UniSearch: Rethinking Search System with a Unified Generative Architecture

Anonymous ACL submission

Abstract

Modern search systems use multi-stage pipelines (recall, pre-ranking, ranking), but their complexity hinders holistic optimization. Recent advances in generative recommendation inspire unified generative search, yet existing methods are not fully end-to-end, leading to objective misalignment and limited generalization. We propose UniSearch, a unified generative framework for short-video search that replaces traditional pipelines with an end-to-end architecture comprising a Search Generator and a Video Encoder. These components are jointly optimized to enhance learning and improve generation accuracy. Additionally, we introduce Search Preference Optimization (SPO), which uses reward modeling and real user feedback to better align results with user preferences. Experiments on industrial-scale datasets and online A/B tests in short-video and live search scenarios demonstrate UniSearch's effectiveness, achieving the largest recent single-experiment improvement in live search performance.

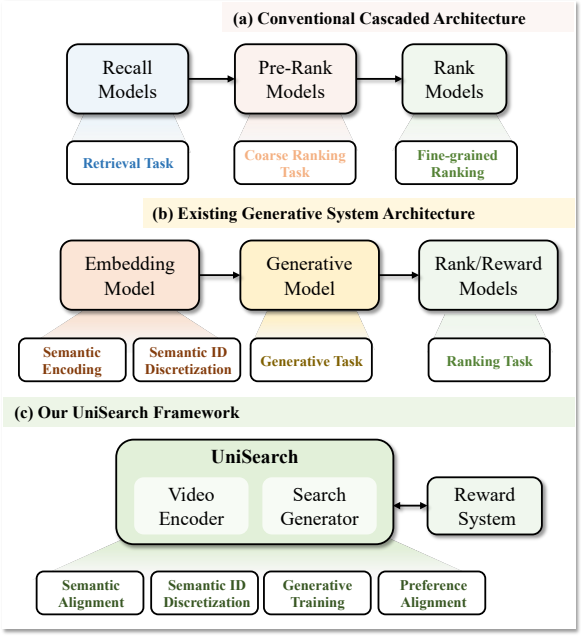

Figure 1: A schematic illustration of existing search architecture and our UniSearch. (a) Conventional Cascaded Architecture, (b) Existing Generative System Architecture and (c) Our UniSearch Framework.

1 Introduction

Search systems are essential in modern applications, helping users access relevant content from massive item collections (Bai et al., 2025; Lin et al., 2024). Over the past two decades, they have evolved from inverted indexing to deep neural models. However, most industrial search engines still rely on multi-stage cascaded architectures (MCA), which filter candidates in a coarse-to-fine manner through recall, pre-ranking, and ranking, as illustrated in Figure 1(a).

Despite years of refinement, cascaded systems have inherent drawbacks. First, stage-specific models optimized with different objectives produce misaligned training signals, limiting end-to-end relevance modeling (Zheng et al., 2024b). Second, the multi-stage design increases inference latency and maintenance cost because each stage requires dedicated modeling and system support (Deng et al., 2025). These limitations motivate *unified generative solutions* that replace the pipeline with a single end-to-end framework.

Recent advances in large-scale generative models have made such solutions increasingly feasible (Floridi and Chiriatti, 2020; Yang et al., 2025a; Wang et al., 2025b; Touvron et al., 2023; Liu et al., 2024a). In recommendation systems, generative models have shown promise in replacing cascaded pipelines (Zhou et al., 2025; Deng et al., 2025). However, most existing approaches are still not fully end-to-end, as shown in Figure 1(b): they first train a video encoder to produce embeddings and tokens, then train a generator to predict these tokens, leading to inconsistent optimization and suboptimal quality. Extending generative models to search is more challenging because search requires

cross-domain generation, mapping text queries directly to items. Existing work has mainly focused on early stages such as recall (Yang et al., 2023; Liu et al., 2024b), leaving unified generative modeling for the full search pipeline largely unexplored.

To address these challenges, we propose UniSearch, an end-to-end generative search framework for industrial-scale deployment. As shown in Figure 1(c), UniSearch consists of a *Search Generator* and a *Video Encoder*. The Generator follows an encoder–decoder architecture that takes query text and user features as input and autoregressively generates semantic IDs (SIDs) of relevant items. The Video Encoder learns latent embeddings from multimodal signals (*e.g.*, text, visual, and statistical features) and discretizes them into SIDs with VQ-VAE (Van Den Oord et al., 2017), creating a shared representation space between queries and items.

UniSearch further introduces a unified pre-training strategy that integrates semantic tokenization and autoregressive generation, jointly optimizing query and item representations. Residual contrastive learning with a coarse-to-fine strategy aligns query and video semantics, while VQ-VAE discretization jointly optimizes semantic IDs with the generative model. This removes the objective mismatch in prior work and yields stronger representations and higher-quality generation. Beyond supervised pre-training, UniSearch includes an online post-training phase, *Search Preference Optimization* (SPO), which refines the generation policy using a reward system that combines system-estimated signals (*e.g.*, relevance and content quality) with real user feedback (*e.g.*, clicks and watch time). For efficient and scalable inference, UniSearch uses a Trie-based mechanism to constrain generation to valid SID paths, preventing invalid outputs and improving efficiency. Deployed on a Chinese short-video platform, UniSearch integrates reinforcement learning, optimized inference with TensorRT and KV-cache, a dynamic Trie server, and a reward feedback system into a complete end-to-end generative search solution.

We evaluate UniSearch on large-scale datasets and real-world traffic. Offline experiments show superior relevance and efficiency over cascaded baselines, and online A/B tests further validate its effectiveness in live and short-video search. Our contributions are summarized as follows: (1) UniSearch is a **generative search solution deployed at scale**, replacing traditional cascaded architectures with a unified model that improves both retrieval quality and computational efficiency. (2) We propose a **unified training framework** that jointly integrates tokenization and generation, leading to more effective item encoding and higher-quality generation than conventional multi-stage training. (3) We deploy UniSearch in the live and short-video search systems of a Chinese short-video platform. Large-scale online A/B tests verify its practical effectiveness and efficiency for hundreds of millions of active users.

