}$, where $\mathbf{E}^f$ is the feature representation subset that belong to the f-th field $f \in \{1, 2, ..., F\}$. And $|\mathbf{E}^f|$ is the number of features belonging to field f.

Feature interaction layer captures effective interaction information with complex interaction encoders, represented by $f_\theta(\cdot)$, θ is the trained parameter. The encoder $f_\theta(\cdot)$ produces a compact representation h_i.

Prediction layer outputs the final prediction probability $\hat{y}_i = \mathbf{p}(y_i = 1|\mathbf{h}_i)$ based on the compact representation $\mathbf{h}_i$.

Finally, the CTR model are trained with the Logloss $\mathcal{L}_C$:

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

3.2 The Alignment and Uniformity Loss

Computing contrastive loss is usually based on either positive or negative sample pairs [7]. For the multi-field categorical data in the CTR prediction task, there are inherently positive and negative sample pairs. Specifically, the features belonging to the same field contain similar characteristics and significant differences exist between the features of different fields and the feature. Therefore, in the latent space for feature representation, features in the same field should be as similar as possible, while the distance between features in distinct fields should be as far as possible. Hence, we construct two training constraints: intra-field

alignment loss and inter-field uniformity loss, adding feature optimization for all features in each batch of the training process, effectively solving the problems of under-optimizing low-frequency features and embedding parameters.

Intra-field Alignment Loss Alignment encourages the feature's representations of the same field to be as similar as possible. However, suppose any two features in the domain are compared during the loss computation. In that case, the computational consumption rises quadratically with the number of features, thus creating a computational disaster for large-scale datasets, as CL4CTR [13] has done. To solve this problem, we design an approach with only linear complexity. Specifically, we calculate a virtual center anchor for each field, computed by averaging the normalized representation for all features in the same fields. For example, the virtual center anchor for f-th field is calculated by:

$$\mathscr{V}_f = \frac{1}{|\mathbf{E}^f|} \sum\nolimits_{i=1}^{|\mathbf{E}^f|} \mathbf{E}^f_{(i)} / \left\| \mathbf{E}^f_{(i)} \right\|_2, \tag{2}$$

where $\mathbf{E}^f_{(i)}$ is the i-th feature belonging to field f. Then, AU4CTR strengthens the intra-field alignment by minimizing the distance between any feature $\mathbf{E}^f_{(i)}$ to its corresponding virtual center anchor $\mathscr{V}_f$. Thus, the number of comparisons is reduced to the number of features. The alignment loss is formulated as follows:

$$\mathcal{L}_A = \frac{1}{N} \sum\nolimits_{f=1}^{F} \sum\nolimits_{i=1}^{|\mathbf{E}^f|} \left\| \mathbf{E}^f_{(i)} - \mathscr{V}_f \right\|_2^2. \tag{3}$$

Inter-field Uniformity Loss Uniformity aims to maximize the feature distance among different fields. The inter-field uniformity loss is also calculated based on the virtual center anchors. It avoids calculating the L2 distances between pairs of features from different fields one by one, which greatly reduces the computational complexity. Formally, the uniformity loss is calculated by:

$$\mathcal{L}_U = \log \left[\frac{1}{K} \sum\nolimits_{\mathscr{V}_{f_1}, \mathscr{V}_{f_2} \in \mathbb{V}} e^{-\left\| \mathscr{V}_{f_1} - \mathscr{V}_{f_2} \right\|_2^2} \right], \tag{4}$$

where $\mathbb{V} = [\mathscr{V}_1, \mathscr{V}_2, ..., \mathscr{V}_F]$ is the collection for all F virtual center anchors.

3.3 The Interaction Alignment Loss

As illustrated in Fig.2(c), the interaction alignment loss is computed through three steps. First, a perturbation operation is applied to the instance embedding $\mathbf{E}$ for generating a duplicate embedding $\tilde{E}$. Then, the FI encoder $f_{\theta_1}(\cdot)$ and its duplicate FI encoder $f_{\theta_2}(\cdot)$ with different parameters are used to generate two compact interaction representations $\mathbf{h}$ and $\tilde{\mathbf{h}}$, respectively. Finally, the interaction alignment loss is calculated by aligning the perturbed interaction

representation $\tilde{\mathbf{h}}$ with the original one $\mathbf{h}$. The three steps are formulated by:

$$\tilde{\mathbf{E}} = Per(\mathbf{E}) = \mathbf{E} \cdot \mathbf{I}, \mathbf{I} \sim \text{Bernoulli}(p) \in \mathbb{R}^{F \times d}, \tag{5}$$

$$\mathbf{h} = f_{\theta_1}(\mathbf{E}), \tilde{\mathbf{h}} = f_{\theta_2}(\tilde{\mathbf{E}}), \tag{6}$$

$$\mathcal{L}_I = \frac{1}{B} \sum_{i=1}^{B} \left\| SG(f_{\theta_1}(\mathbf{E})) - f_{\theta_2}(\tilde{\mathbf{E}}) \right\|_2^2. \tag{7}$$

In Equ.5, p is the hyper-parameter, called mask ratio. Notably, the $SG(\cdot)$ representation stop-gradient operator, which are used to prevent collapse solutions. Meanwhile, the utilization of $SG(\cdot)$ avoids optimizing the parameters of the FI encoder $f_{\theta_1}(\cdot)$ within the base CTR model. By optimizing the interaction alignment loss $\mathcal{L}_I$, the information density and generalization ability of the relevant elements in the feature representation can be significantly improved.

3.4 Objective Function

In the training process, we jointly optimize the CTR prediction loss $\mathcal{L}_C$ with three CL losses ($\mathcal{L}_A$, $\mathcal{L}_U$ and $\mathcal{L}_I$). Meanwhile, we add two weight coefficients, λ_{a-u} and λ_i, to balance CL losses. Formally, the objective function is given by:

$$\mathcal{L} = \mathcal{L}_C + \lambda_{a-u} \cdot (\mathcal{L}_A + \mathcal{L}_U) + \lambda_i \cdot \mathcal{L}_I. \tag{8}$$

3.5 Cohesion and Separation

Here, we introduce two metrics, cohesion and separation, to quantify the alignment and uniformity for multi-field categorical data in CTR prediction task. Cohesion measures how closely related the representations are within the same field, indicating the internal consistency of a cluster; Separation assesses how distinct or well-separated one field cluster is from others, reflecting the clarity of boundaries between different clusters. Formally, they are calculated by:

$$Cohesion = \frac{1}{F} \sum\nolimits_{f=1}^{F} \left(\frac{1}{K_2} \sum\nolimits_{\mathbf{v}_i, \mathbf{v}_j \in \mathbf{E}^f} sim\left(\mathbf{v}_i, \mathbf{v}_j\right) \right), \tag{9}$$

$$Separation = \frac{1}{K} \sum\nolimits_{\mathscr{v}_{f_1}, \mathscr{v}_{f_2} \in \mathbb{V}} \left\| \mathscr{v}_{f_1} - \mathscr{v}_{f_2} \right\|_2^2. \tag{10}$$

4 Experiments

4.1 Experimental Setup

Datasets. We conduct experiments on six datasets, i.e., Avazu, Malware, ML1M, ML-tag, SafeDriver, Frappe. The main statistics of these six datasets and more detailed descriptions can be found in the public code link. And we use AUC and Logloss (LL) to evaluate the prediction performance.

Table 1. Overall performance comparison in the six datasets.

