Ads Recommendation in a Collapsed and Entangled World

Junwei Pan
Tencent Inc.
jonaspan@tencent.com

Wei Xue
Tencent Inc.
weixue@tencent.com

Ximei Wang
Tencent Inc.
messixmwang@tencent.com

Haibin Yu
Tencent Inc.
nathanhbyu@tencent.com

Xun Liu
Tencent Inc.
reubenliu@tencent.com

Shijie Quan
Tencent Inc.
justinquan@tencent.com

Xueming Qiu
Tencent Inc.
xuemingqiu@tencent.com

Dapeng Liu
Tencent Inc.
rocliu@tencent.com

Lei Xiao
Tencent Inc.
shawnxiao@tencent.com

Jie Jiang
Tencent Inc.
zeus@tencent.com

ABSTRACT

We present Tencent's ads recommendation system and examine the challenges and practices of learning appropriate recommendation representations. Our study begins by showcasing our approaches to preserving prior knowledge when encoding features of diverse types into embedding representations. We specifically address sequence features, numeric features, and pre-trained embedding features. Subsequently, we delve into two crucial challenges related to feature representation: the *dimensional collapse* of embeddings and the *interest entanglement* across different tasks or scenarios. We propose several practical approaches to address these challenges that result in robust and disentangled recommendation representations. We then explore several training techniques to facilitate model optimization, reduce bias, and enhance exploration. Additionally, we introduce three analysis tools that enable us to study feature correlation, dimensional collapse, and interest entanglement. This work builds upon the continuous efforts of Tencent's ads recommendation team over the past decade. It summarizes general design principles and presents a series of readily applicable solutions and analysis tools. The reported performance is based on our online advertising platform, which handles hundreds of billions of requests daily and serves millions of ads to billions of users.

CCS CONCEPTS

• **Information systems** → **Display advertising**; • **Computing methodologies** → **Neural networks**; **Factorization methods**.

KDD '24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 979-8-4007-0490-1/24/08…$15.00
https://doi.org/10.1145/3637528.3671607

KEYWORDS

Recommendation Systems, Representation Learning, Dimensional Collapse, Disentangled Learning, User Interest Modeling

ACM Reference Format:
Junwei Pan, Wei Xue, Ximei Wang, Haibin Yu, Xun Liu, Shijie Quan, Xueming Qiu, Dapeng Liu, Lei Xiao, and Jie Jiang. 2024. Ads Recommendation in a Collapsed and Entangled World. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3637528.3671607

1 INTRODUCTION

The online advertising industry, valued at billions of dollars, is a remarkable example of the successful application of machine learning. Various advertising formats, including sponsored search advertising, contextual advertising, display advertising, and micro-video advertising, heavily rely on the accurate, efficient, and reliable prediction of ads click-through or conversion rates using learned models.

Over the past decade, deep learning has achieved remarkable success in diverse domains, including computer vision (CV) [23, 34], natural language processing (NLP) [1, 16, 62], and recommendation systems [41, 77]. The effectiveness of deep learning critically depends on the selection of appropriate data representations [3, 67, 68]. Researchers have extensively explored various aspects of representation learning in CV and NLP. These investigations have focused on topics such as priors [62], smoothness and the curse of dimensionality [5], depth and abstraction [4], disentangling factors of variations [68], and the uniformity of representations [28, 30].

In the field of recommendation systems, numerous works have focused on representation learning techniques to handle various types of features [9, 19, 32, 79, 81], capture feature correlations through explicit or implicit feature interactions [12, 20, 37, 45, 51, 60, 66], address the entangled interest within users' complex behaviors [69], particularly in multi-task [43, 59] or multi-scenario [7, 54, 83] settings, and enhance data representation through self-supervised

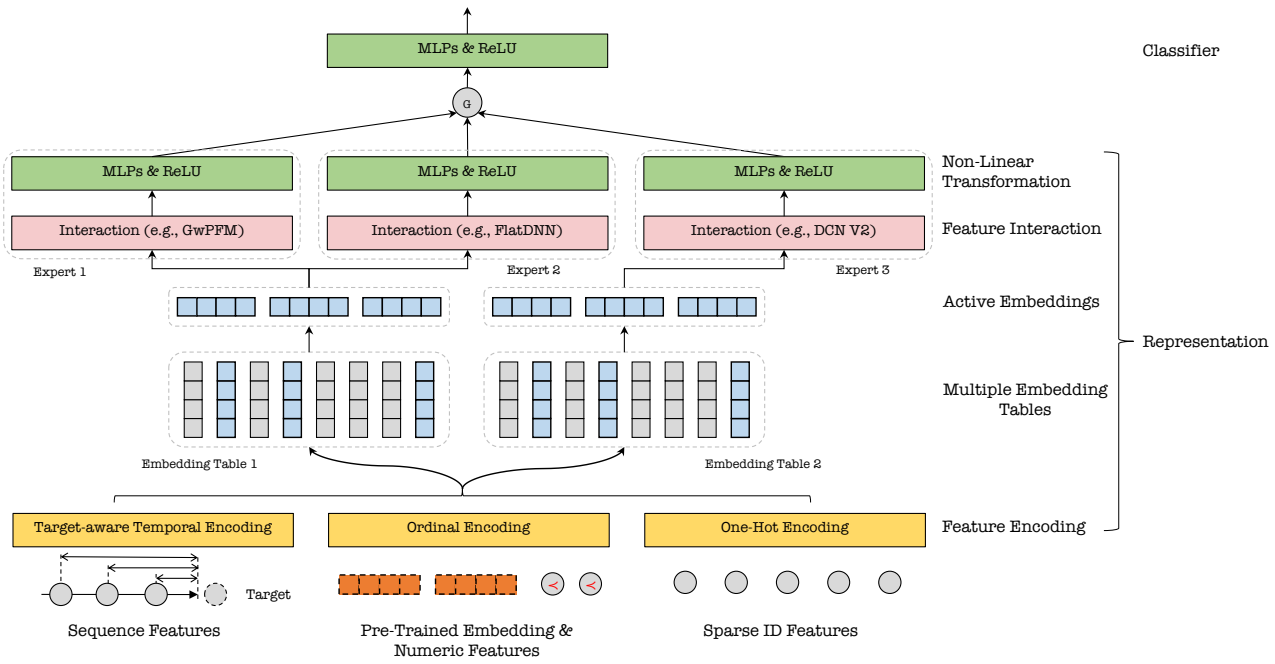


Figure 1: Architecture of our Heterogeneous Mixture-of-Experts with Multi-Embedding for single-task learning, which consists of four key modules: feature encoding, multi-embedding lookup, experts (feature interactions and MLPs), and classification towers.

learning [63, 84]. Despite the progress made in these representation-oriented works, several fundamental questions regarding representation learning in large-scale real-world ads recommenders remain unanswered.

- *Priors for Representation:* Real-world systems encompass various types of features from diverse sources, including sequence features (*e.g.*, user click/conversion history), numeric features (*e.g.*, semantic-preserving ad IDs), and embedding features from pre-trained external models (*e.g.*, GNN or LLM). Preserving the inherent priors of these features when encoding them in recommendation systems is crucial.
- *Dimensional Collapse:* The encoding process maps all features into embeddings, typically represented as K-dimensional vectors, and are learned during model training. However, we observe that the embeddings of many fields tend to occupy a lower-dimensional subspace instead of fully utilizing the available K-dimensional space. Such dimensional collapse not only leads to parameter wastage but also limits the scalability of recommendation models.
- *Interest Entanglement:* User responses in ads recommender systems are determined by complex underlying factors, particularly when multiple tasks or scenarios are learned simultaneously. Existing shared-embedding approaches [6, 43, 59] may fail to disentangle these factors adequately, as they rely on a single entangled embedding for each feature.

This paper presents our practices for addressing these challenges. The remaining sections of the paper are organized as follows: Section 2 provides an overview of our model architecture, giving a high-level understanding of the system. Section 3 focuses on the encoding techniques used to integrate temporal, ordinal, and distance priors of different feature types into the representation. Section 4 delves into the root causes of the embedding dimensional collapse and proposes several solutions to mitigate this issue. Section 5 explores the challenge of interest entanglement across various tasks and scenarios and our solutions. Section 6 presents various model training techniques. Finally, Section 7 introduces a set of off-the-shelf tools designed to facilitate the analysis of feature correlations, dimensional collapse, and interest entanglement. Due to space limitations, this paper cannot provide a detailed description of each approach. For more in-depth information, please refer to the corresponding paper cited in each section.