2 Related Works

2.1 Search System

Traditional search systems follow a cascaded pipeline with recall, pre-ranking, and ranking stages (Zhang et al., 2022; Xu et al., 2024; Khandagale et al., 2025; Xiong et al., 2024; Ye, 2024). Modern systems typically use BERT-based (Devlin et al., 2019) discriminative models with stage-specific architectures and objectives: recall often relies on dual-tower contrastive retrieval, pre-ranking adds lightweight interaction for candidate filtering, and ranking uses fully interactive single-tower models for precise relevance estimation. Although effective, such pipelines require heavy engineering and suffer from objective mismatch across stages, which can hurt end-to-end performance. We instead use a single generative model to unify objectives and replace the cascade.

2.2 Generative Retrieval

Autoregressive LLMs (OpenAI et al., 2024) have advanced recommendation and retrieval (Deng et al., 2025; Zhou et al., 2025; Zhai et al., 2024; Han et al., 2025; Yang et al., 2025b; Huang et al., 2025; Zheng et al., 2025; Li et al., 2025, 2024b; Liu et al., 2024b; Mustar et al., 2020; Pang et al., 2025; Rajput et al., 2023; Tang et al., 2024; Tay et al., 2022; Wang et al., 2025b; Yin et al., 2020; Zheng et al., 2024a; Li et al., 2024a). Generative retrieval formulates retrieval as sequence generation, directly producing document identifiers, passages, or answers instead of scoring candidates (Wang et al., 2025b; Tang et al., 2024). This paradigm avoids limitations of traditional indexing and often surpasses dual-encoder retrieval. In recommendation, prior work often represents items as discrete paths and generates them as retrieval targets (Tay et al., 2022; Rajput et al., 2023; Liu et al., 2024b; Zheng et al., 2024a; Pang et al., 2025). More recent work extends generation to full pipelines, includ-

ing OneRec (Deng et al., 2025; Zhou et al., 2025), OneSug (Guo et al., 2025), OneSearch (Chen et al., 2025), OnePOI (Wang et al., 2025a), and OneLoc (Wei et al., 2025). However, many methods still depend on pre-tokenized candidate items, limiting generalization and adaptability to dynamic content. Our framework removes fixed tokenization and enables open-ended generation of item identifiers.

3 Methodology

This section introduces the UniSearch architecture, its training scheme (unified pre-training and preference alignment post-training), and the inference process and deployment.

3.1 UniSearch Architecture

Unlike prior cascaded search systems that use different models across stages, UniSearch adopts a unified architecture for both training and inference. This design reduces objective mismatch and system complexity, enabling robust and efficient end-to-end learning. As shown in Figure 2(a), UniSearch has two components: a **Search Generator** and a **Video Encoder**.

Search Generator. The Generator follows an encoder–decoder paradigm. Its bidirectional Transformer encoder processes the user query and auxiliary user features (*e.g.*, GSU sequences encoding historical behaviors). We prepend a special `<cls>` token, whose final representation is used as the global query embedding q to summarize search intent. Conditioned on the encoder outputs, the decoder autoregressively generates tokens corresponding to semantic IDs of relevant videos, converting a natural-language query into a structured semantic representation of search results.

Video Encoder. The Video Encoder is a unidirectional Transformer that learns latent embeddings and semantic IDs for each video. Each item is represented by textual metadata, multimodal content features, and side statistical features. These are concatenated with k learnable tokens and fed into the Transformer, producing k latent embeddings $D = \{d^{(1)}, d^{(2)}, \ldots, d^{(k)}\}$. A VQ-VAE module then maps them into a codebook space to obtain the semantic ID (SID) sequence $S = \{s^{(1)}, s^{(2)}, \ldots, s^{(k)}\}$. This yields a compact and interpretable item representation for consistent indexing and retrieval in the generative search framework.

Together, the Generator and Video Encoder bridge textual queries and item representations within a single unified framework.

3.2 Unified Pre-training

To enable cooperative learning between the Generator and Video Encoder, we design a **unified pre-training framework** that jointly learns video encoding and query-to-video generation. Unlike prior generative recommendation methods that separate item tokenization and generation into two disjoint stages, UniSearch optimizes both objectives in a single pipeline, ensuring semantic consistency and avoiding task misalignment.

Following (Xie et al., 2025), the pre-training data are sampled from large-scale search logs, where each record contains a query, user features, a candidate video set $\{d_1, d_2, \ldots, d_N\}$, and a label set $\{l_1, l_2, \ldots, l_N\}$. The candidate set includes both exposed items and unexposed negative items for the query, while the label set provides graded relevance annotations reflecting both query–video relevance and video quality.

Residual Contrastive Learning for Semantic Alignment. The first objective of UniSearch pre-training is to align user queries with the latent representations of relevant items. Let q denote the query embedding produced by the Generator encoder and D_i the latent representation of the i-th candidate video encoded by the Video Encoder. We introduce *residual contrastive learning*, where each video embedding is represented as a sequence of residual components $d_i^{(n)}$. At step n, the current residual $d_i^{(n)}$ is optimized against the query embedding q_i while accumulating information from previous residuals, as defined in Equation 1. This residual formulation encourages each token to capture complementary information, reducing redundancy across tokens, and improving codebook utilization in discrete representation learning.

$$\mathcal{L}_{\text{contrast}} = \sum_{n=1}^{k} \mathcal{L}\Big(q_i, \text{sg}\big[\sum_{m<n} d_i^{(m)}\big] + d_i^{(n)}\Big), \quad (1)$$

$$Z(q) = \sum_{d^- \in \mathcal{N}} \exp(\text{sim}(q, d^-)/\tau), \quad (2)$$

$$\mathcal{L}(q, d) = -\log \frac{\exp(\text{sim}(q, d)/\tau)}{\exp(\text{sim}(q, d)/\tau) + Z(q)}, \quad (3)$$

where $\text{sg}(\cdot)$ denotes stop-gradient, τ is the temperature, and $\mathcal{N}$ is the negative sample set. Instead of cosine similarity, we use L2 similarity

