

INFNet: A Task-aware Information Flow Network for Large-Scale Recommendation Systems

Kaiyuan Li
likaiyuan03@kuaishou.com
Kuaishou Technology
Beijing, China

Dongdong Mao
maodongdong@kuaishou.com
Kuaishou Technology
Beijing, China

Yongxiang Tang
tangyongxiang@kuaishou.com
Kuaishou Technology
Beijing, China

Yanhua Cheng
chengyanhua@kuaishou.com
Kuaishou Technology
Beijing, China

Yanxiang Zeng
zengyanxiang@kuaishou.com
Kuaishou Technology
Beijing, China

Chao Wang
wangchao32@kuaishou.com
Kuaishou Technology
Beijing, China

Xialong Liu
zhaolei16@kuaishou.com
Kuaishou Technology
Beijing, China

Peng Jiang
jiangpeng@kuaishou.com
Kuaishou Technology
Beijing, China

Abstract

Feature interaction has long been a cornerstone of ranking models in large-scale recommender systems due to its proven effectiveness in capturing complex dependencies among features. However, existing feature interaction strategies face two critical challenges in industrial applications: (1) The vast number of categorical and sequential features makes exhaustive interaction computationally prohibitive, often resulting in optimization difficulties. (2) Real-world recommender systems typically involve multiple prediction objectives, yet most current approaches apply feature interaction modules prior to the multi-task learning layers. This late-fusion design overlooks task-specific feature dependencies and inherently limits the capacity of multi-task modeling.

To address these limitations, we propose the **Information Flow Network (INFNET)**, a task-aware architecture designed for large-scale recommendation scenarios. INFNET distinguishes features into three token types, categorical tokens, sequence tokens, and task tokens, and introduces a novel dual-flow design comprising heterogeneous and homogeneous alternating information blocks. For heterogeneous information flow, we employ a cross attention mechanism with proxy that facilitates efficient cross-modal token interaction with balanced computational cost. For homogeneous flow, we design type-specific Proxy Gated Units (PGUs) to enable fine-grained intra-type feature processing.

Extensive experiments on multiple offline benchmarks confirm that INFNET achieves state-of-the-art performance. Moreover, INFNET has been successfully deployed in a commercial online advertising

system, yielding significant gains of **+1.587% in Revenue (REV)** and **+1.155% in Click-Through Rate (CTR)**.

CCS Concepts

• **Information systems** → **Recommender systems**.

Keywords

CTR-prediction; feature interaction; multi-task

ACM Reference Format:

Kaiyuan Li, Dongdong Mao, Yongxiang Tang, Yanhua Cheng, Yanxiang Zeng, Chao Wang, Xialong Liu, and Peng Jiang. 2018. INFNet: A Task-aware Information Flow Network for Large-Scale Recommendation Systems. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 10 pages. <https://doi.org/XXXXXXX.XXXXXXX>

1 Introduction

Short video platforms such as TikTok and Kwai have surged in popularity in recent years, offering users a broad spectrum of interactive behaviors, such as clicks, scrolls, comments, and likes, as illustrated in Fig. 1. In large-scale industrial recommenders, two trends are converging: an explosion in the scale and heterogeneity of high-dimensional features, and the growing adoption of multi-task learning (MTL) to optimize numerous business objectives [1, 3, 8, 13, 24, 33].

In practice, a single model may process thousands of sequential features, hundreds of sparse categorical fields, and simultaneously serve dozens of downstream tasks. User behavior is typically represented by a mix of categorical identifiers (e.g., user ID, item ID) and rich sequence signals (e.g., clicked or liked item histories, augmented with contextual attributes such as creators or topics) [6, 10, 17, 20, 31, 32]. At the same time, modern MTL recommenders aim to jointly optimize diverse objectives ranging from CTR and watch time to revenue within a single unified framework. This scale and diversity introduce a central challenge: the feature interaction stage must be (i) expressive enough to capture complex

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Conference acronym 'XX, Woodstock, NY

© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-XXXX-X/18/06
<https://doi.org/XXXXXXX.XXXXXXX>

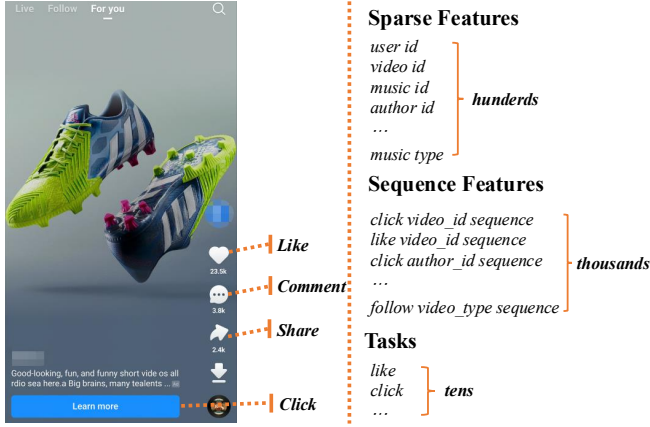


Figure 1: Massive features and multi-task at Kwai.

dependencies among heterogeneous inputs, (ii) efficient enough to satisfy strict latency and memory constraints, and (iii) sufficiently task-aware to mitigate negative transfer across objectives.

From the feature interaction perspective, early architectures such as FM [16] and DCN [22, 23] handle high-cardinality sparse features well through linear or cross-layer embeddings, but largely ignore sequential signals or compress them with coarse pooling. In production, sequence data are often truncated and flattened with side information, producing extremely large token sets directly feeding these into deep interaction layers incurs prohibitive costs, while aggressive pooling discards fine-grained temporal and contextual cues. Attention-based models such as DIN [35] and DIEN [34] alleviate this by applying target-aware attention to select relevant history for a single target item. However, their unidirectional, single-target design limits the modeling of broader cross-feature and cross-field dependencies, especially when multiple attributes jointly influence relevance.

From the MTL perspective, representative approaches such as MMoE [14], PLE [19], STEM [18], and HoME [24] introduce expert-based routing to separate shared and task-specific representations. Yet these methods typically perform feature interaction before task routing [29], meaning that the way two features interact is fixed regardless of which task is being optimized. This neglects task-aware dependencies during the interaction stage, leading to suboptimal representations and negative transfer. In large-scale deployments where objectives are optimized jointly under strict efficiency constraints, this misalignment can degrade both head and tail performance and slow convergence.

