OneSearch-V2: A Latent Reasoning Enhanced Self-distillation Generative Search Framework

Ben Chen[*,†], **Siyuan Wang**[*], **Yufei Ma**[*], **Zihan Liang**[*],
Han Li, **Chenyi Lei**, **Wenwu Ou**, **Kun Gai**

Kuaishou Technology
benchen4395@gmail.com

Abstract

Generative Retrieval (GR) has emerged as a promising paradigm for modern search systems, offering end-to-end joint optimization and lower serving overhead compared to multi-stage cascaded architectures. OneSearch, as a representative industrially deployed generative framework, has delivered substantial commercial benefits. However, three key limitations constrain further improvement: shallow understanding of complex queries, insufficient personalized intent reasoning over user context, and a separately trained reward model that adapts slowly to emerging queries and is prone to sampling bias and reward hacking. To address these challenges, we propose **OneSearch-V2** with three key innovations: (1) a thought-augmented query understanding module that generates compact keyword-based chains-of-thought (CoTs), overcoming the shallow-matching limitation of single-pass SID generation; (2) a reasoning-internalized self-distillation pipeline that encodes keyword-guided reasoning into the existing model weights via information-asymmetric supervision, without any extra parameters, special tokens, or inference-time CoT generation; (3) a behavior-feedback preference alignment system that replaces the separate reward model with composite rewards built from real user interactions, and introduces a token-position marginal advantage (TPMA) mechanism for position-aware credit assignment over hierarchical SID sequences. Extensive offline evaluations demonstrate OneSearch-V2's strong query understanding and personalized intent modeling capabilities. Online A/B tests further validate its business effectiveness, yielding +3.98% item CTR, +1.17% PV CTR, +2.90% PV CVR, +2.07% buyer volume, and +2.11% order volume. Manual evaluation further confirms gains in search experience quality, with +1.37% in page good rate and +1.65% in query-item relevance. Importantly, OneSearch-V2 achieves these gains *without any additional inference cost or serving latency*, while also mitigating information bubbles and long-tail sparsity.

1 Introduction

Large language models (LLMs) are rapidly reshaping search and recommendation systems. Compared to applying or distilling LLMs into isolated modules such as recall, relevance, and ranking [36, 41, 6, 26], an increasingly promising direction is to replace the multi-stage cascaded pipeline with end-to-end generative retrieval, as exemplified by OneRec for video recommendation [5, 40], OneSug for query suggestion [7], OneLoc for local life services [29], and MTGR for advertisement [8]. For e-commerce search, which requires jointly modeling query-item relevance and user-item collaborative

[*]Equal Contribution.
[†]Corresponding Author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

preference, the representative generative framework is OneSearch [4], which directly optimizes the final business objective at substantially lower computational cost than cascaded pipelines.

However, as user preferences grow diverse and queries become more complex, three limitations increasingly constrain OneSearch's performance:

1) *Insufficient understanding of complex queries.* Typical short queries (2-3 keywords) often underspecify the retrieval target, e.g., "indoor fitness equipment" could map to treadmills, dumbbells, yoga mats, or resistance bands. Long-tail queries further suffer from lexical disparity (e.g., "relieve fatigue, no supplements") and intent ambiguity (e.g., "what swimming essentials?"). These complex queries demand stronger semantic understanding and reasoning capabilities. However, OneSearch takes the raw query as input and generates target items in a single forward pass under strict latency constraints. Although it incorporates category-level supervision during training (SFT Stage 1), the model still lacks the capacity for deep comprehension of these ambiguous queries.

2) *Insufficient personalized intent reasoning over user context.* Effective e-commerce search requires reasoning over user-specific context, yet OneSearch relies on co-occurrence patterns and log-fitting objectives that produce shallow matching. For example, a user who recently browsed minimalist home decor and searches "seasonal fresh flowers" likely wants single-color, simple arrangements rather than the overall best-selling bouquets; the model should reason about current season, user style, and available species jointly, rather than simply recalling co-occurring best-sellers.

3) *Slow-adapting reward system with distributional bias.* OneSearch's reward model is trained offline on historical behavior logs and refreshed only periodically, so it adapts slowly to newly emerging queries and is susceptible to sampling bias and reward hacking—pathologies that jointly cause the generative policy to overfit narrow historical preferences and reinforce the long-tail distributional bias inherent in the search system.

To bridge these gaps, we introduce **OneSearch-V2**, a generative search framework enhanced by latent reasoning and self-distillation (comparison with V1 in Appendix A), with three key contributions:

1) **Thought-augmented query understanding module.** We employ LLMs to generate explicit CoT reasoning for each $\langle \text{query}, \text{user} \rangle$ pair and distill it into compact keyword-based CoTs that maximize information density while preserving critical content (cf. [26]). These serve as semantic alignment training data and privileged teacher-side input for the self-distillation pipeline.

2) **Reasoning-internalized self-distillation pipeline.** We propose a self-distillation mechanism that endows the generative model with latent reasoning capabilities without adding trainable parameters, special tokens, or decoding-time overhead, unlike existing latent-reasoning methods [22, 9, 37]. The reasoning ability is encoded entirely into existing model weights and internalized as intuition. To mitigate the representation instability caused by information asymmetry between teacher and student, we further combine R-Drop for output consistency and FGM for input-space robustness.

3) **Behavior-feedback preference alignment system.** We replace the reward-model-based pipeline with direct alignment driven by real user feedback. Composite rewards from query-item relevance, posterior CTR, and click/order signals balance semantic matching and conversion, while a token-position marginal advantage (TPMA) mechanism provides position-aware credit assignment over hierarchical SID sequences, enabling flexible reward tuning and real-time updates.

Offline evaluations confirm OneSearch-V2 achieves substantially higher recall and ranking than V1. Online A/B tests yield +3.98% item CTR, +1.17% PV CTR, +2.90% PV CVR, +2.07% buyer volume, and +2.11% order volume, with manual evaluation showing +1.37% page good rate and +1.65% query-item relevance. Crucially, these gains incur no additional inference cost or latency while mitigating information bubbles and long-tail sparsity.

2 Methodology

OneSearch-V2 inherits the SID tokenization of OneSearch-V1, which uses keyword-based hierarchical quantization on unimodal textual representations (see Appendix C for a detailed justification). In this section, we focus on the three core components of V2, as illustrated in Fig. 1: a thought-augmented query understanding module (§2.1), a reasoning-internalized self-distillation pipeline (§2.2), and a behavior-feedback preference alignment system (§2.3). Table 1 summarizes how these components are integrated into the overall four-stage training procedure of V2.

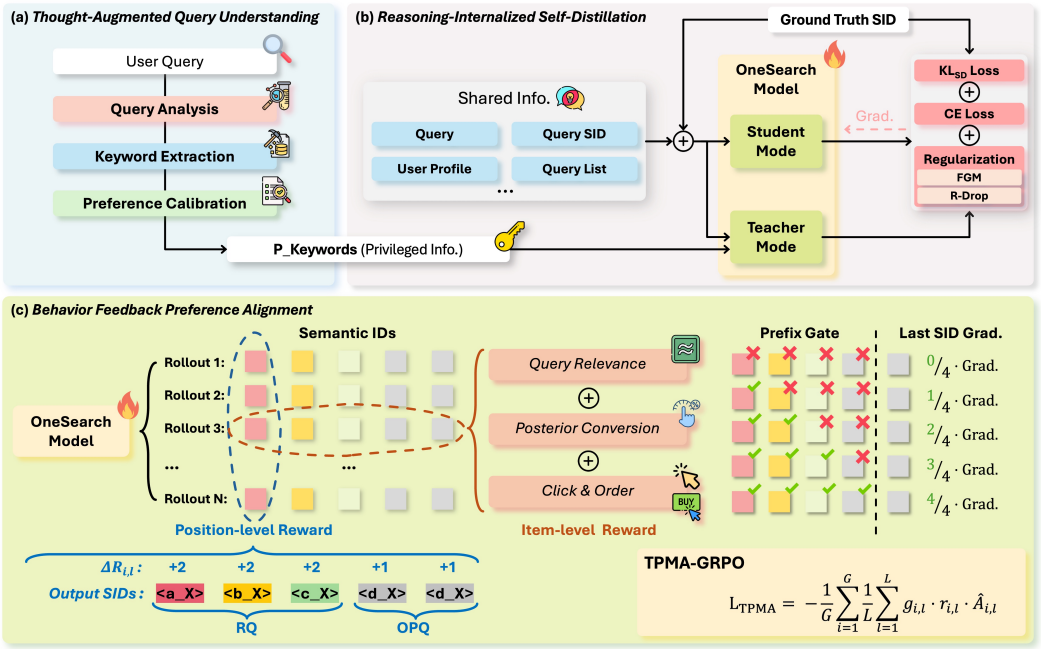

Figure 1: The overall framework of OneSearch-V2, comprising (a) a thought-augmented query understanding module, (b) a reasoning-internalized self-distillation pipeline, and (c) a behavior-feedback preference alignment system.

Table 1: Overall training procedure of OneSearch-V2, including four coherent stages.

Stage	Objective	Training Components
SFT Stage 1	Semantic alignment	$q/i \leftrightarrow$ SID; $q/i \mapsto$ category; SID $\mapsto$ category; CoT tasks
SFT Stage 2	Co-occurrence	$q \leftrightarrow i$; $\text{SID}_q \leftrightarrow \text{SID}_i$
SFT Stage 3	User personalization	$\{\text{uid}, q, \text{SID}_q, \text{Seq}_q, \text{Seq}_{\text{short}}, \text{Seq}^{\text{emb}}_{\text{long}}, (\text{keywords})\} \mapsto \text{SID}_i$
RL Stage	Preference alignment	$\{\text{user}, q, \text{seq. feat.}, \text{item}_{\text{clk/order}}, \text{item}_{\text{rollout}}\} \mapsto$ rank score

2.1 Thought-Augmented Query Understanding

E-commerce queries pose two distinct challenges under OneSearch-V1. *Head queries* with underspecified intent (e.g., "indoor fitness equipment") induce a broad candidate space (*"which to retrieve"*), while *tail queries* with heterogeneous formulations (e.g., "relieve fatigue, no supplements") suffer from intent ambiguity and sparse signals (*"what can be retrieved"*). Although OneSearch-V1 partially alleviates these issues through aligned representations, it still lacks the capacity for deep query comprehension. We address this challenge with a thought-augmented query understanding schema: an LLM first produces explicit CoT reasoning for each query, which is then distilled into compact, information-dense keyword sets under intent, category, and attribute constraints. These keyword-based CoTs serve as supplementary semantic signals during training.