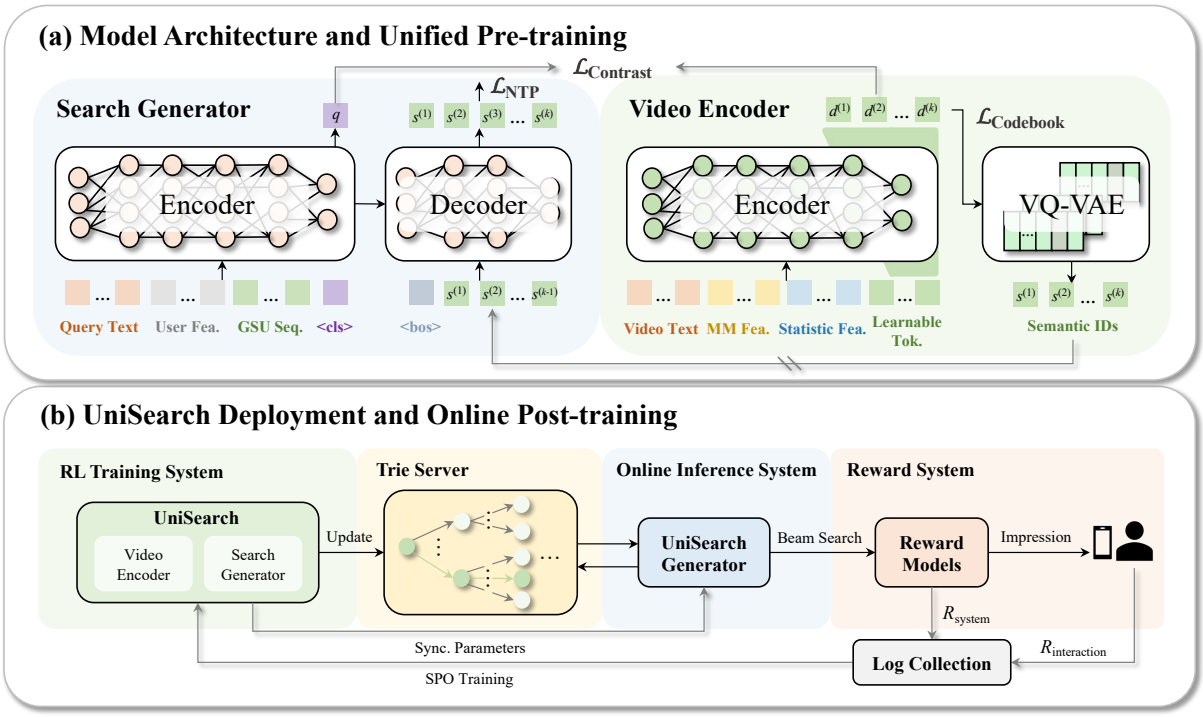

Figure 2: The overall architecture of UniSearch, where "Seq." denotes "Sequence", "Fea." stands for "Feature", "Learned Tok." refers to learned tokens for documents, and "MM Fea." represents multimodal features.

$\text{sim}(q, d) = 1 - \|q - d\|_2^2$, which better matches the residual accumulation mechanism and preserves the geometric structure of embeddings.

Coarse-to-Fine Strategy. To further strengthen semantic alignment, we design a coarse-to-fine training strategy that mimics the cascaded ranking process in traditional search systems. The first residual token $d^{(1)}$ learns an easier coarse-grained matching task, ensuring robust recall by distinguishing relevant from irrelevant items at a broad level. Subsequent tokens $d^{(2)}, \ldots, d^{(k)}$ progressively handle more complex tasks, refining the representation toward fine-grained ranking and personalization. This is achieved by gradually increasing the difficulty of the negative sample set $\mathcal{N}$: $d^{(1)}$ is trained with simple in-batch negatives, while later residuals use increasingly hard negatives sampled from semantically similar but irrelevant items.

Compared with residual quantization methods such as RQ-Kmeans that rely on two-stage residual clustering, our approach supports end-to-end training, reducing clustering loss and avoiding mismatched objectives. The residual formulation not only decomposes learning into increasingly difficult subtasks but also prevents token redundancy, leading to more effective codebook use. As a result, UniSearch captures both strong recall and refined ranking within a single unified framework, outperforming approaches optimized only for recall.

Discretization into Semantic IDs. While semantic alignment yields meaningful item representations, the Video Encoder must also discretize latent embeddings into semantic IDs for generative modeling. We employ VQ-VAE (Van Den Oord et al., 2017) for this purpose. Unlike prior works that rely on post-clustering methods such as balanced K-means or RQ-Kmeans (Deng et al., 2025; Guo et al., 2025), VQ-VAE updates the codebook jointly during training, enabling efficient online learning and inference. For each embedding $d_i^{(n)}$, the VQ-VAE encoder performs nearest-neighbor lookup in the learnable codebook to obtain a quantized embedding $e_i^{(n)}$ and corresponding semantic ID $s_i^{(n)}$. The quantized embeddings $E_i = \{e_i^{(1)}, e_i^{(2)}, \ldots, e_i^{(k)}\}$ are further required to reconstruct the original embeddings D_i. Codebook loss is introduced to ensure $e_i^{(n)}$ approximates d_i, as shown in Equation 4:

$$\mathcal{L}_{\text{codebook}} = \sum_{n=1}^{k} \alpha_1 ||\text{sg}[d_i^{(n)}] - e_i^{(n)}||_2^2 + \alpha_2 ||d_i^{(n)} - \text{sg}[e_i^{(n)}]||_2^2. \quad (4)$$

Specifically, we adopt the SimVQ strategy (Zhu et al., 2024) to further stabilize discretization and avoid codebook collapse. With the VQ module, videos are discretized into semantic IDs in a fully end-to-end manner without the inconsistencies of offline clustering.

Reject-Sampling Generative Training. Once items are mapped into semantic IDs, the Generator is trained to autoregressively generate these sequences using a next-token prediction objective with cross-entropy loss. To improve generation quality, we employ reject-sampling during training: low-quality items (determined by labels) are filtered out, and losses for items of different levels are reweighted accordingly, as shown in Equation 5:

$$\mathcal{L}_{\text{NTP}} = -w_i \sum_{n=1}^{k} \log p\big(s_i^{(n)} | q, u, s_i^{(<n)}\big), \quad (5)$$

where w_i is a reweighting factor determined by label l_i. This ensures that the Generator prioritizes high-relevance, high-quality items when producing search results.

