

# HyFormer: Revisiting the Roles of Sequence Modeling and Feature Interaction in CTR Prediction

Yunwen Huang\*

huangyunwen.eleanor@bytedance.com

ByteDance AML  
Beijing, China

Shiyong Hong\*

hongshiyong.66@bytedance.com

ByteDance Search  
Beijing, China

Xijun Xiao\*

xiaoxijun@bytedance.com

ByteDance AML  
Beijing, China

Jinqui Jin\*

jinjinqui.02@bytedance.com

ByteDance Search  
Beijing, China

Xuanyuan Luo

xuanyuanluo@bytedance.com

ByteDance AML  
Hangzhou, China

Zhe Wang

zhewang.tim@gmail.com

ByteDance Search  
Beijing, China

Zheng Chai†

chaizheng.cz@bytedance.com

ByteDance AML  
Hangzhou, China

Shikang Wu†

wushikang@bytedance.com

ByteDance Search  
Beijing, China

Yuchao Zheng

zhengyuchao.yc@bytedance.com

ByteDance AML  
Hangzhou, China

Jingjian Lin

linjingjian000@gmail.com

ByteDance Search  
Beijing, China

## Abstract

Industrial large-scale recommendation models (LRMs) face the challenge of jointly modeling long-range user behavior sequences and heterogeneous non-sequential features under strict efficiency constraints. However, most existing architectures employ a decoupled pipeline: long sequences are first compressed with a query-token based sequence compressor like LONGER, followed by fusion with dense features through token-mixing modules like RankMixer, which thereby limits both the representation capacity and the interaction flexibility. This paper presents **HyFormer**, a unified hybrid transformer architecture that tightly integrates long-sequence modeling and feature interaction into a single backbone. From the perspective of sequence modeling, we revisit and redesign query tokens in LRM s, and frame the LRM modeling task as an alternating optimization process that integrates two core components: *Query Decoding* which expands non-sequential features into *Global Tokens* and performs long sequence decoding over layer-wise key-value representations of long behavioral sequences; and *Query Boosting* which enhances cross-query and cross-sequence heterogeneous interactions via efficient token mixing. The two complementary

mechanisms are performed iteratively to refine semantic representations across layers. Extensive experiments on billion-scale industrial datasets demonstrate that HyFormer consistently outperforms strong LONGER and RankMixer baselines under comparable parameter and FLOPs budgets, while exhibiting superior scaling behavior with increasing parameters and FLOPs. Large-scale online A/B tests in high-traffic production systems further validate its effectiveness, showing significant gains over deployed state-of-the-art models. These results highlight the practicality and scalability of HyFormer as a unified modeling framework for industrial LRMs.

## CCS Concepts

- Information systems → Recommender systems.

## Keywords

Feature Interaction, Large Recommendation Models, Long Sequence Modeling, Scaling Law

## ACM Reference Format:

Yunwen Huang\*, Shiyong Hong\*, Xijun Xiao\*, Jinqui Jin\*, Xuanyuan Luo, Zhe Wang, Zheng Chai†, Shikang Wu†, Yuchao Zheng, and Jingjian Lin. 2025. HyFormer: Revisiting the Roles of Sequence Modeling and Feature Interaction in CTR Prediction. 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>

\*These authors contributed equally.

†Corresponding Authors.

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, or 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

© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-x-xxxx-xxxx-x/YYY/MM

<https://doi.org/XXXXXXX.XXXXXXX>

## 1 Introduction

Modern industrial large-scale recommendation models (LRMs) operate in increasingly complex environments, where accurate prediction relies on jointly modeling long-range user behavior histories and rich heterogeneous features, including user profiles, contextual signals, and cross features. As user engagement grows over

extended time horizons and feature spaces continue to expand, effectively integrating long sequential signals with high-dimensional non-sequential information has become a central challenge for large-scale recommendation and search systems. To address this challenge, recent industrial architectures have largely converged on a separated scaling paradigm that combines *long sequence modeling*[1, 18, 30] with *feature interaction* [6, 10, 19, 21]. Within this paradigm, long user behavior sequences are encoded by dedicated sequence transformers to capture temporal dependencies and user interests, and the compressed sequence token(s) are mixed with other heterogeneous features through token-mixing or interaction modules to enable cross-feature reasoning. This “Long Sequence Modeling, Then Heterogeneous Feature Interaction” pipeline has proven effective and has become the dominant design choice for scaling up modern industrial LRM. Despite strong empirical performance, this prevailing paradigm fundamentally enforces a compressed, late-fusion, and unidirectional interaction pattern. As sequence lengths and model capacities continue to increase, this two-stage design reveals fundamental limitations that restrict both modeling expressiveness and scalability.

- Sequence transformers in existing architectures often rely on overly simplified query representations [5, 27, 28] during sequence compression. In practice, the query tokens used to aggregate long behavior sequences are usually derived from a limited subset of candidate-related or global features, constraining the amount of contextual information that can be leveraged when modeling long-term user interests. However, directly increasing the number of query tokens would lead to a significant degradation in serving efficiency under KV-Cache and M-Falcon mechanisms [2, 23].
- Interactions between sequence-compressed tokens and heterogeneous non-sequential tokens typically occur only at late stages of the model. Under the current paradigm, cross-feature reasoning is deferred until after sequence compression, leading to shallow and implicit interactions between different token types. This delayed fusion limits the model’s ability to capture fine-grained dependencies across multiple behavior sequences and heterogeneous feature groups, and prevents early-layer representations from benefiting from cross-domain contextual information.
- Since interaction modules operate only on compressed sequence representations, increasing model capacity or sequence length primarily improves isolated components rather than enhancing joint representations. As a result, scaling up depth or parameters leads to a lower scaling efficiency, where performance improvements increase at a slower rate with respect to additional computational budgets, as computation is less effectively translated into richer joint representations.

These limitations motivate a fundamental rethinking of how long-range sequence modeling and heterogeneous feature interaction should be integrated. Rather than treating sequence encoding and token mixing as two loosely coupled stages, a unified modeling framework is needed to enable deeper, earlier, and bidirectional interactions between sequential and non-sequential signals.

