

GRank: Towards Target-Aware and Streamlined Industrial Retrieval with a Generate-Rank Framework

Yijia Sun*

sunyijia@kuaishou.com
Kuaishou Technology
Beijing, China

Qiang Luo‡

luoqiang@kuaishou.com
Kuaishou Technology
Beijing, China

Shanshan Huang*

huangshanshan@kuaishou.com
Kuaishou Technology
Beijing, China

Zhiyuan Guan†

guanzhiyuan03@kuaishou.com
Kuaishou Technology
Beijing, China

Kun Gai

gai.kun@qq.com
Unaffiliated
Beijing, China

Ruiming Tang‡

tangruiming@kuaishou.com
Kuaishou Technology
Beijing, China

Guorui Zhou†

zhouguorui@kuaishou.com
Kuaishou Technology
Beijing, China

Abstract

Industrial-scale recommender systems rely on a cascade pipeline in which the *retrieval* stage must return a high-recall candidate set from *billions* of items under tight latency. Existing solutions either (i) suffer from limited expressiveness in capturing fine-grained user-item interactions, as seen in decoupled dual-tower architectures that rely on separate encoders, or generative models that lack precise target-aware matching capabilities, or (ii) build *structured indices* (tree, graph, quantization) whose item-centric topologies struggle to incorporate dynamic user preferences and incur prohibitive construction and maintenance costs.

We present GRANK, a novel *structured-index-free* retrieval paradigm that seamlessly unifies target-aware learning with user-centric retrieval. Our key innovations include: (1) A *target-aware Generator* trained to perform personalized candidate generation via GPU-accelerated MIPS, eliminating semantic drift and maintenance costs of structured indexing; (2) A *lightweight but powerful Ranker* that performs fine-grained, candidate-specific inference on small subsets; (3) An end-to-end multi-task learning framework that ensures semantic consistency between generation and ranking objectives.

*These authors contributed equally to this work.

†This author conducted the research while at Kuaishou Technology. He is now with Shanghai Jiaotong University.

‡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, xxxx, xxxx

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

ACM ISBN 978-1-4503-XXXX-X/2018/06

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

Extensive experiments on two public benchmarks and a billion-item production corpus demonstrate that GRank improves Recall@500 by over 30% and 1.7× the P99 QPS of state-of-the-art tree- and graph-based retrievers.

GRANK has been **fully deployed in production** in our recommendation platform since Q2 2025, serving 400 million active users with 99.95% service availability. Online A/B tests confirm significant improvements in core engagement metrics, with Total App Usage Time increasing by 0.160% in the main app and 0.165% in the Lite version.

CCS Concepts

- Information systems → Retrieval models and ranking; Personalization.

Keywords

Industrial-Scale Retrieval, Generate→Rank Workflow, Target-Aware Cross-Attention

ACM Reference Format:

Yijia Sun, Shanshan Huang, Zhiyuan Guan, Qiang Luo, Ruiming Tang, Kun Gai, and Guorui Zhou. 2025. GRank: Towards Target-Aware and Streamlined Industrial Retrieval with a Generate-Rank Framework. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 9 pages. <https://doi.org/XXXXXXX.XXXXXXX>

1 Introduction

Recommender systems are pivotal components of modern internet services, typically employing a multi-stage cascade architecture (Retrieval → Pre-ranking → Ranking → Re-ranking). Within this pipeline, the **retrieval stage** serves as the initial funnel, responsible for efficiently screening a massive item corpus (ranging from millions to billions of items) to identify a candidate set of potentially relevant items for a user. The quality of this retrieved candidate set fundamentally determines the performance ceiling of subsequent stages.

However, practical industrial deployments face significant challenges in the retrieval stage:

Limitation 1: Target-Agnostic Retrieval. Mainstream retrieval methods, including dual-tower architectures[2, 5, 10, 15] and generative sequential models[14, 18, 20] , encode users into fixed representations that are reused across all candidates. While enabling efficient retrieval, this design precludes fine-grained, candidate-specific interactions, fundamentally limiting their ability to discern subtle user intents from noisy behavioral histories.

Limitation 2: Structured Indices with Constraints. Despite addressing the expressiveness gap through structured indices (trees, graphs, quantization), target-aware retrieval methods introduce fundamental methodological limitations and significant engineering challenges.

At the methodological level, structured indices exhibit two primary constraints: First, their inherent *item-centric* design paradigm excludes personalized user signals during candidate expansion, creating potential misalignment between retrieval paths and actual user interests. Second, technical implementations present specific limitations—tree indices suffer from cascading errors due to offline hierarchical partitioning, graph indices are constrained by static similarity metrics that poorly adapt to complex embedding spaces, and quantization techniques experience objective misalignment from their Euclidean geometric assumptions.

The engineering challenges are equally consequential: structural imbalance across all index types leads to unpredictable P99 latency, requiring substantial over-provisioning of computational resources to maintain service availability. Furthermore, the complex and time-consuming processes for index construction and updates create significant bottlenecks for rapid iteration in production environments where user distributions evolve continuously.

Our Solution: We propose **GRank**, a novel *structured-index-free* retrieval framework that introduces a two-stage **Generate**→**Rank** paradigm. GRank elegantly circumvents the aforementioned limitations by decoupling *approximate pruning* from *exact reranking* within an end-to-end differentiable architecture, completely eliminating the need for structured index maintenance.

- **Stage-1: Generator:** A generator empowered by target-aware learning during training, which produces personalized candidate subsets via efficient MIPS sweeping over the full corpus.
- **Stage-2: Ranker:** A lightweight cross-attention ranker that performs fine-grained, candidate-specific scoring on the small candidate subset.

This cohesive workflow achieves both high efficiency and precision while avoiding the challenges of structured indices.

Key Contributions:

- **A Universal Two-Stage Architecture** that seamlessly unifies candidate generation and precise ranking in an end-to-end trainable framework, applicable to diverse retrieval systems.
- **Target-Aware Modeling without Indexing Overhead** through a novel training paradigm that infuses target-aware signals into the generator, boosting representation quality without compromising inference efficiency.
- **Large-Scale Empirical Validation** demonstrating over 30% improvement in Recall@500 and 1.7× higher QPS across multiple datasets.