Overall. Through the above unified pre-training objectives, UniSearch jointly optimizes query–item semantic alignment, item discretization, and high-quality sequence generation, providing a consistent and robust foundation for subsequent post-training and inference. The overall training loss is:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{contrast}} + \lambda_2 \mathcal{L}_{\text{codebook}} + \lambda_3 \mathcal{L}_{\text{NTP}}, \quad (6)$$

where $\lambda_{1,2,3}$ are hyper-parameters that balance loss scales and stabilize training.

3.3 Post-training with SPO

After pre-training, UniSearch can already generate usable search results. To further improve result quality and better align with user preferences, we introduce an online reinforcement learning framework called Search Preference Alignment. This post-training stage is conducted in a controlled online environment, as shown in Figure 2(b).

Specifically, the deployed UniSearch model (see Section 3.4) generates N candidate result sequences via Beam Search. These candidates are fed into a Reward System, which assigns each result a reward score based on factors such as relevance and video quality. The top-M results are then exposed to users. Subsequent user interactions (*e.g.*, clicks, likes, or downloads) are collected as an additional reward signal. The combined rewards are used to update UniSearch through Search Preference Optimization, aligning the model output distribution with user preferences in an online, data-driven way.

Reward System. Modern industrial search engines have been refined for years, especially in ranking. To leverage this, we reuse the existing fine-ranking module of the production search system as the Reward System. It takes the query, user context, and candidate video features as input, and outputs predictive scores over dimensions such as query–item relevance, video quality, and expected user satisfaction. These scores form part of the reward signal. After results are shown, explicit user feedback (*e.g.*, click-through, watch time, likes, or downloads) provides another direct measure of search quality. The final reward is a weighted combination of system-estimated scores and observed interactions:

$$R = \gamma_1 R_{\text{system}} + \gamma_2 R_{\text{interaction}}, \quad (7)$$

where $\gamma_{1,2}$ balance the contributions of system-predicted and user-behavioral signals.

Search Preference Optimization. From online logs, we obtain G generated results for a query, denoted as $\{S_1, S_2, \ldots, S_G\}$, together with reward scores $\{R_1, R_2, \ldots, R_G\}$. For each candidate $S_i = \{s_i^{(1)}, s_i^{(2)}, ..., s_i^{(k)}\}$, UniSearch assigns a generation probability $p_i = \pi_\theta(S_i \mid q, u)$ conditioned on query q and user context u. Following GRPO (Liu et al., 2024a), we compute the relative advantage of each candidate by normalizing its reward against the batch expectation:

$$A_i = \frac{R_i - \text{mean}(\{R_1, R_2, \ldots, R_G\})}{\text{std}(\{R_1, R_2, \ldots, R_G\})}. \quad (8)$$

This encourages the model to favor results above the batch average and suppress worse ones. The final optimization objective is:

$$L_{SPO}(\theta) = -\frac{1}{G}\sum_{i=1}^{G}\frac{1}{k}\sum_{n=1}^{k}\Big(\frac{\pi_\theta\big(s_i^{(n)}|q,u,s_i^{(<n)}\big)}{\pi_{\theta_{\text{no_grad}}}\big(s_i^{(n)}|q,u,s_i^{(<n)}\big)}A_i - \beta \mathbb{D}_{\text{KL}}[\pi_\theta || \pi_{ref}]\Big), \quad (9)$$

where the first term increases the likelihood of preference-aligned generations weighted by their relative advantages, and the second regularizes the updated policy π_θ toward the reference policy π_{ref} from pre-training. The coefficient β controls the strength of KL regularization.

This formulation aligns UniSearch with both system-estimated and user-observed preferences while maintaining stability and avoiding catastrophic deviation from the pre-trained model.

3.4 Inference and Deployment

To enhance inference efficiency, UniSearch leverages a prefix tree (Trie) structure to constrain the generation of valid semantic ID sequences. A dedicated Trie service maintains permissible paths, ensuring only valid outputs. At each decoding step, UniSearch queries the Trie for feasible continuation tokens, eliminating invalid IDs and reducing computational overhead. To handle large video corpora, the Trie integrates prefix-tree search with binary search, minimizing memory overhead in a massive semantic ID space.

UniSearch has been deployed in real-world industrial search scenarios, integrating four key components into a unified system (Figure 2 (b)). First, a reinforcement learning (RL) system fine-tunes UniSearch with online preference alignment and synchronizes updated parameters to the inference service. Second, an online inference system handles real-time requests, leveraging TensorRT and KV caching for low-latency responses. Third, the Trie server ensures efficient constrained generation by maintaining valid semantic ID paths, enhancing accuracy and scalability. Lastly, a reward system evaluates results using ranking scores and user interaction signals, providing feedback to guide training. UniSearch's end-to-end design ensures scalable, low-latency inference while enabling continuous improvement in search quality through online learning and preference optimization.

4 Experiments

In this section, we present the implementation details of UniSearch, and both offline and online experimental analyzes.

4.1 Implementation Details

Datasets and Metrics. We construct large-scale training and test sets from real search logs of a Chinese short-video platform. Experiments are conducted on two evaluation subsets: the *ranking subset* (RK), which contains top-ranked videos previously exposed by the production system, and the *click subset* (CK), which treats user-clicked items as positives. We report Recall@300 and Mean Reciprocal Rank (MRR). More details are provided in Appendix A.

Details of UniSearch. We uses BART (Lewis et al., 2020) as the Search Generator and BERT (Devlin et al., 2019) as the Video Encoder. For live search with about 500K candidates, we use a lightweight 6-layer configuration with hidden size 768 and a three-level codebook ($k = 3$) with 512 entries per level. For short-video search with about one billion candidates, we use a larger 12-layer configuration with 8192 entries per codebook level. Additional details are deferred to the appendix.

4.2 Main Results

To verify the effectiveness of UniSearch, we conduct extensive experiments on the live-streaming search dataset and compare it with existing generative search frameworks. All methods are evaluated under the same settings for fair comparison.

Baselines. As discussed in the Introduction, existing generative search methods are not fully end-to-end. They typically adopt a two-stage pipeline: (1) training an embedding model and constructing a codebook using methods such as VQ-VAE (Van Den Oord et al., 2017), FSQ (Mentzer et al., 2024), or RQ-Kmeans; and (2) training a generative model on tokenized items. We reproduce these methods as baselines, with results shown in Table 1 (first five rows). Although they achieve reasonable performance (*e.g.*, a 6-layer baseline with RQ-Kmeans reaches an MRR of 14.81 on CK), their reliance on multiple models and mismatched objectives often leads to suboptimal results.