In this paper, we propose **HyFormer**, a *hybrid transformer* architecture that unifies sequence modeling and feature interaction within a single backbone. HyFormer introduces a set of **global**

**tokens** that serve as a shared semantic interface between long behavior sequences and heterogeneous features. Through a stacked design, HyFormer alternates between two lightweight yet expressive mechanisms. The *Query Decoding* module uses global query tokens to attend over layer-wise key-value representations of long behavioral sequences, allowing global context to directly shape sequence representations. The *Query Boosting* module further strengthens cross-query and cross-sequence interactions via efficient token mixing, progressively enriching semantic representations across layers. This design enables a bidirectional flow of information between sequence modeling and feature interaction components, overcoming the limitations of traditional decoupled pipelines. Extensive experiments on billion-scale industrial datasets demonstrate that HyFormer consistently outperforms strong sequence-based and token-mixing baselines under comparable parameter and FLOPs budgets. Moreover, HyFormer exhibits superior scaling behavior with respect to model FLOPs and parameters, and achieves significant gains in large-scale online A/B tests deployed in high-traffic production systems.

In summary, this paper makes the following contributions:

- We identify fundamental limitations of the prevailing decoupled sequence modeling and feature interaction paradigm in large-scale industrial recommender systems, and analyze how its unidirectional and late-fusion design constrains modeling capacity and scalability.
- We propose HyFormer, a unified hybrid transformer architecture that enables bidirectional, layer-wise interaction between long-range behavioral sequences and heterogeneous features through Query Decoding and Query Boosting, achieving state-of-the-art performance and scalability in real-world industrial settings.
- We empirically verify the effectiveness and its superior scaling performance of the proposed models on a billion-scaled industrial dataset. Currently, HyFormer has been fully deployed at Bytedance, serving billions of users each day.

## 2 Related Work

### 2.1 Traditional Recommendation Paradigms

Modern industrial LRM are typically built upon two major components: behavior-sequence modeling and feature-interaction networks. In this paradigm, user behavior histories are first encoded by dedicated sequence models, whose outputs are then consumed by downstream interaction modules together with heterogeneous non-sequential features. Recent industrial systems have substantially advanced the scalability of sequence modeling along this direction. Methods such as SIM [12], ETA [4], TWIN [3, 13], TransAct [17], and LONGER [2] extend sequence encoders to hundreds or thousands of events through efficient attention mechanisms, hierarchical aggregation, KV caching, and serving-friendly designs. These works demonstrate clear power-law scaling trends in modeling long-range user behaviors under large-scale traffic, while largely preserving a two-stage architecture that decouples sequence encoding from feature interaction.

On the feature-interaction side, early models such as DeepFM[7], xDeepFM[11], and DCNv2[16] model low-order or bounded-degree

feature crosses at scale but suffer from diminishing returns as interaction depth increases. Recent scaling studies like Wukong[25] and RankMixer[29] highlight that cross-module expansion becomes a key driver of industrial performance. These models represent the current state of large-scale feature-interaction design, yet the interaction stack and sequence encoder remain loosely coupled in most production pipelines, resulting in late fusion and preventing unified optimization across heterogeneous signals.

## 2.2 Unified Recommendation Architectures

To reduce the fragmentation between sequence modeling and feature interaction, recent studies explore unified architectures that handle heterogeneous signals within a single backbone. Hierarchical generative architectures such as HSTU[24] represent a unified recommendation paradigm by performing sequence transduction conditioned on contextual and candidate signals. InterFormer[22] bridges the gap between sequence encoders and interaction networks by introducing learnable interaction tokens that enable bidirectional signal exchange. MTGR [8] further pushes unification by re-organizing user, behavior, real-time, and candidate features into heterogeneous tokens and encoding them with a shared Transformer-style backbone, enabling both sequence information and cross features to be modeled coherently. Following MTGR, OneTrans[26] shares a similar direction by using a single Transformer to jointly capture sequence dependencies and high-order feature interaction, while simplifying the Transformer structure with a pyramid-compression style. This work can be regarded as a simplified version compared to MTGR.

As MTGR[8] and OneTrans[26] simply increase the number of query tokens as the number of all the non-sequence tokens, it would be readily observed a significant drop in serving efficiency in practice (see Section 4). Besides, a unified transformer structure for modeling feature interaction is generally insufficient in industrial-scale LRM [29]. Overall, unified architectures represent a step toward dissolving the long-standing separation between sequence models and feature-interaction stacks, though achieving full unification with minimal architectural overhead remains an open challenge.

## 3 Methodology

### 3.1 Problem Statement

Let  $\mathcal{U}$  and  $\mathcal{I}$  denote the user and item spaces. For a user  $u \in \mathcal{U}$ , denote the raw behavioral history as  $S = [i_1^{(u)}, \dots, i_K^{(u)}]$  with each  $i_t^{(u)} \in \mathcal{I}$ , and let  $u$  represent the accompanying non-sequential descriptors such as profile attributes, contextual signals, and cross features. Given a candidate item  $v \in \mathcal{I}$ , the goal is to estimate the probability that user  $u$  engages with item  $v$ :

$$P(y = 1 | S, u, v) \in [0, 1], \quad (1)$$

where  $y \in \{0, 1\}$  indicates whether the interaction occurs.

The model parameters are learned from historical data  $\mathcal{D} = \{(S, u, v, y)\}$  by minimizing the standard binary cross-entropy objective:

$$\mathcal{L} = -\frac{1}{|\mathcal{D}|} \sum_{(S, u, v, y) \in \mathcal{D}} \left[ y \log \hat{y} + (1 - y) \log(1 - \hat{y}) \right], \quad (2)$$

where  $\hat{y} = f_\theta(S, u, v)$  denotes the predicted engagement probability produced by the LRM.

### 3.2 Overall Framework

Traditional LRM architectures generally adopt a pipelined design by performing sequence modeling like LONGER [2] first, and the query token containing the compressed sequence information is then used for subsequence feature interaction like RankMixer[29]. As discussed before, this separate pipeline generally results in an insufficient modeling for both sequence modeling and heterogeneous feature interaction. To overcome the limitation, this work proposes a unified hybrid framework that jointly models non-sequential (NS) tokens and long behavioral sequences through a stack of HyFormer layers.

The overall architecture of HyFormer is presented in Figure 1. As shown in the figure, each HyFormer layer integrates two complementary mechanisms: (1) *Query Decoding*, which expands non-sequential and sequential features into multiple semantic **global tokens** (i.e., sequence queries) via MLP-based query generation and performs cross-attention over long-sequence K/V pairs, enabling global information to directly shape the representation of sequence tokens; and (2) *Query Boosting*, which applies MLP-Mixer-style token mixing to strengthen interactions among decoded queries and non-sequence tokens.

By tightly coupling global heterogeneous-feature mixing with efficient long-sequence modeling, the proposed framework achieves richer heterogeneous interactions, deeper utilization of sequential structure, and more favorable performance and computation cost compared with existing separate pipelined architectures.