GRank has been fully deployed in production since Q2 2025, processing 50 billion daily requests with 99.95% availability. Online A/B tests confirm significant engagement improvements: **Total App Usage Time** increased by 0.160% in the main app and 0.165% in the Lite version.

2 Related Work

2.1 Efficient yet Target-Agnostic Retrieval Paradigms

Dual-Tower Architectures: Early retrieval systems predominantly employed *dual-tower architectures* (e.g., DSSM [10], YouTube Recommendations [5]), which independently encode users and items into embeddings $\mathbf{u} \in \mathbb{R}^d$ and $\mathbf{v} \in \mathbb{R}^d$, then measure relevance via a simple inner product:

$$\text{score}(\mathbf{u}, \mathbf{v}) = \mathbf{u}^\top \mathbf{v}, \quad (1)$$

enabling Maximum Inner-Product Search (MIPS) on GPUs [13] or specialized ANN libraries [9] with logarithmic query time. Despite sub-millisecond latency, Eq. (1) is *first-order* and *target-agnostic*: the user representation \mathbf{u} is frozen for all candidates, precluding any candidate-specific refinement. Moreover, the model is oblivious to higher-order crosses between user and item features, limiting expressiveness and often yielding a noisy candidate pool [6].

Generative Retrieval: Recent advancements have reformulated retrieval as a *generative modeling* problem, leveraging sequence Transformers [21] to capture temporal user dynamics $\partial \mathbf{u} / \partial t$. This paradigm shift began with models like SASRec [14], which employed unidirectional self-attention to model user sequences for next-item prediction. BERT4Rec [20] extended this by introducing bidirectional context encoding, while subsequent works like GPTRec [18] fully embraced an autoregressive framework, generating sequences of future item IDs. Further innovations, such as Kuaiformer [16], introduced techniques like multi-interest query tokens and an adaptive long-sequence compression to enhance representation learning.

Despite their architectural differences and progressive sophistication, these generative approaches share a fundamental *target awareness deficiency*:

- Scoring still reduces to $\mathbf{u}^\top \mathbf{v}$ where \mathbf{u} is a *fixed* summary of the user’s history, computed independently of candidate \mathbf{v} .
- Lack explicit interaction mechanisms $g(\mathbf{u}, \mathbf{v})$ enable candidate-aware history focusing.
- Ineffective noise suppression as attention weights α_{ij} are computed *a priori* without candidate context.

Formally, both dual-tower and generative paradigms compute relevance as $\text{score}(\mathbf{u}, \mathbf{v}) = \phi(\text{Enc}_{\text{user}}(S), \mathbf{v})$, where Enc_{user} produces a candidate-invariant representation. This *target-agnostic* design fundamentally limits precision when discerning subtle intent shifts from noisy behavioral signals [22].

2.2 Target-Aware Retrieval via Structured Indexing

To address the target awareness problem, *index-based retrieval* methods incorporating explicit interaction modeling have emerged. Key approaches include Tree-Based Indexing(such as TDM [24],

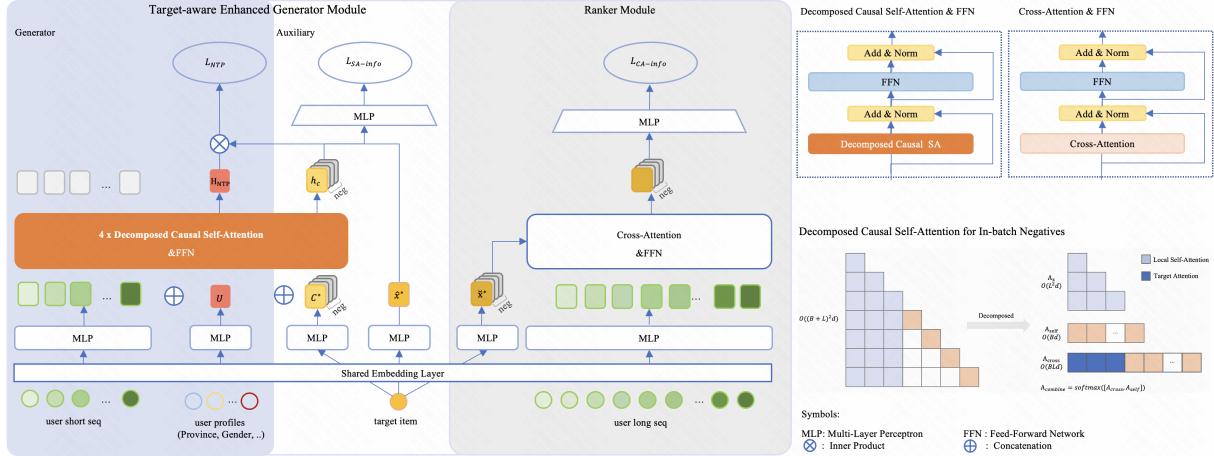


Figure 1: GRank Framework Architecture. The framework comprises two tightly-coupled modules optimized through three complementary losses: (1) Target-Aware Enhanced Generator (optimized by \mathcal{L}_{NTP}) for short-term preference modeling and Auxiliary (optimized by $\mathcal{L}_{\text{SA-Info}}$) for target injection; (2) Ranker (optimized by $\mathcal{L}_{\text{CA-Info}}$) employing efficient cross-attention for fast inference. The Auxiliary module injects target embedding C^* and in-batch negatives C_j during training only. The Generator utilizes our novel decomposed causal self-attention (detailed in bottom-right) to efficiently process user sequences alongside candidate items while maintaining mathematical equivalence. All components are jointly optimized through multi-task learning.

JTM[23], BSAT[26], MISS[8]), Graph-Based Indexing(NANN [3], DR[7]), and Quantization-Based Indexing(StreamingVQ [1]).

Tree-Based Indexing. Tree-based Deep Models propose tree structures to hierarchically search candidates. TDM [24] pioneers tree-based retrieval, reducing complexity to $O(\log N)$. However, its tree hierarchy \mathcal{T} is *decoupled* from model objective $\mathcal{L}_{\text{model}}$, causing *semantic drift* as partitions \mathcal{P}_k misrepresent the semantic space.

