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 substantial gains in live search performance over a strong production baseline.

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.

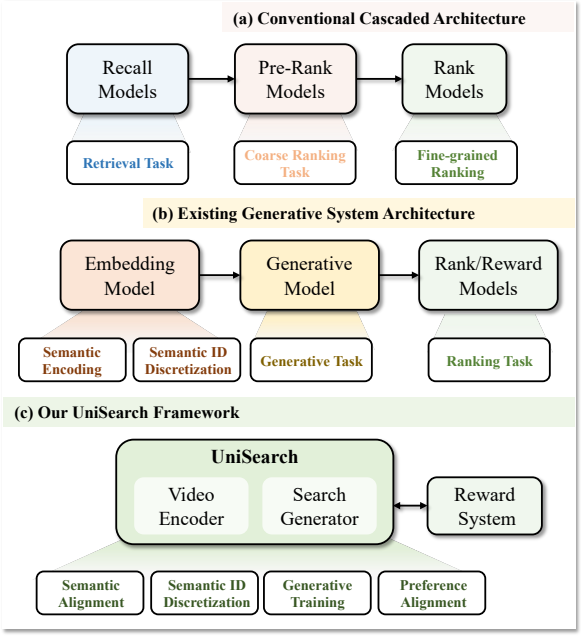

Figure 1: A schematic illustration of existing search architecture and our UniSearch. (a) Conventional Cascaded Architecture, (b) Existing Generative System Architecture and (c) Our UniSearch Framework.

Recent advances in large-scale generative models have made such solutions increasingly feasible (Floridi and Chiriatti, 2020; Yang et al., 2025; Wang et al., 2025; Touvron et al., 2023; Liu et al., 2024a). In recommendation systems, generative models have shown promise in replacing cascaded pipelines (Zhou et al., 2025). However, many existing approaches in the search (like OneSearch (Chen et al., 2025)) and recommendation (like OneRec (Deng et al., 2025)) setting still adopt a two-stage pipeline (Figure 1(b)) that first trains a video encoder to produce embeddings and tokens, and then trains a generator to predict these tokens, leading to inconsistent optimization and suboptimal qual-

ity. 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. *Codes are in the attachment*.

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 mismatches between stages, which can affect end-to-end performance.

2.2 Generative Retrieval

Generative retrieval formulates retrieval as sequence generation, directly producing document identifiers, passages, or answers instead of scoring candidates (Tay et al., 2022; Rajput et al., 2023). Recent work has explored balanced identifier learning (Liu et al., 2024b), semantic item tokenization for recommendation (Zheng et al., 2024a), and preference optimization for generative retrieval (Pang et al., 2025; Li et al., 2024a). Beyond document retrieval, generative formulations have also been applied to query suggestion (Mustar et al., 2020; Yin et al., 2020) and unified retrieval-QA settings (Li et al., 2024b).

Recent studies further extend generative paradigms to industrial recommendation and search systems. OneRec (Deng et al., 2025), OneSug (Guo et al., 2026), and OneSearch (Chen et al., 2025) unify multiple stages in recommendation, query suggestion, and e-commerce search, respectively. Large-scale generative

ranking architectures such as HSTU (Zhai et al., 2024), MTGR (Han et al., 2025), and generative advertising systems (Zheng et al., 2025) further demonstrate the scalability of this paradigm. Closest to our work, GenRet (Sun et al., 2023) and ETEGRec (Liu et al., 2025) jointly optimize tokenization and generation, but focus on document retrieval and recommendation. UniSearch instead targets dynamic live-streaming and short-video search, proposing a unified generative search paradigm for continuously evolving content.

3 Methodology

This section introduces the UniSearch architecture, its training scheme, 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 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., 2026), 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