### 3.3 Query Generation

**3.3.1 Input Tokenization.** Following the tokenization strategy in RankMixer[29], input tokens can be organized either by *semantic grouping* or by *automatic splitting*. Semantic grouping partitions tokens according to their intrinsic meanings (e.g., user, context, or behavior semantics), while auto-split flattens all features into a single embedding and applies uniform splitting without explicit semantic distinctions. In practice, given the clear semantic roles of input features in our setting, HyFormer adopts semantic grouping to preserve structured inductive bias and improve interpretability.

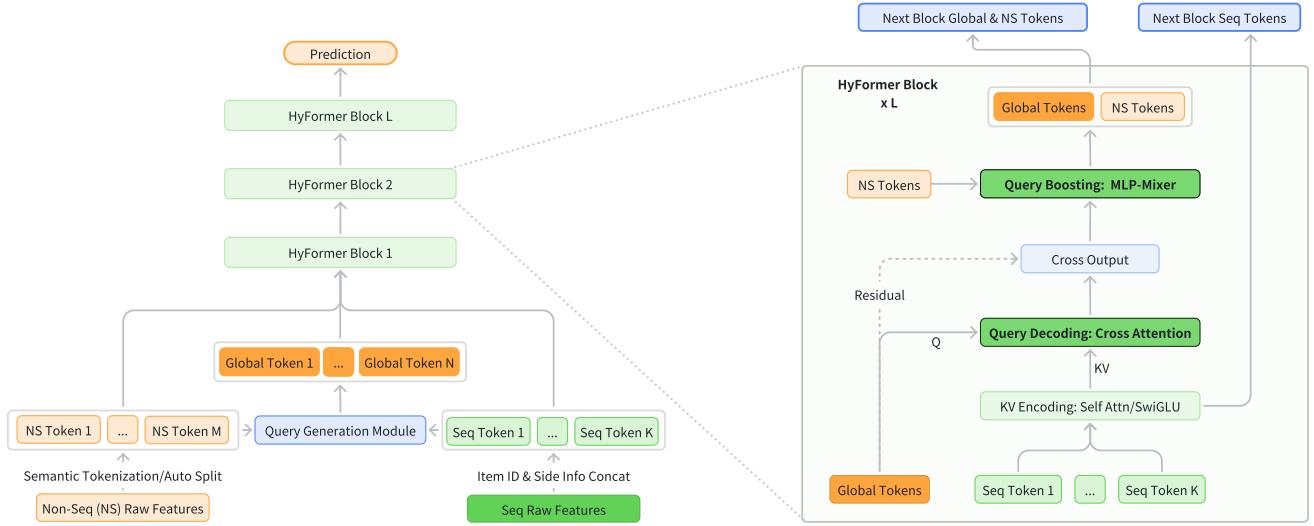
**3.3.2 Query Generation.** The Query Generation module converts heterogeneous non-sequential features into semantic query tokens used for decoding long behavioral sequences. All non-sequential feature vectors  $F_1, F_2, \dots, F_M \in \mathbb{R}^{1 \times D}$  are concatenated and mapped through a lightweight feed-forward network. In addition, a global sequence-level summary is obtained via pooling over the behavioral sequence representations and treated as an additional shared input, analogous to non-sequential features.

The queries are generated by combining non-sequential features with the pooled sequence summary through a lightweight projection:

$$Q = [\text{FFN}_1(\text{Global Info}), \dots, \text{FFN}_N(\text{Global Info})] \in \mathbb{R}^{N \times D}, \quad (3)$$

where

$$\text{Global Info} = \text{Concat}(F_1, \dots, F_M, \text{MeanPool}(Seq)). \quad (4)$$



**Figure 1: Overview of the proposed HyFormer architecture.** The new arch introduces global tokens that derived from original "candidate item" in sequence modeling, and revisits the roles of long-sequence modeling and feature interaction by boosting the query capacity of long-sequence via MLP-Mixer-based feature interaction. It frames the LRM modeling task as an alternating optimization process through the alternation of *Query Decoding* and *Query Boosting* modules.

To maintain serving efficiency, the module supports feature selection and optional query compression, keeping the number of generated queries stable while preserving sufficient representational capacity for downstream decoding.

In deeper HyFormer layers, queries are not regenerated through MLPs. Instead, each layer reuses the queries from the previous layer, effectively using deeper cross-attention outputs as updated queries to interrogate the long sequence with progressively richer semantics.

### 3.4 Query Decoding

The Query Decoding module is responsible for transforming non-sequential features into semantic queries and extracting target-aware information from long behavioral sequences through cross attention. With the layer-wise key-value representations of the long sequence produced by the Sequence Representation Encoding module, Query Decoding module decodes the K/V representation with the multiple query tokens from the Query Generation Module via the multi-query cross attention mechanism.

**3.4.1 Sequence Representation Encoding.** HyFormer supports multiple sequence encoding strategies with different capacity-efficiency trade-off. Given the behavioral sequence  $S$ , each strategy produces layer-wise key-value representations  $(K_l^{(s)}, V_l^{(s)})$  for subsequent decoding.

(i) *Full Transformer Encoding*[15]. At the highest modeling capacity, a standard Transformer encoder is applied:

$$H_l = \text{TransformerEnc}_l(S), \quad (5)$$

which captures fine-grained interactions and long-range dependencies via full self-attention.

(ii) *LONGER [2]-style Efficient Encoding*. To improve efficiency for long sequences, full self-attention is replaced by cross-attention between a compact short sequence and the full history:

$$H_l = \text{CrossAttn}(S_{\text{short}}, S, S), \quad (6)$$

where  $S_{\text{short}}$  is a compact short sequence with length  $L_H \ll L_S$ . Here,  $S_{\text{short}}$  serves as the query, while  $S$  is used as both keys and values. This formulation replaces full self-attention and reduces the computational complexity from  $\mathcal{O}(L_S^2)$  to  $\mathcal{O}(L_H L_S)$ .

(iii) *Decoder-style Lightweight Encoding*. For latency-critical scenarios, sequence representations are transformed using attention-free feed-forward operations:

$$H_l = \text{SwiGLU}_l(S), \quad (7)$$

trading contextual capacity for minimal computational cost.

Across all variants, the resulting representations are linearly projected to obtain layer-specific key-value states:

$$K_l = H_l W_l^K, \quad V_l = H_l W_l^V. \quad (8)$$

Key-value states are recomputed at each layer, allowing sequence features to evolve jointly with decoder depth while supporting flexible deployment configurations.