To address these challenges, we propose a **Task-Aware Information Flow Network (INFNET)** that introduces task proxy tokens to inject task-awareness directly into the feature interaction stage. As shown in Fig. 2, all inputs are represented as categorical tokens, sequence tokens, and task tokens. We explicitly decouple the interaction process into (i) homogeneous flows, which perform within-type refinement using lightweight gated units, and (ii) heterogeneous flows, which enable cross-type exchange via cross attention with token proxies. In the latter, proxies condense large token sets into compact anchors, ensuring that cross-type

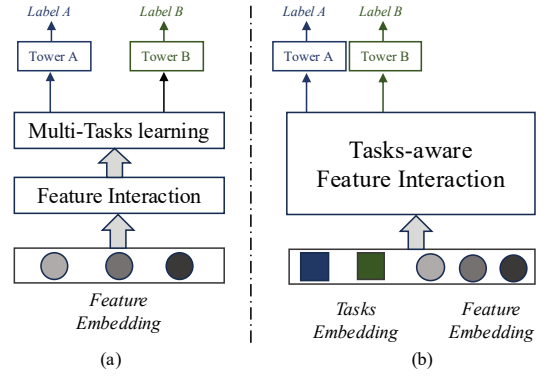


Figure 2: (a) Prior multi-task pipelines typically perform feature interaction before routing; (b) INFNET integrates task-aware interaction via task tokens and structured information flows.

complexity scales with the number of proxies rather than the raw token cardinality. Crucially, we introduce both task tokens and task-shared proxy tokens, allowing early-stage task-aware modulation of feature interactions while preserving cross-task generalization. This unified, structured information flow ensures that the target objective influences heterogeneous feature interactions from the very beginning of the pipeline.

This design yields a compact yet expressive backbone that (i) meets industrial efficiency budgets through low-overhead within-type refinement and proxy-based cross-type exchange, (ii) mitigates negative transfer via early-stage task conditioning, and (iii) unifies heterogeneous features into a single, structured interaction flow. Extensive experiments on both public and large-scale industrial datasets validate the effectiveness of INFNET; in a production advertising system, INFNET achieves +1.587% Revenue (REV) and +1.155% Click-Through Rate (CTR) gains, while maintaining stable performance across traffic segments and favorable training and serving characteristics.

Contributions. Our key contributions are as follows:

- We introduce INFNET, a task-aware information flow architecture that unifies categorical, sequence, and task tokens in a single interaction space, enabling downstream objectives to guide feature interactions from the earliest stage rather than only after feature fusion.
- We design a structured interaction mechanism with two complementary flows: (i) homogeneous flows using lightweight gated units for within-type refinement, and (ii) heterogeneous flows using cross attention with token proxies to enable fine-grained cross-type exchange without quadratic cost growth.
- We propose task-specific and task-shared proxy tokens that explicitly inject task-awareness into the interaction process, mitigating negative transfer across objectives while preserving beneficial cross-task generalization.
- We demonstrate both algorithmic and system-level gains: consistent improvements on public and large-scale industrial datasets, with +1.587% Revenue (REV) and +1.155% CTR in a

production advertising system, all under strict latency and memory budgets.

2 Related Work

We review two closely related lines of research feature interaction and multi-task recommendation highlighting their evolution, design trade-offs, and limitations in industrial settings.

Feature Interaction. Feature interaction modeling has progressed from explicit factorization to structured and sequence-aware designs. Early factorization methods such as FM [16] and its field-aware extensions [9, 15] are efficient and interpretable but restricted to second-order relations. Deep hybrids like DeepFM [7], xDeepFM [12], and DCN/DCN-V2 [23] enhance expressiveness via high-order semantics or polynomial expansions, but still rely on flat, global interactions that can be inefficient under strict latency and memory budgets. Recent structured designs such as xDeepInt [25] and FuXi- α [26] allow controllable capacity allocation across fields, improving scalability but focusing mainly on static or aggregated features and often neglecting fine-grained sequential patterns.

Sequence-aware models, including DIN, DIEN, and DSIN [4], employ target-aware attention for behavior modeling, while temporal methods like TiSASRec [11] and TIN [36] encode timing signals for better interest extraction. Cross-modal designs such as InterFormer [28] and HSTU [29] bridge sequence and non-sequence features. Despite these advances, sequences are often pooled too early, weakening conditioning, and task signals are typically injected late, limiting task-aware crossing in multi-objective scenarios.

Multi-Task Recommendation. Multi-task recommendation addresses the need to optimize multiple objectives jointly. Expert-based architectures such as MMoE [14], PLE [19], STEM [18], and HoME [24] separate shared and task-specific components to improve representation sharing. However, most follow a two-stage pipeline first performing feature interaction, then applying task-specific routing overlooking that interaction patterns can be task-dependent. This late integration of task information risks negative transfer and suboptimal representations, especially when serving constraints demand both high efficiency and strong task alignment.

These limitations motivate our approach, which unifies categorical, sequential, and task proxy tokens into a structured information flow, enabling early-stage task-aware interactions while controlling computational cost for industrial-scale deployment.

3 Methodology

In this section, we present the task-aware Information Flow Network (INFNet) and its overall architecture in Fig. 3. We first introduce feature pre-processing and tokenization, then describe cross-/within-type interaction via a stackable INFNet block, and finally detail the multi-task head and the optimization objective.

3.1 Feature Pre-processing

We unify all inputs into three token types: categorical, sequence, and task and each contains *original tokens* (full representation) and *proxy tokens* (compact queries). All embeddings share the same dimension d , ensuring compatibility in subsequent interaction layers and aligning with the information flow design.

Categorical Feature Tokens and Proxy Tokens Categorical inputs include user/item IDs, static attributes, and bucketized continuous variables. For the j -th field with value $v_j \in \{1, \dots, V_j\}$ and embedding table $\mathbf{E}_j^{\text{cat}} \in \mathbb{R}^{V_j \times d}$:

$$\mathbf{c}_j = \mathbf{E}_j^{\text{cat}}[v_j] \in \mathbb{R}^d. \quad (1)$$

Stacking all M fields yields:

$$\mathbf{C} = [\mathbf{c}_1 \parallel \dots \parallel \mathbf{c}_M]^\top \in \mathbb{R}^{M \times d}. \quad (2)$$

To form categorical proxies, we first *flatten* all field embeddings into a single vector, then project it through a shared MLP ϕ_{cat} and reshape it into m proxy tokens:

$$\tilde{\mathbf{C}} = \text{Reshape}(\phi_{\text{cat}}(\text{Flatten}(\mathbf{C}))) \in \mathbb{R}^{m \times d}. \quad (3)$$

This preserves global cross-field context before compression, yielding more informative proxies under the same token budget.

Sequence Feature Tokens and Proxy Tokens User behaviors are grouped into F action-specific sequences (e.g., clicks, likes, plays). We concatenate all behavior-specific sequences to form a unified sequence token matrix:

$$\mathbf{S} = [\mathbf{s}_{1,1}, \dots, \mathbf{s}_{1,n_1}, \dots, \mathbf{s}_{F,1}, \dots, \mathbf{s}_{F,n_F}] \in \mathbb{R}^{L \times d}, \quad (4)$$

where $L = \sum_{a=1}^F n_a$ is the total number of behavior tokens after padding/truncation, and each $\mathbf{s}_{a,t}$ is obtained from its embedding table $\mathbf{E}_a^{\text{seq}}$. To form proxies, each sequence token passes through a shared projection ϕ_{seq} and is pooled within its behavior type (sum pooling):

$$\tilde{\mathbf{S}} = \begin{bmatrix} \sum_{t=1}^{n_1} \phi_{\text{seq}}(\mathbf{s}_{1,t}) \\ \vdots \\ \sum_{t=1}^{n_F} \phi_{\text{seq}}(\mathbf{s}_{F,t}) \end{bmatrix} \in \mathbb{R}^{F \times d}. \quad (5)$$

This preserves per-type temporal semantics while keeping a bounded number of query tokens in heterogeneous flows.