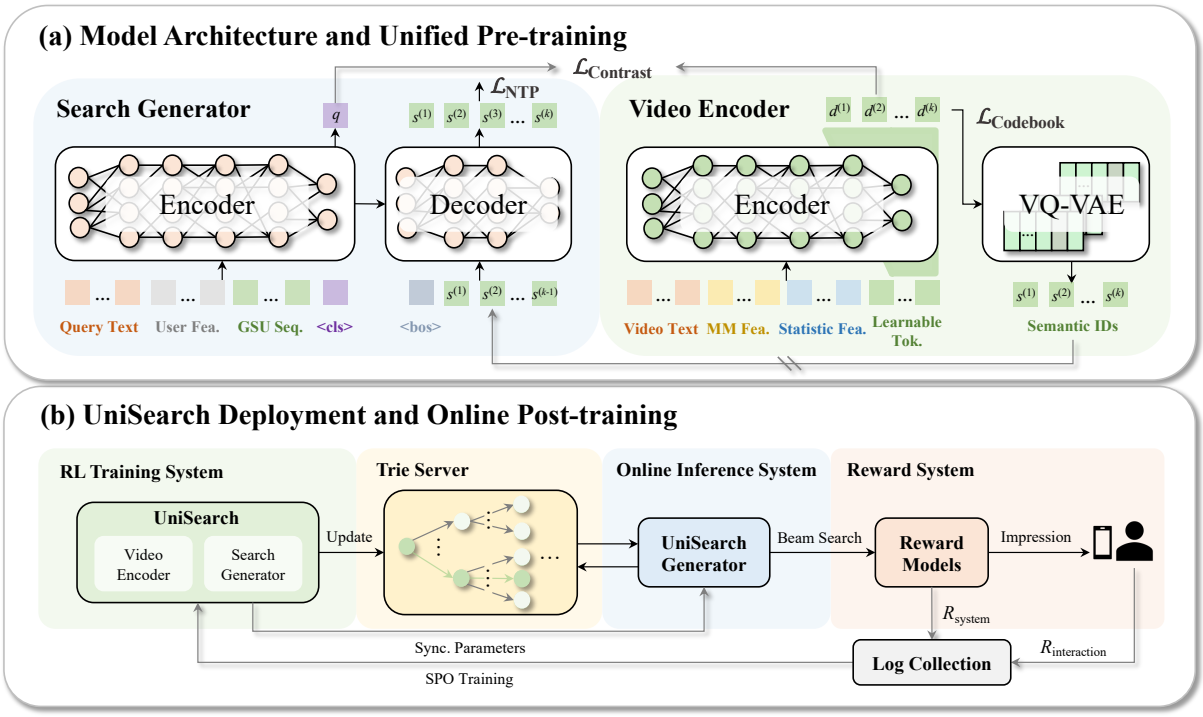

Figure 2: The overall architecture of UniSearch, where "Seq." denotes "Sequence", "Fea." stands for "Feature", "Learned Tok." refers to learned tokens for documents, and "MM Fea." represents multimodal features.

$\text{sim}(q, d) = 1 - \|q - d\|_2^2$, 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., 2026), 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., 2025) 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 π_θ is the current policy under optimization, and π_{ref} is a frozen reference policy initialized from the pre-trained model. 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.

We emphasize that SPO follows the GRPO-style RL framework (Liu et al., 2024a) and is not a new

algorithm; its main novelty lies in the search-aware composite reward design and its integration into an online generative search loop. This formulation aligns UniSearch with both system-estimated and user-observed preferences while avoiding catastrophic deviation from the pre-trained model.

3.4 Inference and Deployment

To improve inference efficiency, UniSearch adopts a prefix tree (Trie) to constrain generation to valid semantic ID sequences. At each decoding step, UniSearch queries the Trie for feasible continuation tokens, preventing invalid paths and reducing decoding overhead. To support large-scale video corpora, the Trie combines prefix-tree traversal with binary search for memory-efficient lookup in a massive semantic ID space.

UniSearch has been deployed in real-world industrial search scenarios through a unified system (Figure 2(b)). An RL system continuously aligns the model with online preferences and synchronizes updated parameters to inference services. The online inference system serves real-time requests with TensorRT and KV caching for low-latency decoding, while the Trie server maintains valid semantic ID paths for constrained generation. A reward system evaluates ranking quality and user interaction signals to support continual optimization. Together, these components enable scalable, low-latency retrieval with continuous online improvement. More details are shown in Appendix B.

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 and D.

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 Appendix A.

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 , their reliance on multiple models and mismatched objectives 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.

Effect of Search Preference Optimization. Considering the resource consumption of online experiments, we further apply SPO 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 better aligned with user preferences.

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

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**

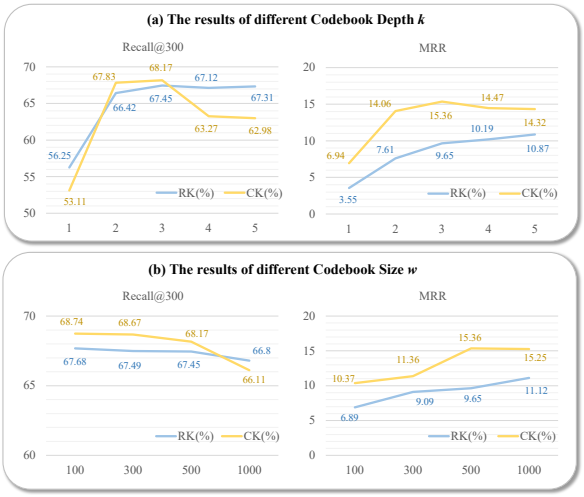

Figure 3: The results of different Codebook Depth k and Codebook Size w on RK and CK test set.