**3.4.2 Query Decoding via Cross-Attention.** Given the sequence-specific query tokens and the corresponding layer-wise key-value representations, HyFormer performs Query Decoding through cross-attention. For each behavioral sequence  $S$  at layer  $l$ , the decoded query representations are obtained as:

$$\tilde{Q}_{(l)} = \text{CrossAttn}(Q_{(l)}, K_{(l)}, V_{(l)}), \quad (9)$$

where  $\text{CrossAttn}(\cdot)$  denotes a standard multi-head cross-attention operation, and  $Q_{(l)} \in \mathbb{R}^{N \times D}$  represents the query token used at layer  $l$ .

This decoding step allows global, non-sequential features to directly attend to long behavioral sequences, injecting contextual signals into the sequence-aware query representations. The decoded query  $\tilde{Q}_{(l)}$  are then used as the semantic interface for subsequent interaction and boosting modules.

### 3.5 Query Boosting

The Query Boosting module enhances query representations before they are fed into the subsequent cross-attention layer. After the decoding step, the queries already encode sequence-aware information, but their interactions with static non-sequential heterogeneous features remain underexplored. Query Boosting addresses this limitation by explicitly mixing information across query tokens and injecting additional non-sequence-feature signals.

With the decoded output, the unified query representation is defined as

$$Q = [\tilde{Q}_{(l)}, F_1, \dots, F_M] \in \mathbb{R}^{T \times D}, \quad (10)$$

where  $T = N + M$ ,  $\tilde{Q}_{(l)} \in \mathbb{R}^{N \times D}$  denotes the set of decoded query tokens obtained at layer  $l$ , and the remaining  $M$  tokens correspond to non-sequential feature embeddings.

Specifically, the boosting module applies an MLP-Mixer-style [14] lightweight token-mixing operation inspired by RankMixer [29] to enrich the decoded queries. Each query token  $q_t \in Q$  is first partitioned into  $T$  channel subspaces:

$$q_t = [q_t^{(1)} \| q_t^{(2)} \| \dots \| q_t^{(T)}], \quad q_t^{(h)} \in \mathbb{R}^{D/T}. \quad (11)$$

For each subspace index  $h \in \{1, \dots, T\}$ , MLP-Mixer aggregates information from all token positions by concatenating the corresponding subspaces:

$$\tilde{q}_h = \text{Concat}(q_1^{(h)}, q_2^{(h)}, \dots, q_T^{(h)}) \in \mathbb{R}^D. \quad (12)$$

Collecting all mixed tokens yields the token-mixed representation

$$\hat{Q} = [\tilde{q}_1, \tilde{q}_2, \dots, \tilde{q}_T] \in \mathbb{R}^{T \times D}. \quad (13)$$

The mixed queries are further refined by a lightweight per-token feed-forward module:

$$\tilde{Q} = \text{PerToken-FFN}(\hat{Q}), \quad (14)$$

where  $\text{PerToken-FFN}(\cdot)$  applies an independent feed-forward transformation to each query token, enabling subspace-specific refinement while preserving linear computational complexity.

Finally, a residual connection is applied to stabilize optimization and preserve the original decoded semantics:

$$Q_{\text{boost}} = Q + \tilde{Q}. \quad (15)$$

The boosted queries are then passed to the next HyFormer layer, allowing deeper layers to interrogate long behavioral sequences with progressively richer and more expressive representations.

### 3.6 HyFormer Module

The HyFormer module is constructed by stacking multiple layers, each consisting of a *Query Decoding* block followed by a *Query Boosting* block. At each layer, semantic queries interact with the long behavioral sequence via cross-attention, and the resulting sequence-aware representations are further refined to serve as inputs to deeper layers.

Formally, at layer  $l$ , the *Query Decoding* block takes the incoming global queries  $Q^{(l-1)}$  and performs cross-attention over the layer-wise key-value representations ( $K^{(l)}, V^{(l)}$ ) derived from the long sequence:

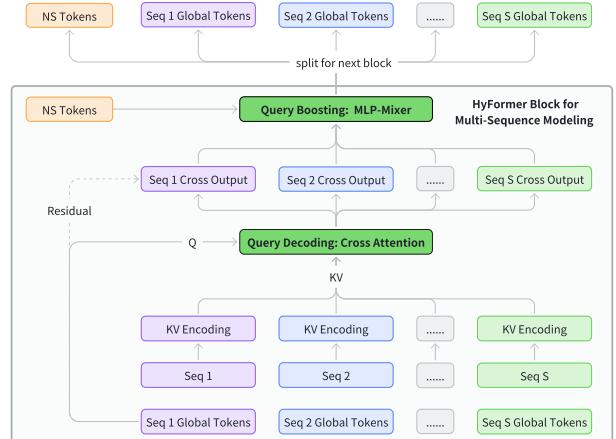
$$\hat{Q}^{(l)} = \text{CrossAttn}(Q^{(l-1)}, K^{(l)}, V^{(l)}). \quad (16)$$

The decoded queries  $\hat{Q}^{(l)}$  are then concatenated with the non-sequential tokens and passed to the *Query Boosting* block, which applies a lightweight token-wise transformation to enrich the query representations:

$$\tilde{Q}^{(l)} = \text{QueryBoost}(\text{Concat}(\hat{Q}^{(l)}, \text{NS Tokens})), \quad (17)$$

By stacking multiple such layers, HyFormer progressively refines the semantic queries, enabling deeper layers to abstract the long sequence with increasingly expressive representations. The output of the top HyFormer layer is fed into downstream MLPs for final predictions, enabling efficient and flexible integration of heterogeneous non-sequential features with long behavioral sequences in LRMAs.

### 3.7 Multi-Sequence Modeling



**Figure 2: Multi-Sequence Modeling in HyFormer.**

In industrial recommendation scenarios, user behaviors are often organized as multiple heterogeneous sequences, for example, video-watch sequence and product-purchase sequence. As practical multi-sequences are generally with different feature spaces and semantic representations, we empirically find that simple sequence-merge that adopted by MTGR[8] or OneTrans[26] would lead to a significant degradation in performance (see Section 4). Thus, instead of merging different sequences into a single unified stream, HyFormer

processes each behavior sequence independently in each HyFormer block for both efficiency and effectiveness. As shown in Figure 2, for each sequence, a dedicated set of query tokens is constructed and used to perform Query Decoding over the corresponding sequence representations. This design preserves sequence-specific semantics during decoding, while enabling cross-sequence interaction to be handled later through query-level token mixing, without requiring explicit sequence concatenation.