Task Feature Tokens and Proxy Tokens Each real task i is represented by an original task tokens, initialized as learnable vectors:

$$\mathbf{T} \in \mathbb{R}^{N_{\text{task}} \times d}, \quad (6)$$

serving as task-specific keys/values and forming the basis for the corresponding prediction heads.

In addition, inspired by expert networks [18, 19], we introduce N_s shared task tokens to capture common knowledge across tasks. These shared tokens can be viewed as a special form of task proxy tokens, analogous to shared experts in mixture-of-experts architectures, providing a compact query set:

$$\tilde{\mathbf{T}} \in \mathbb{R}^{N_s \times d}. \quad (7)$$

They participate as queries in heterogeneous attention, enabling early task-aware conditioning without increasing the overall query count. Meanwhile, the original \mathbf{T} tokens remain in the within-type refinement stage, ensuring sufficient task-specific expressive capacity.

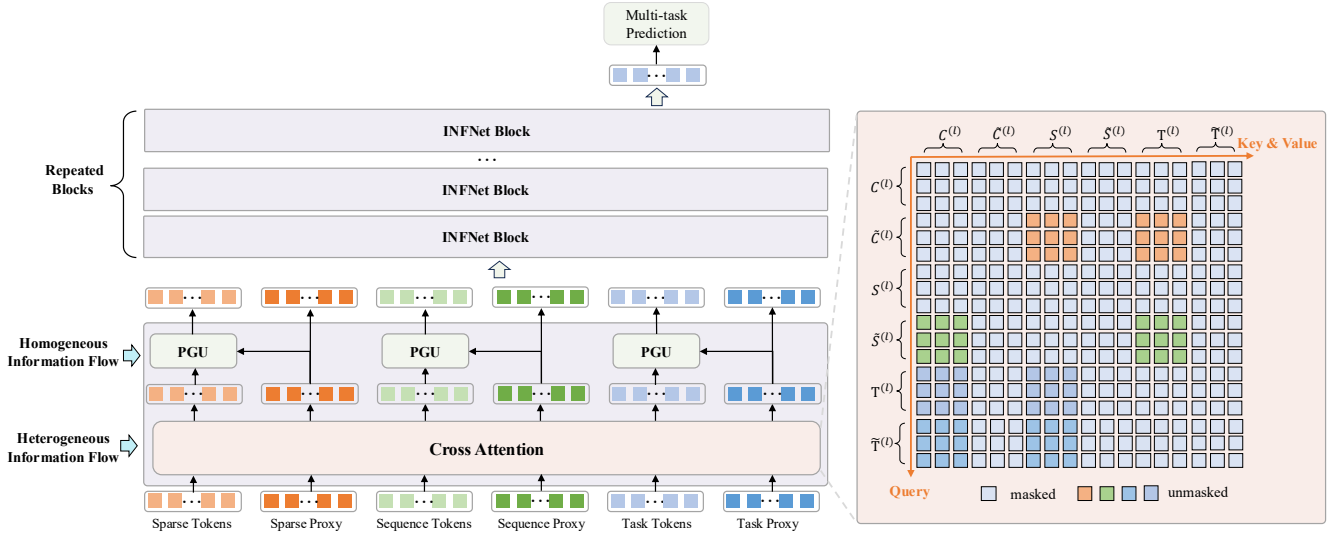


Figure 3: The overall architecture of our task-aware Information Flow Network (INFNet). Inputs are tokenized into categorical C with its proxy set \tilde{C} , sequence S with its proxy set \tilde{S} , and task T with its proxies \tilde{T} . Each block alternates proxy-guided heterogeneous fusion (cross attention: proxies as Q , raw tokens share K, V) and homogeneous refinement (PGU for C, S, T). Stacking N blocks yields fused proxies $\tilde{C}^{(N+1)}, \tilde{S}^{(N+1)}$, and $\tilde{T}^{(N+1)}$; the multi-task head feeds the final-layer task tokens into task-specific MLPs.

Initialization for the Interaction Stack The first INFNet block receives:

$$\begin{aligned} C^{(0)} &= C, & \tilde{C}^{(0)} &= \tilde{C}, \\ S^{(0)} &= S, & \tilde{S}^{(0)} &= \tilde{S}, \\ T^{(0)} &= T, & \tilde{T}^{(0)} &= \tilde{T}. \end{aligned}$$

Proxies act as queries in cross-type cross attention, while originals serve as keys/values and undergo within-type refinement, consistent with the subsequent information flow design.

3.2 Information Flow

As shown in Fig. 3, we split the information flow into two stages: heterogeneous and homogeneous feature interactions, encapsulated in a stackable INFNet Block. We treat the input tensors as layer (0); the superscript (l) indexes blocks, and after stacking N blocks we denote outputs with superscript ($N+1$). Given inputs to the l -th block categorical tokens $C^{(l)}$, categorical proxies $\tilde{C}^{(l)}$, sequence tokens $S^{(l)}$, sequence proxies $\tilde{S}^{(l)}$, and task tokens $T^{(l)}$, task proxies $\tilde{T}^{(l)}$, we first apply cross attention for pairwise heterogeneous interactions among them. Next, Proxy Gated Unit (PGU) handles homogeneous interactions within each feature type using its proxy, and residual connections are used for efficiency. The outputs $\tilde{C}^{(l+1)}$, $\tilde{S}^{(l+1)}$, and $\tilde{T}^{(l+1)}$ are fed to the next block or the prediction head. $CA(\cdot)$ and $PGU(\cdot)$ denote cross attention and proxy gated unit, respectively. Proxies act as attention bottlenecks: they are used as queries to limit fan-out, while richer (non-proxy) tokens can serve as keys/values to preserve information.

3.2.1 Heterogeneous Feature Interactions. To enable information exchange between different feature types, we adopt a cross-attention

mechanism that allows one set of tokens to query another. Intuitively, CA can be viewed as an information flow from a key-value set (K, V) to a query set Q , where the query decides what information to retrieve based on content similarity. Specifically, given Q, K , and V , cross attention mechanism is defined as:

$$CA(Q, K, V) = \text{softmax}\left(\frac{QW_Q(KW_K)^T}{\sqrt{d_k}}\right)(VW_V), \quad (8)$$

where d_k denotes the dimensionality of the key and query vectors. This formulation preserves the flexibility of content-based addressing while avoiding the quadratic complexity of self-attention across all tokens.