JTM [23] alleviates the gap by co-training \mathcal{T} and θ with heuristic clustering. However, the clustering itself is a limitation: user interest \mathcal{D}_u can hardly perfectly aligns with item clusters C_i , and the $O(N \log N)$ index rebuild is prohibitive for billion-scale item corpora C [25].

Graph-Based Indexing. NANN [3] resorts to HNSW [17] for candidate searching with model-generated edge weights. However, the construction of \mathcal{G} often relies on pre-defined metrics (e.g. cosine similarity), which impose strong geometric assumptions on the item embedding space. This deviation is exacerbated by non-linear encoders, causing retrieval paths to diverge from ranking objectives [12].

Quantization-Based Indexing. StreamingVQ [1] employs quantization clustering to compress high-dimensional embeddings into short codes using a codebook $\mathcal{B} = \{c_1, \dots, c_K\}$. This enables sub-linear retrieval via table lookups. However, quantization indices typically assume Euclidean space ($\text{sim}(v, c_k) = -\|v - c_k\|_2^2$). Complex encoders (e.g., those using attention) break this assumption, exacerbating the mismatch between the quantization objective \mathcal{L}_{VQ} and $\mathcal{L}_{\text{rank}}$, further reducing retrieval accuracy [11]. Additionally, codebook maintenance $\mathcal{B}(t)$ and joint model-codebook training incur significant engineering overhead.

Fundamental Limitation: Item-Centric expansion. A core limitation shared by all structured indexing methods is their inherent *item-centric candidate expansion*. The expansion logic follows a pre-defined path based on item proximity, inherently excluding *personalized user preference* signals p_u . This results in a *semantic deviation* during retrieval—the system’s path diverges from the user’s true intent [19]. For example, a user’s dynamic preference cannot re-route the query, causing relevant candidates to be pruned early and never reached [4].

The discussed limitations of both target-agnostic models and target-aware but item-centric indexing methods motivate our work GRANK, which unifies target-aware precision with efficient retrieval through an index-free workflow: (1) target-aware training cultivates a high-quality generator for personalized MIPS retrieval; (2) a light-weight ranker performs fine-grained, candidate-specific scoring on small candidate subsets. This avoids semantic drift and maintenance costs while delivering precise personalization.

3 Methodology

This section details the GRank framework, structured as follows. We begin by formalizing the retrieval task in §3.1. We then present our offline training methodology in §3.2, which jointly optimizes the Generator and Ranker modules in an end-to-end manner. Finally, §3.3 describes the efficient two-stage online inference pipeline. An overview of the complete training and serving architecture is illustrated in Figure 1 and Figure 2, respectively.

3.1 Problem Formulation

Let $\mathcal{S}_u = \{(i_1, f_1), \dots, (i_n, f_n)\}$ be a user’s chronological interaction sequence with items $i_t \in \mathcal{I}$ and features $f_t \in \mathbb{R}^d$ (watch duration, engagement type, author embedding). Given a target item i^* , the

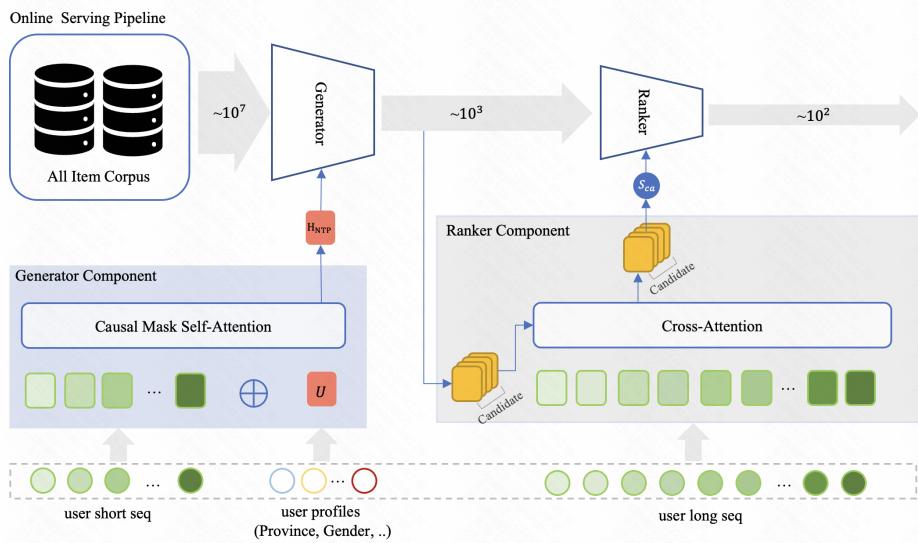


Figure 2: GRank Serving Architecture. Stage-1 (Generator): Leverages user latent interests to generate a personalized candidate subset. Stage-2 (Ranker): Employs a target-aware cross-attention scorer for fine-grained relevance estimation.

retrieval task is to estimating the relevance score $\mathcal{E}(i^* | \mathcal{S}_u)$, which defines the engagement probability:

$$p(i^* | \mathcal{S}_u) \propto \exp(\mathcal{E}(i^* | \mathcal{S}_u)). \quad (2)$$

The objective is to efficiently retrieve the top- K items from corpus \mathcal{I} that maximize this probability.

3.2 Offline Training

We first map user-item interactions into unified dense representations (§3.2.1). Built upon these embeddings, the GRank framework introduces two tightly-coupled modules (Figure 1): (1) **Target-Aware Enhanced Generator**, integrating **Generator** for short-term preference modeling and **Auxiliary** for target injection to enhance both Generator and Ranker; (2) **Ranker** for long-term preference modeling.

3.2.1 Input Representation. This section outlines the foundational input processing steps shared by GRank’s core components.

Shared Item Embedding. We use a shared embedding matrix $E \in \mathbb{R}^{|\mathcal{I}| \times d}$ to map item IDs to dense vectors $e_t = E(i_t)$.