4.3 Ablation Study

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

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

(1) **Baseline** optimizes all tokens toward the same objective, weakening token diversity and yielding the worst Recall@300 and MRR. (2) **Baseline w/ CF** introduces a coarse-to-fine (CF) task, improving ranking by progressively learning from easier to harder objectives. However, correlated coarse and fine tasks may produce similar token representations, leading to the *path collapsing problem*, where multiple candidates share a decoding path, reducing discrimination and precision. (3) **Baseline w/ RCL** incorporates residual contrastive learning (RCL), progressively refining representations via residual summation. This encourages complementary token semantics, mitigates path collapse, and substantially improves MRR.

Our full UniSearch combines CF and RCL: CF provides progressive supervision, while the residual design preserves token diversity and prevents collapse. Together, they improve end-to-end discretization, reduce clustering loss, and achieve the best performance.

Ablation on Codebook Depth and Size. Codebook depth and size are key hyperparameters affecting retrieval quality and efficiency.

(1) **Depth.** Fixing the codebook size at 512, Figure 3(a) shows that performance saturates when depth $k > 3$, with limited gains in Recall@300 and MRR. Deeper codebooks may increase candidate-space dispersion, weaken generalization, and introduce higher decoding latency. We therefore set $k = 3$. (2) **Size.** Fixing depth at $k = 3$, Figure 3(b) shows divergent trends: Recall@300 decreases as size increases, while MRR steadily improves. Larger codebooks enable finer semantic partitioning, improving ranking precision but reducing recall due to fragmented candidate coverage. We set the size to $w = 512$ to balance recall and precision.

Reward Design and Comparison. To verify that SPO gains arise from the search-aware compos-

Table 3: Ablation of reward design and comparison with DPO on UniSearch-6.

Variant	MRR-CK ↑	MRR-RK ↑
w/o RL	68.17	15.36
DPO	68.21	15.57
SPO (γ_1=1.0, γ_2=0.0)	68.33	15.90
SPO (γ_1=0.0, γ_2=1.0)	66.81	14.11
SPO (γ_1=0.75, γ_2=0.25)	**68.48**	**16.04**

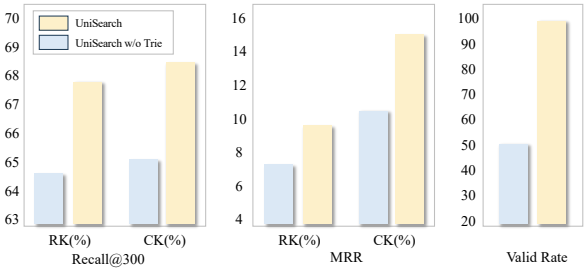

Figure 4: The results of incorporating the Trie on RK and CK test set, and Valid Rates of path to candidates.

ite reward rather than the RL algorithm itself, we ablate reward weights (γ_1, γ_2) in Eq. 7 and compare against DPO (Rafailov et al., 2023) using the same UniSearch-6L and online preference data (Table 3). (1) Combining system-estimated and user-interaction rewards (γ_1=0.75, γ_2=0.25) outperforms either signal alone and DPO. (2) Using only interaction rewards (γ_1=0, γ_2=1) degrades below the no-RL baseline, likely due to reward hacking on noisy short-horizon clicks.

Impact of the Trie Constraint. Generated path validity (*i.e.*, whether paths correspond to real candidates) is critical for system reliability. Unconstrained generation may produce invalid paths and empty results. Figure 4 shows that the Trie significantly improves Recall@300 and MRR, while increasing path validity from 51.3% to 99.8%. This ensures that nearly all generated results map to valid candidates, improving reliability and efficiency. In practice, the Trie must also be updated dynamically to reflect changes in live streams and candidate identifiers.

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, UniSearch (6 Layer) fully replaces the original multi-stage cascading architecture. For short-video search, we adopt progressive deployment by introducing UniSearch (12 Layer) as an additional retrieval source in ranking while gradually phasing out existing recall sources. We conduct a 5-day online A/B test, where both control and experimental groups receive 10% of real search traffic. Due to commercial sensitivity, absolute online metrics and latency are omitted. All reported improvements are statistically significant ($p < 0.05$) and consistently validated across multiple testing windows.

Table 4: 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 reflecting user satisfaction and engagement; results are summarized in Table 4.