2.1.1 Keyword-based CoT

We formulate a three-step pipeline, with detailed diagram and prompt templates in Appendix H:

1. **Query Analysis.** We formulate an analysis scheme comprising four components. (i) *Intent understanding*, which identifies the primary retrieval target (merchandise, shop, or livestream host); (ii) *category identification*, which performs taxonomy matching from coarse to fine granularity; (iii) *attribute recognition*, which extracts the attribute key-value pairs from the query; and (iv) *topic recommendation*, which speculates potential candidate topics satisfying the user's need.

2. **Keyword Extraction.** For merchandise-intent queries, we extract keywords from the full analysis under intent, category, and attribute constraints, refine them via synonym merging and redundancy removal, and rank them by popularity. Other intents are handled by dedicated matching engines.
3. **Preference Calibration.** Leveraging user profile and behavioral signals (e.g., recent queries and interacted items), the LLM filters or augments the keyword set toward individual preferences. Session-interacted items are injected as weak supervision during training, ensuring ground-truth items are reflected. This yields personalized keywords as privileged teacher-side input for §2.2.

2.1.2 Training Paradigm Refinement

The three-step pipeline produces two types of keyword-based CoTs: query-level $\langle \text{query}, \text{keywords} \rangle$ from Step 2 and personalized $\langle \text{query}, \text{user}, \text{keywords} \rangle$ from Step 3. We integrate them into SFT Stage 1 of OneSearch-V1 (Table 1) as four CoT tasks: (i) query $\mapsto$ keywords, (ii) $\text{SID}_q \mapsto \text{SID}_{\text{kw}}$, (iii) $\langle \text{query}, \text{user} \rangle \mapsto$ P_keywords, and (iv) $\langle \text{SID}_q, \text{user} \rangle \mapsto \text{SID}_{\text{P_kw}}$. Tasks (i)(ii) expose the model to query-level semantic decomposition, while tasks (iii)(iv) further inject personalization signals. CoT tasks are used only during training. LLM-based CoT generation is performed asynchronously per distinct query (or $\langle \text{query}, \text{user} \rangle$ pair) and cached for streaming training; at inference the model receives only the raw query and generates SIDs in a single pass with no additional latency.

2.2 Reasoning-Internalized Self-Distillation

An intuitive approach would be to train OneSearch to first generate keywords and then candidate SIDs. However, the representational heterogeneity between discrete SIDs and textual keywords poses a challenge for small-scale generative models: as shown in §3.2 (Table 3), this direct CoT substantially degrades performance. A simpler alternative is to inject the keyword-based CoTs as supplementary signals at the input layer—a retrieval-augmented (RAG) variant that does improve retrieval and ranking effectiveness. However, it has two practical drawbacks. First, every request must invoke the thought-augmented module, causing non-trivial latency under strict online constraints. Second, keyword coverage is bounded: searching "hotel essentials" with keywords limited to towels and toothbrushes may miss disposable slippers. This raises a key question: *can we retain the reasoning benefits of keyword-based CoTs without the inference cost or coverage limits?*

We address this by proposing a **reasoning-internalized self-distillation** mechanism that transfers the explicit reasoning capability into the model's parameters, converting deliberate, keyword-guided CoTs into fast, intuition-like inference. Unlike prior latent reasoning approaches that introduce additional trainable tokens [9] or continuous thought vectors [22, 37] into the decoding process, our method requires *no architectural modification, no extra parameters, and no inference overhead*. The reasoning ability is encoded entirely into existing model weights through an information-asymmetric distillation pipeline, detailed below.

2.2.1 Self-Distillation Formulation

Our self-distillation leverages *information asymmetry*, where the teacher observes a superset of the student's input, while the student learns to fit the teacher's output distribution despite less information.

Let $\mathcal{M}_\theta$ denote the model parameterized by θ, shared by teacher and student, which eliminates the memory overhead of a separate teacher in conventional knowledge distillation. For each training sample, the teacher and student share all features except the keyword-based CoTs $\mathbf{kw}$ from §2.1.1:

$$x^{(T)} = (x^{(S)}, \mathbf{kw}), \quad \text{where } x^{(S)} = (\text{uid},\ q,\ \text{SID}_q,\ \text{Seq}_q,\ \text{Seq}_{\text{short}},\ \text{Seq}^{\text{emb}}_{\text{long}}). \tag{1}$$

Both the teacher and student produce output logits over the target sequence $y = (y_1, \ldots, y_L)$:

$$z^{(T)} = \mathcal{M}_\theta(y \mid x^{(T)}), \quad z^{(S)} = \mathcal{M}_\theta(y \mid x^{(S)}). \tag{2}$$

Since θ is shared, the difference between $z^{(T)}$ and $z^{(S)}$ arises solely from the presence or absence of keyword information in the input. The distillation objective encourages the student to close this gap:

$$\mathcal{L}_{\text{KL}} = \frac{1}{|\mathcal{V}|} \sum_{t \in \mathcal{V}} \text{KL}\Big(\text{softmax}\big(z_t^{(T)}/\tau\big) \,\Big\|\, \text{softmax}\big(z_t^{(S)}/\tau\big)\Big) \cdot \tau^2, \tag{3}$$

where $\mathcal{V}$ is the set of valid (non-padding) token positions, τ is the distillation temperature, and gradients flow only through the student as teacher logits are detached.

The base training objective for the student combines the cross-entropy loss with the distillation signal:

$$\mathcal{L}_{\text{base}} = \mathcal{L}_{\text{CE}}(z^{(S)}, y) + \alpha_{\text{KL}} \cdot \mathcal{L}_{\text{KL}}, \tag{4}$$

where α_{KL} controls the relative strength of the distillation signal.

2.2.2 Mitigating Representation Instability

The information asymmetry between teacher and student sharpens the loss surface around keyword-absent inputs, making small embedding perturbations cause large output-distribution shifts. This manifests as inconsistent predictions under dropout noise and brittle outputs under input perturbations. We address them with two complementary regularizers.

Output consistency via R-Drop. To stabilize the student's predictions under dropout noise, we apply R-Drop regularization [15]: two forward passes are performed with independent dropout masks, and a symmetric KL penalty $\mathcal{L}_{\text{R-Drop}}$ pulls their output distributions together (details in Appendix D.1). This discourages fragile pathways that depend on specific dropout realizations.

Input robustness via FGM. The Fast Gradient Method (FGM) [19] computes an adversarial perturbation r_{adv} along the gradient of the current loss, and performs a second forward-backward pass on perturbed embeddings $e + r_{\text{adv}}$ to produce auxiliary loss $\mathcal{L}_{\text{adv}}$ (full expression in Appendix D.2), smoothing the input space and reducing sharp decision boundaries.

Total objective. Combining the distillation loss (Eq. 3), R-Drop, and FGM, the student objective is:

$$\mathcal{L}_{\text{SDFT}} = \mathcal{L}_{\text{CE}} + \alpha_{\text{KL}}\mathcal{L}_{\text{KL}} + \alpha_{\text{R}}\mathcal{L}_{\text{R-Drop}} + \mathcal{L}_{\text{adv}}, \tag{5}$$

where $\mathcal{L}_{\text{adv}} = \mathcal{L}_{\text{CE}}(\tilde{z}^{(S)}, y) + \alpha_{\text{KL}}\tilde{\mathcal{L}}_{\text{KL}} + \alpha_{\text{R}}\tilde{\mathcal{L}}_{\text{R-Drop}}$ is the same three-term objective re-evaluated on the FGM-perturbed embeddings $e + r_{\text{adv}}$, with $\tilde{\cdot}$ denoting quantities computed on perturbed inputs. We additionally replace the cross-entropy with focal loss [17] to counter the long-tail class imbalance in the SID vocabulary; this is an orthogonal improvement ablated in §3.2.

2.3 Behavior-Feedback Preference Alignment

Motivated by the limitations of RM-based RL (sampling bias, reward hacking [5, 40]) and the hierarchical nature of SID generation, OneSearch-V2 replaces the separately trained RM with a direct behavior-feedback alignment system that (1) constructs composite rewards from real user interactions, (2) introduces a **token-position marginal advantage (TPMA)** mechanism for position-aware credit assignment, and (3) supports streaming updates for newly emerging queries.

2.3.1 Composite Reward Design

For each rollout o_i, we compute an item-level reward $R_{\text{item}}(o_i) = R_{\text{Rel}}(o_i) + R_{\text{CTR}}(o_i) + R_{\text{C\&O}}(o_i)$ from three complementary signals. **(i)** R_{Rel}: the existing relevance model scores each item into four tiers (0-Irrelevant to 3-Excellent). **(ii)** R_{CTR}: the calibrated CTR from OneSearch-V1, clipped to $[0, 1]$, preventing high-CTR but irrelevant items from dominating. **(iii)** $R_{\text{C\&O}}$: a behavior signal that rewards clicked/ordered items with V_c/V_o ($V_o > V_c$), and 0 otherwise. This additive design in $R_{\text{item}}(o_i)$ balances relevance and conversion while avoiding signal sparsity.