2 BRIEF SYSTEM OVERVIEW

The overall architecture of our ads recommendation model for single-task learning is illustrated in Fig. 1. For the multi-task learning model architecture, please refer to Fig. 4. Our model follows the widely adopted Embedding & Explicit Interaction framework [41, 77], which consists of four key modules: feature encoding, multi-embedding lookup, experts (feature interactions and MLPs), and classification towers. In the feature encoding module, we apply specific encoding methods tailored to various feature types in our system. Next, based on the encoded IDs obtained from the feature encoding module, multiple embeddings are looked up from individual embedding tables for each feature. Within the expert module, embeddings from the same table are explicitly interacted with one another. The outputs of the expert module are then passed through Multi-Layer Perceptrons (MLPs) with non-linear transformations.

The classification towers receive the gate-weighted sum of the outputs from the experts. Finally, the sigmoid activation function is applied to generate the final prediction.

In the case of single-task learning, such as Click-Through Rate (CTR) prediction, our model employs a single tower, as depicted in Fig. 1. However, in the context of multi-task learning (MTL), such as Conversion Rate (CVR) prediction, where each conversion type is treated as an individual task [48], our model utilizes multiple towers and corresponding gates. Each tower is dedicated to a specific group of conversion types, allowing for task-specific predictions. To address the challenge of interest entanglement that arises in the MTL setting, further evolution of the model architecture to disentangle user interest is presented in Section 5.

Our team is responsible for ads recommendation across all modules, including retrieval and pre-ranking, CTR prediction (pCTR), (shallow) conversion prediction (pCVR) of various conversion types, deep conversion prediction (pDCVR), and Long-time Value prediction (pLTV). There are many commonalities regarding the model design principle among these modules, and we mainly discuss the pCTR and pCVR as representative modules for single-task and multi-task learning, respectively. Our models serve various ads recommendation scenarios within Tencent, encompassing Moments (social stream), Channels (micro-video stream), Official Accounts (subscription), Tencent News, Tencent Video (long-video platform), and Demand Side Platform.

3 FEATURE ENCODING

In industrial ads recommendation systems, features are generated from many sources and belong to different types, such as sequence, numeric, and embedding features. When encoding these features, we'd like to preserve their inherent temporal, ordinal, or distance (similarity) priors as much as possible.

3.1 Sequence Features

A user's history behaviors reflect her interest, making them critical in recommendations. One key characteristic of such features is that there are strong semantic and temporal correlations between these behaviors and the target [82]. For example, given a target ad, those behaviors that are either semantically related (*e.g.*, belonging to the same category with the target ad) or temporally close to the target are more informative to predict the user's response to the target item.

We propose Temporal Interest Module (TIM) [82] to learn the quadruple semantic-temporal correlation between *(behavior semantic, target semantic, behavior temporal, target temporal)*. Specifically, in addition to the semantic encoding [17, 80, 81], TIM leverages Target-aware Temporal Encoding for each behavior, *e.g.*, the *relative position* or *time interval* between each behavior and target. Furthermore, to capture the *quadruple correlation*, TIM employs Target-aware Attention and Target-aware Representation to interact behaviors with the target in both attention and representation, resulting in *explicit 4-way interaction*(shown in Fig. 2(a)). Mathematically, the encoding of user behavior sequence $\mathcal{H}$ can be formulated as:

$$\boldsymbol{u}_{\text{TIM}} = \sum_{X_i \in \mathcal{H}} \alpha(\tilde{\boldsymbol{e}}_i, \tilde{\boldsymbol{v}}_t) \cdot (\tilde{\boldsymbol{e}}_i \odot \tilde{\boldsymbol{v}}_t) \tag{1}$$

where $\alpha(\tilde{\boldsymbol{e}}_i, \tilde{\boldsymbol{v}}_t)$ denotes the *target-aware attention* between each behavior i and target t, $(\tilde{\boldsymbol{e}}_i \odot \tilde{\boldsymbol{v}}_t)$ denotes the *target-aware representation*, and $\tilde{\boldsymbol{e}}_i = \boldsymbol{e}_i \oplus \boldsymbol{p}_{f(X_i)}$ denotes the *temporally encoded embedding* of the i-th behavior, which is an element-wise summation of semantic embedding $\boldsymbol{e}_i$ and target-aware temporal encoding $\boldsymbol{p}_{f(X_i)}$, *i.e.*, the embedding of either the relative position of each behavior regarding the target, or the discretized time interval. Please note that the target-aware representation $\tilde{\boldsymbol{e}}_i \odot \tilde{\boldsymbol{v}}_t$ acts like a feature interaction layer to explicitly interact the behavior feature with the target, as done in other FM-based explicit feature interaction models [31, 37, 49, 52, 66]. The importance of such explicit behavior-target interaction in the representation was also emphasized in a recent work [74].

Deployment Details. In practice, we adopt both relative position and time interval for temporal encoding. The output of TIM is concatenated with the output of the feature interaction module, *e.g.*, DCN V2 [66] or GwPFM (will be discussed later). We apply TIM on the user's click/conversion category sequence features in various click and conversion prediction tasks across multiple scenarios. TIM brings a 1.93% Gross Merchandise Value (GMV) lift in WeChat pCTR and a 2.45% GMV lift in Game and e-Commerce pLTV. We observe the model learns *much stronger decaying patterns in the time interval embeddings than the relative position embedding*. This is because users' clicks on ads are pretty sparse, making time intervals more informative than relative positions.

3.2 Numeric Features

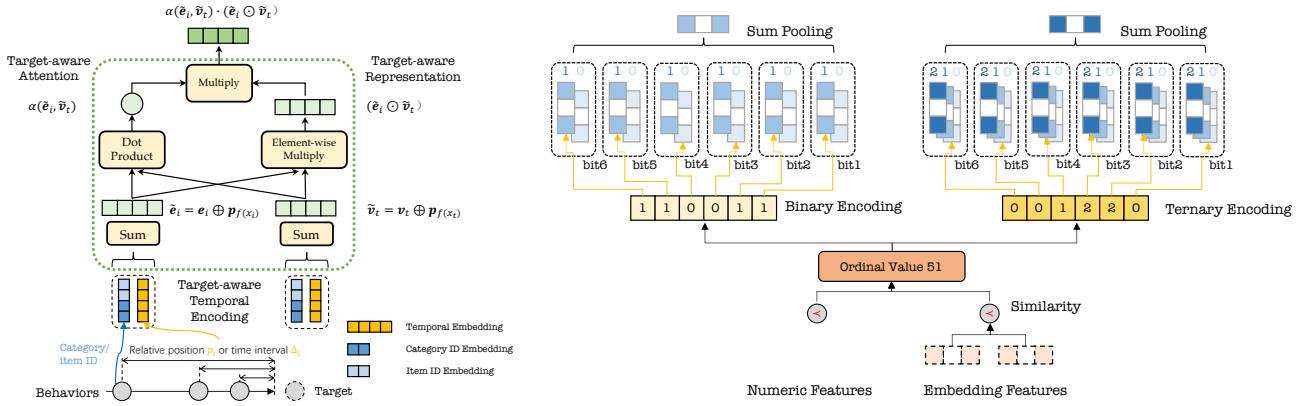
Unlike independent ID features, there is inherent partial order between numeric/ordinal features, such as Age_20 $\prec$ Age_30. To preserve these ordinal priors, we adopt a simplified variant of the NaryDis encoding [9], namely the Multiple Numeral Systems Encoding (MNSE). It encodes numeric features by getting codes according to multiple numeral systems (*i.e.*, binary, ternary, decimal) and then assigns learnable embeddings to these codes, as shown in Fig. 2(b).