Datasets	Avazu		Malware		ML1M		SafeDriver		ML-tag		Frappe		ΔAUC	ΔLL
Models	AUC↑	LL↓	AUC↑	LL↓	AUC↑	LL↓	AUC↑	LL↓	AUC↑	LL↓	AUC↑	LL↓	↑	↓
FM	0.7829	0.3785	0.7107	0.6196	0.8006	0.5343	0.6255	0.1580	0.9415	0.2809	0.9631	0.2142	-0.0159	8.79%
FwFM	0.7857	0.3772	0.7367	0.5980	0.8061	0.5269	0.6298	0.1562	0.9468	0.2636	0.9702	0.1891	-0.0073	4.06%
DIFM	0.7887	0.3748	0.7424	0.5926	0.8104	0.5211	0.6365	0.1543	0.9477	0.2627	0.9768	0.1762	-0.0028	2.07%
IPNN	0.7890	0.3745	0.7433	0.5921	0.8108	0.5200	0.6344	0.1557	0.9496	0.2649	0.9792	0.1747	-0.0022	2.42%
OPNN	0.7892	0.3746	0.7436	0.5918	0.8113	0.5190	0.6354	0.1510	0.9497	0.2656	0.9802	0.1674	-0.0017	1.26%
DNN	0.7886	0.3748	0.7424	0.5928	0.8103	0.5210	0.6346	0.1535	0.9484	0.2658	0.9789	0.1729	-0.0027	1.32%
CNv2	0.7893	0.3745	0.7383	0.5959	0.8116	0.5213	0.6359	0.1522	0.9459	0.2710	0.9787	0.1716	-0.0033	2.02%
DeepFM	0.7891	0.3745	0.7424	0.5925	0.8108	0.5206	0.6360	0.1531	0.9489	0.2743	0.9772	0.1795	-0.0025	3.73%
DCN	0.7890	0.3745	0.7434	0.5917	0.8114	0.5191	0.6357	0.1522	0.9505	0.2594	0.9780	0.1703	-0.0019	0.93%
xDeepFM	0.7893	0.3744	0.7442	0.5911	0.8112	0.5193	0.6384	0.1520	0.9489	0.2715	0.9795	0.1726	-0.0013	1.92%
DCNv2	0.7898	0.3743	0.7433	0.5920	0.8125	0.5176	0.6424	0.1508	0.9532	0.2507	0.9782	0.1688	-	-
NON	0.7889	0.3747	0.7426	0.5925	0.8117	0.5175	0.6412	0.1511	0.9536	0.2503	0.9786	0.1698	-0.0005	-0.13%
$CL4CTR_{DCNv2}$	OOM	OOM	OOM	OOM	0.8144	0.5151	0.6438	0.1498	0.9543	0.2474	0.9794	0.1583	-	-
$AU4CTR_{DCNv2}$	**0.7906**	**0.3734**	**0.7459**	**0.5896**	**0.8161**	**0.5134**	**0.6454**	**0.1479**	**0.9555**	**0.2449**	**0.9805**	**0.1573**	0.0024	-2.08%
Rel.Imp	0.0008	-0.24%	0.0017	-0.33%	0.0017	-0.33%	0.0016	-1.28%	0.0012	-0.28%	0.0011	-0.64%	-	-

Comparison methods. We compare AU4CTR with three groups of representative models: a) The shallow methods, i.e., FM [9], FwFM [6], DIFM [4]; b) The high-order methods, i.e., IPNN [8], OPNN [8], DNN [1], CNv2 [15]; c) The parallel/ensemble methods, i.e., DeepFM [2], xDeepFM [3], DCN [14], NON [5], DCNv2 [15], CL4CTR [13]. We choose DCNv2 as the base model for following descriptions, represented by $AU4CTR_{DCNv2}$.

Implementation Details. We implement our AU4CTR and other baselines with Pytorch. We conduct most experiments following CL4CTR [13]. More detailed experimental setting can be found on the public code[4].

4.2 Performance Comparison

From Table 1, we know that $AU4CTR_{DCNv2}$ consistently outperforms all baselines across all six datasets. Specifically, $AU4CTR_{DCNv2}$ significantly outperforms the strongest baselines, with AUC improvements ranging from 0.0008 to 0.0017 (ranging from 0.24% to 1.28% in terms of LL). These improvements validate the effectiveness of AU4CTR. Additionally, $AU4CTR_{DCNv2}$ achieves the highest average performance boost (ΔAUC=0.0024, ΔLL=2.08%), demonstrating the strong generalization ability of AU4CTR across various datasets.

4.3 The Compatibility Ability of AU4CTR

From Table 2, we assess the compatibility of AU4CTR. The results show that all enhanced models significantly outperform their baseline counterparts. Meanwhile, we note that although some base models don't perform very well, their results gain a huge improvement after the integration of AU4CTR. It demonstrates that sub-optimal representation relying solely on the base supervised loss ($\mathcal{L}_C$) limits the performance of base models, and it is necessary to boost the quality of feature representations with additional operations.

[4] The code is available at https://github.com/ProjectCodeWfy/AU4CTR.