### 3.8 Training and Deployment Optimization

**3.8.1 GPU Pooling for Long-Sequence.** User long-sequence features can be extremely large, incurring significant data-transfer overhead (e.g., Host-to-Device memory copies) and high memory pressure on the host. Fortunately, the number of truly unique feature IDs in such sequences is limited (typically 25% of the total tokens). We exploit this sparsity to deduplicate features, substantially reducing transfer costs and host-memory footprint. Specifically, before graph execution, features are stored in a compressed embedding-table. During execution, we build a high-performance forward operator that reconstructs the original sequence features directly on the GPU. In the backward pass, the companion backward operator aggregates gradients from the sequence features into gradients for the embedding table. These gradients are then propagated upstream to update the sparse parameters.

**3.8.2 Asynchronous AllReduce.** To mitigate idle time introduced by synchronous gradient aggregation, the system enables asynchronous AllReduce, allowing the gradient synchronization of step  $k$  to overlap with the forward and backward computation of step  $k+1$ . This design effectively eliminates communication bubbles and maximizes GPU utilization. The trade-off, however, is the introduction of one-step staleness for dense parameters: since their gradients are only available after the asynchronous reduction completes, the update rule becomes  $W_k = W_{k-1} + g_{k-1}$  indicating that dense parameters at step  $k$  use gradients from the previous step. In contrast, sparse parameters can be updated immediately after their local gradients are computed, yielding  $W_k = W_{k-1} + g_k$  and thus effectively staying one step ahead of the dense parameter updates. Although this hybrid update schedule introduces a small degree of temporal inconsistency between dense and sparse parameter states, empirical results indicate that this staleness does not degrade convergence quality or model performance in practice.

## 4 Experiments

### 4.1 Experimental Setting

**4.1.1 Datasets.** We evaluate our model on the Click Rate (CTR) prediction task in the Douyin Search System, a real-world and large-scale industrial search recommendation scenario at ByteDance. The dataset is derived from a subset of online user interaction logs spanning 70 consecutive days and comprises 3 billion samples. Each sample incorporates user features, query features, document features, cross-features, and several sequential features. The three primary sequences used in the model are defined as follows:

- Long-term sequence: The user’s long-term search and click behavior sequence, the length of which can be tailored as needed, with an upper bound of 3000 adopted in this study.

- Search sequence: The user’s top-50 search behavior items, filtered by the Query Search module.
- Feed sequence: The user’s top-50 feed behavior items, filtered by the Query Search module.

**4.1.2 Baselines.** We compared our model against several strong baselines, which can be categorized into two architectural paradigms: Traditional Two-Stage Models and Unified-Architecture Models.

Traditional Two-Stage Models follow the prevalent mainstream design where sequence modeling and feature interaction are separated into two sequential stages. Specifically, sequential representations are first generated through a dedicated sequence modeling module and subsequently crossed with token-level representations of other features. For long-sequence modeling, we used the LONGER[2] or Full Transformer[15] architecture. To capture interactions between tokenized features, we employed several established architectures designed for feature interaction, including RankMixer[29], Full Transformer[15], and Wukong[25].

Unified-Block Models, in contrast, adopt a joint modeling approach where both sequential and non-sequential features are tokenized and processed simultaneously within a single model block. This integrates sequence modeling and heterogeneous feature interaction into one unified stage. An example is MTGR[8], which tokenizes all features and models them jointly using a transformer-style backbone. Similarly, OneTrans[26] follows a comparable simplified design, as it adopts a pyramid-compressed structure as backbone. In our implementation of the MTGR/OneTrans models, we performs MTGR/OneTrans (LONGER) for only cross-attention between non-sequential and sequential features, without the calculation of inner-sequence self-attention. Besides, we perform MTGR/OneTrans (Full Transformer) with full self-attention in sequence to achieve better performance with increased FLOPs.

**4.1.3 Evaluation Metrics.** For offline evaluation, we employ Query-level AUC (Area Under the Curve) which calculates the AUC[9] for samples within each query and then averages the results across all queries. To assess the relative efficacy, we adopt the relative improvement (Imp.)[20] as our key metric for measuring gains in AUC. We also report the number of dense parameters and training FLOPs, with the latter computed using a batch size of 2048.

**4.1.4 Implementation Details.** For the convenience of our experiment, the recommendation model is cold-started for offline evaluation and warmed up with checkpoints for online evaluation. All baselines utilize the same 2048 batch size and optimizer settings. The input token count for all MLMixer modules is aligned to 16. In the multi-sequence HyFormer implementation, it comprises 13 non-sequential tokens and 3 global tokens—one per sequence—summing to a total of 16 tokens. All models are trained with the same hyperparameter tuning, and experiments are conducted on a 64-GPUs cluster.

### 4.2 Overall Performance

**4.2.1 Comparison of existing methods.** Our proposed HyFormer architecture achieves the highest AUC among all evaluated models, outperforming both traditional two-stage models (termed BaseArch) and other unified-block models (termed UniArch). Within the BaseArch group, performance varies significantly with component choice:

**Table 1: Overall Performance on Industrial Dataset**

<b>Sequence Modeling</b>	<b>Feature Interaction</b>	<b>AUC↑</b>	<b>AUC Imp.</b>	<b>Params (<math>\times 10^6</math>)</b>	<b>FLOPs (<math>\times 10^{12}</math>)</b>
<b>BaseArch: Traditional Two-Stage Models</b>					
LONGER[2]	RankMixer[14, 29]	0.6478	–	386	3.5
LONGER[2]	Full Transformer[15]	0.6472	-0.41%	416	6.2
LONGER[2]	Wukong[25]	0.6465	-0.88%	385	5.2
Full Transformer[15]	RankMixer[14, 29]	0.6481	+0.20%	388	6.6
Full Transformer[15]	Full Transformer[15]	0.6474	-0.27%	418	9.3
Full Transformer[15]	Wukong[25]	0.6468	-0.68%	387	8.3
<b>UniArch: Unified-Block Models</b>					
MTGR/OneTrans (w/ LONGER)[8, 26]		0.6480	+0.14%	406	6.6
MTGR/OneTrans (w/ Full Transformer)[8, 26]		0.6483	+0.34%	450	21.9
<b>HyFormer (Ours)</b>		<b>0.6489</b>	<b>+0.74%</b>	418	3.9

for feature interaction, RankMixer [29] consistently outperforms Self-Attention and Wukong[25], while for sequence modeling, incorporating full self-attention within the sequence generally yields gains. Notably, the best-performing BaseArch combination that employs a Full Transformer for sequence modeling with RankMixer still falls short of HyFormer, owing to its inherent limitation of unidirectional information flow. Furthermore, it is evident from the table that HyFormer demonstrates superior computational efficiency. Despite achieving the highest accuracy, it requires only  $3.9 \times 10^{12}$  total FLOPs, including forward and backward propagation during training. This computational cost is significantly lower than that of most competitors, including other high-performing models such as MTGR[8]. The overall performance results highlight the inherent limitations of the traditional two-stage paradigm.