Having defined the cross attention operator, we now detail the specific information flow patterns in our architecture, namely: flow to categorical, flow to sequence, and flow to task. Each pattern specifies a distinct query-key-value configuration, determining how cross-type information is injected.

Information flow to categorical. In the information flow to categorical, categorical proxy tokens serve as the query in cross attention, while sequence and task tokens act as the key and value in their respective cross attention. The outputs of these cross attention are summed to produce the new result $\tilde{C}^{(l)}$ of this information flow.

$$\tilde{C}^{(l+1)} = CA(\tilde{C}^{(l)}, S^{(l)}, S^{(l)}) + CA(\tilde{C}^{(l)}, T^{(l)}, T^{(l)}). \quad (9)$$

Information flow to sequence. Similarly, in the information flow to sequence, sequence proxies serve as the query in cross attention, while categorical and task tokens act as the key and value in their respective cross attention. The outputs are summed to produce $\tilde{S}^{(l)}$.

$$\tilde{S}^{(l+1)} = CA(\tilde{S}^{(l)}, C^{(l)}, C^{(l)}) + CA(\tilde{S}^{(l)}, T^{(l)}, T^{(l)}). \quad (10)$$

Information flow to task. Different from the flows to categorical and sequence tokens, both task proxies and real-task tokens are few in number, and each serves a distinct role: task proxies capture shared cross-task patterns, while real-task tokens focus on task-specific nuances. To fully exploit their capacity and allow these roles to complement each other, we perform cross attention for both, using categorical and sequence tokens as keys/values:

$$\begin{aligned}\tilde{\mathbf{T}}^{(l+1)} &= \text{CA}(\tilde{\mathbf{T}}^{(l)}, \mathbf{C}^{(l)}, \mathbf{C}^{(l)}) + \text{CA}(\tilde{\mathbf{T}}^{(l)}, \mathbf{S}^{(l)}, \mathbf{S}^{(l)}), \\ \hat{\mathbf{T}}^{(l+1)} &= \text{CA}(\mathbf{T}^{(l)}, \mathbf{C}^{(l)}, \mathbf{C}^{(l)}) + \text{CA}(\mathbf{T}^{(l)}, \mathbf{S}^{(l)}, \mathbf{S}^{(l)}).\end{aligned}\quad (11)$$

Here, $\hat{\mathbf{T}}^{(l+1)}$ serves as an intermediate output enriched with task-specific information, thereby endowing the task tokens with explicit and concrete semantics.

3.2.2 Homogeneous Feature Interactions. After heterogeneous integration, we further refine features of the same type. We adopt PGU for homogeneous flows to mirror the channel-wise specialization role of MLPs in Transformers: Cross-Attention handles global, cross-type information exchange, while PGU refines intra-type representations with proxy-conditioned, channel-wise modulation—avoiding redundant token-to-token attention and ensuring deployment-friendly complexity in large-scale systems.

Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ denote the tokens of a given type and $\tilde{\mathbf{X}} \in \mathbb{R}^{\tilde{n} \times d}$ its \tilde{n} proxy tokens from the heterogeneous stage. We first flatten $\tilde{\mathbf{X}}$ along the token dimension to form $\tilde{\mathbf{X}}_f \in \mathbb{R}^{\tilde{n}d}$, map it through a lightweight MLP to produce a *channel-wise* gating vector $\mathbf{g} \in \mathbb{R}^d$, and then apply a Sigmoid activation to bound the gate values before broadcasting to all tokens for element-wise modulation:

$$\text{PGU}(\mathbf{X}, \tilde{\mathbf{X}}) = \mathbf{X} \odot \sigma(\text{MLP}(\tilde{\mathbf{X}}_f)), \quad \in \mathbb{R}^{n \times d}. \quad (12)$$

In our architecture, categorical, sequence, and task tokens are all refined using their own proxies:

$$\begin{aligned}\mathbf{C}^{(l+1)} &= \text{PGU}(\mathbf{C}^{(l)}, \tilde{\mathbf{C}}^{(l)}), \\ \mathbf{S}^{(l+1)} &= \text{PGU}(\mathbf{S}^{(l)}, \tilde{\mathbf{S}}^{(l)}), \\ \mathbf{T}^{(l+1)} &= \text{PGU}(\hat{\mathbf{T}}^{(l+1)}, \tilde{\mathbf{T}}^{(l)}).\end{aligned}\quad (13)$$

This channel-wise design decouples PGU’s complexity from the token length n , making it efficient even for very long sequences while preserving proxy-conditioned modulation.

3.3 Optimization Objective

Given the enhanced per-task representation $\mathbf{T}_i^{(N)}$ obtained in the multi-task modeling stage, each task prediction is produced by a task-specific MLP followed by a Sigmoid activation:

$$\hat{y}_i = \sigma(\text{MLP}_i(\mathbf{T}_i^{(N)})). \quad (14)$$

This head shares architecture across tasks but maintains separate parameters, mapping the task proxy representation to a calibrated probability.

To optimize multiple tasks within a single network, we adopt a weighted multi-task loss:

$$\mathcal{L} = \sum_{i=1}^{N_{\text{task}}} \lambda_i \mathcal{L}_i(\hat{y}_i, y_i). \quad (15)$$

where \hat{y}_i and y_i denote the predicted and ground-truth labels for task i , respectively. \mathcal{L}_i is the task-specific loss function. In this paper, we choose the binary cross-entropy:

$$\mathcal{L}_i(\hat{y}_i, y_i) = -[y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)]. \quad (16)$$

The weight λ_i (default = 1) balances gradient magnitudes across tasks and can be tuned via grid search or uncertainty-based methods when tasks vary in size or difficulty. The network parameters Θ are optimized end-to-end by minimizing $\mathcal{L}(\Theta)$ over the training set, enabling both shared and task-specific learning.

4 Experiment

In this section, we evaluate INFNET by comparing it with feature interaction models and multi-task recommendation models. We begin by introducing the experimental setup and analyze the experimental results.

4.1 Experimental Setup

Dataset. We evaluate INFNET on three public short-video benchmarks KuaiRand-Pure, KuaiRand-27K [5], and QB-Video [27]—plus a large-scale internal dataset. As summarized in Table 2, KuaiRand-Pure is relatively compact, KuaiRand-27K expands to a much larger item and log scale with the same users, while QB-Video adopts a different schema and task setup. This mix covers diverse sparsity levels, feature richness, and task counts, offering a varied testbed for evaluation.

Baselines. We compare INFNET against two baseline families: feature interaction models and multi-task recommendation models, using common instantiations from their original papers and adapting sequence handling only when required.

Feature interaction models.