Personalized Query Token. We construct a personalized query vector by concatenating user demographics with aggregated behavior sequences:

$$U = [u; \text{Sum}(\text{Seq}_{rs}); \text{Sum}(\text{Seq}_{click}); \text{Sum}(\text{Seq}_{long_view})],$$

where $u \in \mathbb{R}^d$ encodes user attributes, and Seq_* are item sparse embedding of corresponding behavior sequences.

3.2.2 Target-Aware Enhanced Generator. The Target-Aware Enhanced Generator produces high-quality user representations for retrieval by integrating short-term preference modeling with target-aware training through a dual-component architecture:

- **Generator:** Models short-term preferences via causal Transformer decoder

- **Auxiliary:** Injects target-awareness and provides sequence supervision during training, omitted during inference

Input Encoding & Model Structure. Each historical interaction (i_t, f_t) is encoded as $x_t = \text{MLP}([e_t; f_t]) \in \mathbb{R}^d$. The generator takes the truncated sequence of recent interactions $x_{T-L+1:T}$ and the personalized query token U as primary inputs.

The target item i^* and in-batch negative samples(\mathcal{B}) are processed through an auxiliary MLP:

$$C^* = \text{MLP}_{\text{aux}}(e^*), \quad C_j = \text{MLP}_{\text{aux}}(e_j) \quad \forall j \in \mathcal{B} \quad (3)$$

The complete candidate set $C = [C^*, C_1, \dots, C_{|\mathcal{B}|}]$ is appended to form:

$$H^{(0)} = [x_{T-L+1:T}; U; C]. \quad (4)$$

This design enables the model to leverage target-specific signals while facilitating InfoNCE loss computation through batch-level interactions. The subsequent section (§3.2.3) will detail our optimized attention mechanism that efficiently processes this extended sequence while maintaining mathematically equivalent.

We model the sequence using a N -layer pre-norm causal Transformer decoder:

$$H^{(N)} = \text{Transformer}(H^{(0)}, M), \quad (5)$$

where mask M enforces causality within user sequences while allowing full candidate-sequence interaction.

The user representation is extracted from the personalized query token:

$$h_u = H_U^{(N)}. \quad (6)$$

Training Objectives. The generator is optimized through two complementary losses:

- **Generator Loss (\mathcal{L}_{NTP}):** The user representation h_u is L2-normalized and trained with InfoNCE loss:

$$\hat{x}^* = \text{MLP}_{\text{ntp}}(e^*), \quad \hat{x}_j = \text{MLP}_{\text{ntp}}(e_j) \quad \text{for } j \in \mathcal{B}. \quad (7)$$

Table 1: Performance Comparison of SOTA Methods on Different Datasets. The best and second-best results highlighted in bold font and underlined. Note: TDM was only evaluated on the Industry Dataset due to its deployment complexity, using our existing infrastructure.

Method	User Behavior		MovieLens		Industry Datasets	
	Recall@50	NDCG@50	Recall@50	NDCG@50	Recall@500	NDCG@500
DSSM	0.2711	0.1911	0.1940	0.0792	0.1068	0.0360
Kuaiformer	<u>0.3270</u>	<u>0.2347</u>	0.2538	0.0906	0.1151	0.0386
TDM	-	-	-	-	<u>0.1766</u>	0.0535
NANN	0.3264	0.1765	<u>0.2633</u>	<u>0.1017</u>	0.1326	0.0466
Streaming VQ	0.3220	0.2262	0.2554	0.0907	0.1296	<u>0.0603</u>
GRank	0.4348	0.2780	0.3350	0.1250	0.2346	0.0775
Relative Gain	+33.0%	+18.4%	+27.2%	+22.9%	+32.8%	+28.5%

The loss is computed as:

$$\mathbf{H}_{\text{NTP}} = \mathbf{h}_u / \|\mathbf{h}_u\|_2, \quad \mathcal{L}_{\text{NTP}} = -\log \frac{\exp(\langle \mathbf{H}_{\text{NTP}}, \hat{\mathbf{x}}^* \rangle / \tau)}{\sum_{j \in \mathcal{B}} \exp(\langle \mathbf{H}_{\text{NTP}}, \hat{\mathbf{x}}_j \rangle / \tau)}, \quad (8)$$

where τ is a softmax temperature.

- **Auxiliary Score Loss ($\mathcal{L}_{\text{SA-info}}$):** The candidates corresponding Transformer outputs are scored:

$$s_{\text{sa}} = \text{MLP}_{\text{sa}}(\mathbf{h}_{C^*}), \quad s_{\text{sa}_j} = \text{MLP}_{\text{sa}}(\mathbf{h}_{C_j}) \quad \text{for } j \in \mathcal{B}. \quad (9)$$

The loss is computed as:

$$\mathcal{L}_{\text{SA-info}} = -\log \frac{\exp(s_{\text{sa}})}{\sum_{j \in \mathcal{B}} \exp(s_{\text{sa}_j})}. \quad (10)$$

Training-Serving Consistency. The causal mask ensures that while historical tokens can attend to the target during training, C^* is masked out during inference, preventing information leakage and maintaining strict training-serving consistency.

3.2.3 Optimized Training via Decomposed Causal Self-Attention. While the generator architecture provides target-aware modeling capabilities, the conventional approach of concatenating user sequences with candidate items for self-attention results in $O((L + B)^2 d)$ complexity, creating a computational bottleneck during training.

We address this through a mathematically equivalent decomposition that maintains causal integrity while enabling efficient parallel computation.