Unified architectures like HyFormer and MTGR demonstrate that integrating sequence modeling and feature interaction into a cohesive design enhances overall effectiveness. However, as evidenced by the results in the table, MTGR/OneTrans[8, 26] relies on Self-Attention for feature interaction—an approach that often degrades AUC and significantly compromises computational efficiency in the interaction module[29]. HyFormer, therefore, distinguishes itself by achieving the best accuracy without resorting to such costly substitutions or complex modeling on the sequence key-value side. This validates its core design principle of iterative query decoding and boosting within a unified block. Besides, MTGR/OneTrans combines Global Tokens and Seq Tokens as keys, while exclusively employing Global Tokens as queries. This design facilitates Global Tokens to attend more readily to themselves rather than sequence tokens. In contrast, HyFormer enforces a separated information flow: it first compresses and absorbs concrete sequence item information into Global Tokens, and then conducts interaction between different abstract Global Tokens, with this two-step process repeatedly stacked across layers. Furthermore, the hybrid architecture of HyFormer offers greater flexibility for future scaling. It allows for independent adjustment of interaction layers/dimensions and sequence modeling layers/dimensions, providing a more adaptable framework than methods that rigidly bind feature interaction and sequence modeling within a single standard attention layer.

**4.2.2 Ablation Study.** Table 2 presents an ablation study on the primary contributors to HyFormer’s performance improvement. First, we ablate the components of the query. The HyFormer query is generated from three sources: global non-sequential features, multiple sequences pooling tokens, and the original target features. Experiments show that reverting the query to its original, target-feature-only state severely limits subsequent deep feature interaction, causing a 0.34% AUC decline. Removing the cross-sequence pooling tokens from the full query also leads to a 0.20% AUC loss, confirming that inter-sequence interaction contributes meaningfully within the HyFormer structure.

Second, we evaluate the overall architectural change. Restoring the baseline architecture (LONGER + RankMixer), which applies sequential modeling followed by separate feature interaction, shows that even with enriched query information, the lack of deepened interaction caps the gains, yielding only a 0.14% AUC improvement(-0.74% vs -0.60%). In contrast, within the HyFormer framework which is designed to strengthen interaction throughout the model, expanding query information delivers a significantly larger 0.34% AUC gain.

Third, we conduct an ablation study on the multi-sequence modeling strategy within HyFormer. Two principal paradigms exist for handling multiple sequences: merging sequences into one via dimension alignment and concatenation for joint modeling, or keeping sequences separate and modeling them independently. HyFormer adopts the latter approach, using distinct query tokens for each sequence. In our experiments, it is observed that the sequence merging and query sharing resulted in a significant AUC loss of 0.27%. Consequently, this presents the advantages of the HyFormer in expanding queries and enabling broader feature interaction. Additionally, merging forces disparate sequences to share global tokens, ignoring their distinctiveness. The resulting representations capture far less differentiated information than HyFormer’s separate modeling of each sequence. We speculate that this inherent limitation of sequence merging also partly explains why models like MTGR and OneTrans underperform compared to HyFormer.

In summary, the HyFormer architecture provides a versatile multi-sequence modeling framework by employing independent

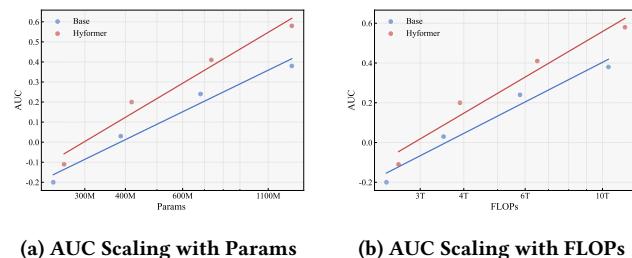
**Table 2: Ablation Study on HyFormer Components**

Configuration	AUC↑	AUC Imp.	Params ( $\times 10^6$ )	FLOPs ( $\times 10^{12}$ )
<b>Ablation of Query Global Context</b>				
HyFormer	0.6489	-	418	3.9
Query w/o Seq Pooling Tokens	0.6486	-0.20%	415	3.9
Query w/o Nonseq and Seq Pooling Tokens	0.6484	-0.34%	414	3.8
<b>Ablation of Query Boosting</b>				
HyFormer	0.6489	-	418	3.9
HyFormer w/o Global Tokens	0.6484	-0.34%	414	3.8
BaseArch w/ Global Tokens	0.6480	-0.60%	505	3.6
BaseArch w/o Global Tokens	0.6478	-0.74%	387	3.5
<b>Ablation of Multi-Sequence Modeling</b>				
HyFormer	0.6489	-	418	3.9
HyFormer + Merge Seq	0.6485	-0.27%	397	3.9

tokens for different sequences, thereby eliminating the need for forced alignment of side information or sparse dimensions across sequences. This design not only preserves the inherent distinctions between sequences to a great extent, but also enables the adaptive allocation of more global tokens to more important sequences, which has yielded measurable gains in our offline experiments.

### 4.3 Scaling Analysis

In this section, we present the scaling analysis of model performance with respect to sequence side information, FLOPs, and the number of parameters. As shown in the overall performance in Table 1, under the paradigm of first performing sequential modeling and then performing heterogeneous feature interaction, LONGER + RankMixer achieves the best performance and is currently the production baseline. Therefore, we use it as the control group as BaseArch in the scaling experiments to compare the scaling performance of the HyFormer architecture.

**Figure 3: Scaling performance with respect to FLOPs and model parameters.**

**4.3.1 Parameters & FLOPs.** We examine the scaling law of the HyFormer architecture across model sizes ranging from 200M to 1B+ parameters. The results are shown in Figure 3(a). As can be observed, while HyFormer initially outperforms the baseline LONGER + RankMixer model, it maintains strong scaling benefits overall, exhibiting a steeper slope than the baseline. This indicates that the

bidirectional information flow, enabled by the alternating stacked layers of the LONGER and RankMixer in HyFormer, allows it to achieve significantly greater gains from increasing depth compared to the baseline at similar parameter scales. A similar pattern emerges when scaling law is analyzed in terms of computational cost (FLOPs). As shown in Figure 3(b), AUC increases steadily with FLOPs, following a strong power-law trend. This indicates that increasing computational resources enables the model to process sequences with richer information, benefiting from the expansion of the initial query and the repeated enhancement of the query through feature interaction in MLP-Mixer, ultimately leading to greater AUC improvement.