Model	ML1M		SafeDriver		ML-tag		Frappe	
	AUC	Logloss	AUC	Logloss	AUC	Logloss	AUC	Logloss
FwFM	0.8061	0.5269	0.6298	0.1562	0.9468	0.2636	0.9702	0.1891
+AU4CTR	0.8112	0.5209	0.6356	0.1480	0.9524	0.2611	0.9785	0.1657
IPNN	0.8108	0.5200	0.6344	0.1557	0.9496	0.2649	0.9793	0.1747
+AU4CTR	0.8136	0.5159	0.6419	0.1516	0.9547	0.2451	0.9803	0.1642
DNN	0.8103	0.5210	0.6346	0.1535	0.9484	0.2658	0.9789	0.1729
+AU4CTR	0.8128	0.5180	0.6358	0.1502	0.9566	0.2403	0.9803	0.1668
DCN	0.8114	0.5191	0.6357	0.1522	0.9505	0.2594	0.9780	0.1703
+AU4CTR	0.8122	0.5172	0.6410	0.1508	0.9539	0.2471	0.9800	0.1651
DCNv2	0.8125	0.5176	0.6424	0.1508	0.9518	0.2483	0.9782	0.1688
+AU4CTR	0.8161	0.5134	0.6475	0.1495	0.9555	0.2449	0.9805	0.1573
Ave.Imp	**0.0038**	**-1.04%**	**0.0049**	**-2.11%**	**0.0049**	**-5.85%**	**0.0027**	**-6.23%**

Table 2. Compatibility analysis.

Model		ML1M		ML-tag		Frappe	
	Loss Function	AUC	LL	AUC	LL	AUC	LL
DNN	$\mathcal{L}_C$	0.8103	0.5210	0.9484	0.2658	0.9789	0.1729
	$+\mathcal{L}_I$	0.8137	0.5161	0.9526	0.2543	0.9794	0.1676
	$+(\mathcal{L}_A+\mathcal{L}_U)$	0.8144	0.5153	0.9536	0.2482	0.9797	0.1658
	$+\mathcal{L}_I+(\mathcal{L}_A+\mathcal{L}_U)$	**0.8151**	**0.5134**	**0.9542**	**0.2478**	**0.9803**	**0.1668**
IPNN	$\mathcal{L}_C$	0.8108	0.5200	0.9496	0.2649	0.9792	0.1747
	$+\mathcal{L}_I$	0.8133	0.5161	0.9526	0.2521	0.9801	0.1665
	$+(\mathcal{L}_A+\mathcal{L}_U)$	0.8128	0.5163	0.9561	0.2453	0.9800	0.1668
	$+\mathcal{L}_I+(\mathcal{L}_A+\mathcal{L}_U)$	**0.8136**	**0.5159**	**0.9568**	**0.2438**	**0.9806**	**0.1642**
DCNv2	$\mathcal{L}_C$	0.8125	0.5176	0.9518	0.2483	0.9782	0.1688
	$+\mathcal{L}_I$	0.8155	0.5139	0.9527	0.2460	0.9795	0.1643
	$+(\mathcal{L}_A+\mathcal{L}_U)$	0.8156	0.5141	0.9547	0.2459	0.9798	0.1625
	$+\mathcal{L}_I+(\mathcal{L}_A+\mathcal{L}_U)$	**0.8161**	**0.5134**	**0.9555**	**0.2449**	**0.9805**	**0.1573**

Table 3. Ablation study.

Model	DCNv2		$AU4CTR_{DCNv2}$	
Dataset	Cohesion	Separation	Cohesion	Separation
Avazu	0.0631	0.0155	0.5566	0.0537
Malware	-0.1129	0.0047	0.9391	0.0082
ML1M	-0.0020	0.0005	0.7464	0.0135
SafeDriver	-0.1968	0.0107	0.9452	0.0632
ML-tag	0.1294	0.0025	0.7559	0.0424
Frappe	0.2193	0.0011	0.8612	0.0284

Table 4. The cohesion and separation for the learned representation.

Training Time Per Epoch (s)						
Datasets	Avazu	Malware	ML-tag	ML1M	SafeDriver	Frappe
DCNv2	311.6	39.2	39.8	13.9	32.4	14.2
+CL4CTR	OOM	OOM	1020	365.4	35.9	57.8
+AU4CTR	401.2	118.2	40.2	14.5	32.9	14.5
Imp.(%)	-	-	96.1%	96.0%	8.4%	74.9%
Memory Cost (MB)						
Datasets	Avazu	Malware	ML-tag	ML1M	SafeDriver	Frappe
DCNv2	1136	1134	620	602	806	624
+CL4CTR	OOM	OOM	6892	5048	3346	2396
+AU4CTR	1584	1380	772	736	1168	784
Imp.(%)	-	-	88.8%	85.4%	65.1%	67.3%

Table 5. Comparisons of the training time and memory cost.