Traditional Approach The baseline computes full self-attention on the concatenated sequence of user history and candidate items:

$$\mathbf{S} = [\mathbf{X}; \mathbf{C}], \quad \mathbf{A} = \text{softmax} \left(\frac{\mathbf{S} \mathbf{W}_Q \mathbf{W}_K^\top \mathbf{S}^\top}{\sqrt{d}} \odot \mathbf{M} \right). \quad (11)$$

Optimized Decomposition Our method decouples attention computation into three coordinated components:

1. User Sequence Self-Attention:

$$\mathbf{Q}_X = \mathbf{X} \mathbf{W}_Q, \quad \mathbf{K}_X = \mathbf{X} \mathbf{W}_K, \quad \mathbf{V}_X = \mathbf{X} \mathbf{W}_V \quad (12)$$

$$\mathbf{A}_X = \text{softmax} \left(\frac{\mathbf{Q}_X \mathbf{K}_X^\top}{\sqrt{d}} \odot \mathbf{M}_X \right), \quad \mathbf{O}_X = \mathbf{A}_X \mathbf{V}_X \quad (13)$$

2. Candidate Attention Decomposition:

$$\mathbf{Q}_C = \mathbf{C} \mathbf{W}_Q, \quad \mathbf{K}_C = \mathbf{C} \mathbf{W}_K, \quad \mathbf{V}_C = \mathbf{C} \mathbf{W}_V \quad (14)$$

Cross-Scores (Candidates → User Sequence):

$$\mathbf{A}_{\text{cross}} = \frac{\mathbf{Q}_C \mathbf{K}_X^\top}{\sqrt{d}} \in \mathbb{R}^{B \times L} \quad (15)$$

Candidate Self-Correlation Scores:

$$\mathbf{A}_{\text{self}} = \frac{\sum_{i=1}^d \mathbf{Q}_{C,:,i} \odot \mathbf{K}_{C,:,i}}{\sqrt{d}} \in \mathbb{R}^{B \times 1} \quad (16)$$

3. Attention Combination and Output:

$$\mathbf{A}_{\text{combined}} = \text{softmax}([\mathbf{A}_{\text{cross}}, \mathbf{A}_{\text{self}}]), \quad \mathbf{V}_{\text{combined}} = [\mathbf{V}_X; \mathbf{V}_C] \quad (17)$$

$$\mathbf{O}_C = \mathbf{A}_{\text{combined}} \mathbf{V}_{\text{combined}}, \quad \mathbf{O}_{\text{final}} = [\mathbf{O}_X; \mathbf{O}_C] \quad (18)$$

Complexity Analysis This decomposition reduces complexity from $O((L + B)^2 d)$ to $O(L^2 d + BLd + Bd)$, achieving 82% FLOPs reduction under typical configurations ($B = 300, L = 64, d = 128$) while maintaining mathematical equivalence with the traditional approach.

3.2.4 Ranker Module. To capture long-term user preference, the Ranker processes an extended history of the latest 1,000 items, $\mathcal{S}_u^{\text{long}}$.

Sequence and Candidate Encoding. We first transform both the historical sequence and candidate item through separate MLPs to obtain enriched representations:

$$\mathbf{H}_u^{\text{long}} = \text{MLP}_{\text{long}}(\mathbf{E}[i_{T-999}, i_{T-998}, \dots, i_T]) \in \mathbb{R}^{1000 \times d}, \quad (19)$$

$$\check{\mathbf{x}}^* = \text{MLP}_{\text{query}}(\mathbf{x}^*) \in \mathbb{R}^d. \quad (20)$$

These non-linear transformations capture higher-order feature interactions.

Cross-Attention Scoring. We then apply a single-head cross-attention mechanism where the candidate serves as query and the historical sequence provides keys and values:

$$\mathbf{z} = \text{Attention}(\mathbf{Q} = \check{\mathbf{x}}^* \mathbf{W}_Q, \mathbf{K} = \mathbf{H}_u^{\text{long}} \mathbf{W}_K, \mathbf{V} = \mathbf{H}_u^{\text{long}} \mathbf{W}_V) \quad (21)$$

$$= \text{Softmax} \left(\frac{\mathbf{Q} \mathbf{K}^\top}{\sqrt{d_k}} \right) \mathbf{V}. \quad (22)$$

This allows the model to attend to historically relevant items when scoring the candidate.

Output and Optimization. The context vector \mathbf{z} is projected to a final relevance score:

$$s_{\text{ca}} = \text{MLP}_{\text{ca}}(\mathbf{z}) \in \mathbb{R}, \quad (23)$$

which is optimized using an InfoNCE loss with in-batch negatives:

$$\mathcal{L}_{\text{CA-info}} = -\log \frac{\exp(s_{\text{ca}})}{\sum_{j \in \mathcal{B}} \exp(s_{\text{ca}_j})}. \quad (24)$$

3.2.5 Multi-Task Loss. The three components—Generator, Auxiliary, and Ranker—are trained jointly within a multi-task learning framework. The overall training objective is

$$\mathcal{L}_{\text{total}} = \lambda_0 \mathcal{L}_{\text{NTP}} + \lambda_1 \mathcal{L}_{\text{SA-info}} + \lambda_2 \mathcal{L}_{\text{CA-info}}, \quad (25)$$

where each \mathcal{L}_* denotes InfoNCE, and the weights $\{\lambda_i\}$ are automatically tuned by Bayesian optimization on the validation set.

3.3 Online Inference

Online inference adopts a two-stage latency-sensitive cascade as shown in Figure 2.

Stage-1: NTP Generator. The vector \mathbf{H}_{NTP} is used as a query for Maximum Inner Product Search (MIPS) over a quantized embedding index, retrieving a top- $k_1 = 2,000$ candidate set C_1 in $\mathcal{O}(\log |\mathcal{I}|)$ time.

Stage-2: Ranker. For each $i \in C_1$, we compute the cross-attention score s_{CA} and rerank the candidates to obtain the final top- $k_2 = 500$ items. This computation is purely feedforward and highly parallelizable, enabling high-throughput, low-latency inference when batched on GPU—making it well-suited for real-time serving (see §4.3 for detailed benchmarks).

4 Experiments

We conducted rigorous experiments to answer five research questions (RQs).

- **RQ1:** How effective is GRank compared with state-of-the-art models in the industry?
- **RQ2:** Does the two-stage architecture meet industrial latency constraints?
- **RQ3:** What is the contribution of each core architectural component to the overall performance?
- **RQ4:** How do hyper-parameter selections enable the optimal trade-off between efficiency and performance in industrial deployment?
- **RQ5:** Does GRank bring significant online gains in our short-video service?

4.1 Experimental Setup

4.1.1 Datasets and Evaluation Protocol. We evaluate GRank on one industrial and two public datasets (Table 2): **MovieLens-20M** (movie recommendations), **Taobao UserBehavior** (e-commerce), and **Kuaishou Industrial** (short-video platform). All datasets are split chronologically: Kuaishou uses 6 days for training and the next day for testing, while the public datasets follow standard 90%/10% split.