2.3.2 Token-Position Marginal Advantage

Standard GRPO [21] computes a single sequence-level advantage $\hat{A}_i$ per rollout applied uniformly to every token (details in Appendix E). However, SID have hierarchical semantics: the first token encodes the coarsest category, while later tokens refine finer details. A rollout with a correct first token but wrong second differs fundamentally from the reverse, yet GRPO assigns identical credit, conflating positional contributions and weakening the signal for fine-grained tokens. To address GRPO's credit assignment issue, we propose **TPMA-GRPO**, which breaks rewards into position-level marginal contributions and controls gradient flow based on prefix correctness.

Prefix reward. For each rollout o_i, we define the *prefix reward* at position l as the maximum weighted cumulative match against any ground-truth target SID in $\mathcal{T} = \mathcal{S}_{\text{click}} \cup \mathcal{S}_{\text{order}}$:

$$R_{i,l} = \max_{t \in \mathcal{T}} \sum_{k=1}^{l} w_k \cdot \mathbf{1}[o_i^k = t^k], \quad w_k = \begin{cases} 2, & k \leq 3 \text{ (RQ tokens)}, \\ 1, & k > 3 \text{ (OPQ tokens)}, \end{cases} \tag{6}$$

where o_i^k and t^k denote the k-th token of rollout o_i and target t respectively. The position weight w_k captures SID hierarchy: the three RQ tokens (positions 1–3) encode coarse-grained category information and carry greater semantic weight than the two OPQ tokens (positions 4–5) that capture fine-grained attribute residuals, guiding the GR model to prioritize query-conforming items.

Position-level advantage. Instead of GRPO's single sequence-level advantage, we normalize $\Delta R_{i,l}$ independently at each position across the G rollouts within a group:

$$\hat{A}_{i,l} = \frac{\Delta R_{i,l} - \text{mean}_{j \in [G]}(\Delta R_{j,l})}{\text{std}_{j \in [G]}(\Delta R_{j,l}) + \delta}. \tag{7}$$

Each token is thus credited only against the same position of other rollouts, providing precise attribution across the coarse-to-fine hierarchy.

Prefix gate. Gradient signals for later positions are meaningful only when the prefix is correct: optimizing a fine-grained token within an already-wrong branch is counterproductive. We introduce a *prefix gate* $g_{i,l} \in [0, 1]$ that scales gradients by prefix accuracy:

$$g_{i,l} = \begin{cases} 1, & l = 1, \\ R_{i,l-1} \,/\, W_{l-1}, & l \geq 2, \end{cases} \qquad W_{l-1} = \sum_{k=1}^{l-1} w_k. \tag{8}$$

When the prefix matches perfectly ($R_{i,l-1} = W_{l-1}$), the gate opens fully ($g = 1$); if incorrect ($R_{i,l-1} = 0$), the gate closes ($g = 0$), suppressing gradients for later tokens. This induces a natural curriculum: the model first learns correct coarse tokens before being trained on fine-grained ones.

TPMA-GRPO loss. To incorporate conversion-oriented signals, we combine the position-level advantage with a group-normalized item-level advantage:

$$\hat{A}_{i,l}^{\text{final}} = \hat{A}_{i,l} + w_{\text{item}} \cdot \hat{A}_i^{\text{item}}, \quad \hat{A}_i^{\text{item}} = \frac{R_{\text{item}}(o_i) - \text{mean}_j(R_{\text{item}}(o_j))}{\text{std}_j(R_{\text{item}}(o_j)) + \delta}, \tag{9}$$

where w_{item} balances prefix matching and conversion signals, enabling the model to learn both *what* to generate (via TPMA) and *how valuable* the generation is (via R_{item}). The final loss integrates the combined advantage, prefix gate, and per-token importance ratio $r_{i,l}$:

$$\mathcal{L}_{\text{TPMA}} = -\frac{1}{G} \sum_{i=1}^{G} \frac{1}{L} \sum_{l=1}^{L} g_{i,l} \cdot r_{i,l} \cdot \hat{A}_{i,l}^{\text{final}}. \tag{10}$$

Unlike GRPO, we skip explicit clipping of $r_{i,l}$ as the prefix gate inherently controls gradient magnitude. When $g_{i,l} \to 0$, gradients vanish, avoiding explosion. This approach is similar to GBPO in OneRec-V2 [40], which achieves stability via structural gating instead of truncation.

3 Experiments

We evaluate OneSearch-V2 with offline benchmarks (§3.1), controlled ablations (§3.2), large-scale online A/B tests (§3.3), and further analysis of deployment behavior (Appendix G).

Datasets and baselines. We train on three months of clicked and purchased ⟨query, item⟩ pairs from a mall search platform, evaluating on the most recent day's logs, with 30K page views, 30K clicks, 7,229 orders (detailed in §3.1). For offline comparison, all models share the same pretrained backbone and training data as OneSearch-V1; online A/B tests compare against the deployed V1.

Evaluation metrics. We report Hit Rate (HR@10) and Mean Reciprocal Rank (MRR@10) over the top-10 generated items per page view. Results are averaged across all test samples.

Table 2: Incremental contribution of each component on the offline test set. "\+" denotes cumulatively adding a component on top of all rows above; "+" denotes adding a single alternative to the SFT-complete model (the row above the horizontal separator). Best results in bold, second-best underlined.

Method	Order (7,229)		Click (30k)	
	HR@10	MRR@10	HR@10	MRR@10
OneSearch (V1 baseline)	0.2046	0.0985	0.2231	0.0728
\+ CoT tasks	0.2094	0.1008	0.2266	0.0731
\+ self-distillation	0.2163	0.1017	0.2398	0.0757
\+ R-Drop	0.2168	0.1045	0.2398	0.0760
\+ FGM	0.2180	0.1047	0.2422	0.0766
\+ focal loss	0.2214	0.1048	0.2471	0.0788
+ PARS (V1 reward model)	0.2221	0.1067	<u>0.2538</u>	0.0809
+ vanilla GRPO	0.2248	0.1106	0.2481	0.0798
+ TPMA (ours)	<u>0.2265</u>	<u>0.1136</u>	0.2498	<u>0.0815</u>
OneSearch-V2 (DPO + TPMA)	**0.2314**	**0.1151**	**0.2568**	**0.0833**

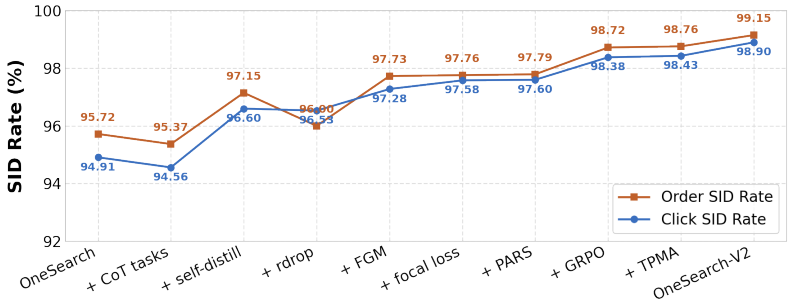

Figure 2: The SID rate (the fraction of top-10 generated SIDs that map to valid items) across the incremental variants of OneSearch-V2, evaluated under unconstrained beam decoding (no trie search).

Implementation details. To ensure architecture-agnostic effectiveness, we test three pretrained backbones: BART-B [13] (encoder-decoder), GPT-2 [3] (decoder-only), and Qwen3-0.6B [31] (decoder-only). Unless stated otherwise, main results use BART-B, with cross-architecture results in Appendix F. We use Qwen3-32B [31] to generate keyword-based CoTs. Beam search size is 512 for quality-latency balance. Batch size is 512 for standard SFT and 2,048 for listwise DPO (due to higher sample consumption). For self-distillation (SFT Stage 3), we use self-mode with shared teacher-student weights, distillation temperature $\tau = 1.0$, KL weight $\alpha_{\text{KL}} = 0.1$, and R-Drop coefficient $\alpha_{\text{R}} = 0.5$. FGM perturbation magnitude is $\epsilon = 0.6$, and focal loss parameters are $\alpha = 2$, $\gamma = 3$. For TPMA-GRPO, $V_o = 4$ and $V_c = 3$ (satisfying $V_o > V_c$ in §2.3.1) with a batch size of 256. Supervised training stages run sequentially, while the RL stage operates in a streaming fashion, refreshing user logs and keyword-based CoTs in near real-time.

3.1 Offline Performance

Semantic alignment and self-distillation (Table 2, upper block). Adding CoT tasks confirms that thought-augmented query understanding mitigates semantic ambiguity. Self-distillation delivers the largest gain in this block (+0.69% order HR@10, +1.32% click HR@10 over CoT tasks alone), confirming reasoning has been internalized into model weights. R-Drop, FGM, and focal loss each add consistent marginal improvements as complementary regularizers.

Preference alignment (Table 2, lower block). We compare three alignment strategies on the SFT model: PARS (Preference-Adaptive Reward System, the separately trained RM used in OneSearch-V1), vanilla GRPO, and TPMA. TPMA outperforms both PARS and vanilla GRPO on three of four metrics (+2.0% relative HR@10, +6.5% relative MRR@10 over PARS on orders). On click HR@10, TPMA is marginally below PARS (0.2498 vs. 0.2538), but the full OneSearch-V2 (DPO + TPMA)

Table 3: Effect of keyword-based CoT integration across query types. Upper block: adding four CoT tasks during SFT Stage 1. Lower block: injecting CoTs at SFT Stage 3 as *direct CoT* (generation) or *RAG* (input augmentation). Metrics: HR/MRR@10. Best results in bold.

Model	Order		Click		Head		Tail	
	HR	**MRR**	**HR**	**MRR**	**HR**	**MRR**	**HR**	**MRR**
Baseline	0.2046	0.0985	0.2231	0.0728	0.2362	0.0817	0.1952	0.0733
\+ CoT tasks	0.2094	0.1008	0.2266	0.0731	0.2419	0.0829	0.1963	0.0734
+ direct CoT	0.0898	0.0189	0.1013	0.0146	0.1116	0.0180	0.0809	0.0120
+ RAG	**0.2139**	**0.1011**	**0.2327**	**0.0743**	**0.2438**	**0.0845**	**0.1973**	**0.0779**

recovers this gap and achieves the best result across all metrics, confirming that position-aware credit assignment and real interaction rewards outperform a separately trained reward model.