These results suggest that the architectural design of HyFormer prioritizes scaling efficiency, yielding greater gains per parameter via enriched heterogeneous feature interactions, which results in a steeper performance scaling curve.

**Table 3: Scaling with Sequence Sparse Dim**

Seq Length	Arch	Seq Sparse Dim	AUC↑	AUC Imp.	AUC Imp. Gap
1k	BaseArch	64	0.6478	-	-
		224	0.6484	+0.41%	-
	HyFormer	64	0.6489	-	-
		224	0.6497	+0.54%	+0.13%
3k	BaseArch	64	0.6486	-	-
		224	0.6490	+0.27%	-
	HyFormer	64	0.6499	-	-
		224	0.6507	+0.53%	0.26%

**4.3.2 Sparse Dim.** We also analyzed how model performance varies with the expansion of the sequence token input dimension (sparse embedding dim), i.e., the richness of sequence side information. Our experiments show that, regardless of sequence length, enriching sequence side information consistently brings greater benefits to the HyFormer framework than to the baseline LONGER + RankMixer framework. As shown in Table 3, for sequences of length 1000, expanding the sparse dimension width from the original 64 dimensions with three side information types (item ID, search query textnet classification, and timestamp) to 224 dimensions with seven types (adding search query ID, author ID, event ID, and playtime) yielded a AUC imp. of 0.41% for the baseline, compared to a 0.54% gain for HyFormer. The improvement for HyFormer is significantly larger, a trend that holds across other sequence lengths in experiments. Furthermore, the performance gap between HyFormer and the BaseArch widens as sequences grow longer, with the additional gain from dimension expansion increasing from 0.13% at 1k sequence length to 0.26% at 3k.

These results indicate that expanding the sequence key/value information delivers greater value within the HyFormer framework, and this advantage becomes more pronounced with longer sequences. The benefit stems from HyFormer’s ability to integrate richer global information into sequence queries, coupled with the bidirectional information flow between its LONGER and Mixer modules, which collectively enable more thorough feature interaction.

#### 4.4 Online A/B Tests

This section presents the online A/B test results for HyFormer model on the Douyin Search platform, where it was evaluated against strong existing RankMixer baseline. For online evaluation, we employ three key metrics: Average Watch Time Per User, Video Finish Play Count Per User, and Query Change Rate. In particular, the Query Change Rate quantifies the probability of a user manually refining a search query to a more specific one (e.g., from “iPhone” to “iPhone 17 Pro”), which is calculated as follows:

$$\text{query change rate} = \frac{N_{\text{reform}}}{N_{\text{total}}} \quad (18)$$

where  $N_{\text{reform}}$  is the number of distinct user-query pairs with query reformulation, and  $N_{\text{total}}$  is the total number of distinct user-query pairs. This metric serves as an indicator of a negative search experience for users.

As shown in Table 4, the online A/B Test confirms substantial improvements across key metrics: a 0.293% increase in average watch time per user, a 1.111% growth in video finish play count per user and a 0.236% decrease in query change rate. These significant gains demonstrate the practical value and effectiveness of HyFormer in a real-world, billion-user platform environment.

**Table 4: Online A/B Test Results on Douyin**

Online Test Metrics	Gain
Average Watch Time Per User ↑	+0.293%
Video Finish Play Count Per User ↑	+1.111%
Query Change Rate ↓	-0.236%

## 5 Conclusions

In this paper, we propose the HyFormer architecture. In contrast to the prevalent “Long Sequence Modeling, Then Feature Interaction” paradigm which first performs sequential modeling and then conducts heterogeneous feature interaction in a unidirectional flow, HyFormer introduces Global Tokens to redefine the roles of long-sequence modeling and feature interaction by boosting the query capacity via feature interaction. The architecture alternates between two core components: Query Decoding and Query Boosting. From a sequential modeling perspective, this corresponds to an iterative optimization process that alternates between decoding long sequences using the Global Tokens and enhancing the Global Tokens through cross-feature interaction. This design provides a novel and effective framework for more thorough sequence modeling and feature interaction, while also providing a flexible paradigm for multi-sequence modeling. Extensive offline and online experiments validate the superiority of upgrading from a unidirectional information flow to a bidirectional, co-evolutionary paradigm, and also raise the scaling ceiling for future LRM in industry.