Advantages of Unified Pre-training. In contrast, UniSearch unifies representation learning, codebook construction, and generative learning in a single framework. This joint optimization removes objective inconsistency and enables better query understanding and higher-quality generation. As shown in Table 1, UniSearch consistently outperforms baseline methods in offline evaluation. At the same model scale, UniSearch-6 Layer significantly surpasses the 6-layer baselines in MRR and achieves Recall@300 close to 12-layer baselines. This suggests that the unified objective produces more relevant results and improves overall generation quality. UniSearch also scales well: both 12 Layer and 24 Layer yield substantial gains over baselines of the same scale, demonstrating the robustness of the framework at larger capacities.

Effect of Search Preference Optimization. We further apply Search Preference Optimization to both 6-layer and 12-layer UniSearch models. Results are shown in the last two rows of Table 1. SPO brings moderate gains in Recall@300 and substantial improvements in MRR, indicating that it helps the model generate results that are not only relevant but also higher-quality and better aligned

Table 1: Performance comparison of different model on live-streaming search dataset.

	Model			Recall@300(%)↑		MRR(%)↑	
	Embed. Model	**Codebook**	**Gen. Model**	**RK**	**CK**	**RK**	**CK**
Existing Generative Search Arch.	BERT-6 Layer	FSQ	BART- 6 Layer	55.42	62.26	8.36	9.93
		VQ-VAE		64.83	63.67	7.98	9.50
		RQ-Kmeans		65.56	64.12	9.24	14.81
	BERT-12 Layer	RQ-Kmeans	BART-12 Layer	67.79	67.90	12.13	15.68
	BERT-24 Layer		BART-24 Layer	69.05	69.27	13.84	16.80
UniSearch (Ours)	UniSearch- 6 Layer			67.45	68.17	9.65	15.36
	UniSearch-12 Layer			68.83	70.11	12.35	15.73
	UniSearch-24 Layer			69.78	72.03	14.62	16.02
	UniSearch- 6 Layer (w/ Online SPO)			68.29	68.48	12.19	16.04
	UniSearch-12 Layer (w/ Online SPO)			69.53	70.25	13.56	16.97

with user preferences. In particular, SPO encourages the model to prioritize items more likely to be preferred by users, such as popular or high-quality videos, thereby improving user experience.

4.3 Ablation Study

We conduct ablation studies on the training scheme, codebook configuration, and inference strategy.

Ablation on Unified Pre-training. To isolate the contributions of unified pre-training, we compare three ablated UniSearch variants in Table 2.

(1) Baseline treats each token independently and optimizes all tokens toward the same objective, weakening token diversity and yielding the worst Recall@300 and MRR. (2) Baseline w/ CF adds a coarse-to-fine (CF) task. By learning from easier to harder objectives, it improves ranking and achieves clear gains in recall and accuracy. However, because coarse and fine tasks are correlated, token representations often become similar, causing the *path collapsing problem*, where multiple candidate items share one decoding path, reducing discriminative power and precision. (3) Baseline w/ RCL incorporates residual contrastive learning (RCL). Rather than optimizing the final objective in one step, RCL progressively refines representations through residual summation. This encourages tokens to capture complementary semantics, mitigates path collapse, and substantially improves MRR. Our full UniSearch combines CF and RCL. CF provides a path from easy to hard objectives, while the residual design preserves token diversity and prevents collapse. Together, they enable more effective end-to-end discretization, reduce clustering loss, and achieve the best overall performance.

Ablation on Codebook Depth and Size. Codebook depth and size are key hyperparameters in candidate tokenization, directly affecting retrieval quality and efficiency.

Table 2: Analysis results of Coarse-to-Fine strategy (CF) and Residual Contrastive Learning (RCL), and baseline model refers to UniSearch without CF and RCL.

Method	**Recall@300**(%) ↑		**MRR**(%) ↑	
	RK	**CK**	**RK**	**CK**
Baseline	63.54	63.15	6.55	12.16
Baseline w/ CF	64.63	65.61	7.01	12.97
Baseline w/ RCL	66.18	65.41	8.59	14.71
UniSearch	**67.45**	**68.17**	**9.65**	**15.36**

(1) To study depth, we fix the codebook size at 512. Although deeper codebooks may capture richer semantics with longer token sequences, Figure 3(a) shows that performance saturates once depth k exceeds 3, with diminishing returns on both Recall@300 and MRR. One possible reason is that deeper codebooks increase dispersion in the candidate space and weaken generalization. They also prolong decoding and increase latency. Considering both effectiveness and efficiency, we use $k = 3$. (2) We then fix the depth at $k = 3$ and vary the size of each codebook layer. Figure 3(b) shows divergent trends: Recall@300 decreases as size grows, while MRR steadily improves. Larger codebooks appear to encourage finer semantic partitioning, improving ranking precision but reducing recall by fragmenting candidate coverage. To balance recall and precision, we set the codebook size to $w = 512$, which performs well on both metrics.

Impact of the Trie Constraint. The validity of generated paths (*i.e.*, whether they correspond to actual candidate items) is critical for system reliability. Without constraints, invalid generations may produce empty result sets and threaten search

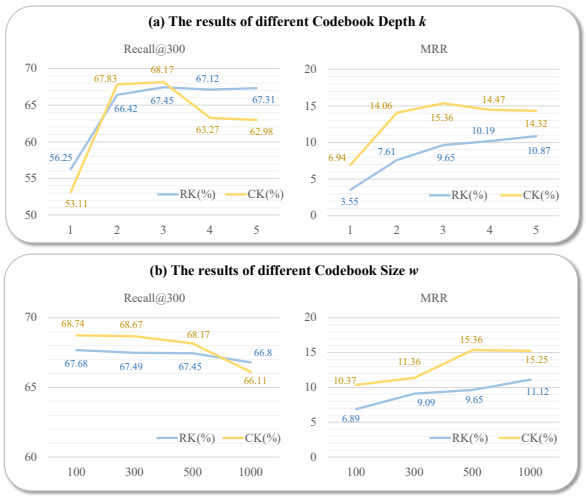

Figure 3: The results of different Codebook Depth k and Codebook Size w on RK and CK test set.