- **FM** [16]: models second-order interactions via factorized embeddings and linear terms, offering a strong lightweight baseline for sparse high-cardinality inputs.
- **DIN** [35]: applies target-aware attention over a user’s history, improving relevance modeling without heavy sequence encoders.
- **DIEN** [34]: builds on DIN by modeling interest extraction and temporal evolution with attention-updated GRUs and an auxiliary loss.
- **DCN v2** [23]: enhances cross networks with low-rank residual mixing (CrossNet-Mix) for efficient high-order feature crossing.
- **GDCN** [21]: uses gated cross layers and field-level dimension optimization to filter redundant capacity while improving expressiveness.
- **WuKong** [30]: deepens and widens FM blocks under a scaling rule to capture any-order interactions at scale.
- **HSTU** [29]: treats recommendation as generative sequential transduction with hierarchical units, excelling on very long contexts.

Multi-task recommendation models.

- **Shared-Bottom** [2]: shares a common encoder across tasks with separate towers; simple but prone to negative transfer.

Table 1: The performance comparison between the baselines and INFNET. The best value in each row is in bold, and the second best is underlined. All improvements are statistically significant (p -value < 0.05).

		Metric	Feature-interaction baselines							Multi-task baselines						
			FM	DIN	DIEN	DCNv2	GDCN	WuKong	HSTU	Shared-Bottom	MMoE	OMoE	PLE	STEM	INFNET	
KuaiRand-pure	click	AUC	0.7206	0.7526	0.7252	0.7649	0.7652	0.7706	0.7709	0.7644	0.7639	0.7632	0.7648	<u>0.7714</u>	0.7736	
		gAUC	0.6367	0.6570	0.6441	0.6669	0.6655	0.6704	0.6674	0.6634	0.6624	0.6609	0.6633	<u>0.6728</u>	0.6757	
	like	AUC	0.7020	0.8157	0.8218	0.7872	0.7909	0.8801	0.8837	0.7924	0.8047	0.7858	0.8621	<u>0.8898</u>	0.8960	
		gAUC	0.5021	0.5352	0.4942	0.5925	0.6160	0.6320	0.6015	0.5568	0.5484	0.5440	0.6060	<u>0.6437</u>	0.6464	
	long-view	AUC	0.6973	0.7532	0.7184	0.7556	0.7602	0.7697	0.7698	0.7628	0.7623	0.7620	0.7648	<u>0.7700</u>	0.7727	
		gAUC	0.6206	0.6630	0.6428	0.6669	0.6694	0.6743	0.6713	0.6692	0.6688	0.6678	0.6716	<u>0.6763</u>	0.6784	
QB-video	click	AUC	0.9646	0.9702	0.9634	0.9699	0.9693	0.9737	<u>0.9740</u>	0.9684	0.9689	0.9692	0.9690	0.9699	0.9749	
		gAUC	0.9196	0.9273	0.9125	0.9280	0.9277	0.9300	<u>0.9308</u>	0.9263	0.9268	0.9269	0.9268	0.9270	0.9320	
	like	AUC	0.6832	0.9077	0.9021	0.9095	0.9093	0.9070	0.9078	0.8937	0.8836	0.8896	0.8996	<u>0.9171</u>	0.9225	
		gAUC	0.5404	0.5647	0.5410	0.5964	0.5920	0.6428	0.5860	0.5793	0.5784	0.5762	0.5842	0.6020	<u>0.6143</u>	
	follow	AUC	0.5940	0.8628	0.8431	0.8657	0.8745	0.8543	0.8409	0.8361	0.8447	0.8394	0.8747	<u>0.8870</u>	0.8881	
		gAUC	0.6042	0.5944	0.5714	0.5439	0.5996	0.6338	0.6578	0.6179	0.6065	0.6087	0.5544	0.5602	<u>0.6401</u>	
	share	AUC	0.6148	0.7388	0.6842	0.6786	0.6272	<u>0.7628</u>	0.7115	0.7022	0.6828	0.6824	0.7538	0.7057	0.8091	
		gAUC	0.6693	0.7012	<u>0.7245</u>	0.5898	0.6257	0.6888	0.7378	0.6500	0.6480	0.6498	0.5567	0.6618	0.7194	
	KuaiRand-27k	click	AUC	0.7311	0.7438	0.7345	0.7365	0.7115	0.7436	0.7293	0.7632	<u>0.7686</u>	0.7678	0.7667	0.7616	0.7820
			gAUC	0.6383	0.6475	0.6411	0.6452	0.6256	0.6409	0.6390	0.6563	<u>0.6641</u>	0.6620	0.6603	0.6551	0.6826
like		AUC	0.8760	0.9129	0.8696	0.8676	0.7914	0.8885	0.8768	0.8812	0.9113	0.9116	<u>0.9206</u>	0.9181	0.9254	
		gAUC	0.5990	0.6544	0.5699	0.5816	0.5418	0.6075	0.5523	0.6041	0.6471	0.6543	<u>0.6767</u>	0.6608	0.7039	
long-view		AUC	0.7647	0.7808	0.7731	0.7787	0.7435	0.7732	0.7608	0.7867	<u>0.7945</u>	0.7896	0.7926	0.7875	0.8038	
		gAUC	0.6824	0.6968	0.6897	0.6969	0.6668	0.6788	0.6801	0.6927	<u>0.7034</u>	0.6987	0.7003	0.6945	0.7203	

Table 2: Statistics of datasets used for experiments.

Dataset	kuairand-pure	kuairand-27k	QB-video
# Users	27,285	27,285	34,240
# Items	7,551	32,038,725	130,637
# Interactions	1,436,609	322,278,385	1,726,886
# Categorical Features	89	89	6
# Sequence Features	28	28	12
# Tasks	3	3	4

- **MMoE** [14]: combines multiple experts via task-specific gates to reduce interference and adapt to task difficulty.
- **OMoE** [14]: a lighter single-gate variant of MMoE with less specialization capability.
- **PLE** [19]: separates shared and task-specific experts progressively, mitigating the seesaw effect.
- **STEM** [18]: uses shared and task-specific embeddings with gating to balance personalization and cross-task sharing.

Evaluation Metric. We report three standard metrics.

- **AUC**: area under the receiver operating characteristic curve; measures overall ranking quality. Higher is better.
- **gAUC**: user-level averaged AUC that weights users more uniformly to reflect personalization quality. Higher is better.

Unless specified, we compute metrics on the test split and select checkpoints based on validation performance.

Experiment Settings. For a fair comparison, all methods are evaluated under the same data processing pipeline and a consistent hyperparameter tuning protocol. The batch size is fixed at 4096. Embedding dimensions are tuned over {8, 16, 32, 64, 96, 128}. All

models are implemented using the fuxiCTR¹ [37] framework, with necessary extensions for specific architectures. The optimizer is selected from {Adam, Adagrad}, and the initial learning rate is searched over {1e−4, 3e−4, 1e−3}, combined with optional linear warmup or cosine decay schedules. L2 regularization is tuned from {0, 1e−7, 1e−6, 1e−5}, and dropout rates are searched in {0.0, 0.1, 0.2}. Sequence lengths are tuned per dataset according to validation performance. Early stopping is applied when the validation AUC shows no improvement for five consecutive evaluations. All reported results are averaged over three runs with different random seeds to mitigate randomness and improve robustness.