4.1.2 Evaluation Metrics and Baselines. We report **Recall@K** and **NDCG@K** (retrieval from full corpus), and **QPS** (maximum throughput under P99 latency ≤ 100 ms).

Baselines represent key retrieval paradigms: **DSSM** (dual-tower), **Kuaiformer** (generative), **TDM** (tree-based), **NANN** (graph-based),

Table 2: Dataset statistics after preprocessing

	UserBehavior	MovieLens-20M	Kuaishou
Users	964K	138K	108M
Items	4.2M	27K	42M
Interactions	1.7M	9.3M	4B
Avg. Seq. Len.	101	144	980

and **StreamingVQ** (quantization). All use identical training data, embeddings ($d = 128$), and hardware configurations.

4.1.3 Implementation Details. GRank uses embedding dimension $d = 128$, 4-layer causal self-attention, 1-layer cross-attention, sequence lengths of 64 (short-term) and 1000 (long-term), candidate set size 2000.

4.2 Offline Evaluation (RQ1)

As summarized in Table 1, GRank achieves state-of-the-art performance across all datasets, demonstrating the effectiveness of its unified, index-free paradigm.

Our results demonstrate two key findings: First, GRank significantly outperforms target-agnostic models including generative (Kuaiformer) and dual-tower (DSSM) approaches, improving Recall@50 by 33.0% on the User Behavior dataset. All target-aware methods (TDM, NANN, StreamingVQ) consistently outperform those baselines on industrial data, confirming the necessity of target-aware modeling.

Second, GRank achieves superior performance even compared to sophisticated target-aware systems that rely on structured indices, with 32.8% higher Recall@500 than tree-based TDM on the Industry Dataset. The fact that GRank attains this superior accuracy without any structured index highlights the sufficiency of a well-trained generator combined with a lightweight ranker.

These consistent gains across diverse datasets confirm GRank’s effectiveness in bridging the accuracy-efficiency gap in large-scale retrieval.

4.3 Online Service Performance (RQ2)

This section evaluates whether GRank’s two-stage architecture meets industrial latency constraints (RQ2). All experiments used production servers with identical GPU configurations, measuring QPS under the P99 latency constraint of ≤ 100 ms. Reported latencies include the full pipeline from feature extraction to final scoring.

As shown in Table 3, GRank achieves a QPS of 767, significantly outperforming NANN and TDM while maintaining superior recall. The suboptimal throughput of NANN and TDM can be attributed to path uncertainty, candidate set size fluctuations, and multi-round retrieval overhead. In contrast, GRank’s efficiency stems from its two-stage design: the generator rapidly reduces candidate sets via efficient MIPS, avoiding complex tree/graph traversals, while the lightweight ranker applies precise target-aware scoring with minimal overhead. This approach avoids the latency variability of structured indices while delivering higher accuracy than target-agnostic models.

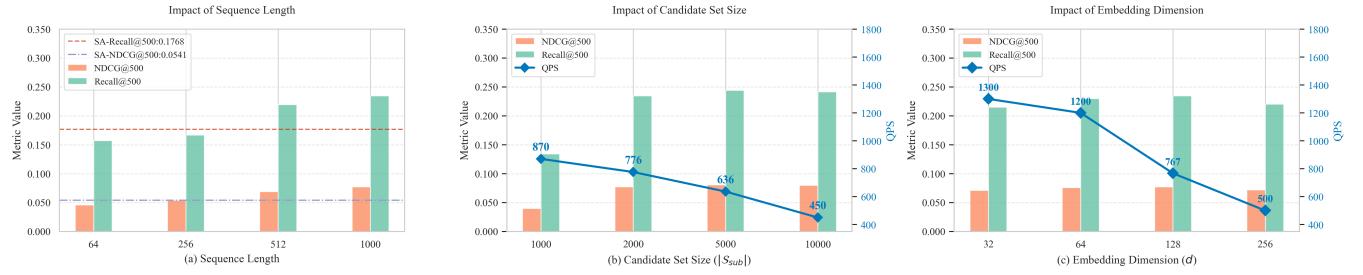


Figure 3: Hyper-parameter analysis on the efficiency-performance trade-off. (a) Sequence length: CA Recall@500 increases monotonically with L , peaking when $L=1000$, significantly outperforming the SA module while maintaining lower latency. (b) Candidate size: diminishing returns begin at $k_1=2000$; recall improves by only 0.1 pp at $k_1=5000$ while QPS drops 18%, making 2000 the pragmatic optimum. (c) Top embedding dimension: recall saturates at $d_{top}=128$ and falls at 256 owing to memory pressure; $d_{top}=64$ offers a high-ROI fallback, trading 3% recall for 56% QPS gain.

Table 3: End-to-end latency, throughput and recall comparison. SLA threshold is 100ms. All latency values include full retrieval service pipeline (feature processing + core retrieval logic).

Method	Recall @500	NDCG @500	QPS	P50 Lat. (ms)	P99 Lat. (ms)
DSSM	0.1068	0.0360	1300	50	61
Kuaiformer	0.1151	0.0386	772	59	87
TDM	0.1766	0.0535	273	55	96
NANN	0.1326	0.0466	384	70	100
StreamVQ	0.1296	0.0603	450	48	75
GRank	0.2346	0.0775	767	48	73

4.4 Ablation Study on Architectural Components (RQ3)

Our ablation study addresses three pivotal questions through corresponding model variants:

- **Pure Generator:** How does the generator perform alone as a retrieval baseline? — Uses only the first-stage generator without ranker.
- **Gen+SA:** Does self-attention provide sufficient performance gains? — Generator with 4-layer self-attention only.
- **Gen+CA:** Can cross-attention offer a better efficiency-accuracy trade-off? — Generator with cross-attention only.

The full GRank model then answers: can a hybrid architecture synergize both to achieve the optimal balance?