Complementarity of DPO and GRPO. Listwise DPO [24] and TPMA-GRPO optimize complementary objectives: DPO refines the policy toward real preference pairs, while GRPO pushes generation toward reward-aligned regions. In online deployment, DPO is applied first to learn base preferences, followed by TPMA to balance multiple reward objectives. The final OneSearch-V2, combining all components, achieves the best performance across all metrics.

Generation validity. Fig. 2 shows the valid SID rate for each cumulative variant, with nearly every innovation improving results. The final model achieves 98.90% on click and 99.15% on order, demonstrating enhanced generation quality while maintaining structural validity.

3.2 Ablation Study

We ablate three questions: (1) do keyword-based CoT tasks help across query frequencies; (2) which components of self-distillation drive the gains; and (3) does self-distillation genuinely outperform alternative reasoning-internalization strategies.

CoT tasks across query frequency (Table 3). We define *head* as queries with daily PV ≥ 100 (combining the *top* and *middle* buckets of §G.3) and *tail* as PV < 100. CoT auxiliary tasks improve consistently across order, click, head, and tail subsets. In contrast, explicit CoT at inference time (+ *direct CoT*) collapses performance well below baseline (order HR@10: 0.2046 $\rightarrow$ 0.0898), consistent with findings that explicit reasoning steps can harm small generative models [34]. The RAG variant improves results but is impractical due to latency and bounded keyword coverage.

Components of self-distillation (Table 4). Self-distillation is the primary performance driver (+0.69% order HR@10, +1.32% click HR@10). The three regularizers yield moderate improvements individually but combine super-additively on top of self-distillation, addressing distinct instability facets: dropout-induced output variance (R-Drop), input-space brittleness (FGM), and long-tail class imbalance (focal loss).

Self-distillation vs. alternatives (Table 5). Three observations emerge. First, the self-distilled student SELF-DISTILL (S) outperforms the keyword-augmented teacher BASE (T) despite never seeing keywords at inference, confirming that reasoning is encoded into the weights rather than injected through input features. Second, SELF-DISTILL (S) slightly surpasses its own teacher-side evaluation SELF-DISTILL (T); in self-mode distillation teacher and student share parameters while gradients are driven entirely by the student's no-keyword loss, so optimization favors robustness under information-deficient conditions, allowing the student to generalize beyond the keyword-augmented teacher. Third, all four alternative internalization paradigms—special tokens, CODI-style hidden-state alignment, EMA-mode, and joint-mode—underperform our self-mode, suggesting that fully-shared weights with input-level asymmetry is the most effective paradigm for our generative search setting. A deeper analysis of why latent-token methods fall short is deferred to Appendix G.4.

Table 4: Individual contribution of R-Drop, FGM, and focal loss. Upper block: each regularizer added directly to the CoT-task baseline. Lower block: each added on top of self-distillation. The combined effect (self-distill + all three regularizers) is reported in Table 2.

Method	Order (7,229)		Click (30k)	
	HR@10	MRR@10	HR@10	MRR@10
Baseline (CoT tasks only)	0.2094	0.1008	0.2266	0.0731
+ R-Drop	0.2124	0.1020	0.2292	0.0733
+ FGM	0.2109	0.1011	0.2279	0.0732
+ focal loss	0.2074	0.1010	0.2237	0.0723
Self-distillation	0.2163	0.1017	0.2398	0.0757
+ R-Drop	0.2168	0.1045	0.2398	0.0760
+ FGM	0.2168	0.1050	0.2380	0.0757
+ focal loss	0.2161	0.1042	0.2385	0.0753

Table 5: Comparison of self-distillation against keyword-augmented baselines and alternative internalization strategies. "(S)" and "(T)" denote evaluation without and with keyword input, respectively. Upper block: direct supervision baselines. Middle block: alternative internalization paradigms evaluated on the student side. Lower block: our self-mode.

Method	Order (7,229)		Click (30k)	
	HR@10	MRR@10	HR@10	MRR@10
Base (S) *(no keywords)*	0.2094	0.1008	0.2266	0.0731
Base (T) *(keywords as input)*	0.2139	0.1011	0.2327	0.0743
Special-token distill [27]	0.2092	0.0999	0.2335	0.0739
Latent + CODI [22]	0.2105	0.0985	0.2269	0.0714
EMA-mode [23]	0.2097	0.1009	0.2317	0.0746
Joint-mode [38]	0.2156	0.1016	0.2348	0.0748
Self-Distill (T)	0.2155	0.1015	0.2397	0.0756
Self-Distill (S) (ours)	**0.2163**	**0.1017**	**0.2398**	**0.0757**

3.3 Online Results

To verify that offline gains translate into real production value, we deployed OneSearch-V2 on a large-scale commercial mall search platform and conducted rigorous A/B tests against the currently deployed OneSearch-V1. Across key business metrics, OneSearch-V2 yields statistically significant improvements ($p < 0.05$) of +3.98% item CTR, +1.17% PV CTR, +2.90% PV CVR, +2.07% buyers, and +2.11% orders. Manual evaluation on 3,200 query-item pairs further confirms improved search experience, with +1.37% page good rate, +0.55% item quality, and +1.65% query-item relevance (per-variant breakdown in Appendix G.2, Table 12). Notably, these gains are concentrated in categories dominated by ambiguous head queries (e.g., Clothing, Sportswear, Cosmetics) and on cold items, exactly the regimes where keyword-based reasoning is expected to help most. Ablation on three progressively enabled deployed variants (V2_RAG, V2_Reason, V2 full) further isolates the contribution of each component and confirms a clear per-module monotonic trend; the full online breakdown is in Appendix G.1 (Table 11). Further analysis of online gains from product category to user/query/item-level segment is presented in Appendix G.3, and a discussion of why self-distillation outperforms alternative internalization strategies is given in Appendix G.4.

4 Limitations

We note three limitations of this work. First, our evaluation is conducted on a single commercial e-commerce search platform; although the core challenges are common across modern e-commerce systems, cross-platform validation remains future work. Second, the thought-augmented query understanding module relies on a capable teacher LLM (Qwen3-32B) for offline CoT generation; deployments without access to a sufficiently strong teacher may see reduced gains, and investigating

smaller or domain-adapted teachers is a natural extension. Third, our online A/B tests measure short-term business metrics over a fixed observation window, leaving long-horizon effects such as user retention and the downstream impact on platform diversity to future longitudinal studies.

5 Conclusion

This paper presents OneSearch-V2, a reasoning-enhanced generative search framework addressing critical limitations in complex query understanding, personalized reasoning, and adaptive preference alignment. Through thought-augmented query understanding, reasoning-internalized self-distillation, and direct behavior-feedback optimization, V2 achieves substantial improvements while maintaining deployment efficiency. Rigorous online A/B tests demonstrate significant conversion gains, particularly on ambiguous queries, long-tail queries, low-activity users, and cold items—the regimes where keyword-based reasoning helps most.

Acknowledgments and Disclosure of Funding

We thank Kuaishou Technology for supporting this research work.

Funding. This work was supported by Kuaishou Technology. The authors received no other third-party funding or support for this work during the 36 months preceding submission.

Competing interests. The authors' affiliation with Kuaishou Technology is disclosed in the author information. The authors declare no additional financial relationships or competing interests relevant to this work during the 36 months preceding submission.

References

[1] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 2023.

[2] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, and et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[3] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, and Amanda Askell et al. Language models are few-shot learners. *https://arxiv.org/abs/2005.14165*, 2020.

[4] Ben Chen, Xian Guo, Siyuan Wang, Zihan Liang, Yue Lv, Yufei Ma, Xinlong Xiao, and et al. Onesearch: A preliminary exploration of the unified end-to-end generative framework for e-commerce search, 2025. URL `https://arxiv.org/abs/2509.03236`.

[5] Jiaxin Deng, Shiyao Wang, Kuo Cai, Lejian Ren, Qigen Hu, Weifeng Ding, Qiang Luo, and Guorui Zhou. Onerec: Unifying retrieve and rank with generative recommender and iterative preference alignment, 2025. URL `https://arxiv.org/abs/2502.18965`.

[6] Chenhe Dong, Shaowei Yao, Pengkun Jiao, Jianhui Yang, Yiming Jin, Zerui Huang, Xiaojiang Zhou, Dan Ou, Haihong Tang, and Bo Zheng. Taosr1: The thinking model for e-commerce relevance search. *arXiv preprint arXiv:2508.12365*, 2026.

[7] Xian Guo, Ben Chen, Siyuan Wang, Ying Yang, Chenyi Lei, and et al. Onesug: The unified end-to-end generative framework for e-commerce query suggestion. *CoRR*, abs/2506.06913, 2025. doi: 10.48550/ARXIV.2506.06913. URL `https://doi.org/10.48550/arXiv.2506.06913`.

[8] Ruidong Han, Bin Yin, Shangyu Chen, He Jiang, Fei Jiang, Xiang Li, Chi Ma, Mincong Huang, Xiaoguang Li, Chunzhen Jing, Yueming Han, MengLei Zhou, Lei Yu, Chuan Liu, and Wei Lin. Mtgr: Industrial-scale generative recommendation framework in meituan. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, CIKM '25, page 5731–5738. ACM, November 2025.

[9] Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason E Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. In *Second Conference on Language Modeling*, 2025.

[10] Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, and Andreas Krause. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*, 2026.

[11] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, July 2021. URL `https://doi.org/10.5281/zenodo.5143773`. If you use this software, please cite it as below.

[12] Jian Jia, Jingtong Gao, Ben Xue, Junhao Wang, Qingpeng Cai, Quan Chen, Xiangyu Zhao, Peng Jiang, and Kun Gai. From principles to applications: A comprehensive survey of discrete tokenizers in generation, comprehension, recommendation, and information retrieval. *arXiv preprint arXiv:2502.12448*, 2025.