4.4 Ablation Study

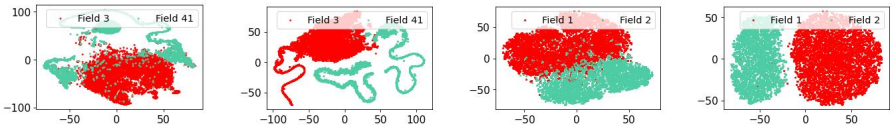
In Table 3 show the impact of $\mathcal{L}_A + \mathcal{L}_U$ and $\mathcal{L}_I$ on the performance of AU4CTR by gradually removing each from AU4CTR. The findings are as follows: First, incorporating either the $\mathcal{L}_A + \mathcal{L}_U$ or $\mathcal{L}_I$ contrastive loss alone results in performance improvements over the original CTR prediction models. Secondly, the complete AU4CTR framework, which includes all $\mathcal{L}_A + \mathcal{L}_U$ and $\mathcal{L}_I$ achieves the best performance on three datasets. It confirm that three CL signals are all effective, and their combined use is essential for optimal performance.

4.5 Study of Alignment and Uniformity

We conduct visualization and quantitative analyses to validate AU4CTR can improve two core properties of feature representation: alignment and uniformity.

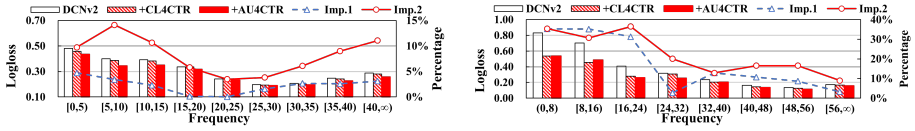
Visualization Analysis. Fig.3 visualize all feature representations of the DCNv2 before and after the integration of AU4CTR. We choose two categorical fields and create the 2D plots using t-SNE. We can find that the learned representation by base DCNv2 (Figures 3(a) and 3(c)), their visualized plots of the two fields are intermixed, lacking distinct clusters and boundaries. In contrast, after integrating AU4CTR, the feature representations from different fields become easily distinguishable with clear boundaries,(Figures 3(b) and 3(d)).

Quantitative analysis. Table 4 calculate the cohesion and separation metrics for both the DCNv2 model and the $CL4CTR_{DCNv2}$. It clearly show substantial increases in both cohesion and separation, with values several times higher than those of the base DCNv2 model, indicating AU4CTR significantly enhances the intra-field similarity and the inter-field distance between features.



(a) Malware (before) **(b)** Malware (after) **(c)** ML1M (before) **(d)** ML1M (after)

Fig. 3. Visualization of feature representation learned by DCNv2 model before and after implementing AU4CTR framework on Malware and ML1M datasets.



(a) The Logloss of ML-tag **(b)** The Logloss of Frappe

Fig. 4. Performance w.r.t **feature frequency**. Imp.1 and Imp.2 indicate the relative improvement of CL4CTR and AU4CTR compared to DCNv2.

4.6 Comparisons with CL4CTR

We compare AU4CTR with SOTA CL4CTR [13] across the following aspects:

Comparisons of training efficiency. Table 5 shows the training time per epoch and memory usage. Applying CL4CTR results in a massive increase in training time and memory usage. In contrast, AU4CTR substantially reduces training time and memory usage, with an average improvement of 68.8% and 76.6%, respectively. It benefits from utilizing virtual center nodes while computing the alignment and uniformity losses, dramatically reducing the number of comparisons while giving all features a clear goal in the learning process.

Optimization of different frequency features. In Fig.4, we calculate the Logloss based on the different feature frequency subsets. First, subsets containing low-frequency features typically perform worse on DCNv2. Second, after applying CL4CTR and AU4CTR, the prediction accuracy is improved, with a relatively more significant accuracy boost. However, the accuracy of low-frequency subsets still lags behind that of high-frequency subsets, indicating CL4CTR and AU4CTR cannot eliminate the suboptimal results caused by feature distribution imbalance. Finally, AU4CTR achieves the best results on most subsets.