Table 4: Ablation Results of Hybrid Attention Architecture

Config	Generator Recall@2000	Recall @500	NDCG @500	QPS
Pure Generator	0.1512	0.1190	0.0405	1333
Gen+SA Rank	0.3427	0.1768	0.0541	340
Gen+CA Rank	0.1820	0.1363	0.0502	767
Full GRank	0.3685	0.2346	0.0775	767

The ablation results conclusively demonstrate that our two-stage design with hybrid attention mechanisms achieves the best trade-off between retrieval quality and computational efficiency.

- **Universal Offline-Enhancement Paradigm for Generator.**

To further elucidate the performance gains of GRank, we dissect the behaviour of the Generator itself. Since the 2000 candidates produced in the first stage form the *absolute* ceiling for any subsequent ranker, the jump of Generator Recall@2000 from 0.1512 to 0.3685 *significantly contributes* to the final Recall@500 of 0.2346. This observation reveals that injecting a target-aware ranking loss during training substantially strengthens the user representation of a dual-tower model, while leaving the tower architecture *completely untouched* at inference—introducing **zero** additional latency. Consequently, the “**offline optimise, transparent deploy**” paradigm acts as a drop-in, model-agnostic performance booster that is instantly applicable to *any* representation-based or dual-tower retrieval system, delivering higher accuracy with no extra parameters, no extra engineering, and no extra cost at serving time.

- **Second-Stage Rescoring is Critical for Accuracy Gains.** The removal of rescoring drastically impairs performance (Recall@500: -50.7%), highlighting its necessity. All rescoring variants deliver substantial metric gains over the pure generator, confirming the value of second-stage refinement, though at the cost of reduced QPS. The full GRank model achieves the most favorable accuracy-efficiency trade-off.

- **CA-SA Synergy Outperforms Isolated Pathways.** Our ablation shows that combining CA and SA outperforms either approach alone, with each mechanism offering distinct advantages:

- **Generator with Self-Attention (Gen+SA):** This configuration achieves a Recall@500 of 0.1768, representing a significant improvement over the pure generator baseline. However, the quadratic computational complexity ($O(4 * n^2)$) of 4 layers self-attention leads to substantial latency overhead.

- **Generator with Cross-Attention (Gen+CA):** This approach also demonstrates notable improvement in Recall@500 (0.1363) over the baseline. More importantly, CA’s linear complexity ($O(n)$) enables efficient processing, particularly advantageous

Table 5: Online A/B Test Results of GRank Across Different Scenarios

Applications	Overall App Metrics				Recall Pathway Metrics		
	Total App Usage Time	Usage Time per User	Video Watch Time	Effective Interests	Show Ratio	Avg. Watch Time	UV Coverage
Kuaishou Single Page	+0.160%	+0.139%	+0.347%	+0.527%	+21.92%	+16.26%	+16.50%
Kuaishou Lite Single Page	+0.165%	+0.118%	+0.233%	+0.384%	+20.31%	+11.69%	+12.29%

for long-sequence modeling tasks. Nevertheless, its accuracy remains inferior to the SA-enhanced variant.

- **Hybrid CA-SA synergy optimizes trade-offs:** The complete GRank framework leverages both attention mechanisms: CA for efficient global modeling and SA for capturing fine-grained sequential dependencies during training and influence the update dynamics of shared sparse representations. Crucially, SA is excluded from inference, preserving the latency benefits of CA while maintaining superior accuracy.

4.5 Balancing Efficiency and Performance through Hyper-Parameter Selection (RQ4)

We systematically examine how hyper-parameters govern the trade-off between efficiency and performance in industrial deployment. Rather than merely reporting sensitivity, we identify optimal configurations that achieve the best practical balance for our production environment.

4.5.1 Sequence Length: Long-Range Modeling with Linear Complexity. The analysis focuses on the CA module’s sequence length. To maximize efficiency, scoring is performed exclusively by the CA module during inference. As shown in Figure 3.a, the performance improves consistently with increasing L : when $L = 1,000$, Recall@500 reaches 0.2345, significantly surpassing the SA module’s performance(0.1768) while maintaining substantially lower latency. This demonstrates that increasing sequence length effectively compensates for excluding the SA module at inference, achieving an optimal balance where enhanced effectiveness meets practical efficiency constraints.

4.5.2 Candidate Size: Diminishing Returns in Recall-Throughput Trade-off. The generator candidate size k_1 presents a critical trade-off between recall and throughput, as illustrated in Figure 3.b. While larger candidate sets improve recall, the marginal gain diminishes rapidly beyond $k_1 = 2\,000$ (only 4.27% improvement at $k_1 = 5\,000$), while QPS drops by 18%. This makes $k_1 = 2\,000$ the pragmatically optimal choice for our industrial setting.

4.5.3 Embedding Dimension: Performance Saturation under Memory Constraints. We evaluate the selection of the top embedding dimension d_{top} under production memory constraints. As shown in Figure 3.c, recall increases monotonically with increasing d_{top} and saturates at 128, but drops significantly at 256 due to memory constraints that limit candidate pool size. Meanwhile, $d_{top} = 64$ maintains comparable performance while significantly improving system throughput. Therefore, $d_{top} = 128$ is optimal for accuracy-critical scenarios, while $d_{top} = 64$ provides an ideal alternative for throughput-sensitive deployments requiring efficiency trade-offs.

4.6 Online Results (RQ5)

We conducted a one-week A/B test on Kuaishou’s single-column recommendation platforms by replacing the original KUAIFORMER recall with GRANK while maintaining identical experimental conditions. The results in Table 5 demonstrate GRANK’s effectiveness across both business and technical dimensions.

Overall App Performance. GRANK delivers consistent improvements in core user engagement metrics, with particular strength in video consumption and preference modeling. The framework effectively translates retrieval enhancements into measurable business value, with both main and lite versions showing positive trends across all key indicators.

Recall Quality. GRANK improves recall effectiveness, with higher show ratio, UV coverage, and average watch time per session, reflecting improved content relevance and increased user reach. These consistent improvements demonstrate GRANK’s ability to enhance the entire recommendation ecosystem through superior retrieval.

The correlated improvements across both business metrics and technical indicators establish that GRANK’s architectural innovations directly translate to enhanced user experience and platform value.