4.2 Performance Comparison

Table 1 summarizes the overall comparison of INFNET against both feature interaction and multi-task recommendation baselines on the public datasets. The main observations are as follows.

- **Against feature interaction models.** INFNET consistently achieves higher AUC and gAUC on multiple targets such as click, like, long-view, and share. This indicates that the bidirectional heterogeneous information flow in INFNET integrates categorical fields with multi-behavior sequences more effectively, leading to stronger high-order interactions while keeping computational cost manageable.
- **Against multi-task models.** INFNET shows clear gains over Shared-Bottom, MMoE, OMoE, PLE, and STEM, especially on challenging objectives and on KuaiRand-27K where behavior sparsity and noise are more pronounced. The cross attention with token proxies, together with hybrid task representations that include both shared and real task tokens,

¹<https://github.com/reczoo/FuxiCTR>

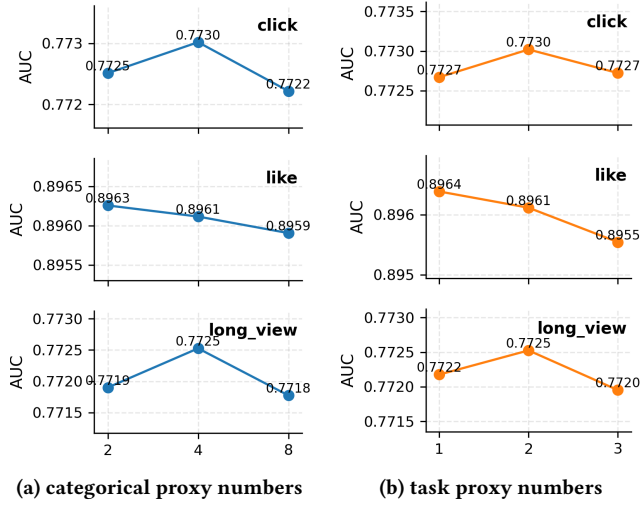


Figure 4: Performance of INFNet on the KuaiRand-pure dataset with different hyper-parameters.

strengthens commonality learning across tasks while capturing task-specific nuances, which reduces negative transfer and improves overall multi-task performance.

- **Robustness across settings.** We observe consistent improvements across a range of embedding sizes and sequence truncation lengths, suggesting that INFNet maintains stable advantages under different capacity and context budgets.

In summary, INFNet delivers consistent improvements over strong baselines in both the feature interaction and multi-task regimes, supporting its effectiveness and generalizability for large-scale short-video recommendation.

4.3 Ablation Study

We evaluate INFNet by removing each major component in turn while keeping data processing, optimization, and checkpoint selection identical to the main setup. Experiments are conducted on KuaiRand-pure and QB-video to cover short-video recommendation and multi-behavior advertising.

Ablated variants: **w/o1** – Removes task tokens (learnable task vectors), preventing early integration of multi-task information. **w/o2** – Removes homogeneous interaction (PGU updates for categorical, sequence, and task tokens), weakening within-type modeling. **w/o3** – Removes heterogeneous interaction (cross attention across feature types), limiting cross-type fusion.

As shown in Table 3, the full INFNet consistently outperforms all variants. (1) **w/o1** shows universal drops, especially on share (+0.0344 AUC), confirming that task vectors improve alignment with task-specific objectives. (2) **w/o2** notably hurts sequence-dependent tasks (e.g., long-view, follow, like), indicating that per-type refinement improves feature quality before cross-type mixing. (3) **w/o3** causes the largest losses on most tasks, highlighting the importance of aligning categorical and sequential evidence under task context.

Table 3: Performance comparison of INFNet and its variants. Best results in bold.

Dataset	Task	Metric	w/o1	w/o2	w/o3	INFNet
kuairand-pure	click	AUC	0.7720	0.7716	0.7705	0.7736
	like	AUC	0.8925	0.8926	0.8899	0.8960
	long-view	AUC	0.7707	0.7713	0.7701	0.7727
QB-video	click	AUC	0.9747	0.9741	0.9741	0.9749
	like	AUC	0.9185	0.9183	0.9168	0.9225
	follow	AUC	0.8842	0.8852	0.8829	0.8881
	share	AUC	0.7747	0.7868	0.7854	0.8091

Overall, task vectors inject early task awareness, homogeneous flows strengthen per-type representations, and heterogeneous flows enable effective cross-type fusion. The full model benefits from all three for the best performance.

4.4 Effect of Proxy Token Counts

We study how the number of proxy tokens influences INFNet’s multi-task performance on KuaiRand-pure, fixing task proxies = 4 and shared task tokens = 2 unless otherwise noted. To isolate the effect of each factor, we vary one at a time under the same data pipeline, optimization schedule, and checkpoint selection.

As shown in Fig. 4a–4b, increasing categorical/sequential proxies from 2 to 4 improves AUC by enabling finer-grained representation of their respective feature spaces. More proxies allow the model to preserve diverse subspaces of categorical and sequential information, reducing the compression of multiple signals into a single vector and enhancing alignment between task intent and relevant regions. However, pushing the number to 8 leads to a drop in performance, consistent with over-parameterization: excessive proxies introduce redundancy, dilute attention focus, and increase optimization difficulty.

A similar pattern emerges for shared task tokens. Increasing from 1 to 2 yields gains on both click and long-view metrics by better summarizing cross-task patterns, while still allowing sufficient capacity for task-specific tokens to specialize. Setting this value to 3 brings diminishing or negative returns, consistent with over-sharing: too many shared carriers blur task boundaries and compete with real task tokens, weakening per-task discrimination. Overall, categorical/sequential proxies = 4 and shared task tokens = 2 offer a good trade-off between representation richness and stability, and serve as robust defaults in our training setup.

4.5 Visualization Analysis

As shown in Fig. 5, we visualize the final layer’s cross attention where task query all categorical and sequential tokens (x-axis: token positions; y-axis: query channels). Shared-task channels exhibit smoother, low-frequency patterns covering broad regions, serving as carriers of cross-task context, while real-task channels show sharper peaks aligned with a few salient positions (e.g., ~60–70, ~90), emphasizing influential tokens from behavior sequences or key categorical fields. Dark “corridors” indicate tokens with limited utility.

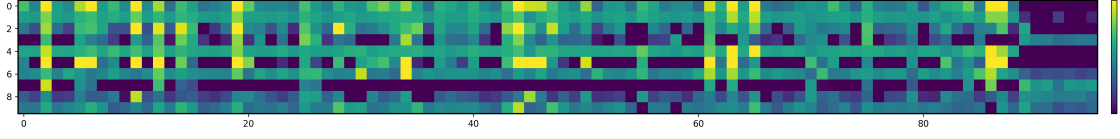


Figure 5: Visualization of the final-layer cross attention between task tokens and their proxies. The x-axis indicates feature or sequence positions, the y-axis corresponds to query channels for individual and shared tasks, and the color intensity reflects attention weights (brighter indicates stronger attention).