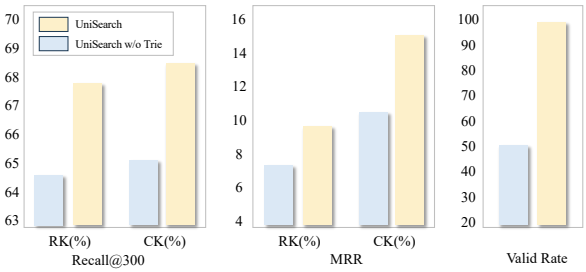

Figure 4: The results of incorporating the Trie on RK and CK test set, and Valid Rates of path to candidates.

availability. We therefore incorporate a Trie-based constraint during inference.

As shown in Figure 4, the Trie markedly improves both Recall@300 and MRR, while path validity rises from 51.3% to 99.8%. This ensures that nearly all generated results map to valid candidates, improving reliability and efficiency. In live search, dynamic Trie updates are also necessary because the valid path set changes as live streams start or end, and evolving content can alter valid identifiers for the same stream.

4.4 Online Testing

We evaluate UniSearch by deploying it to the live-streaming and short-video search systems of a Chinese short-video platform and conducting online A/B tests against the legacy production system.

Online A/B Testing Setup. For live-streaming search, we fully replace the original Multi-stage Cascading Architecture with UniSearch. In contrast, short-video search serves much higher traffic and therefore requires a more conservative deployment strategy. We thus adopt progressive integration: UniSearch is introduced as an additional retrieval source in the ranking stage, while several existing recall sources are gradually phased out. This incremental deployment lets us evaluate UniSearch under real production constraints while minimizing risks to system performance and user experience.

Table 3: The improvements of UniSearch in online A/B test compared to the production baseline.

UniSearch -Live	**TPC ↑**	**CTR ↑**	**CQR ↓**	**FCP ↓**
	+3.31%	+0.202%	-0.382%	-0.107%
UniSearch -Video	**VPD ↑**	**PVD ↑**	**CQR ↓**	**LPC ↑**
	+0.213%	+0.993%	-0.602%	+0.830%

Impact on Business Metrics. We evaluate UniSearch using business-critical metrics that reflect user satisfaction and engagement. Results are summarized in Table 3.

(1) Live-streaming search. UniSearch yields a +3.31% improvement in total play counts (TPC), a +0.202% increase in page click-through rate (CTR), and reductions of -0.382% and -0.107% in change query rate (CQR) and position of first click (PFC), respectively. TPC is the most important metric because it reflects both platform vitality and user engagement, and this gain is the largest seen in recent years. The increase in CTR indicates stronger engagement, while the reductions in CQR and PFC suggest that users find satisfactory content faster and with fewer query reformulations. Further analysis is provided in Appendix B. (2) Short-video search. UniSearch yields a +0.213% improvement in video playback duration (VPD), a +0.993% increase in page view depth (PVD), a -0.602% reduction in CQR, and a +0.830% increase in long-play count per page view (LPC). These gains indicate that UniSearch not only better satisfies immediate search intent, but also encourages deeper exploration and more diverse content consumption.

5 Conclusion

We propose **UniSearch**, a unified generative search framework that replaces traditional multi-stage architectures with an end-to-end solution. By jointly optimizing the Search Generator and Video Encoder, UniSearch unifies tokenization, representation learning, and result generation, improving semantic modeling and output quality. Search Preference Optimization further aligns results with user preferences. Experiments and real-world deployment validate its effectiveness.

Limitations

Currently, UniSearch generates candidates in a *point-wise* manner using beam search, which may limit result diversity. Future work will focus on enhancing *list-wise generation* to improve both diversity and ranking accuracy, as well as exploring finer-grained reward methods to further strengthen alignment with user preferences.

References

Yimeng Bai, Shunyu Zhang, Yang Zhang, Hu Liu, Wentian Bao, Enyun Yu, Fuli Feng, and Wenwu Ou. 2025. Unconstrained monotonic calibration of predictions in deep ranking systems. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Ben Chen, Xian Guo, Siyuan Wang, Zihan Liang, Yue Lv, Yufei Ma, Xinlong Xiao, Bowen Xue, Xuxin Zhang, Ying Yang, and 1 others. 2025. Onesearch: A preliminary exploration of the unified end-to-end generative framework for e-commerce search. *arXiv preprint arXiv:2509.03236*.

Jiaxin Deng, Shiyao Wang, Kuo Cai, Lejian Ren, Qigen Hu, Weifeng Ding, Qiang Luo, and Guorui Zhou. 2025. Onerec: Unifying retrieve and rank with generative recommender and iterative preference alignment. *arXiv preprint arXiv:2502.18965*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Luciano Floridi and Massimo Chiriatti. 2020. Gpt-3: Its nature, scope, limits, and consequences. *Minds and Machines*, 30:681–694.

Xian Guo, Ben Chen, Siyuan Wang, Ying Yang, Chenyi Lei, Yuqing Ding, and Han Li. 2025. Onesug: The unified end-to-end generative framework for e-commerce query suggestion. *arXiv preprint arXiv:2506.06913*.

Ruidong Han, Bin Yin, Shangyu Chen, He Jiang, Fei Jiang, Xiang Li, Chi Ma, Mingcong 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. *arXiv preprint arXiv:2505.18654*.

Yanhua Huang, Yuqi Chen, Xiong Cao, Rui Yang, Mingliang Qi, Yinghao Zhu, Qingchang Han, Yaowei Liu, Zhaoyu Liu, Xuefeng Yao, Yuting Jia, Leilei Ma, Yinqi Zhang, Taoyu Zhu, Liujie Zhang, Lei Chen, Weihang Chen, Min Zhu, Ruiwen Xu, and Lei Zhang. 2025. Towards large-scale generative ranking. *arXiv preprint arXiv:2505.04180*.

Sujay Khandagale, Bhawna Juneja, Prabhat Agarwal, Aditya Subramanian, Jaewon Yang, and Yuting Wang. 2025. Interactrank: Personalized web-scale search pre-ranking with cross interaction features. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 287–295.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Mingming Li, Huimu Wang, Zuxu Chen, Guangtao Nie, Yiming Qiu, Guoyu Tang, Lin Liu, and Jingwei Zhuo. 2024a. Generative retrieval with preference optimization for e-commerce search. *arXiv preprint arXiv:2407.19829*.