5 Conclusion

This paper presents a model-agnostic framework AU4CTR, designed to enhance feature representation by directly optimizing two essential properties: alignment and uniformity. Specifically, AU4CTR introduces three contrastive representation learning losses: the intra-field alignment loss, the inter-field uniformity loss, and the interaction alignment loss. Finally, extensive experiments further validate the effectiveness and compatibility of the AU4CTR framework.

Acknowledgment

This research was supported by the National Key R&D Program of China (No.2022YFF0712400), the Leading Innovation and Entrepreneurship Team of Zhejiang Province of China (Grant No.2023R01008), and the Zhejiang Province Key Research and Development Plan (No.2024SSYS0010).

References

1. Covington, P., Adams, J., Sargin, E.: Deep neural networks for youtube recommendations. In: Proceedings of the 10th ACM conference on recommender systems. pp. 191–198 (2016)
2. Guo, H., Tang, R., Ye, Y., Li, Z., He, X.: Deepfm: a factorization-machine based neural network for ctr prediction. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence (2017)
3. Lian, J., Zhou, X., Zhang, F., Chen, Z., Xie, X., Sun, G.: xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In: Proceedings of the 24th ACM SIGKDD International Conference. pp. 1754–1763 (2018)
4. Lu, W., Yu, Y., Chang, Y., Wang, Z., Li, C., Yuan, B.: A dual input-aware factorization machine for ctr prediction. In: IJCAI. pp. 3139–3145 (2020)
5. Luo, Y., Zhou, H., Tu, W.W., Chen, Y., Dai, W., Yang, Q.: Network on network for tabular data classification in real-world applications. In: Proceedings of the 43rd International ACM SIGIR Conference. pp. 2317–2326 (2020)
6. Pan, J., Xu, J., Ruiz, A.L., Zhao, W., Pan, S., Sun, Y., Lu, Q.: Field-weighted factorization machines for click-through rate prediction in display advertising. In: Proceedings of the 2018 World Wide Web Conference. pp. 1349–1357 (2018)
7. Pu, S., Zhao, K., Zheng, M.: Alignment-uniformity aware representation learning for zero-shot video classification. In: CVPR. pp. 19968–19977 (2022)
8. Qu, Y., Fang, B., Zhang, W., Tang, R., Niu, M., Guo, H., Yu, Y., He, X.: Product-based neural networks for user response prediction over multi-field categorical data. ACM Transactions on Information Systems (TOIS) **37**(1), 1–35 (2018)
9. Rendle, S.: Factorization machines. In: 2010 IEEE International conference on data mining. pp. 995–1000. IEEE (2010)
10. Wang, F., Gu, H., Li, D., Lu, T., Zhang, P., Gu, N.: Mcrf: Enhancing ctr prediction models via multi-channel feature refinement framework. In: DASFAA. pp. 359–374. Springer (2022)
11. Wang, F., Gu, H., Li, D., Lu, T., Zhang, P., Gu, N.: Towards deeper, lighter and interpretable cross network for ctr prediction. In: CIKM. pp. 2523–2533 (2023)
12. Wang, F., Wang, Y., Li, D., Gu, H., Lu, T., Zhang, P., Gu, N.: Enhancing ctr prediction with context-aware feature representation learning. In: Proceedings of the 44th International ACM SIGIR Conference. pp. 343–352 (2022)
13. Wang, F., Wang, Y., Li, D., Gu, H., Lu, T., Zhang, P., Gu, N.: Cl4ctr: A contrastive learning framework for ctr prediction. In: WSDM. pp. 805–813 (2023)
14. Wang, R., Fu, B., Fu, G., Wang, M.: Deep & cross network for ad click predictions. In: Proceedings of the ADKDD'17, pp. 1–7 (2017)
15. Wang, R., Shivanna, R., Cheng, D., Jain, S., Lin, D., Hong, L., Chi, E.: Dcn v2: Improved deep & cross network and practical lessons for web-scale learning to rank systems. In: Proceedings of the web conference 2021. pp. 1785–1797 (2021)
16. Yu, Y., Wang, Z., Yuan, B.: An input-aware factorization machine for sparse prediction. In: IJCAI. pp. 1466–1472 (2019)