5 Conclusion

We propose **GRank**, a target-aware retrieval framework whose primary contribution is a *universal* two-stage **Generate**→**Rank** architecture. This paradigm is readily applicable to diverse retrieval systems (e.g., dual-tower, generative), achieving significant improvements in representation quality and recall accuracy through co-training of target-aware generators and lightweight rankers, with only modest computational overhead. For structured-index-based systems (e.g., tree- or graph-based), GRank offers a software-level alternative that eliminates index maintenance while achieving superior accuracy and greatly reducing serving complexity and cost.

Extensive experiments demonstrate over 30% improvement in Recall, validating the framework’s effectiveness. GRank has been deployed in production since Q2 2025, now serving 400 million users and handling 50 billion daily requests, confirming its scalability and real-world impact.

References

- [1] Xingyan Bin, Jianfei Cui, Wujie Yan, Zhichen Zhao, Xintian Han, Chongyang Yan, Feng Zhang, Xun Zhou, Qi Wu, and Zuoqiao Liu. 2025. Real-time Indexing for Large-scale Recommendation by Streaming Vector Quantization Retriever. *arXiv preprint arXiv:2501.08695* (2025).
- [2] Yukuo Cen, Jianwei Zhang, Xu Zou, Chang Zhou, Hongxia Yang, and Jie Tang. 2020. Controllable Multi-Interest Framework for Recommendation. *arXiv:2005.09347 [cs.IR]* <https://arxiv.org/abs/2005.09347>
- [3] Rihan Chen, Bin Liu, Han Zhu, Yaoxuan Wang, Qi Li, Buting Ma, Qingbo Hua, Jun Jiang, Yunlong Xu, Hongbo Deng, and Bo Zheng. 2022. Approximate Nearest

- Neighbor Search under Neural Similarity Metric for Large-Scale Recommendation. arXiv:2202.10226 [cs.IR] <https://arxiv.org/abs/2202.10226>
- [4] Chih-Yi Chiu, Amorntip Prayoonwong, and Yin-Chih Liao. 2019. Learning to index for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence* 42, 8 (2019), 1942–1956.
- [5] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. 191–198.
- [6] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. 191–198.
- [7] Weihao Gao, Xiangjun Fan, Chong Wang, Jiankai Sun, Kai Jia, Wenzhi Xiao, Ruofan Ding, Xingyan Bin, Hui Yang, and Xiaobing Liu. 2021. Deep Retrieval: Learning A Retrievable Structure for Large-Scale Recommendations. arXiv:2007.07203 [cs.IR] <https://arxiv.org/abs/2007.07203>
- [8] Chengcheng Guo, Junda She, Kuo Cai, Shiyao Wang, Qigen Hu, Qiang Luo, Kun Gai, and Guorui Zhou. 2025. MISS: Multi-Modal Tree Indexing and Searching with Lifelong Sequential Behavior for Retrieval Recommendation. arXiv:2508.14515 [cs.IR] <https://arxiv.org/abs/2508.14515>
- [9] Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. 2020. Accelerating Large-Scale Inference with Anisotropic Vector Quantization. arXiv:1908.10396 [cs.LG] <https://arxiv.org/abs/1908.10396>
- [10] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. 2333–2338.
- [11] Herve Jegou, Matthijs Douze, and Cordelia Schmid. 2010. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence* 33, 1 (2010), 117–128.
- [12] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data* 7, 3 (2019), 535–547.
- [13] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with GPUs. arXiv:1702.08734 [cs.CV] <https://arxiv.org/abs/1702.08734>
- [14] Wang-Cheng Kang and Julian McAuley. 2018. Self-Attentive Sequential Recommendation. arXiv:1808.09781 [cs.IR] <https://arxiv.org/abs/1808.09781>
- [15] Chao Li, Zhiyuan Liu, Mengmeng Wu, Yuchi Xu, Pipei Huang, Huan Zhao, Guoliang Kang, Qiwei Chen, Wei Li, and Dik Lun Lee. 2019. Multi-Interest Network with Dynamic Routing for Recommendation at Tmall. arXiv:1904.08030 [cs.IR] <https://arxiv.org/abs/1904.08030>
- [16] Chi Liu, Jiangxia Cao, Rui Huang, Kai Zheng, Qiang Luo, Kun Gai, and Guorui Zhou. 2024. KuaFormer: Transformer-Based Retrieval at Kuaishou. *arXiv preprint arXiv:2411.10057* (2024).
- [17] Yu. A. Malkov and D. A. Yashunin. 2018. Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs. arXiv:1603.09320 [cs.DS] <https://arxiv.org/abs/1603.09320>
- [18] Aleksandr V. Petrov and Craig Macdonald. 2023. Generative Sequential Recommendation with GPTRec. arXiv:2306.11114 [cs.IR] <https://arxiv.org/abs/2306.11114>
- [19] Steffen Rendle, Li Zhang, and Yehuda Koren. 2019. On the difficulty of evaluating baselines: A study on recommender systems. *arXiv preprint arXiv:1905.01395* (2019).
- [20] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. arXiv:1904.06690 [cs.IR] <https://arxiv.org/abs/1904.06690>
- [21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. arXiv:1706.03762 [cs.CL] <https://arxiv.org/abs/1706.03762>
- [22] 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.
- [23] Han Zhu, Daqing Chang, Ziru Xu, Pengye Zhang, Xiang Li, Jie He, Han Li, Jian Xu, and Kun Gai. 2019. Joint Optimization of Tree-based Index and Deep Model for Recommender Systems. arXiv:1902.07565 [cs.IR] <https://arxiv.org/abs/1902.07565>
- [24] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning Tree-based Deep Model for Recommender Systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*. ACM, 1079–1088. doi:10.1145/3219819.3219826
- [25] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning tree-based deep model for recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1079–1088.
- [26] Jingwei Zhuo, Ziru Xu, Wei Dai, Han Zhu, Han Li, Jian Xu, and Kun Gai. 2020. Learning Optimal Tree Models Under Beam Search. arXiv:2006.15408 [stat.ML] <https://arxiv.org/abs/2006.15408>

Received 21 August 2025; revised 21 August 2025; accepted 21 August 2025