Xiaoxi Li, Jiajie Jin, Yujia Zhou, Yuyao Zhang, Peitian Zhang, Yutao Zhu, and Zhicheng Dou. 2025. From matching to generation: A survey on generative information retrieval. *ACM Transactions on Information Systems*, pages 1–62.

Xiaoxi Li, Yujia Zhou, and Zhicheng Dou. 2024b. Unigen: A unified generative framework for retrieval and question answering with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8688–8696.

Juexin Lin, Sachin Yadav, Feng Liu, Nicholas Rossi, Praveen R Suram, Satya Chembolu, Prijith Chandran, Hrushikesh Mohapatra, Tony Lee, Alessandro Magnani, and 1 others. 2024. Enhancing relevance of embedding-based retrieval at walmart. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 4694–4701.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024a. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Yuxuan Liu, Tianchi Yang, Zihan Zhang, Minghui Song, Haizhen Huang, Weiwei Deng, Feng Sun, and Qi Zhang. 2024b. Asi++: Towards distributionally balanced end-to-end generative retrieval. *arXiv preprint arXiv:2405.14280*.

Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. 2024. Finite scalar quantization: Vq-vae made simple.

Agnès Mustar, Sylvain Lamprier, and Benjamin Piwowarski. 2020. Using bert and bart for query suggestion. In *Joint Conference of the Information Retrieval Communities in Europe*, volume 2621.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, and 1 others. 2024. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Ming Pang, Chunyuan Yuan, Xiaoyu He, Zheng Fang, Donghao Xie, Fanyi Qu, Xue Jiang, Changping Peng, Zhangang Lin, Ching Law, and Jingping Shao. 2025. Generative retrieval and alignment model: A new paradigm for e-commerce retrieval. *arXiv preprint arXiv:2504.01403*.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, and 1 others. 2023. Recommender systems with generative retrieval. *Advances in Neural Information Processing Systems*, 36:10299–10315.

Yubao Tang, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Wei Chen, and Xueqi Cheng. 2024. Listwise generative retrieval models via a sequential learning process.

Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, and 1 others. 2022. Transformer memory as a differentiable search index. *Advances in Neural Information Processing Systems*, 35:21831–21843.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Aaron Van Den Oord, Oriol Vinyals, and 1 others. 2017. Neural discrete representation learning. *Advances in Neural Information Processing Systems*, 30.

Dongsheng Wang, Yuxi Huang, Shen Gao, Yifan Wang, Chengrui Huang, and Shuo Shang. 2025a. Generative next poi recommendation with semantic id. *arXiv preprint arXiv:2506.01375*.

Zhibo Wang, Xiaoze Jiang, Zhiheng Qin, Enyun Yu, and Han Li. 2025b. Personalized query auto-completion for long and short-term interests with adaptive detoxification generation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD '25), August 3–7, 2025, Toronto, ON, Canada*.

Zhipeng Wei, Kuo Cai, Junda She, Jie Chen, Minghao Chen, Yang Zeng, Qiang Luo, Wengcong Zeng, Ruiming Tang, Kun Gai, and Guorui Zhou. 2025. Oneloc: Geo-aware generative recommender systems for local life service. *arXiv preprint arXiv:2508.14646*.

Ao Xie, Jiahui Chen, Quanzhi Zhu, Xiaoze Jiang, Zhiheng Qin, Enyun Yu, and Han Li. 2025. Crops: Improving dense retrieval with cross-perspective positive samples in short-video search. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Haoyi Xiong, Jiang Bian, Yuchen Li, Xuhong Li, Mengnan Du, Shuaiqiang Wang, Dawei Yin, and Sumi Helal. 2024. When search engine services meet large language models: Visions and challenges. *IEEE Transactions on Services Computing*.

Enqiang Xu, Yiming Qiu, Junyang Bai, Ping Zhang, Dadong Miao, Songlin Wang, Guoyu Tang, Lin Liu, and MingMing Li. 2024. Optimizing e-commerce search: Toward a generalizable and rank-consistent pre-ranking model. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2875–2879.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Tianchi Yang, Minghui Song, Zihan Zhang, Haizhen Huang, Weiwei Deng, Feng Sun, and Qi Zhang. 2023. Auto search indexer for end-to-end document retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6955–6970.

Yuhao Yang, Zhi Ji, Zhaopeng Li, Yi Li, Zhonglin Mo, Yue Ding, Kai Chen, Zijian Zhang, Jie Li, Shuanglong Li, and Lin Liu. 2025b. Sparse meets dense: Unified generative recommendations with cascaded sparse-dense representations. *arXiv preprint arXiv:2503.02453*.

Cheng Ye. 2024. Exploring a learning-to-rank approach to enhance the retrieval augmented generation (rag)-based electronic medical records search engines. *Informatics and Health*, 1(2):93–99.

Di Yin, Jiwei Tan, Zhe Zhang, Hongbo Deng, Shujian Huang, and Jiajun Chen. 2020. Learning to generate personalized query auto-completions via a multi-view multi-task attentive approach. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2998–3007.

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 preprint arXiv:2402.17152*.

Shunyu Zhang, Yaobo Liang, Ming Gong, Daxin Jiang, and Nan Duan. 2022. Multi-view document representation learning for open-domain dense retrieval. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5990–6000.

Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, Ming Chen, and Ji-Rong Wen. 2024a. Adapting large language models by integrating collaborative semantics for recommendation. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 1435–1448. IEEE.

Kai Zheng, Haijun Zhao, Rui Huang, Beichuan Zhang, Na Mou, Yanan Niu, Yang Song, Hongning Wang, and Kun Gai. 2024b. Full stage learning to rank: A unified framework for multi-stage systems. In *Proceedings of the ACM Web Conference 2024*, page 3621–3631.

Zuowu Zheng, Ze Wang, Fan Yang, Jiangke Fan, Teng Zhang, and Xingxing Wang. 2025. Ega: A unified end-to-end generative framework for industrial advertising systems. *arXiv preprint arXiv:2505.17549*.