(1) **Live search.** UniSearch improves total play counts (TPC) by +3.31%, page click-through rate (CTR) by +0.202%, while reducing change query rate (CQR) and position of first click (PFC) by -0.382% and -0.107%, respectively.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. Additional analysis is provided in Appendix C. (2) **Short-video search.** UniSearch improves video playback duration (VPD) by +0.213%, page view depth (PVD) by +0.993%, and long-play count per page view (LPC) by +0.830%, while reducing CQR by -0.602%. These results indicate improved intent satisfaction and deeper content exploration.

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

UniSearch is an early step toward unified generative retrieval, with three main limitations: *tokenizer scalability*, *limited interpretability*, and *point-wise decoding*.

Deployment Scope and Tokenizer Scalability. At present, end-to-end generative search is better suited to vertical scenarios or complementary retrieval rather than fully replacing multi-stage cascaded systems. The main bottleneck is tokenizer capacity: as corpus scale and heterogeneity grow, the codebook must preserve fine-grained distinctions while remaining compact for low-latency decoding. Scaling tokenization to *billion-scale heterogeneous corpora* remains an open challenge.

Interpretability. Unlike cascaded systems with intermediate ranking signals, UniSearch offers limited transparency. We currently analyze failures through *PID-to-SID stability*, *PID collisions*, and query-to-SID decoding behavior, but diagnosing individual failures remains more difficult than in cascaded pipelines.

Point-wise Generation. UniSearch generates candidates independently via beam search, which may reduce result diversity. Future work includes *list-wise decoding* and finer-grained reward design to jointly optimize diversity and ranking quality.

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*, pages 937–947.

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, Mingyue Cheng, Chenyi Lei, Yuqing Ding, and Han Li. 2026. Onesug: The unified end-to-end generative framework for e-commerce query suggestion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 14774–14782.

Ruidong Han, Bin Yin, Shangyu Chen, He Jiang, Fei Jiang, Xiang Li, Chi Ma, Mincong Huang, Xiaoguang Li, Chunzhen Jing, and 1 others. 2025. Mtgr: Industrial-scale generative recommendation framework in meituan. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 5731–5738.

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, 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*.

Enze Liu, Bowen Zheng, Cheng Ling, Lantao Hu, Han Li, and Wayne Xin Zhao. 2025. Generative recommender with end-to-end learnable item tokenization. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 729–739.

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. In *International Conference on Learning Representations*, volume 2024, pages 51772–51783.

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.

Ming Pang, Chunyuan Yuan, Xiaoyu He, Zheng Fang, Donghao Xie, Fanyi Qu, Xue Jiang, Changping Peng, Zhangang Lin, Zheng Luo, and 1 others. 2025. Generative retrieval and alignment model: A new paradigm for e-commerce retrieval. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 413–421.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, pages 53728–53741.

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. In *Advances in Neural Information Processing Systems*, volume 36, pages 10299–10315.

Weiwei Sun, Lingyong Yan, Zheng Chen, Shuaiqiang Wang, Haichao Zhu, Pengjie Ren, Zhumin Chen, Dawei Yin, Maarten Rijke, and Zhaochun Ren. 2023. Learning to tokenize for generative retrieval. In *Advances in Neural Information Processing Systems*, volume 36, pages 46345–46361.

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. In *Advances in Neural Information Processing Systems*, volume 35, pages 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 Koray Kavukcuoglu. 2017. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, volume 30.

Zhibo Wang, Xiaoze Jiang, Zhiheng Qin, and Enyun Yu. 2025. 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*, pages 5018–5028.

Ao Xie, Jiahui Chen, Quanzhi Zhu, Xiaoze Jiang, Zhiheng Qin, Enyun Yu, and Han Li. 2026. Crops: Improving dense retrieval with cross-perspective positive samples in short-video search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 34061–34069.

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*, 17(6):4558–4577.

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. 2025. 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.

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, Jiayuan He, and 1 others. 2024. Actions speak louder than words: Trillion-parameter sequential transducers for generative recommendations. In *International Conference on Machine Learning*, pages 58484–58509.

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.

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*, pages 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. 2025. Addressing representation collapse in vector quantized models with one linear layer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22968–22977.

A Implementation Details

Datasets. For live-streaming search, we collect 80M 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.

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:

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

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.

Loss-balancing Hyperparameters. For the unified pre-training loss in Eq. 6 we set λ_1=1.0 for the residual contrastive loss, λ_2=0.25 for the codebook loss, and λ_3=1.0 for the next-token-prediction loss. Inside the codebook loss in Eq. 4 we use the standard VQ-VAE setting α_1=1.0, α_2=0.25. The contrastive temperature is τ=0.07. For SPO (Eq. 7) we use γ_1=0.75, γ_2=0.25, KL coefficient β=0.04, and group size G=8,. The reweighting factor w_i in Eq. 5 is set proportional to the graded relevance label l_i and clipped to $[0.5, 2.0]$. These values were selected on a held-out subset of the training period and kept fixed across all reported experiments.