For example, a feature value "51" is transformed into code "{6_1, 5_1, 4_0, 3_0, 2_1, 1_1}" according to binary system, and "{6_0, 5_0, 4_1, 3_2, 2_2, 1_0}" according to ternary system. All codes are projected to embeddings and then sum-pooled to get the final encoding result. To improve computation efficiency, we remove the inter- and intra-attention in the original NaryDis [9]. Given a continuous feature with value v, the encoding result is:

$$\begin{gathered} f_{\text{MNS}}(v) = [\sum_{k=1}^{K_2} \mathbf{X}^{(2)}_{2k+\mathbb{B}_k}, \sum_{k=1}^{K_3} \mathbf{X}^{(3)}_{3k+\mathbb{C}_k}, \ldots, \sum_{k=1}^{K_n} \mathbf{X}^{(n)}_{nk+\mathbb{N}_k}] \\ \mathbb{B} = \text{func\_binary}(v),\ \mathbb{C} = \text{func\_ternary}(v), \ldots \end{gathered} \tag{2}$$

where $\mathbf{X}^{(2)}_{2k+\mathbb{B}_k}$ and $\mathbf{X}^{(3)}_{3k+\mathbb{C}_k}$ are the embedding dictionaries for binary and ternary systems respectively, whose lengths of encodings are K_2 and K_3. func_binary and func_ternary are the binarization and ternarization functions that transform the continuous feature v into their corresponding encodings.

Deployment Details. In an advertising system, ads are often indexed by discrete identifiers (Ad IDs), which are self-incremental or random and contain little information. However, each ad is associated with a creative containing abundant visual semantics. We



(a) Temporal Interest Module

(b) Multiple Numeric Systems Encoding

Figure 2: Illustration of Temporal Interest Module (left) for sequence features and Multiple Numeral Systems Encoding (right) for numeric and pre-trained embedding features.

replace the self-incremental or random Ad IDs with novel Visual Semantic IDs to preserve the visual similarity between ads. We achieve this by obtaining visual embeddings from ad images using a vision model and applying hashing algorithms like Locality-Sensitive Hashing (LSH) [64] to preserve visual similarity. The Visual Semantic IDs serve as numeric features, and we apply Minimum Norm Scaling (MNS) to preserve their ordinal priors. This replacement leads to a 1.13% GMV lift in Moments pCVR, with a larger lift of 1.74% for new ads. Additionally, the coefficient of variation in prediction scores among similar ads exposed to the same user is significantly reduced from 2.44% to 0.30%, substantiating that our approach can preserve the visual similarity priors.

3.3 Embedding Features

Besides the main recommendation model, we may train a separate model, such as LLM or GNN, to learn embeddings for entities (users or items). Such embeddings capture the relationship between users and items from a different perspective, *e.g.*, a Graph Model or a Self-Supervised Language Model, and can be trained on a larger or different dataset and hence should provide extra information to the recommendation models. The key challenge in leveraging such pre-trained embedding directly in our recommendation system is the *semantic gap* between the embedding space of the external models and the recommenders. That is, these embedding captures different semantics from the collaborative semantics of the ID embeddings in recommendation models [36, 79].

We propose a Similarity Encoding Embedding approach to mitigate such a semantic gap. Take GNN for example. Once we train a GNN model and get the pre-trained embeddings $\bar{\boldsymbol{e}}_u, \bar{\boldsymbol{e}}_i$ for each user-item pair (u, i), we first calculate the similarity $w_{\text{sim}}(u, i)$ based on their GNN embeddings using the corresponding similarity function, *i.e.*, cosine in GraphSage [22]. Formally,

$$w_{\text{sim}}(u, i) = \text{sim}(\bar{\boldsymbol{e}}_u, \bar{\boldsymbol{e}}_i). \tag{3}$$

Such a similarity score is an ordinal value. Hence, similar to numeric features, we can use the Multiple Numeral Systems Encoding mentioned before to transform it into a learnable embedding $\boldsymbol{e}_{\text{sim}}(u, i) = f_{\text{MNS}}(w_{\text{sim}}(u, i))$. After that, the encoded embedding is simultaneously co-trained with the other ID embeddings in recommenders. Thus, the similarity priors in the original space are retained via the similarity score and encoding. Then, such priors are transferred to the recommenders by aligning the similarity encoding embedding $\boldsymbol{e}_{\text{sim}}(u, i)$ with the recommendation ID embeddings.

Furthermore, such an embedding encoding strategy has also been developed to incorporate large-language model (LLM) knowledge into our recommendation system. An LLM model is first transformed into an encoder-only architecture and trained with base proxy tasks like next-sentence prediction. Through such general pre-training, the LLM encoder can encode semantic embedding. After that, the LLM model is finetuned with high-quality positive and negative user-item pairs from the ads domain. Such contrastive alignment enables the LLM to generate high-quality pre-trained user embeddings $\bar{\boldsymbol{e}}_u$ and ad embeddings $\bar{\boldsymbol{e}}_i$. With such LLM similarity priors, like GNN embeddings, we can then adopt Similarity Encoding Embedding for space alignment.

Deployment Details. We train a GraphSage [22] upon a user-ad/content bipartite graph, with clicks in both ad and content recommendation domains as the edges. We then adopt the Similarity Encoding Embedding on the GNN embeddings and concatenate the resulting representation with that of the feature interaction layer. GNN embeddings are successfully deployed in many scenarios, leading to +1.21%, +0.59%, and 1.47% GMV lift on Moments, Channel, and Applet pCTR. In addition, incorporating LLM also leads to +2.55% GMV lift on Channel pCVR and +1.41% GMV lift on Channel pCTR during online A/B test.

4 TACKLING DIMENSIONAL COLLAPSE

After encoding, all features are transformed into embeddings and then interact with each other explicitly through FM-like models [20, 31, 33, 37, 49, 51, 52, 58, 66]. However, one key side effect of

explicit feature interaction is that some dimensions of embeddings collapse [21]. In this section, we'll first explain the dimensional collapse phenomenon and then present two different multi-embedding approaches and a collapse-resilient feature interaction function to mitigate it.

4.1 Embedding Dimensional Collapse

Recent work [1, 18, 78] has demonstrated that large-scale models especially transformer-based models with billions, even trillions, of parameters can achieve remarkable performance (*e.g.*, GPT-4 [1], LLaMA [61]). Inspired by these works, we explore how to scale up ads recommendation models. Usually, embeddings dominate the number of model parameters. For example, more than 99.99% of parameters in our production model are from feature embeddings. Therefore, we start to scale up our model by enlarging the embedding size K, *e.g.*, increasing K from 64 to 192. However, it doesn't bring significant performance lift and sometimes even leads to performance deterioration.

We investigate the learned embedding matrix of each field by singular spectral analysis [30], and observe dimensional collapse. That is, many singular values are very small, indicating that embeddings of many fields end up spanning a lower-dimensional subspace instead of the entire available embedding space [21, 28]. The dimensional collapse of embeddings results in a vast waste of model capacity since many embedding dimensions collapse and are meaningless. Furthermore, the fact that many embeddings have already collapsed makes it infeasible to scale up models by simply increasing dimension length [2, 21].

We study the root cause of the dimensional collapse and find it's due to the explicit feature interaction module, namely, fields with collapsed dimension make the embeddings of other fields collapse. For example, some fields such as Gender have very low cardinality N_{Gen}, making their embeddings only able to span a N_{Gen}-dimension space. As N_{Gen} is much smaller than embedding size K, the interaction between these low-dimension embeddings and the possibly high-dimensional embedding (in K-dimensional) of remaining fields make the latter collapse to an N_{Gen}-dimensional subspace.

4.2 Multi-Embedding Paradigm

We propose a *multi-embedding paradigm* [21] to mitigate embedding dimensional collapse when scaling up ads recommenders. Specifically, for each feature, instead of looking up only one embedding in the existing single-embedding paradigm, we learn multiple embedding tables, and look up several embeddings from these table for each feature. Then, all feature embeddings from the same embedding table interact with each other in the corresponding expert I. Formally, a recommendation model with T embedding tables is defined as:

$$\begin{aligned}
\boldsymbol{e}_i^{(t)} &= \left(\boldsymbol{E}_i^{(t)}\right)^\top \mathbf{1}_{x_i},\ \forall i \in \{1, 2, ..., N\},\\
h^{(t)} &= I^{(t)}\left(\boldsymbol{e}_1^{(t)}, \boldsymbol{e}_2^{(t)}, ..., \boldsymbol{e}_N^{(t)}\right),\\
h &= \frac{1}{T}\sum_{t=1}^{T} g^{(t)} h^{(t)}, \quad \hat{y} = F(h),
\end{aligned}$$

where t stands for the index of the embedding table, g denotes the gating function for each expert, and $F(\cdot)$ denotes the final classifier. One requirement is that there should be non-linearities such as ReLU within the interaction expert I; otherwise, the model is equivalent to the single-embedding paradigm [21]. An overall architecture is shown in Figure 1.