Guorui Zhou, Jiaxin Deng, Jinghao Zhang, Kuo Cai, Lejian Ren, Qiang Luo, Qianqian Wang, Qigen Hu, Rui Huang, Shiyao Wang, and 1 others. 2025. Onerec technical report. *arXiv preprint arXiv:2506.13695*.

Yongxin Zhu, Bocheng Li, Yifei Xin, Zhihua Xia, and Linli Xu. 2024. Addressing representation collapse in vector quantized models with one linear layer. *arXiv preprint arXiv:2411.02038*.

A Implementation Details

Datasets. For live-streaming search, we collect $\tilde{8}$0M user search sessions spanning the period from July 9 to July 23, 2025. Data from July 9 to July 22 are used for training, while logs from July 23 form the held-out test set. All offline experiments reported in Sections 4.2 and 4.3 are conducted on this live-streaming dataset. Following the same procedure, we also construct a training/testsing dataset for short-video search, which is used in Section 4.4 to validate the generalization of UniSearch beyond live scenarios.

Training and Inference Details. We use Adam with a batch size of 64. During pre-training, the initial learning rate is set to 2×10^{-4} with cosine decay. For online post-training, we use a fixed learning rate of 1×10^{-5}. During inference, beam search is applied with beam size 256.

Evaluation Metrics. To evaluate the performance of our model, we adopt Recall@300 and Mean Reciprocal Rank (MRR) as evaluation metrics. For Recall@300, let T_i denote the number of true positive instances among the top 300 retrieved results (*i.e.*, instances that are both predicted and labeled as positive), let P_i denote the total number of positive instances in the dataset, and let N represent the total number of samples:

$$\text{Recall@300} = \frac{1}{N} \sum_{i=1}^{N} \frac{T_i}{P_i}. \tag{10}$$

For the MRR metric, K_i denotes the position of the ground-truth result for the i-th sample in the ranked list of candidates:

$$\text{MRR} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{K_i}. \tag{11}$$

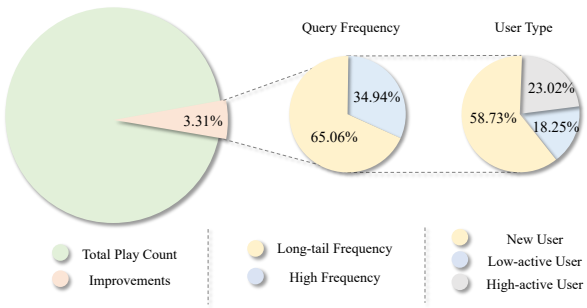

Figure 5: The analysis of the improvements in Total Play Count (TPC) from query frequency and user type.

B Further Analysis

From the online experiments reported in Section 4.4, UniSearch achieves substantial improvements in key business metrics for both live-streaming and short-video search. To better understand the underlying reasons, we perform a detailed analysis of the Live Search scenario, examining the contributions of query frequency, user activity status, and the diversity of generated results.

Enhanced Performance on Long-tail Queries. Given that TPC is the most critical metric for Live Search, we conduct a fine-grained decomposition of its gain. As shown in Figure 5, 65.06% of the TPC improvement is driven by long-tail queries—nearly double the contribution from head queries. This underscores the importance of effectively modeling long-tail traffic in industrial search. Long-tail queries are typically characterized by their sparse occurrence and semantic complexity, making them challenging for traditional systems like MCA to handle. UniSearch, by contrast, benefits from unified pre-training and improved semantic representations, enabling it to capture subtle user intents and generalize more effectively across diverse and low-frequency queries.

Stronger Appeal to New Users. We further segment the TPC gains by user activity status. As illustrated in Figure 5, 58.73% of the increase originates from new users, accounting for more than half of the total gain. This suggests that UniSearch delivers results that are particularly attractive to users who are less familiar with the platform. A plausible explanation is that existing users have

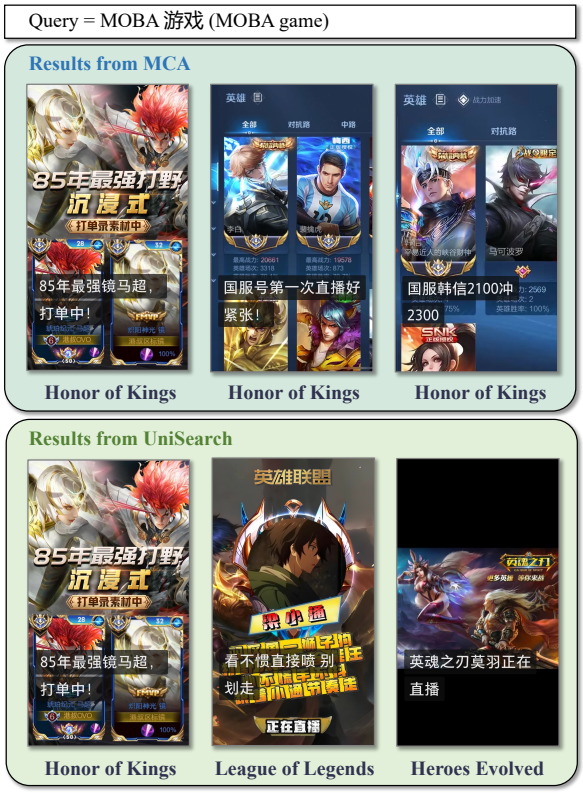

Figure 6: Search results comparison between the Multi-stage Cascading Architecture (MCA) and our UniSearch.

already adapted to the behavioral patterns of the legacy MCA system, while new users are more sensitive to improved semantic understanding and richer result sets. This improvement in new-user engagement is especially valuable, as it directly contributes to expanding the active search user base and sustaining long-term growth.

Richer and More Diverse Results. Case studies further validate that UniSearch generates results that are both semantically relevant and diverse. For example, Figure 6 presents the query "MOBA game". Under MCA, the top three results across users are consistently "Honor of Kings", showing strong bias toward a single popular item. In contrast, UniSearch surfaces additional relevant games such as "League of Legends" and "Heroes Evolved". This broader coverage ensures that users with different preferences all find satisfactory content. Without such diversity, users whose interests are not reflected (*e.g.*, those preferring "League of Legends") may abandon the platform and reduce future engagement. By balancing relevance with diversity, UniSearch not only improves user satisfaction but also mitigates the risk of content homogenization.