These patterns match our design goals: real tasks specialize in task-specific cues, and shared tasks provide a broad prior that regularizes and complements them. The attention remains differentiated yet coordinated broad shared coverage does not overwhelm task-specific spikes mirroring our quantitative trends that moderate sharing yields more effective attention than overly concentrated or diffuse maps.

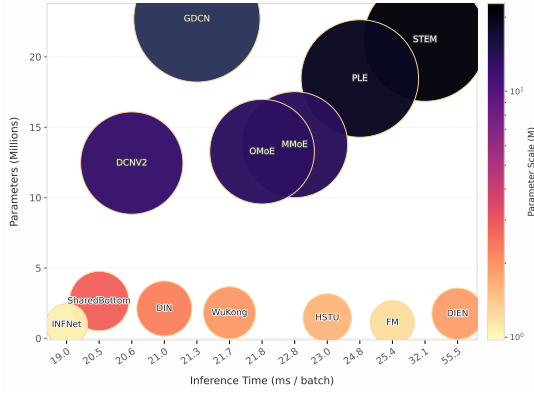


Figure 6: Inference Time and Parameters of INFNET and baseline on the kuairand-pure Dataset.

4.6 Efficiency Analysis

Fig. 6 compares inference time and parameter count of representative models on KuaiRand-pure, illustrating the trade-off between complexity and online efficiency. The x-axis shows per-batch inference time (lower is better), and the y-axis indicates model size (smaller is easier to deploy), with circle size/color reflecting parameter magnitude.

Feature interaction models such as DIN, DIEN, and INFNET achieve low latency and compact sizes, whereas high-order or multi-expert methods (e.g., MMoE, OMoE, PLE, DCNv2, GDCN) incur higher cost due to additional branches, routing, or explicit crossing. Methods like AITM, WuKong, Shared-Bottom, and STEM sit in between.

Overall, deeper explicit-interaction models and multi-expert designs increase both parameters and inference time, while INFNET maintains compactness and low latency through hierarchical feature interaction and task-aware fusion with learnable task vectors, making it practical under strict latency and resource constraints.

4.7 Online Analysis

This section details the deployment of our model as a real-time ranking system for short-video ads in a large-scale online advertising system, targeting shallow signals such as click, play, and completion. Emphasizing task-specific efficiency and revenue impact, we evaluate the model under a streaming training setup from both training and serving perspectives. To ensure a fair assessment, the online experiment uses the same candidate generation and request routing as the production baseline, and we report results after the system reaches a steady state.

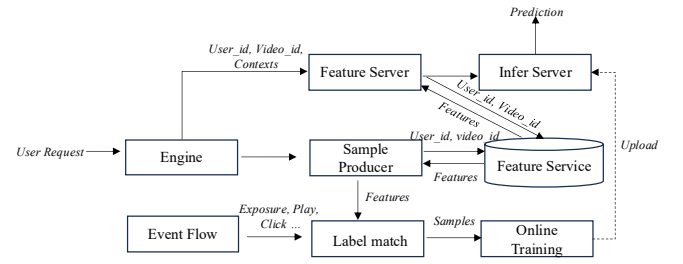


Figure 7: Feature generation logic in the deployment stage of INFNET.

4.7.1 Online Deployment. As the model is trained in a streaming fashion, Fig. 7 illustrates feature generation and label acquisition during deployment. Upon receiving a ranking-stage request, the system splits it into two pipelines: inference and training. In inference, context features are sent to the feature server to retrieve user and video features, which are then scored by the infer server. In training, the sample producer fetches features, and the label matcher aligns callback timestamps to generate labels. The updated model is deployed to the infer server.

Model complexity thus affects both training efficiency and inference performance. To keep the end-to-end cost stable, we reuse cached embeddings for infrequently changing fields, maintain a bounded sequence length for behaviors, and apply consistent snapshotting so that features and model versions remain time-aligned. These practices help maintain predictable throughput under fluctuating traffic.

4.7.2 Online A/B Testing. We deployed our model in the main feed scenario, which serves billions of requests per day. The baseline, approximately a combination of DIN, DCNv2, and PLE, was compared against our INFNET model in an A/B test conducted from 2025-03-10 to 2025-04-10. As shown in Table 4, results show

Table 4: A/B test results of INFNET versus the baseline in the main feed scenario (2025-03-10 to 2025-04-10). P3s, P5s, and PEnd denote the proportion of videos played for at least 3 seconds, 5 seconds, and until the end, respectively. All improvements are statistically significant (p -value < 0.05). Latency is measured under identical hardware and quota settings.

Method	REV (%)	CTR (%)	P3s (%)	P5s (%)	PEnd (%)	Pred. Latency (ms)	Train Latency (min)
Baseline	–	–	–	–	–	18.28	21.21
INFNET	+1.587	+1.155	+0.105	+0.317	+0.351	18.17	20.04

statistically significant improvements (p -value < 0.05) across all key metrics, including a +1.587% increase in expected revenue and +1.155% in click-through rate (CTR). We also observed consistent gains in short-term user retention indicators, where P3s, P5s, and PEnd respectively denote the proportion of videos played for at least 3 seconds, 5 seconds, and until the end: +0.105%, +0.317%, and +0.351%.

As shown by the latency metrics, our model achieves comparable efficiency to the baseline, with a slightly lower prediction latency (18.17 ms vs. 18.28 ms) and reduced training latency (20.04 min vs. 21.21 min). The online ramp-up followed a fixed traffic ratio to avoid seasonality artifacts, and all comparisons were performed under the same hardware and quota settings to ensure comparability.

4.8 Conclusion

This paper proposes INFNET, a large-scale multi-task feature interaction framework for recommendation. It treats task as a learnable feature and integrates sparse, sequential, and task features via bidirectional heterogeneous and homogeneous flows. Extensive offline experiments show significant gains across datasets and objectives. Online results confirm notable improvements in business metrics and user experience with efficient inference. Overall, INFNET demonstrates strong potential and wide applicability in multi-source feature modeling and multi-task learning for large-scale recommendation.