[13] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[14] Xiaopeng Li, Bo Chen, Junda She, Shiteng Cao, You Wang, Qinlin Jia, Haiying He, Zheli Zhou, Zhao Liu, and et al. A survey of generative recommendation from a tri-decoupled perspective: Tokenization, architecture, and optimization. *Preprints*.

[15] xiaobo Liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. R-drop: Regularized dropout for neural networks. In *Advances in Neural Information Processing Systems*, pages 10890–10905. Curran Associates, Inc., 2021.

[16] Zihan Liang, Yufei Ma, Zhipeng Qian, Huangyu Dai, Zihan Wang, Ben Chen, Chenyi Lei, Yuqing Ding, and Han Li. Uniecs: Unified multimodal e-commerce search framework with gated cross-modal fusion. CIKM '25, page 1788–1797, New York, NY, USA, 2025. doi: 10.1145/3746252.3761170. URL `https://doi.org/10.1145/3746252.3761170`.

[17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *arXiv preprint arXiv:1708.02002*, 2018.

[18] Xinyu Lin, Chaoqun Yang, Wenjie Wang, Yongqi Li, Cunxiao Du, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. Efficient inference for large language model-based generative recommendation. In *ICLR*, 2025.

[19] Takeru Miyato, Andrew M. Dai, and Ian Goodfellow. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*, 2017.

[20] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, and et al. Recommender systems with generative retrieval. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 10299–10315. Curran Associates, Inc., 2023. URL `https://proceedings.neurips.cc/paper_files/paper/2023/file/20dcab0f14046a5c6b02b61da9f13229-Paper-Conference.pdf`.

[21] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[22] Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. Codi: Compressing chain-of-thought into continuous space via self-distillation. *arXiv preprint arXiv:2502.21074*, 2025.

[23] Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning. *arXiv preprint arXiv:2601.19897*, 2026.

[24] Yuhui Sun, Xiyao Wang, Zixi Li, YiTian Ding, Tianyang Ling, Jialuo Chen, Tianyi Yu, Zhenlong Yuan, and Jinman Zhao. Listwise direct preference optimization with multi-dimensional preference mixing. *arXiv preprint arXiv:2506.19780*, 2026.

[25] Juntao Tan, Shuyuan Xu, Wenyue Hua, Yingqiang Ge, Zelong Li, and Yongfeng Zhang. Idgenrec: Llm-recsys alignment with textual id learning. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 355–364, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704314. doi: 10.1145/3626772.3657821. URL https://doi.org/10.1145/3626772.3657821.

[26] Jianting Tang, Dongshuai Li, Tao Wen, Fuyu Lv, Dan Ou, and Linli Xu. Large reasoning embedding models: Towards next-generation dense retrieval paradigm. *arXiv preprint arXiv:2510.14321*, 2025.

[27] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *https://arxiv.org/abs/2012.12877*, 2021.

[28] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, and et al. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf.

[29] Zhipeng Wei, Kuo Cai, Junda She, Jie Chen, Minghao Chen, and et al. Oneloc: Geo-aware generative recommender systems for local life service. 2025. URL https://arxiv.org/abs/2508.14646.

[30] Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 641–649. Association for Computing Machinery, 2024.

[31] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[32] Chaoqun Yang, Xinyu Lin, Wenjie Wang, Yongqi Li, Teng Sun, Xianjing Han, and Tat-Seng Chua. Earn: Efficient inference acceleration for llm-based generative recommendation by register tokens. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, KDD '25, page 3483–3494, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400714542. doi: 10.1145/3711896.3736919. URL https://doi.org/10.1145/3711896.3736919.

[33] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 11809–11822. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/271db9922b8d1f4dd7aaef84ed5ac703-Paper-Conference.pdf.

[34] Xinhao Yao, Ruifeng Ren, Yun Liao, Lizhong Ding, and Yong Liu. Compositional generalization from learned skills via cot training: A theoretical and structural analysis for reasoning. *arXiv preprint arXiv:2502.04667*, 2026.

[35] Runyang You, Yongqi Li, Xinyu Lin, Xin Zhang, Wenjie Wang, Wenjie Li, and Liqiang Nie. R^2ec: Towards large recommender models with reasoning. In *NeurIPS*, 2025.

[36] Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, and et al. Actions speak louder than words: Trillion-parameter sequential transducers for generative recommendations, 2024. URL https://arxiv.org/abs/2402.17152.

[37] Yang Zhang, Wenxin Xu, Xiaoyan Zhao, Wenjie Wang, Fuli Feng, Xiangnan He, and Tat-Seng Chua. Reinforced latent reasoning for LLM-based recommendation. In *The Fourteenth International Conference on Learning Representations*, 2026.

[38] Ying Zhang, Tao Xiang, Timothy M. Hospedales, and Huchuan Lu. Deep mutual learning. *https://arxiv.org/abs/1706.00384*, 2017.

[39] Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

[40] Guorui Zhou, Hengrui Hu, Hongtao Cheng, Huanjie Wang, Jiaxin Deng, Jinghao Zhang, Kuo Cai, Lejian Ren, Lu Ren, Liao Yu, Pengfei Zheng, Qiang Luo, and et al. Onerec-v2 technical report, 2025. URL `https://arxiv.org/abs/2508.20900`.

[41] Jie Zhu, Zhifang Fan, Xiaoxie Zhu, Yuchen Jiang, and et al. Rankmixer: Scaling up ranking models in industrial recommenders, 2025. URL `https://arxiv.org/abs/2507.15551`.

A Framework Comparison: OneSearch-V1 vs. V2

To provide a clearer visual distinction between the previous and current generations of our generative search architecture, Fig. 3 illustrates the structural comparison between OneSearch-V1 and OneSearch-V2.

While both frameworks share the foundational unimodal Semantic ID (SID) tokenization to represent items (as justified in Appendix C), OneSearch-V2 introduces three critical architectural and training advancements to overcome V1's bottlenecks. First, it upgrades the straightforward log-fitting sequence-to-sequence training of V1 with thought-augmented query understanding and reasoning-internalized self-distillation. As analyzed in Appendix G.4, this specific self-distillation paradigm allows V2 to internalize complex query reasoning directly into its weights without the latency bottlenecks of explicit CoT generation. Finally, V2 replaces V1's reliance on a periodically updated, separately trained reward model (PARS) with a dynamic, behavior-feedback preference alignment system optimized via TPMA-GRPO. The progressive online impact of these structural differences is further broken down in Appendix G.1.

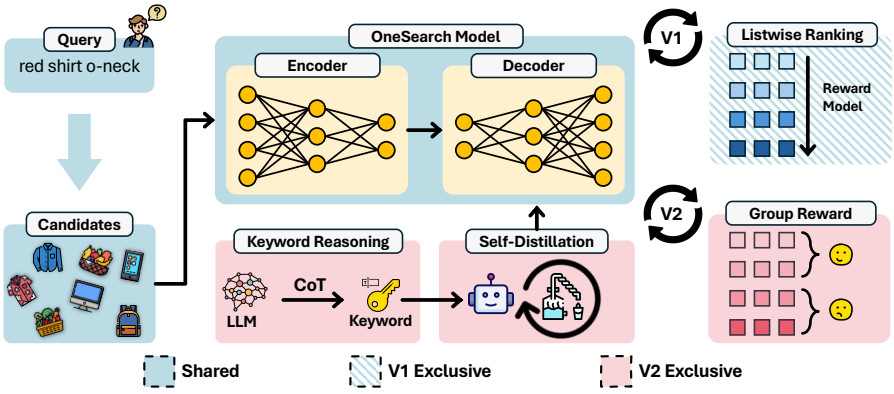

Figure 3: Comparison between OneSearch-V1 and OneSearch-V2. OneSearch-V2 extends the OneSearch framework with thought-augmented query understanding, reasoning-internalized self-distillation, and behavior-feedback preference alignment.

B Related Works

B.1 Generative Retrieval and Recommendation

Generative retrieval and recommendation reframe item retrieval as a sequence generation problem, where a model directly produces discrete Semantic IDs (SIDs) of items in an autoregressive manner. TIGER [20] is the seminal work in this line: it employs a Residual Quantized Variational Autoencoder (RQ-VAE) to compress item content embeddings into hierarchical discrete token sequences, and trains a Transformer-based encoder-decoder to autoregressively predict the next item's SID given a user's interaction history, elegantly unifying semantic content knowledge with collaborative signals through a shared quantized vocabulary. By contrast, IDGenRec [25] represents items by a *textual* ID generator natively in the LLM vocabulary, bridging the semantic gap between generative models and item ID spaces, and further enabling stronger cross-domain transfer.

As these generative models scale to billion-parameter LLMs, their inference capabilities are greatly enhanced, but inference latency also becomes a critical bottleneck in deployment. Lin et al. [18] addressed this by adapting speculative decoding to LLM-based generative recommenders, where a lightweight draft model proposes candidate SID sequences that the large target model verifies in parallel, yielding significant speedup with negligible quality loss. EARN [32] further proposed inserting compact register tokens into the input sequence to cache repetitive intermediate computations across decoding steps, reducing inference latency for deployment-scale systems. Most recently, R^2ec [35] introduced the first unified architecture that intrinsically integrates a reasoning chain into the generative recommendation loop: by optimizing both reasoning and recommend head, it achieves substantial gains on diverse scenarios with competitive inference efficiency. These advances collectively motivate the core question of our work: how to equip generative search models with reasoning capabilities while keeping inference cost practical for online deployment.