The multi-embedding paradigm offers an effective approach to scaling up recommendation models. *Instead of simply increasing the length of a shared embedding for each feature, this paradigm involves learning multiple embeddings for each feature.* By adopting the multi-embedding paradigm, we can achieve *parameter scaling* for recommendation models, which has traditionally been a challenging task [2]: the model's performance improves as the number of parameters increases.

Deployment Details. Almost all pCTR models in our platform adopt the Multi-Embedding paradigm. Specifically, we learn multiple different feature interaction experts, *e.g.*, GwPFM (a variant of FFM, which will be described below), IPNN, DCN V2, or FlatDNN, and multiple embedding tables. One or several experts share one of these embedding tables. We name such architecture *Heterogeneous Mixture-of-Experts with Multi-Embedding*, which differs from DHEN [75] in the sense that [75] employs one shared embedding table while we deploy multiple ones. For example, the Moments pCTR model consists of a GwPFM, IPNN [51], FlatDNN, and two embedding tables. GwPFM and FlatDNN share the first table, while IPNN uses the second one. Switching from a single embedding to the above architecture brings a 3.9% GMV lift in Moments pCTR, which is one of the largest performance lifts during the past decade.

4.3 GwPFM: Yet Another Simplified Approach to Multi-Embedding Paradigm

FFM [31] can also be regarded as another approach of the Multi-Embedding paradigm because FFM also learns multiple embeddings for each feature. In particular, for a dataset with M fields, FFM learns $M-1$ embeddings $\{\boldsymbol{e}_{i,F_l} | F_l \neq F(i)\}$ for each feature x_i. When interacting feature x_i with another feature j, among x_i's embeddings, FFM chooses the one corresponding to the field of j, *i.e.*, $\boldsymbol{e}_{i,F(j)}$, where $F(j)$ denotes the field of feature j.

Even though FFM has been proven more effective than FM, it's not widely deployed in industry due to its huge space complexity since it introduces $M-2$ times more parameters than FM, where M is usually at the magnitude of hundreds in practice. To tackle the high complexity, we propose to decouple the number of embeddings per feature from the number of fields. Specifically, we group fields to P *field parts* and learn P embeddings for each feature, one for each field part. We choose a small P to reduce the total model size. Furthermore, we want to capture the field-pair-wise correlation to improve performance [49]. A straightforward implementation is to assign a weight for each field pair, but it leads to a computation cost of $O(M^2)$, which is unacceptable. To reduce the computation cost, we group fields into *field groups* and learn a weight for each field group pair. We name this method Group-weighted Part-aware Factorization Machines, or GwPFM in short. The formal representation of its interaction is:

$$\Phi = \sum_{i=1}^{\oplus} \sum_{j=i+1}^{\oplus} x_i x_j \langle \boldsymbol{e}_{i,P(j)}, \boldsymbol{e}_{j,P(i)} \rangle r_{G(i),G(j)} \tag{4}$$

where $\oplus$ denotes element-wise summation, $P(i)$ and $G(i)$ denotes the field Part or Group of feature i, and $r_{G(i),G(j)}$ denotes the learnable weights for field group pair $(G(i), G(j))$.

Deployment Details. In practice, we split all fields into two parts: the first one consists of all fields unrelated to the target ads, including all user and context-side fields, while the second part consists of all fields regarding the target ad. We then split all fields of the first part into G groups based on expert knowledge, where G is at the dozens level, usually less than 50. We don't further split fields in the second part to avoid high online inference complexity. That is, all fields in the second part belong to one field group. The GwPFM has been deployed in our production since 2018 and has served many modules and scenarios to this day.

4.4 Beyond Multi-Embedding Paradigm: Collapse Resilient Feature Interaction

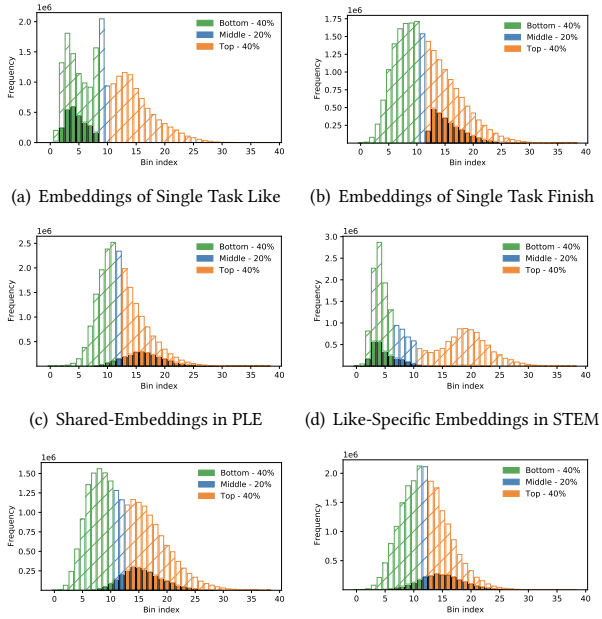
In addition to exploring multiple embeddings, we have conducted further investigation into the interaction function between two feature embeddings. The conventional approach, as employed in FM, conducts an element-wise inner product between the two embeddings: $f(\boldsymbol{e}_i, \boldsymbol{e}_j) = \boldsymbol{e}_i \odot \boldsymbol{e}_j$. However, recent research [30] has revealed that directly calculating the distance between two embeddings can lead to dimensional collapse. To address this issue, researchers verify that adding a projection matrix upon embeddings before computing the inner product can effectively mitigate the collapse [30]. We confirm its efficacy in ads recommendation, that is, incorporating a field-pair wise projection matrix $M_{F(i)\to F(j)} \in \mathcal{R}^{K\times K}$ within feature interaction [21, 29, 58, 66] as done in FiBiNET, FmFM and DCN V2 can also mitigate the dimensional collapse of embeddings in recommendation. Formally, the interaction function between feature embedding pair $(\boldsymbol{e}_i, \boldsymbol{e}_j)$ with a projection matrix $M_{F(i)\to F(j)}$ is defined as

$$f_{\text{Proj}}(\boldsymbol{e}_i, \boldsymbol{e}_j) = (\boldsymbol{e}_i M_{F(i)\to F(j)}) \odot \boldsymbol{e}_j \tag{5}$$

5 TACKLING INTEREST ENTANGLEMENT

User responses in ads recommender systems are driven by their interests under a specific task or scenario. Recently, there has been a trend to train multiple tasks or scenarios together to leverage the information from more tasks/scenarios to enhance prediction accuracy. However, existing work, *e.g.*, MMoE [43] and PLE [59], mainly employ a shared-embedding paradigm [6, 43, 48, 59, 70], learning one embedding representation for each user and ad. This leads to a risk of entangling the learned embedding by the possibly contradictory user interests from various tasks or scenarios [56], resulting in *negative transfer*.

Fig. 3 demonstrates the entanglement of user interest in the public TikTok dataset, which consists of two tasks: Like and Finish. We select a set of contradictory user-item pairs S whose Euclidean distance is among the bottom-40% regarding single task Like embedding (Fig. 3(a)) and among the top-40% regarding single task Finish embedding (Fig. 3(b)). We plot the distance distribution of these pairs regarding the shared embedding from PLE in Fig. 3(c) and observe that PLE learns large distances for most of them, which is similar to the distribution of single task Finish, while contradictory to that of single task Like.



(a) Embeddings of Single Task Like (b) Embeddings of Single Task Finish

(c) Shared-Embeddings in PLE (d) Like-Specific Embeddings in STEM

(e) Finish-Specific Embeddings in STEM (f) Shared Embeddings in STEM

Figure 3: Illustration of interest entanglement between tasks in single-embedding based MTL models and disentanglement in STEM. It shows the distance distribution of the contradictory user-item pair set S (with solid color) as well as the whole user-item pair set (with slash lines) regarding the single task Like (a) and Finish embedding (b), the PLE shared-embedding (c), and the Like (d) and Finish-specific (e) embedding and shared embedding (f) in STEM.

This section presents two approaches to tackle interest entanglement for multi-task/scenario learning and auxiliary learning. In the following discussion, we take multi-task learning as an example, and the same principle applies to multi-scenario learning.