## References

- [1] Fedor Borisyuk, Mingzhou Zhou, Qingquan Song, Siyu Zhu, Birjodh Tiwana, Ganesh Parameswaran, Siddharth Dangi, Lars Hertel, Qiang Charles Xiao, Xiaochen Hou, et al. 2024. LiRank: Industrial Large Scale Ranking Models at LinkedIn. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4804–4815.
- [2] Zheng Chai, Qin Ren, Xijun Xiao, Huizhi Yang, Bo Han, Sijun Zhang, Di Chen, Hui Lu, Wenlin Zhao, Lele Yu, Xionghang Xie, Shiru Ren, Xiang Sun, Yaocheng Tan, Peng Xu, Yuchao Zheng, and Di Wu. 2025. LONGER: Scaling Up Long Sequence Modeling in Industrial Recommenders. *arXiv:2505.04421 [cs.IR]* <https://arxiv.org/abs/2505.04421>
- [3] Jianxin Chang, Chenbin Zhang, Zhiyi Fu, Xiaoxue Zang, Lin Guan, Jing Lu, Yiqun Hui, Dewei Leng, Yanan Niu, Yang Song, and Kun Gai. 2023. TWIN: Two-stage Interest Network for Lifelong User Behavior Modeling in CTR Prediction at Kuaishou. *arXiv:2302.02352 [cs.IR]* <https://arxiv.org/abs/2302.02352>
- [4] Qiwei Chen, Yue Xu, Changhua Pei, Shanshan Lv, Tao Zhuang, and Junfeng Ge. 2022. Efficient long sequential user data modeling for click-through rate prediction. *arXiv preprint arXiv:2209.12212* (2022).
- [5] 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).
- [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] Hui Feng Guo, Ruiming Tang, Yunning Ye, Zhenguo Li, and Xiugiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. *arXiv:1703.04247 [cs.IR]* <https://arxiv.org/abs/1703.04247>
- [8] Ruidong Han, Bin Yin, Shangyu Chen, He Jiang, Fei Jiang, Xiang Li, Chi Ma, Mincong Huang, Xiaoguang Li, Chunzhen Jing, Yueming Han, MengLei Zhou, Lei Yu, Chuan Liu, and Wei Lin. 2025. MTGR: Industrial-Scale Generative Recommendation Framework in Meituan. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM '25)*. ACM, 5731–5738. doi:10.1145/3746252.3761565
- [9] David J Hand and Robert J Till. 2001. A simple generalisation of the area under the ROC curve for multiple class classification problems. *Machine learning* 45, 2 (2001), 171–186.
- [10] Kirill Khrylenko, Artem Matveev, Sergei Makeev, and Vladimir Baikalov. 2025. Scaling recommender transformers to one billion parameters. *arXiv preprint arXiv:2507.15994* (2025).
- [11] 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 (KDD '18)*. ACM, 1754–1763. doi:10.1145/3219819.3220023
- [12] Pi Qi, Xiaoqiang Zhu, Guorui Zhou, Yujing Zhang, Zhe Wang, Lejian Ren, Ying Fan, and Kun Gai. 2020. Search-based User Interest Modeling with Lifelong Sequential Behavior Data for Click-Through Rate Prediction. *arXiv:2006.05639 [cs.IR]* <https://arxiv.org/abs/2006.05639>
- [13] Zihua Si, Lin Guan, Zhongxiang Sun, Xiaoxue Zang, Jing Lu, Yiqun Hui, Xingchao Cao, Zeyu Yang, Yichen Zheng, Dewei Leng, Kai Zheng, Chenbin Zhang, Yanan Niu, Yang Song, and Kun Gai. 2024. TWIN V2: Scaling Ultra-Long User Behavior Sequence Modeling for Enhanced CTR Prediction at Kuaishou. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24)*. ACM, 4890–4897. doi:10.1145/3627673.3680030
- [14] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. 2021. Mlp-mixer: An all-mlp architecture for vision. *Advances in neural information processing systems* 34 (2021), 24261–24272.
- [15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).
- [16] 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 (WWW '21)*. ACM, 1785–1797. doi:10.1145/3442381.3450078
- [17] Xue Xia, Pong Eksombatchai, Nikil Pancha, Dhruvil Deven Badani, Po-Wei Wang, Neng Gu, Saurabh Vishwas Joshi, Nazanin Farahpour, Zhiyuan Zhang, and Andrew Zhai. 2023. TransAct: Transformer-based Realtime User Action Model for Recommendation at Pinterest. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23)*. ACM, 5249–5259. doi:10.1145/3580305.3599918
- [18] Songpei Xu, Shijia Wang, Da Guo, Xianwen Guo, Qiang Xiao, Bin Huang, Guanlin Wu, and Chuanjiang Luo. 2025. Climber: Toward Efficient Scaling Laws for Large Recommendation Models. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM '25)*. ACM, 6193–6200. doi:10.1145/3746252.3761561
- [19] Yi Xu, Chaofan Fan, Jinxin Hu, Yu Zhang, Zeng Xiaoyi, and Jing Zhang. 2025. STORE: Semantic Tokenization, Orthogonal Rotation and Efficient Attention for Scaling Up Ranking Models. *arXiv preprint arXiv:2511.18805* (2025).
- [20] Ling Yan, Wu-Jun Li, Gui-Rong Xue, and Dingyi Han. 2014. Coupled Group Lasso for Web-Scale CTR Prediction in Display Advertising. In *Proceedings of the 31st International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 32)*, Eric P. Xing and Tony Jebara (Eds.). PMLR, Beijing, China, 802–810. <https://proceedings.mlr.press/v32/yan14.html>

- [21] Liren Yu, Wenming Zhang, Silu Zhou, Tao Zhang, Zhixuan Zhang, and Dan Ou. 2025. HHFT: Hierarchical Heterogeneous Feature Transformer for Recommendation Systems. arXiv:2511.20235 [cs.IR] <https://arxiv.org/abs/2511.20235>
- [22] Zhichen Zeng, Xiaolong Liu, Mengyue Hang, Xiaoyi Liu, Qinghai Zhou, Chaofei Yang, Yiqun Liu, Yichen Ruan, Laming Chen, Yuxin Chen, Yujia Hao, Jiaqi Xu, Jade Nie, Xi Liu, Buyun Zhang, Wei Wen, Siyang Yuan, Hang Yin, Xin Zhang, Kai Wang, Wen-Yen Chen, Yiping Han, Huayu Li, Chunzhi Yang, Bo Long, Philip S. Yu, Hanghang Tong, and Jiyan Yang. 2025. InterFormer: Effective Heterogeneous Interaction Learning for Click-Through Rate Prediction. arXiv:2411.09852 [cs.IR] <https://arxiv.org/abs/2411.09852>
- [23] 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).
- [24] Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Michael He, Yinghai Lu, and Yu Shi. 2024. Actions Speak Louder than Words: Trillion-Parameter Sequential Transducers for Generative Recommendations. arXiv:2402.17152 [cs.LG] <https://arxiv.org/abs/2402.17152>
- [25] Buyun Zhang, Liang Luo, Yuxin Chen, Jade Nie, Xi Liu, Daifeng Guo, Yanli Zhao, Shen Li, Yuchen Hao, Yantao Yao, et al. 2024. Wukong: Towards a scaling law for large-scale recommendation. *arXiv preprint arXiv:2403.02545* (2024).
- [26] Zhaoqi Zhang, Haolei Pei, Jun Guo, Tianyu Wang, Yufei Feng, Hui Sun, Shaowei Liu, and Aixin Sun. 2025. OneTrans: Unified Feature Interaction and Sequence Modeling with One Transformer in Industrial Recommender. arXiv:2510.26104 [cs.IR] <https://arxiv.org/abs/2510.26104>
- [27] 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.
- [28] 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.
- [29] Ji Zhu, Zhifang Fan, Xiaoxie Zhu, Yuchen Jiang, Hangyu Wang, Xintian Han, Haoran Ding, Xinmin Wang, Wenlin Zhao, Zhen Gong, Huizhi Yang, Zheng Chai, Zhe Chen, Yuchao Zheng, Qiwei Chen, Feng Zhang, Xun Zhou, Peng Xu, Xiao Yang, Di Wu, and Zuotao Liu. 2025. RankMixer: Scaling Up Ranking Models in Industrial Recommenders. arXiv:2507.15551 [cs.IR] <https://arxiv.org/abs/2507.15551>
- [30] Pablo Zivic, Hernan Vazquez, and Jorge Sánchez. 2024. Scaling Sequential Recommendation Models with Transformers. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2024)*. ACM, 1567–1577. doi:10.1145/3626772.3657816