B.2 Latent Reasoning and Self-Distillation

Explicit chain-of-thought (CoT) reasoning has proven effective in enhancing LLM performance on complex tasks [28, 33, 21], yet the increased token generation incurs prohibitive latency for online deployment. To circumvent this cost, latent reasoning methods internalize reasoning into continuous hidden representations without explicit verbalization. Coconut [9] replaces textual reasoning steps with continuous thought vectors in the latent space. CODI [22] compresses explicit CoT into continuous thought vectors through single-stage self-distillation with L1 hidden-state alignment. Latent-R3 [37] further applies reinforcement learning over continuous latent representations. While these methods avoid explicit CoT at inference time, they typically require architectural modifications such as additional token embeddings, projection layers, or specialized decoding, which complicate deployment.

A closely related line of work achieves reasoning internalization through *information-asymmetric self-distillation*, where the same model serves as both teacher and student, with the asymmetry arising from different input contexts rather than distinct parameter sets. SDFT [23] constructs this asymmetry through in-context learning: the teacher observes few-shot demonstrations while the student sees only the raw query, and alignment is performed via reverse KL divergence. OPSD [39] applies a similar paradigm to mathematical reasoning with reference-solution-augmented teachers and token-level JSD alignment. SDPO [10] further extends this to RL settings, using environment feedback as privileged teacher information to construct dense per-token advantages from the teacher-student logit gap. These methods share a common insight: rich supervision can be extracted from the logit discrepancy between information-advantaged and information-deprived views of the same model, without any external teacher. Our work extends this paradigm to generative retrieval for e-commerce search, where the output space shifts from natural language to discrete SID sequences and the information asymmetry is constructed through keyword-based CoTs derived from query understanding.

B.3 Preference Alignment for GRs

Reinforcement learning (RL) has been extensively explored to align generative retrieval and recommendation models with complex user preferences. OneRec [5] introduces Early Clipped GRPO (ECPO) to optimize a personalized Preference Score derived from a separately trained multi-objective reward model. To mitigate the instability and reward hacking issues associated with reward models,

OneRec-V2 [40] leverages Gradient-Bounded Policy Optimization (GBPO) directly on real-world user feedback signals, such as duration-aware watch time. Similarly, OneSug [7] adopts a reward-weighted ranking strategy based on fine-grained behavior-level weights.

Despite these advancements, existing RL methods for generative retrieval share two critical limitations. First, methods relying on separately trained reward models are susceptible to sampling bias and reward hacking, as these models tend to overfit to a narrow subset of historical logs that only approximate global behavior distributions. Second, GRPO and its variants (e.g., ECPO, GBPO) assign a uniform, sequence-level advantage to every token within a generated SID sequence. However, SID generation follows a strict hierarchical causal structure, progressing from coarse-grained categories to fine-grained item attributes. Under this structure, a correct prefix followed by an incorrect suffix has fundamentally different implications from an entirely incorrect prefix. Uniform advantage assignment conflates these distinct positional contributions, weakening the learning signal for fine-grained token generation. Our work addresses both limitations: we replace the separate reward model with direct behavior feedback and introduce a token-position marginal advantage mechanism that respects the hierarchical nature of SID sequences.

C Analysis on Unimodal vs. Multimodal SID Tokenization

This appendix documents the tokenization inherited from OneSearch-V1, not a new contribution of V2. It is included so that readers unfamiliar with V1 can understand why unimodal keyword-based hierarchical quantization (KHQE) is the natural starting point for e-commerce generative search.

Semantic IDs (SIDs) have emerged as a cornerstone for GR systems due to their efficient and hierarchical semantic representation. Extensive research has investigated efficient SID tokenization [14, 12], which can be broadly categorized into two types: unimodal and multimodal. Here we explore which encoding paradigm is more suitable for e-commerce generative search.

Unlike recommendation systems, search engines must address the critical challenge of aligning queries and items within a unified tokenization to ensure robust semantic constraints. This necessitates careful handling of the representational disparity between unimodal queries and multimodal item contents, as items are characterized by extensive textual descriptions, multiple images showing different perspectives, and even explanatory videos. OneSearch-V1 addresses this by transforming multimodal information into a unimodal representation. Specifically, it employs Qwen-VL [1] to extract core keywords from diverse sources, thereby constructing a unified textual representation. Alternative approaches adopt direct multimodal mapping, either by feeding all sources simultaneously into the model or by encoding individual modalities separately before concatenation. However, these methods face inherent limitations: multiple images may display mutually exclusive attributes (e.g., a dress available in different colors), and the abundance of redundant attributes may introduce extra bias (e.g., number and position of T-shirt buttons). Consequently, core attributes risk being obscured in the multimodal encoding process.

To comprehensively compare the effectiveness of multimodal versus unimodal tokenization, we conducted experiments across multiple model configurations, including: a) Unimodal encoding utilizing text descriptions only, b) Multimodal encoding, containing unified encoding (joint processing) and separate encoding with subsequent concatenation, as well as c) Keyword hierarchical quantization in OneSearch. For experimental simplicity, we collected about 5M online clicked <query, item> pairs, and restricted the item input to only the title and two primary pictures. All embeddings were subsequently tokenized using the unified RQ-OPQ framework. We report two SID-quality metrics alongside retrieval performance: *CUR* (code utilization rate), the fraction of codebook entries assigned to at least one catalog item, which measures codebook efficiency; and *ICR* (item coverage ratio), the fraction of catalog items whose full SID sequence is unique, where a higher ICR indicates fewer SID collisions between distinct items. Higher is better for both. The results are shown in Table 6.

Table 6: Comparison of unimodal, multimodal, and KHQE tokenization approaches. Recall@10 and MRR@10 are evaluated on click data.

Type	Model	Size	CUR	ICR	Recall	MRR
uni-	bge-base [30]	109M	4.54%	96.88%	0.2445	0.1013
	qwen3 [31]	0.6B	**5.11%**	<u>97.56%</u>	<u>0.2468</u>	<u>0.1025</u>
multi-	UniECS [16]	200M	4.54%	94.62%	0.2368	0.1007
	bge-vl [30]	149M	4.23%	94.46%	0.2364	0.1009
	qwen3-vl [2]	2B	4.86%	95.27%	0.2389	0.1012
	CLIP [11]	188M	4.03%	94.16%	0.2358	0.1003
KHQE	bge+kw.	109M	**5.11%**	**99.50%**	**0.2542**	**0.1085**

Unimodal approaches significantly outperform multimodal counterparts at comparable scales—even the smaller bge-base surpasses the larger Qwen3-VL. This gap stems from cross-modal representational discrepancies and redundant attributes that constrain multimodal encoding effectiveness. The separate-then-concatenate strategy performs worst, further confirming these challenges. KHQE achieves optimal results, demonstrating superior core attribute extraction and hierarchical representation. More importantly, its smaller size allows real-time processing of input queries, striking a favorable balance between performance and efficiency. Meanwhile, these findings underscore that developing discriminative encodings for e-commerce search should highlight two critical factors: mitigating cross-modal disparities and enhancing salient information.

D Regularizer Details

D.1 R-Drop Formulation

R-Drop [15] performs two forward passes $z_1^{(S)}, z_2^{(S)}$ of the same input through the student with independent dropout masks, and minimizes the symmetric KL between their output distributions:

$$\mathcal{L}_{\text{R-Drop}} = \frac{1}{2}\Big[\text{KL}(P_1 \| P_2) + \text{KL}(P_2 \| P_1)\Big], \tag{11}$$

where $P_k = \text{softmax}(z_k^{(S)})$ for $k \in \{1, 2\}$, and both KL terms are masked to the valid token set $\mathcal{V}$ defined in §2.2.

D.2 FGM Formulation

FGM [19] perturbs the embedding layer e along the gradient direction of $\mathcal{L}_{\text{base}}$ (Eq. 4):

$$r_{\text{adv}} = \epsilon \cdot \frac{\nabla_e \mathcal{L}_{\text{base}}}{\|\nabla_e \mathcal{L}_{\text{base}}\|_2}, \tag{12}$$

where ϵ controls the perturbation magnitude. A second forward-backward pass on the perturbed embeddings $e + r_{\text{adv}}$ produces $\mathcal{L}_{\text{adv}}$, whose gradients are accumulated before restoring e to its original value.

E GRPO Formulation

Group Relative Policy Optimization (GRPO) [21] eliminates the critic network via within-group advantage normalization. For each input prompt x_u, the current policy π_θ generates G rollouts $\{o_i\}_{i=1}^{G}$. The sequence-level advantage is computed as:

$$\hat{A}_i = \frac{R_i - \mathrm{mean}_{j\in[G]}(R_j)}{\mathrm{std}_{j\in[G]}(R_j) + \delta}, \tag{13}$$

where δ is a constant for numerical stability. The GRPO loss is:

$$\mathcal{L}_{\mathrm{GRPO}} = -\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\min\Big(r_{i,t}\hat{A}_i,\ \mathrm{clip}(r_{i,t}, 1{-}\varepsilon, 1{+}\varepsilon)\hat{A}_i\Big), \tag{14}$$

where $r_{i,t} = \pi_\theta(o_{i,t} \mid x_u, o_{i,<t}) / \pi_{\theta_{\mathrm{old}}}(o_{i,t} \mid x_u, o_{i,<t})$ is the per-token importance ratio.

F Cross-Architecture Generalization

To verify that the proposed innovations generalize across different model architectures, we conduct experiments on both GPT-2 [3] and Qwen3-0.6B [31] (decoder-only) in addition to the BART-B (encoder-decoder) backbone used in the main paper. All models are trained on the same 5M sample dataset under comparable settings.

F.1 Overall Self-Distillation Effectiveness

Table 7 and Table 8 report the incremental results on GPT-2 and Qwen3-0.6B. Both architectures exhibit the same cumulative pattern as BART-B (Table 2), confirming the broad effectiveness of the proposed framework.

Table 7: Cumulative performance of reasoning-internalized self-distillation on **GPT-2**.

Method	Order (7229)		Click (30k)	
	HR@10	MRR@10	HR@10	MRR@10
Baseline	0.2088	0.0993	0.2270	0.0733
\+ self-distill	0.2128	0.1011	0.2325	0.0734
\+ R-Drop	0.2168	0.1012	0.2380	0.0755
\+ FGM	0.2195	0.1030	0.2430	0.0775
\+ focal loss	**0.2230**	**0.1050**	**0.2520**	**0.0802**

Table 8: Cumulative performance of reasoning-internalized self-distillation on **Qwen3-0.6B**.