5.1 AME for Multi-Task Learning

To tackle such an interest entanglement issue, we adopt a Shared and Task-specific EMbedding (STEM) paradigm [56], which *incorporates task-specific embeddings* to learn user's different interest across tasks, along with a shared embedding. The task-specific embeddings disentangle user and item representation (embeddings) across tasks, making preserving the distinct user interest in different tasks possible, as shown in Fig. 3(d) and Fig. 3(e). We then employ a set of experts, each of which is either task-specific or shared across tasks. We propose an All Forward Task-specific Backward gating mechanism [56] for task-specific towers so that each task tower receives the forward from all experts, while only backward gradients to its corresponding expert and the shared one.

However, there are many tasks in real-world ads recommendation systems. For example, each conversion type is usually treated as a task [48] when predicting conversions. There are usually dozens

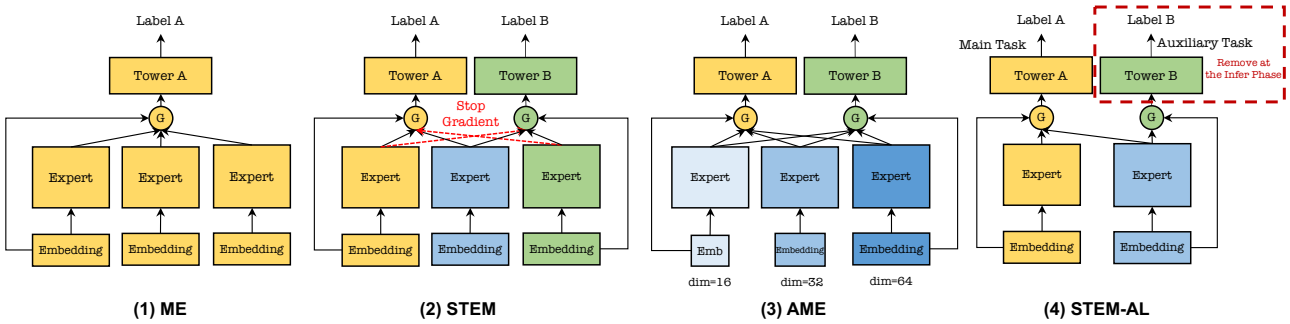


Figure 4: Architecture illustration of various paradigms. Multi-Embedding (ME) is for single-task learning and doesn't disentangle representations. Shared and Task-specific Embedding (STEM) and Asymmetric Multi-Embedding (AME) are both for multi-task learning. STEM disentangles representations via task-specific embeddings, while AME achieves disentanglement through learning multiple embedding tables with different embedding sizes. STEM for Auxiliary Learning (STEM-AL) is for auxiliary learning, which learns task-specific embedding for the main task and a shared embedding updated by multiple tasks.

to hundreds of conversion types, making learning an embedding table for each task infeasible. Therefore, in practice, we group conversion types into groups and treat each group as a task. On the other hand, we decouple the number of embedding tables from the number of conversion groups, *i.e.*, we learn a fixed number of embedding tables regardless of the number of groups. We then rely on the gating mechanism to route between embedding tables and conversion groups. However, these embedding tables may be entangled with each other due to their symmetry.

To this end, we set different embedding sizes for these embedding tables to disentangle them, leading to an Asymmetric Multi-Embedding paradigm, or AME in short, as shown in Figure 4. These embedding tables are disentangled in the sense that small tasks with fewer data need less model capacity and are routed more to the small-size embedding tables via the gating. While the other tasks with more data require larger model capacity and are routed to the large-size embedding tables. Other disentangled learning techniques can also be used [39, 44].

Connections to the Multi-Embedding Paradigm. Multi-Embedding (ME) paradigm is mainly used for single-task learning to tackle the embedding dimensional collapse. In contrast, Shared and Task-specific Embedding (STEM) and Asymmetric Multi-Embedding (AME) are mainly used to disentangle user interest representations across various tasks or scenarios. We try to use AME for single-task learning (*e.g.*, click prediction), but it brings little additional performance gain upon ME. Similarly, using ME for multi-task learning leads to Multi-Embedding MMoE (ME-MMoE) [56], which has been proven less effective than STEM [56] and AME (in our online test) since its embeddings are symmetric and hence may still be entangled.

Deployment Details. Our conversion prediction model learns more than 100 conversion types simultaneously. We group these conversion types into around 30 towers and adopt the asymmetric multi-embedding (AME) paradigm with three embedding tables of embedding sizes 16, 32, and 64, respectively. Compared to the single embedding baseline PLE with embedding size $K = 64$, AME brings 0.32%, 0.24%, and 0.48% average AUC lifts for three representative scenarios (Moments, Official Accounts, and News), leading to 4.2%, 3.9%, and 7.1% GMV lift in our online A/B test. In particular, the AUC lifts in small tasks such as *Pay* are 0.35%, 0.27%, and 0.78%, respectively, which are much larger than that on other large tasks. We also train a PLE model with $K = 128$ to study the effect of model capacity, but its performance is even worse than the baseline PLE with $K = 64$.

5.2 STEM-AL for Auxiliary Learning

In industrial recommenders, sometimes we pay more attention to a main task and want to leverage the signals from other tasks to improve the performance of this main task. For example, the main task in click prediction is to predict the *convertible click*, which leads to the landing page for further conversions. Besides this valuable feedback, we also collect users' other behaviors regarding the ad: like, favorite, comment, dislike, and dwell time (on video ads). We want to enhance the performance of the convertible clicks via Auxiliary Learning upon these additional tasks.

To prevent these auxiliary tasks from entangling a user's interest in the main task, we follow the STEM paradigm [56] and adopt a STEM-based Auxiliary Learning architecture, namely, STEM-AL. In the following discussion, we'll treat Task A as the main task and Task B as the auxiliary one. As shown in Figure 4, different from STEM and AME, which pay equal attention to all tasks, STEM-AL treats Task A as the primary one and treats Task B as an auxiliary task to improve the performance of A. In particular, STEM-AL incorporates two embedding tables and two corresponding interaction experts. The first embedding table, referred to as the *main embedding table*, is exclusively used by the tower of the primary task A. It is only forwarded to and optimized by the primary task, ensuring that the main task's distinctiveness is preserved without interference from other tasks. The second embedding table, known as the *shared embedding table*, is forwarded to and optimized by the towers of both tasks. This shared embedding table allows Task A to benefit from the knowledge and insights from Task B. The auxiliary tower is removed during inference; only the main task's tower remains active.

Deployment Details. We deploy STEM-AL to improve the pCTR in one scenario by samples from other domains. For example, we take the Applet pCTR as the main task and treat Moments pCTR as the auxiliary task. By using STEM-AL, the CTR of the main task can be improved by 1.16%. Further, if we use Moments and Channel pCTR as the auxiliary task, the CTR on Applet can be improved by 2.93%.

6 MODEL TRAINING

In this section, we describe several training challenges for the model training in ads recommendation and present our solution. Commonly, the click or conversion prediction task is formalized as a supervised learning problem with the optimization loss $\mathcal{L} = 1/N \sum_{i=1}^{N} \text{BCE}(y_i, f(x_i))$, here y_i represents labels with $y_i = 1$ denoting a positive label (*e.g.*, click in pCTR) and $y_i = 0$ denoting a negative label (*e.g.*, non-click in pCTR), x_i represents the input, and $\text{BCE}(y_i, f_i) = -y_i \log \sigma(f_i) - (1 - y_i) \log(1 - \sigma(f_i))$ is the Binary Cross Entropy loss.

6.1 Gradient Vanishing and Ranking Loss

Recent work [35, 53] finds out that incorporating an auxiliary ranking loss with BCE loss has shown substantial performance improvement in online advertising. However, the efficacy of this combination form is not fully comprehended. We examine its efficiency from a new perspective [40]: that negative samples suffer from *gradient vanishing* with only BCE loss when the positive feedback is sparse, such as in our pCTR model, where only 0.1% to 2% samples are positive (clicks in pCTR). Instead, after combining BCE with the ranking loss, we show empirically and theoretically that the gradients become significantly larger [40]. This leads to a lower BCE loss on both the validation set (indicating better classification ability) and the training set (indicating a better optimization procedure). We kindly advise readers to refer to [40] for more details.

Deployment Details. The combination of ranking loss with BCE loss is widely deployed in the Moments and Channel pCTR models, with GMV lifts of 0.57% and 1.08%, respectively. It's also deployed in all pLTV models, with an LTV GMV lift of 5.99%. Besides, the prediction bias is also reduced, especially on samples with low prediction scores.