B Additional Deployment Details

System Latency. Despite generating semantic IDs autoregressively, UniSearch is competitive with the cascaded baseline in end-to-end serving cost. In the live-streaming production environment, the original cascaded pipeline takes about 110 ms per request (recall ∼15 ms, pre-ranking ∼30 ms, ranking ∼65 ms), whereas UniSearch (UniSearch-6, deployed configuration) takes about 95 ms per request, a 13.6% reduction. The savings come from removing the inter-stage data exchange of the cascade and from TensorRT-accelerated inference with KV caching combined with Trie-constrained decoding.

Online Indexing and Trie Update. For newly published videos and newly started live streams, the Video Encoder extracts multimodal features online and maps them into the existing codebook to obtain semantic IDs without retraining. The

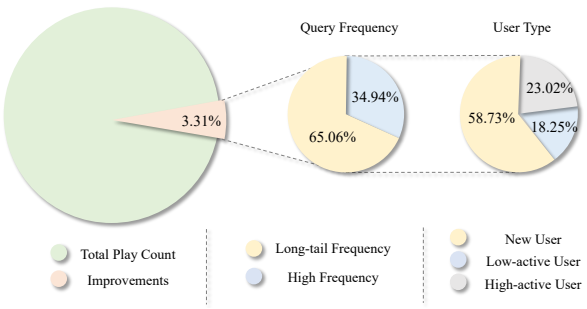

Figure 5: The analysis of the improvements in Total Play Count (TPC) from query frequency and user type.

Trie is incrementally updated so that fresh content becomes searchable in near real time.

Decoding Latency Control. To bound the tail latency of autoregressive decoding, we cap generation at 25 tokens. Since queries in live and short-video search are typically short, truncated cases mainly arise from malformed inputs, resulting in negligible impact on perceived search quality.

Trie Implementation. To improve scalability for massive semantic ID spaces, the Trie integrates prefix-tree traversal with binary search, reducing memory overhead while maintaining efficient constrained decoding.

C 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.

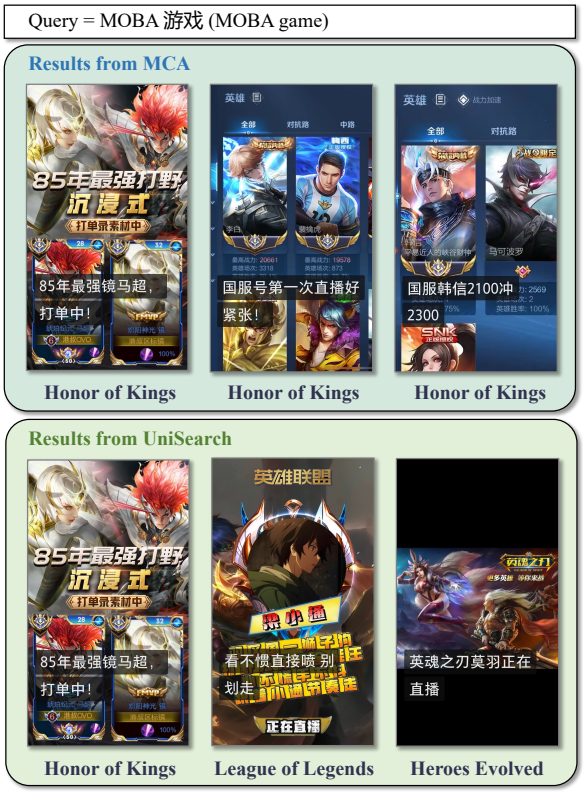

Figure 6: Search results comparison between the Multi-stage Cascading Architecture (MCA) and our UniSearch.

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 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.

D Reproducibility Statement.

Our offline and online experiments are conducted on proprietary search logs from a Chinese short-video platform that contain user interactions, personalized features, and live-stream metadata, none of which can be released for privacy and business reasons. Public academic video-retrieval benchmarks (e.g., MSR-VTT, DiDeMo, MSVD) lack user interactions, personalization signals, and live-stream metadata, so they are not appropriate proxies for our setting; conducting industrial generative search research on private production data is therefore common practice (Chen et al., 2025; Guo et al., 2026; Xie et al., 2026). To partially mitigate this limitation we report all training and SPO hyperparameters above, describe deployment configurations explicitly, and plan to release a desensitized subset of the training data along with model code and weights for several UniSearch variants upon publication.