References

- [1] Qingpeng Cai, Shuchang Liu, Xueliang Wang, Tianyou Zuo, Wentao Xie, Bin Yang, Dong Zheng, Peng Jiang, and Kun Gai. 2023. Reinforcing user retention in a billion scale short video recommender system. In *Companion Proceedings of the ACM Web Conference 2023*. 421–426.
- [2] Rich Caruana. 1997. Multitask learning. *Machine learning* 28 (1997), 41–75.
- [3] Zhongxiang Fan, Zhaocheng Liu, Jian Liang, Dongying Kong, Han Li, Peng Jiang, Shuang Li, and Kun Gai. 2024. Multi-Epoch learning with Data Augmentation for Deep Click-Through Rate Prediction. *arXiv preprint arXiv:2407.01607* (2024).
- [4] 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. *arXiv preprint arXiv:1905.06482* (2019).
- [5] Chongming Gao, Shijun Li, Yuan Zhang, Jiawei Chen, Biao Li, Wenqiang Lei, Peng Jiang, and Xiangnan He. 2022. Kuairand: An unbiased sequential recommendation dataset with randomly exposed videos. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 3953–3957.
- [6] Huan Gui, Ruoxi Wang, Ke Yin, Long Jin, Maciej Kula, Taibai Xu, Lichan Hong, and Ed H Chi. 2023. Hiformer: Heterogeneous feature interactions learning with transformers for recommender systems. *arXiv preprint arXiv:2311.05884* (2023).
- [7] 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).
- [8] Yuezhan Jiang, Changyu Li, Gaode Chen, Peiyi Li, Qi Zhang, Jingjian Lin, Peng Jiang, Fei Sun, and Wentao Zhang. 2024. MMGCL: Meta Knowledge-Enhanced Multi-view Graph Contrastive Learning for Recommendations. In *Proceedings of the 18th ACM Conference on Recommender Systems*. 538–548.
- [9] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware factorization machines for CTR prediction. In *Proceedings of the 10th ACM conference on recommender systems*. 43–50.
- [10] Honghao Li, Yiwen Zhang, Yi Zhang, Hanwei Li, and Lei Sang. 2024. Dcnv3: Towards next generation deep cross network for ctr prediction. *arXiv e-prints* (2024), arXiv–2407.
- [11] Jiacheng Li, Yujie Wang, and Julian McAuley. 2020. Time interval aware self-attention for sequential recommendation. In *Proceedings of the 13th international conference on web search and data mining*. 322–330.
- [12] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1754–1763.
- [13] Xiao Lin, Xiaokai Chen, Linfeng Song, Jingwei Liu, Biao Li, and Peng Jiang. 2023. Tree based progressive regression model for watch-time prediction in short-video recommendation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4497–4506.
- [14] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1930–1939.
- [15] Junwei Pan, Jian Xu, Alfonso Lobos Ruiz, Wenliang Zhao, Shengjun Pan, Yu Sun, and Quan Lu. 2018. Field-weighted factorization machines for click-through rate prediction in display advertising. In *Proceedings of the 2018 world wide web conference*. 1349–1357.
- [16] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International conference on data mining*. IEEE, 995–1000.
- [17] Lei Sang, Qiuzhe Ru, Honghao Li, Yiwen Zhang, Qian Cao, and Xindong Wu. 2024. Feature Interaction Fusion Self-Distillation Network For CTR Prediction. *arXiv preprint arXiv:2411.07508* (2024).
- [18] 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.
- [19] 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 *Proceedings of the 14th ACM conference on recommender systems*. 269–278.

- [20] Zhen Tian, Changwang Zhang, Wayne Xin Zhao, Xin Zhao, Ji-Rong Wen, and Zhao Cao. 2023. UFIN: Universal feature interaction network for multi-domain click-through rate prediction. *arXiv preprint arXiv:2311.15493* (2023).
- [21] Fangye Wang, Hansu Gu, Dongsheng Li, Tun Lu, Peng Zhang, and Ning Gu. 2023. Towards deeper, lighter and interpretable cross network for CTR prediction. In *Proceedings of the 32nd ACM international conference on information and knowledge management*. 2523–2533.
- [22] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*. 1–7.
- [23] 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 2021*. 1785–1797.
- [24] Xu Wang, Jiangxia Cao, Zhiyi Fu, Kun Gai, and Guorui Zhou. 2024. HoME: Hierarchy of Multi-Gate Experts for Multi-Task Learning at Kuaishou. *arXiv preprint arXiv:2408.05430* (2024).
- [25] Yachen Yan and Liubo Li. 2023. xDeepInt: a hybrid architecture for modeling the vector-wise and bit-wise feature interactions. *arXiv preprint arXiv:2301.01089* (2023).
- [26] Yufei Ye, Wei Guo, Jin Yao Chin, Hao Wang, Hong Zhu, Xi Lin, Yuyang Ye, Yong Liu, Ruiming Tang, Defu Lian, et al. 2025. FuXi-*alpha*: Scaling Recommendation Model with Feature Interaction Enhanced Transformer. *arXiv preprint arXiv:2502.03036* (2025).
- [27] Guanghu Yuan, Fajie Yuan, Yudong Li, Beibei Kong, Shujie Li, Lei Chen, Min Yang, Chenyun Yu, Bo Hu, Zang Li, et al. 2022. Tenrec: A large-scale multipurpose benchmark dataset for recommender systems. *Advances in Neural Information Processing Systems* 35 (2022), 11480–11493.
- [28] Zhichen Zeng, Xiaolong Liu, Mengyue Hang, Xiaoyi Liu, Qinghai Zhou, Chaofei Yang, Yiqun Liu, Yichen Ruan, Laming Chen, Yuxin Chen, et al. 2024. InterFormer: Towards Effective Heterogeneous Interaction Learning for Click-Through Rate Prediction. *arXiv preprint arXiv:2411.09852* (2024).
- [29] 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).
- [30] Buyun Zhang, Liang Luo, Yuxin Chen, Jade Nie, Xi Liu, Shen Li, Yanli Zhao, Yuchen Hao, Yantao Yao, Ellie Dingqiao Wen, et al. 2024. Wukong: towards a scaling law for large-scale recommendation. In *Proceedings of the 41st International Conference on Machine Learning*. 59421–59434.
- [31] Pengtao Zhang, Zheng Zheng, and Junlin Zhang. 2023. Fibinet++: Reducing model size by low rank feature interaction layer for ctr prediction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 4425–4429.
- [32] Yang Zhang, Tianhao Shi, Fuli Feng, Wenjie Wang, Dingxian Wang, Xiangnan He, and Yongdong Zhang. 2023. Reformulating CTR prediction: Learning invariant feature interactions for recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1386–1395.
- [33] Zijian Zhang, Shuchang Liu, Jiaao Yu, Qingpeng Cai, Xiangyu Zhao, Chunxu Zhang, Ziru Liu, Qidong Liu, Hongwei Zhao, Lantao Hu, et al. 2024. M3oe: Multi-domain multi-task mixture-of experts recommendation framework. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 893–902.
- [34] 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 *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 5941–5948.
- [35] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1059–1068.
- [36] 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 Web Conference 2024*. 413–422.
- [37] Jieming Zhu, Jinyang Liu, Shuai Yang, Qi Zhang, and Xiuqiang He. 2021. Open benchmarking for click-through rate prediction. In *Proceedings of the 30th ACM international conference on information & knowledge management*. 2759–2769.