6.2 Repeated Exposure and Weighted Sampling

Repetitive exposure, that is, displaying the same or similar ads to users within a short period, can enhance user's perception of specific ads but may also risk harming user experience. To tackle this, we introduced the *Repetitive Exposure Weight* (REW) module to decrease the prediction score of repeated ads for a given user to reduce their exposure. The core idea is to assign higher weights to the repeated impressions (negative samples).

Specifically, for each repeated impression with negative label, we assign a weight $w_{\text{rep}} >= 1$ in the original loss: $\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} w_{\text{rep}} \cdot \text{BCE}(y_i, f(x_i))$. It considers both the repeated count as well as recency: $w_{\text{rep}} = \alpha \cdot w_{\text{count}} + (1 - \alpha) \cdot w_{\text{recency}}$, where w_{count} is proportional to the total number of exposure (time decayed) of the same or similar ads to this user, and w_{recency} considers the time interval between the last repeated impression and the current time. Please note that these weights would lead to bias in the whole model since they upweight negative samples of repeated exposure. We rectify such bias by involving an additional weight $w_{debias} = (\sum_{i=1}^{N}(1-y_i) \cdot w_{\text{rep}} / \sum_{i=1}^{N}(1-y_i))$ for all positive samples.

Deployment Details. The REW module is widely deployed in Tencent Official Accounts, News, and Video pCTR models, reducing the percentage of repetitive ads exposure by 14.7%, 7.8%, and 9.7% respectively.

6.3 Online Learning

We train our pCTR and pCVR models with online learning, where samples are populated in seconds. Online learning for pCVR poses a special challenge due to conversion delay feedback. There is a lot of work [8, 72] to address it. Nevertheless, this method will lead to pronounced model bias due to substantial fluctuations of conversion feedback in our system. For instance, certain advertisers may report all previous conversions at uncertain times, resulting in an exceptionally high observed CVR at that moment, while reporting no conversions at other times. We propose a dynamic online learning method based on the conversion feedback variance in response to these challenges. Specifically, a very small variance means that the observed CVR is close to the history CVR, so we can populate the samples as fast as possible. Otherwise, when the variance is large, we will set a waiting time to ensure the stability of conversion arrival and reduce the risk of high bias due to arrival fluctuation.

Deployment Details. Our approach has been implemented in various scenarios for Tencent Ads, including pCVR models for Tencent Moments, Channel, and Official accounts, with overall GMV lift of 0.3%, 1.49%, 1.14%, and new ad GMV lift of 2.48%, 0.8%, 4.34% respectively, where the new ads refer to ads that have been online within three days. Besides, the bias of new ads in all scenarios has decreased from over 10% to within 1%.

6.4 Exploration with Uncertainty Estimates

So far, our focus is on enhancing the models' ability to accurately predict click or conversion rates, utilizing these scores to rank ads and maximize exploitation while neglecting the exploration. However, extensive research has demonstrated the criticality of striking a balance between exploration and exploitation, particularly for cold-start ads. Consequently, we propose adopting a Bayesian perspective for CTR modeling, wherein instead of predicting a single point estimate for CTR, we predict a distribution that incorporates uncertainty estimations.

To achieve this, we introduce a Gaussian process (GP) prior distribution to represent the unknown true CTR function. We leverage observed data to obtain predictions and uncertainty estimations from the posterior distribution. Combining these uncertainty estimates with well-established bandit algorithms, specifically Thompson Sampling (TS), enables us to manage the exploration-exploitation trade-off and enhance long-term utilities effectively. Formally,

$$\text{pCTR}_{\text{TS}} = \sigma(\hat{f}) \text{ where } \hat{f} \sim \mathcal{N}(\mu(\mathbf{x}^\star), \Sigma(\mathbf{x}^\star)) \tag{6}$$

where $\mu(\mathbf{x}^\star)$ and $\Sigma(\mathbf{x}^\star)$ denotes the mean and variance of the posterior logit value $f(\mathbf{x}^\star)$ for test data point $\mathbf{x}^\star$.

Deployment Details. The GP-based model is deployed in Tencent Moments pCTR, with GMV lift of +1.92%.

7 ANALYSIS TOOLS

In this section, we present several off-the-shelf analysis tools on representation learning to analyze the correlation between features, check whether and to what extent embeddings collapse, and examine the entanglement of user interest. We release the analysis code in this repository: https://github.com/junwei-pan/RecScope.

7.1 Feature Correlation

We can measure both *ground-truth* and *learned* feature correlation on particular samples or feature combinations via mutual information [49, 82]. The ground-truth correlation can be calculated by the mutual information between features X and the user's response(label) Y under certain constraints. In particular, when handling sequential features, we'd like to measure the semantic-temporal correlation between behaviors with specific categories while at specific position X_{con_b}, and the user's response on targets of specific categories Y_{con_t}. After defining the constraints on behaviors and the target, *e.g.*, con_b and con_t, the correlation can be quantified as

$$\text{Cor} = \text{MI}(X_{\text{con}_b}, Y_{\text{con}_t}) \tag{7}$$

For example, we can first select a subset of samples with target category c_t, and then for behaviors with category c_i while at various positions p (or with various time intervals), we can quantify the correlation as

$$\text{Cor} = \text{MI}(X_{C(X)=c_i \wedge P(X)=p}, Y_{C(Y)=c_t}) \tag{8}$$

Using this metric, we do observe both strong semantic correlation, *i.e.*, behaviors belonging to the same category as the target exhibit a stronger correlation than those of other categories, and strong temporal correlation, *i.e.*, there is a compelling correlation decrease from the most recent behaviors to the oldest ones. Please kindly refer to [82] for more details.

7.2 Embedding Dimensional Collapse

Dimensional collapse happens when embedding vectors span in a lower-dimensional subspace. Following the work of [30], we can measure dimensional collapse by conducting a singular value decomposition (SVD) of the embedding matrix of each field. In particular, given an embedding matrix of field i: $\boldsymbol{E}_i \in \mathbb{R}^{N_i \times K}$, after the SVD $\boldsymbol{E}_i = U\Sigma V^*, \Sigma = diag(\sigma^k)$, we can get the singular values σ_k. Dimensional collapse happens when some singular values are small. Besides, we can further quantify the dimensional collapse of an embedding matrix by a new metric: Information Abundance (IA) [21], which is defined as the sum of all singular values normalized by the maximum singular value:

Definition 7.1 (Information Abundance)**.** Consider a matrix $\boldsymbol{E} \in \mathbb{R}^{D \times K}$ and its singular value decomposition $\boldsymbol{E} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V} = \sum_{k=1}^{K} \sigma_k \boldsymbol{u}_k \boldsymbol{v}_k^\top$, then the *information abundance* of $\boldsymbol{E}$ is defined as

$$\text{IA}(\boldsymbol{E}) = \frac{\|\boldsymbol{\sigma}\|_1}{\|\boldsymbol{\sigma}\|_\infty},$$

7.3 Interest Entanglement

User responses in ads recommender systems are driven by many factors behind the users' decision-making processes. We can measure such factor entanglement by selecting a set of contradictory user-item pairs S whose embedding distances are large in one task but low in another. We then plot the distance distribution of S based on: a) embeddings from each single task model, b) the embeddings of a shared-embedding multi-task learning model, *e.g.*, PLE, c) the embeddings of each task as well as the shared embedding in STEM. An example of such analysis is already shown in Fig. 3.

8 RELATED WORK

Feature Encoding. Modeling sequence of user behaviorss have been widely studied [10, 17, 25, 32, 50, 57, 80, 81]. Regarding numerical features, existing work can be categorized into two groups: non-discretization [13, 47, 51] and discretization [9, 19]. Recently, with huge growth in research on LLM, lots of work has been done on how to utilize embeddings learned from these external pre-trained models in recommendation systems [26, 27, 36, 38, 73, 79].

Feature Interactions and Dimensional Collapse. There are numerous work on the backbone architecture with explicit or implicit feature interaction, from the shallow models FM [52], FFM [31], FwFM [49] and FmFM [58], to deep models such as Wide & Deep [12], DeepFM [20], xDeepFM [37], AutoInt [55], DCN V2 [66] and FinalMLP [45]. Refer to [71, 76] for a comprehensive survey.