Method	Order (7229)		Click (30k)	
	HR@10	MRR@10	HR@10	MRR@10
Baseline	0.2195	0.1012	0.2503	0.0769
\+ self-distill	0.2266	0.1060	0.2568	0.0794
\+ R-Drop	0.2275	0.1070	0.2625	0.0800
\+ FGM	0.2295	0.1082	0.2629	0.0809
\+ focal loss	**0.2310**	**0.1089**	**0.2632**	**0.0815**

F.2 Self-Distillation Verification

Table 9 and Table 10 follow the same protocol as Table 5. On both GPT-2 and Qwen3-0.6B, Self-Distill (S) outperforms Base (T) without observing keywords at inference, confirming architecture-agnostic reasoning internalization.

Table 9: Self-distillation verification on **GPT-2**. "(S)" and "(T)" denote student-side and teacher-side evaluation. †Student model trained and evaluated without keyword augmentation. ‡Teacher model trained and evaluated with keyword-augmented data.

Method	Order (7229)		Click (30k)	
	HR@10	MRR@10	HR@10	MRR@10
Base (S)†	0.2088	0.0993	0.2270	0.0733
Base (T)‡	0.2115	0.1008	0.2298	0.0732
Self-Distill (T)	0.2098	0.1002	0.2306	0.0729
Self-Distill (S)	**0.2128**	**0.1011**	**0.2325**	**0.0734**

Table 10: Self-distillation verification on **Qwen3-0.6B**. "(S)" and "(T)" denote student-side and teacher-side evaluation. †Student model trained and evaluated without keyword augmentation. ‡Teacher model trained and evaluated with keyword-augmented data.

Method	**Order (7229)**		**Click (30k)**	
	HR@10	**MRR@10**	**HR@10**	**MRR@10**
Base (S)†	0.2195	0.1012	0.2503	0.0769
Base (T)‡	0.2232	0.1035	0.2550	0.0785
Self-Distill (T)	0.2241	0.1042	0.2533	0.0780
Self-Distill (S)	**0.2266**	**0.1060**	**0.2568**	**0.0794**

G Further Analysis and Discussion

This appendix provides the additional evidence and analysis referenced from the main paper. Sections G.1 and G.2 isolate the online contribution of each proposed component and report the manual-evaluation breakdown behind the summary numbers in §3.3. Section G.3 analyzes where the online gains concentrate, from product category breakdowns to user/query/item-level segment drill-downs. Section G.4 examines why our self-distillation outperforms alternative latent-token and hidden-state alignment strategies. Section G.5 discusses future directions.

G.1 Per-component Online Contribution

To isolate the contribution of each proposed component in production, we deployed three progressively enabled variants of OneSearch-V2 on the live mall search platform and compared each against the serving OneSearch-V1 baseline under identical traffic splits. All three variants share the same deployment paradigm as V1—the raw query is fed to the model and candidate items are produced directly via beam search—so the online gaps reflect only the proposed innovations.

Variants. V2_RAG adds the four CoT tasks of §2.1 to SFT Stage 1 on top of V1. V2_Reason further replaces Stage 3 fine-tuning with the reasoning-internalized self-distillation of §2.2. V2 (full) additionally applies the behavior-feedback preference alignment system of §2.3 (listwise DPO followed by TPMA-GRPO) in place of V1's PARS reward model.

Table 11: Online A/B improvements of three progressively enabled variants of OneSearch-V2 over the deployed OneSearch-V1 on the mall search platform. All numbers are relative gains and statistically significant ($p < 0.05$). Each variant inherits all components of the variant(s) above it.

Variant	Item CTR	PV CTR	PV CVR	Buyer	Order
V2_RAG	+0.52%	+0.77%	+0.63%	+1.04%	+1.07%
V2_Reason	+2.59%	+1.42%	+2.21%	+1.50%	+1.57%
V2 (full)	**+3.98%**	+1.17%	**+2.90%**	**+2.07%**	**+2.11%**

The per-component trend is clear. Thought-augmented query understanding alone (V2_RAG) already delivers consistent improvements on every metric. Internalizing the reasoning through self-distillation (V2_Reason) then amplifies all gains substantially, confirming that the benefit of keyword-based CoT is not limited to input augmentation and can indeed be encoded into the model weights. The full pipeline with behavior-feedback alignment yields the strongest gains on conversion-oriented metrics (Item CTR, PV CVR, Buyer, Order). PV CTR is the only metric on which V2 (full) is slightly below V2_Reason; this is consistent with the design of the composite reward in §2.3.1, which explicitly trades a small amount of exposure-CTR lift for relevance and conversion improvements.

G.2 Manual Evaluation

To complement the automatic A/B metrics, we conduct the same manual evaluation protocol as OneSearch-V1. We randomly sampled 200 queries from live traffic and, for each, collected 16 query-item pairs at identical exposure positions across systems, yielding 3,200 query-item pairs in total. Trained human annotators then rated each pair along three independent axes: (i) *page good rate*, the fraction of result pages judged as overall satisfactory; (ii) *item quality*, a per-item judgment of listing quality (title clarity, image quality, price reasonableness); and (iii) *query-item relevance*, a per-item semantic relevance judgment. All numbers below are relative improvements over the deployed OneSearch-V1.

Table 12: Manual evaluation of online variants on 3,200 query-item pairs, reported as relative gains over OneSearch-V1.

Variant	Page Good Rate	Item Quality	Q-I Relevance
V2_Reason	+1.12%	+0.28%	+1.01%
V2 (full)	**+1.37%**	**+0.55%**	**+1.65%**

The gap between V2_Reason and V2 (full) on query-item relevance (+1.65% vs. +1.01%) is the largest among the three axes, indicating that direct behavior-feedback alignment primarily sharpens semantic match rather than listing presentation.

G.3 Where Do the Online Gains Come From?

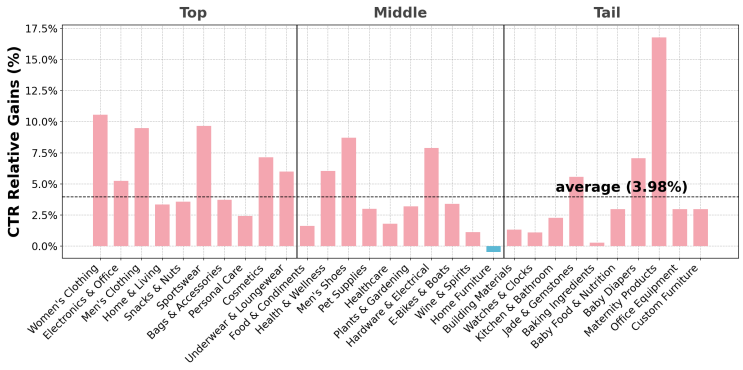

Figure 4: Relative gains in item CTR of OneSearch-V2 over OneSearch-V1 across the top, middle, and tail 10 product categories ranked by page views, aggregated across multiple A/B experiment runs. The dashed line marks the platform-wide average of $+3.98\%$ ($p < 0.05$).

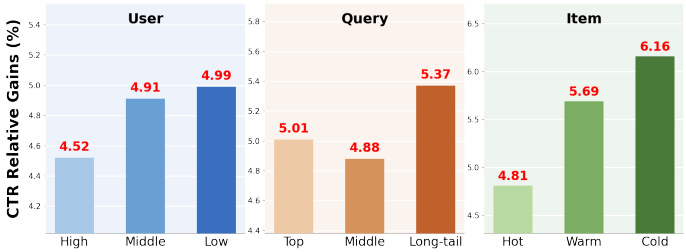

Figure 5: Relative gains in item CTR of OneSearch-V2 over OneSearch-V1, measured within each user-activity, query-frequency, and item-popularity bucket in the most recent A/B experiment.

We further analyze the impact of OneSearch-V2 on item CTR across product categories. Fig. 4 aggregates results across multiple A/B experiment runs, yielding a statistically significant average gain of +3.98% ($p < 0.05$) across all industries. The largest improvements are concentrated in categories within categories characterized by high-pv yet semantically ambiguous queries, such as Clothing, Sportswear, and Cosmetics, demonstrating the more accurate semantic understanding and personalized predictions of the newer model.

To understand which user and item segments benefit most, we further conduct a drill-down analysis on the most recent A/B experiment along three dimensions: query frequency, user activity, and item popularity. For query frequency, we partition queries into top (daily PV $>$ 1,000), middle (100–1,000), and long-tail (PV $<$ 100) buckets. For user activity, low/mid/high-activity users are determined by a composite of search frequency, click/purchase count, and total spending. For item popularity, we define cold items as those published within the last seven days without any interaction, and hot items as the top 10% by sales within each leaf category; all others are warm.

As illustrated in Fig. 5, OneSearch-V2 delivers consistent and substantial CTR improvements across every user segment, query frequency bucket, and item popularity level, validating the robustness and generalizability of the framework. Two patterns are particularly notable. First, *long-tail queries* benefit most among the query frequency buckets (+5.37%), followed by top queries (+5.01%) and middle queries (+4.88%), showing that CoT-enhanced semantic alignment particularly helps ambiguous or rare queries. Second, *cold items* benefit most among the item popularity buckets (+6.16%), substantially above warm items (+5.69%) and hot items (+4.81%); effectively surfacing newly published items is valuable for both merchant satisfaction and platform ecosystem health.

G.4 Why Does Self-Distillation Outperform Latent-Token Approaches?

Table 5 in §3.2 shows that latent-token and hidden-state alignment strategies consistently underperform our self-distillation approach. Since CODI-style hidden-state alignment is the strongest latent-token baseline, we conduct further experiments on it to understand the source of the gap; results are reported in Table 13.