The complete collapse has been widely studied in self-supervised learning (SSL) [11, 67], and Mixtures-of-Experts (MoE) [30]. On the other hand, dimensional collapse has been studied in SSL [28] and contrastive learning [30].

Interest Entanglement under MTL and MDL. Negative transfer has been a critical challenge in Multi-Task Learning (MTL) and Multi-Domain Learning (MDL). Shared-embedding paradigm is widely adopted in either MTL [6, 43, 48, 59, 70] and MDL [7, 54, 83]. Disentangled Representation Learning (DRL) aims to identify and disentangle the underlying explanatory factors [3] in embeddings and has also been widely used in recommendation [39, 44, 68, 69].

Industry Systems. There are already several works on industrial recommender systems [13–15, 24, 42, 46, 65, 74]. We differ from these works in that we pay more attention to the representation, especially from a dimensional collapse and interest entanglement perspective.

9 CONCLUSION

In this paper, we describe an industry ads recommendation system, paying special attention to the representation learning perspective. We present how to encode features with inherent priors, as well as practices to tackle the dimensional collapse and interest entanglement issue. We also showcase several training techniques and analysis tools. We hope this work can shed light on the future development of this research area.

ACKNOWLEDGMENTS

We want to express our sincere gratitude to the following individuals (alphabetical order) for their invaluable contributions: Chen Cai, Chengfei Cai, Genbao Chen, Rong Chen, Xihua Chen, Junwen Cheng, Xi Cheng, Chang Cui, Chao Deng, Guihe Deng, Huiting Deng, Yiming Deng, Zhixiang Feng, Haijie Gu, Weibo Gu, Chaonan Guo, Xian Hu, Ronggeng Huang, Shudong Huang, Renjie Jiang, Tingyu Jiang, Junfeng Kang, Kai Kang, Weijie Kong, Biao Li, Cong Li, Kaixin Li, Lingling Li, Rui Li, Xiaobo Li, Yi Li, Yuxiong Li, Zhaohua Li, Liwei Lin, Wenbo Liu, Yue Liu, Zijun Liu, Hua Lu, Feiheng Luo, Chao Lv, Lei Mu, Zhen Ouyang, Shuai Ren, Xueyu Shi, Xun Song, Jiayu Sun, Yan Tan, Hui Tang, Henghuan Wang, Hongfa Wang, Shaoying Wang, Xiaochen Wang, Yuan Wang, Shifeng Wen, Gengyu Weng, Haiyang Wu, Qinchen Wu, Ruiqian Wu, Zhengtao Wu, Zhiyuan Wu, Datian Xing, Tengfei Xiong, Sheng Xu, Zeen Xu, Yuekui Yang, Zhaohuan Yang, Changan Ye, Chengguo Yin, Qiufang Ying, Jiulong You, Ming Yue, Difei Zeng, Zijian Zeng, Junjie Zhai, Anran Zhang, Haoran Zhang, Jihong Zhang, Kuo Zhang, Linghan Zhang, Ruifeng Zhang, Shangyu Zhang, Tianjin Zhang, Yaqian Zhang, Wenzhe Zhao, Yufei Zheng, Erheng Zhong, Longsha Zhou and Qi Zhou.

REFERENCES

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

[2] Newsha Ardalani, Carole-Jean Wu, Zeliang Chen, Bhargav Bhushanam, and Adnan Aziz. 2022. Understanding Scaling Laws for Recommendation Models. *arXiv preprint arXiv:2208.08489* (2022).

[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* 35, 8 (2013), 1798–1828.

[4] Yoshua Bengio and Olivier Delalleau. 2011. On the expressive power of deep architectures. In *International conference on algorithmic learning theory*. Springer, 18–36.

[5] Yoshua Bengio, Olivier Delalleau, and Nicolas Roux. 2005. The curse of highly variable functions for local kernel machines. *Advances in neural information processing systems* 18 (2005).

[6] Rich Caruana. 1997. Multitask learning. *Machine learning* 28, 1 (1997), 41–75.

[7] Jianxin Chang, Chenbin Zhang, Yiqun Hui, Dewei Leng, Yanan Niu, Yang Song, and Kun Gai. 2023. Pepnet: Parameter and embedding personalized network for infusing with personalized prior information. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3795–3804.

[8] Olivier Chapelle. 2014. Modeling delayed feedback in display advertising. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1097–1105.

[9] Bo Chen, Huifeng Guo, Weiwen Liu, Yue Ding, Yunzhe Li, Wei Guo, Yichao Wang, Zhicheng He, Ruiming Tang, and Rui Zhang. 2022. Numerical Feature Representation with Hybrid N-ary Encoding. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2984–2993.

[10] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. 2019. Behavior sequence transformer for e-commerce recommendation in alibaba. In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*. 1–4.

[11] 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.

[12] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.

[13] 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.

[14] Daniel Crankshaw, Xin Wang, Guilio Zhou, Michael J Franklin, Joseph E Gonzalez, and Ion Stoica. 2017. Clipper: A {Low-Latency} online prediction serving system. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*. 613–627.

[15] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, et al. 2010. The YouTube video recommendation system. In *Proceedings of the fourth ACM conference on Recommender systems*. 293–296.

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[17] 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. In *International Joint Conference on Artificial Intelligence (IJCAI)*. 2301–2307.

[18] Tao Gong, Chengqi Lyu, Shilong Zhang, Yudong Wang, Miao Zheng, Qian Zhao, Kuikun Liu, Wenwei Zhang, Ping Luo, and Kai Chen. 2023. Multimodal-gpt: A vision and language model for dialogue with humans. *arXiv preprint arXiv:2305.04790* (2023).

[19] Huifeng Guo, Bo Chen, Ruiming Tang, Weinan Zhang, Zhenguo Li, and Xiuqiang He. 2021. An embedding learning framework for numerical features in ctr prediction. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2910–2918.

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

[21] Xingzhuo Guo, Junwei Pan, Ximei Wang, Baixu Chen, Jie Jiang, and Mingsheng Long. 2024. On the Embedding Collapse when Scaling up Recommendation Models. *International Conference on Machine Learning (ICML)* (2024).

[22] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems* 30 (2017).

[23] 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.

[24] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, et al. 2014. Practical lessons from predicting clicks on ads at facebook. In *International Workshop on Data Mining for Online Advertising (ADKDD)*. 1–9.

[25] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[26] Yupeng Hou, Zhankui He, Julian McAuley, and Wayne Xin Zhao. 2023. Learning vector-quantized item representation for transferable sequential recommenders. In *Proceedings of the ACM Web Conference 2023*. 1162–1171.

[27] Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. 2022. Towards universal sequence representation learning for recommender systems. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 585–593.

[28] Tianyu Hua, Wenxiao Wang, Zihui Xue, Sucheng Ren, Yue Wang, and Hang Zhao. 2021. On feature decorrelation in self-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 9598–9608.

[29] Tongwen Huang, Zhiqi Zhang, and Junlin Zhang. 2019. FiBiNET: combining feature importance and bilinear feature interaction for click-through rate prediction. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 169–177.

[30] Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. 2021. Understanding Dimensional Collapse in Contrastive Self-supervised Learning. In *ICLR*.

[31] 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 (RecSys)*. 43–50.

[32] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 197–206.

[33] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[34] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *NeurIPS*.

[35] Cheng Li, Yue Lu, Qiaozhu Mei, Dong Wang, and Sandeep Pandey. 2015. Click-through prediction for advertising in twitter timeline. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1959–1968.

[36] Xiangyang Li, Bo Chen, Lu Hou, and Ruiming Tang. 2023. CTRL: Connect Tabular and Language Model for CTR Prediction. *arXiv preprint arXiv:2306.02841* (2023).

[37] 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 and Data Mining (SIGKDD)*. 1754–1763.

[38] Jianghao Lin, Bo Chen, Hangyu Wang, Yunjia Xi, Yanru Qu, Xinyi Dai, Kangning Zhang, Ruiming Tang, Yong Yu, and Weinan Zhang. 2024. ClickPrompt: CTR Models are Strong Prompt Generators for Adapting Language Models to CTR Prediction. In *Proceedings of the ACM on Web Conference 2024*. 3319–3330.

[39] Zhutian Lin, Junwei Pan, Haibin Yu, Xi Xiao, Ximei Wang, Zhixiang Feng, Shifeng Wen, Shudong Huang, Lei Xiao, and Jie Jiang. 2024. Disentangled Representation with Cross Experts Covariance Loss for Multi-Domain Recommendation. *arXiv preprint arXiv:2405.12706* (2024).

[40] Zhutian Lin, Juwei Pan, Shangyu Zhang, Ximei Wang, Xi Xiao, Shudong Huang, Lei Xiao, and Jie Jiang. 2024. Understanding the Ranking Loss for Recommendation with Sparse User Feedback. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*.

[41] Fan Liu, Huilin Chen, Zhiyong Cheng, Anan Liu, Liqiang Nie, and Mohan Kankanhalli. 2022. Disentangled multimodal representation learning for recommendation. *IEEE Transactions on Multimedia* (2022).

[42] Quan Lu, Shengjun Pan, Liang Wang, Junwei Pan, Fengdan Wan, and Hongxia Yang. 2017. A practical framework of conversion rate prediction for online display advertising. In *Proceedings of the ADKDD'17*. 1–9.

[43] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H. Chi. 2018. Modeling Task Relationships in Multi-task Learning with Multi-gate Mixture-of-Experts. In *KDD*. ACM, 1930–1939.

[44] Jianxin Ma, Chang Zhou, Peng Cui, Hongxia Yang, and Wenwu Zhu. 2019. Learning disentangled representations for recommendation. *Advances in neural information processing systems* 32 (2019).

[45] Kelong Mao, Jieming Zhu, Liangcai Su, Guohao Cai, Yuru Li, and Zhenhua Dong. 2023. FinalMLP: An Enhanced Two-Stream MLP Model for CTR Prediction. *arXiv preprint arXiv:2304.00902* (2023).

[46] H Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, et al. 2013. Ad click prediction: a view from the trenches. In *ACM SIGKDD International conference on Knowledge Discovery & Data Mining (KDD)*. 1222–1230.

[47] Maxim Naumov, Dheevatsa Mudigere, Hao-Jun Michael Shi, Jianyu Huang, Narayanan Sundaraman, Jongsoo Park, Xiaodong Wang, Udit Gupta, Carole-Jean Wu, Alisson G Azzolini, et al. 2019. Deep learning recommendation model for personalization and recommendation systems. *arXiv preprint arXiv:1906.00091* (2019).

[48] Junwei Pan, Yizhi Mao, Alfonso Lobos Ruiz, Yu Sun, and Aaron Flores. 2019. Predicting different types of conversions with multi-task learning in online advertising. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2689–2697.

[49] 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 (WWW)*. 1349–1357.

[50] Qi Pi, Guorui Zhou, Yujing Zhang, Zhe Wang, Lejian Ren, Ying Fan, Xiaoqiang Zhu, and Kun Gai. 2020. Search-based user interest modeling with lifelong sequential behavior data for click-through rate prediction. In *ACM International Conference on Information & Knowledge Management (CIKM)*. 2685–2692.

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

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

[53] Xiang-Rong Sheng, Jingyue Gao, Yueyao Cheng, Siran Yang, Shuguang Han, Hongbo Deng, Yuning Jiang, Jian Xu, and Bo Zheng. 2023. Joint Optimization of Ranking and Calibration with Contextualized Hybrid Model. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4813–4822.

[54] Xiang-Rong Sheng, Liqin Zhao, Guorui Zhou, Xinyao Ding, Binding Dai, Qiang Luo, Siran Yang, Jingshan Lv, Chi Zhang, Hongbo Deng, et al. 2021. One model to serve all: Star topology adaptive recommender for multi-domain ctr prediction. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 4104–4113.

[55] 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 (CIKM)*. 1161–1170.

[56] Liangcai Su, Junwei Pan, Ximei Wang, Xi Xiao, Shijie Quan, Xihua Chen, and Jie Jiang. 2024. STEM: Unleashing the Power of Embeddings for Multi-task Recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 9002–9010.

[57] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *ACM International Conference on Information and Knowledge Management (CIKM)*. 1441–1450.

[58] Yang Sun, Junwei Pan, Alex Zhang, and Aaron Flores. 2021. Fm2: Field-matrixed factorization machines for recommender systems. In *Proceedings of the Web Conference 2021*. 2828–2837.

[59] Hongyan Tang, Junning Liu, Ming Zhao, and Xudong Gong. 2020. Progressive Layered Extraction (PLE): A Novel Multi-Task Learning (MTL) Model for Personalized Recommendations. In *RecSys*. ACM, 269–278.

[60] 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. *arXiv preprint arXiv:2304.10711* (2023).

[61] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[62] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

[63] 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.

[64] Hongya Wang, Jiao Cao, LihChyun Shu, and Davood Rafiei. 2013. Locality sensitive hashing revisited: filling the gap between theory and algorithm analysis. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management* (San Francisco, California, USA) *(CIKM '13)*. Association for Computing Machinery, New York, NY, USA, 1969–1978. https://doi.org/10.1145/2505515.2505765

[65] Jun Wang, Weinan Zhang, Shuai Yuan, et al. 2017. Display advertising with real-time bidding (RTB) and behavioural targeting. *Foundations and Trends® in Information Retrieval* 11, 4-5 (2017), 297–435.

[66] Ruoxi Wang, Rakesh Shivanna, Derek Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed Chi. 2021. DCN-V2: Improved deep & cross network and practical lessons for web-scale learning to rank systems. In *Proceedings of the Web Conference (WWW)*. 1785–1797.

[67] Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*. PMLR, 9929–9939.

[68] Xin Wang, Hong Chen, Si'ao Tang, Zihao Wu, and Wenwu Zhu. 2022. Disentangled representation learning. *arXiv preprint arXiv:2211.11695* (2022).

[69] Xin Wang, Hong Chen, Yuwei Zhou, Jianxin Ma, and Wenwu Zhu. 2022. Disentangled representation learning for recommendation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 1 (2022), 408–424.

[70] Enneng Yang, Junwei Pan, Ximei Wang, Haibin Yu, Li Shen, Xihua Chen, Lei Xiao, Jie Jiang, and Guibing Guo. 2023. Adatask: A task-aware adaptive learning rate approach to multi-task learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37. 10745–10753.

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

[72] Shota Yasui, Gota Morishita, Fujita Komei, and Masashi Shibata. 2020. A feedback shift correction in predicting conversion rates under delayed feedback. In *Proceedings of The Web Conference 2020*. 2740–2746.

[73] Zheng Yuan, Fajie Yuan, Yu Song, Youhua Li, Junchen Fu, Fei Yang, Yunzhu Pan, and Yongxin Ni. 2023. Where to go next for recommender systems? id- vs. modality-based recommender models revisited. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2639–2649.

[74] Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Michael He, et al. 2024. Actions Speak Louder than Words: Trillion-Parameter Sequential Transducers for Generative Recommendations. *arXiv preprint arXiv:2402.17152* (2024).

[75] Buyun Zhang, Liang Luo, Xi Liu, Jay Li, Zeliang Chen, Weilin Zhang, Xiaohan Wei, Yuchen Hao, Michael Tsang, Wenjun Wang, et al. 2022. DHEN: A deep and hierarchical ensemble network for large-scale click-through rate prediction. *arXiv preprint arXiv:2203.11014* (2022).

[76] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM computing surveys (CSUR)* 52, 1 (2019), 1–38.

[77] Weinan Zhang, Jiarui Qin, Wei Guo, Ruiming Tang, and Xiuqiang He. 2021. Deep learning for click-through rate estimation. *arXiv preprint arXiv:2104.10584* (2021).

[78] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* (2023).

[79] Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Adapting large language models by integrating collaborative semantics for recommendation. *arXiv preprint arXiv:2311.09049* (2023).

[80] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. In *AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 33. 5941–5948.

[81] 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 *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. 1059–1068.

[82] Haolin Zhou, Junwei Pan, Xinyi Zhou, Xihua Chen, Jie Jiang, Xiaofeng Gao, and Guihai Chen. 2024. Temporal Interest Network for User Response Prediction. In *Companion Proceedings of the ACM on Web Conference 2024*. 413–422.

[83] Jie Zhou, Xianshuai Cao, Wenhao Li, Kun Zhang, Chuan Luo, and Qian Yu. 2023. HiNet: A Novel Multi-Scenario & Multi-Task Learning Approach with Hierarchical Information Extraction. *arXiv preprint arXiv:2303.06095* (2023).

[84] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM international conference on information & knowledge management*. 1893–1902.