Table 13: Analysis of CODI-style configurations on the BART-B backbone. "+Proj" adds a projection layer; "+SD" combines logit-level KL distillation.

Method	Order (7,229)		Click (30k)	
	HR@10	MRR@10	HR@10	MRR@10
Baseline	0.2094	0.1008	0.2266	0.0731
Self-Distill (KL)	**0.2163**	**0.1017**	**0.2398**	**0.0757**
CODI	0.2105	0.0985	0.2269	0.0714
CODI + Proj	0.2092	0.0998	0.2270	0.0717
CODI + Proj + SD	0.2084	0.1002	0.2230	0.0720

Two probable causes emerge. First, our self-distillation provides a position-wise learning signal: at every SID token position, the student receives the teacher's full output distribution, which directly reflects how keyword information shifts the likelihood of each candidate code. CODI-style alignment, by contrast, supervises a single designated token via L1 regression of its hidden states at each transformer layer, providing no explicit reasoning signal at the remaining SID token positions. To isolate this distillation mechanism, we omit CODI's projection layer, which contributes only marginally in the original paper (43.7% vs. 42.5%). Table 13 confirms that restoring it does not close the gap, suggesting the limitation lies in the supervision form itself rather than model capacity. Second, when CODI's L1 loss is combined with our logit-level KL distillation (*CODI + Proj + SD* in Table 13), performance drops below either objective in isolation (0.2084 vs. 0.2105 for CODI and 0.2163 for Self-Distill). A plausible explanation is that the two losses impose competing constraints—L1 pulls hidden activations toward the teacher's layer-wise geometry while KL shapes the output distribution—and the representation that best satisfies one need not best serve the other. Our KL-only formulation sidesteps this tension, letting the model organize its internal representations freely around the prediction objective.

G.5 Future Directions

Future developments should be driven by three core principles: business requirements, scenario diversity, and user-centric needs. We identify several promising directions for further investigation. (1) For long-tail queries with limited historical interactions, more effective beyond-logs training strategies are needed to address the insufficient-sample problem. (2) E-commerce platforms increasingly feature diverse content modalities, including videos, live streams, and traditional item listings; a fundamental challenge is constructing a unified SID tokenization scheme that effectively represents heterogeneous content types while preserving their unique characteristics and cross-modal relationships. (3) The evolution toward agentic search systems represents another promising frontier, requiring innovations in efficient online learning mechanisms that can update model behavior in real-time without compromising latency or stability.

H Prompt Templates for the Reasoning Pipeline

Table 14 presents the complete set of prompt templates designed for the reasoning pipeline (see Fig. 6), covering three core modules: query analysis, keyword extraction, and preference calibration. Each module consists of a system-level role definition and a structured task prompt, collectively enabling the understanding and condensation of complex queries.

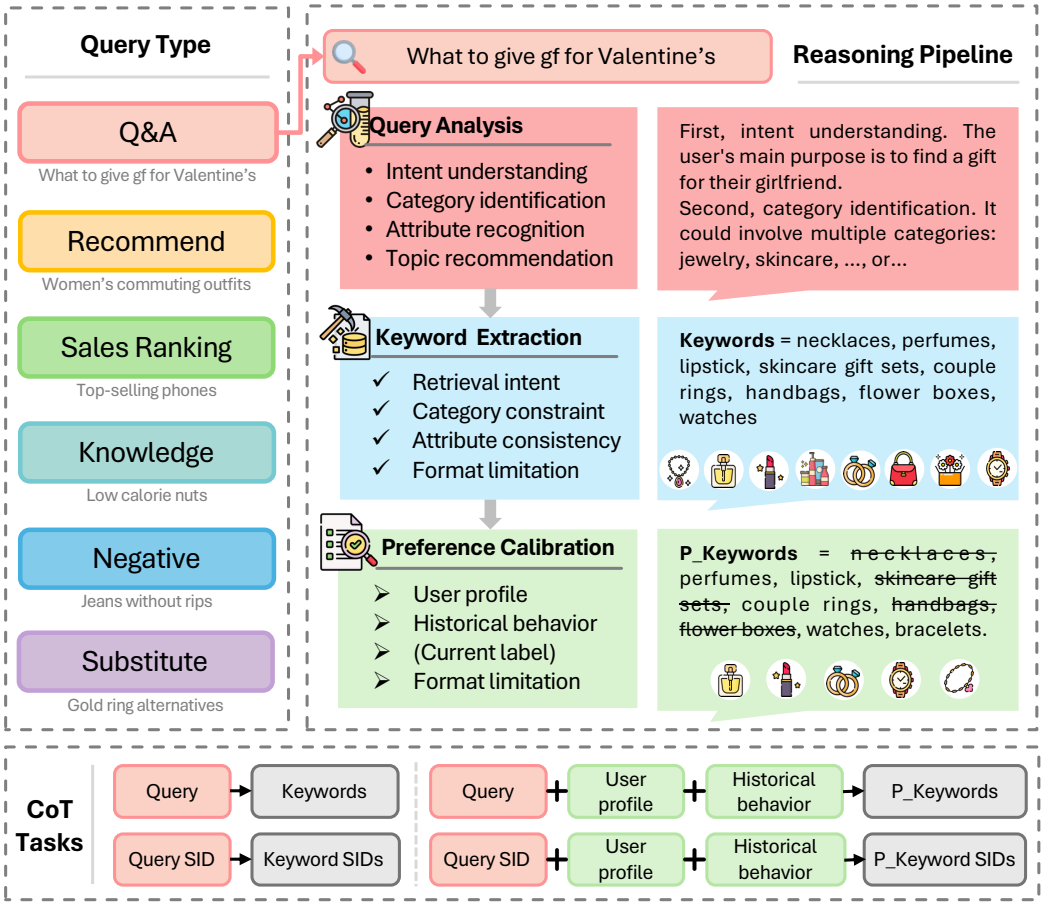

Figure 6: Three-step keyword-based CoT extraction pipeline for diverse complex query types, along with the corresponding CoT tasks.

Table 14: Prompt Templates for the Three-step Reasoning Pipeline

Step 1 — Query Analysis

System Instruction:

You are an AI search assistant for a Chinese e-commerce search platform. Analyze the user's search query across the following four dimensions.

Task Prompt:

Analyze the query along **four dimensions**:

1. Intent Understanding — Identify the user's *single* primary intent.

- *Product Search*: most common (e.g. dress, smartphone).
- *Functional Need*: platform features (e.g. track parcel).
- *Note*: If intent $\neq$ product search, skip remaining steps.

2. Category Identification — Identify one or more product categories.

- *Top-level categories*: women's wear, mobile & electronics, home goods, bags, accessories, men's wear, personal care, snacks, skincare, sports & outdoors, cosmetics, underwear, home apparel, women's shoes, toys, gaming peripherals, fresh produce, instant food, home appliances, etc.
- *Sub-categories*: e.g., women's wear includes T-shirts, skirts, sweatshirts, sweaters, and clothing for middle-aged and elderly women.
- *Multiple categories*: some queries may correspond to multiple categories, e.g. "women's windbreaker" $\rightarrow$ women's wear AND sports & outdoors.
- *Note*: Provide as comprehensive and detailed a range of product categories as possible.

3. Attribute Recognition — Extract attributes *explicitly* stated in the query without any expansion.

- *Common attributes*: entity, model, brand, audience, color, material, style, season, scene, function, price, etc.
- *Note*: The search system must return products that match the query, so strictly retain the attributes that are relevant in the query.

4. Topic Recommendation — Suggest candidate topics satisfying the query, like categories or specific products.

- *Note*: need meet its categories, and attribute constraints. Do **not** over-recommend.
- *Good cases*:
  - "plaid skirt" $\rightarrow$ plaid wrap skirt, plaid A-line skirt.
  - 'La Mer dupe" $\rightarrow$ Estée Lauder serum, SK-II, Lancôme cream.
  - "knitwear, no turtleneck" $\rightarrow$ V-neck knitwear, crew-neck knitwear.
  - "winter fruits" $\rightarrow$ strawberry, red pomelo, orange.
- *Bad cases*:
  - "bicycle accessories" $\rightarrow$ bicycle (wrong category).
  - "knitwear, no turtleneck" $\rightarrow$ turtleneck knitwear (violates constraint).
  - "iPhone 17" $\rightarrow$ iPhone 16 (wrong model).

Keep analysis $\leq$300 words. Please analysis query: `{}`

Step 2 — Keyword Extraction

System Instruction:

You are an AI search assistant for a Chinese e-commerce search platform. Based on the user's search query and the LLM analysis result, extract keywords that are **closely related** to the query.

Continued on next page

Table 14 *(continued)*

Task Prompt:

Rules for the extraction:

1. Source Constraints:

- Extract **only** under "Product Search" intent; otherwise output `Not extractable` and stop.
- Extract **only** from the *Topic Recommendation* section.
- If empty, fall back to keywords from *Attribute Recognition* and *Category Identification*.

2. Extraction Criteria:

- Remove off-query items (e.g. query "Hisense TV" ⇒ exclude "TCL TV").
- Keep specific attributes (e.g. "*plaid* skirt").
- Remove marketing terms (e.g. "bestseller", "good quality").
- Merge synonymous attributes (e.g. "woolens" merged into "wool sweater").
- Preserve model details (e.g. "iPhone 15 Pro Max").

3. Output Format:

- Comma-separated; at most **8 keywords**; Each keyword can have a maximum of 10 Chinese characters.
- by popularity (descending).
- Each keyword must be independently retrievable.

Please extract keywords from the analysis results based on the query:
Query: `{}`
Analysis Result: `{}`

Step 3 — Preference Calibration

System Instruction:

You are an AI search assistant for a Chinese e-commerce search platform. Based on the user's query and behavioral history, extract or supplement keywords from the candidate list that the user is likely interested in.

Task Prompt:

Rules for the calibration:

1. All keywords must be **closely related** to the user's query.
2. Prioritize keywords aligned with the user's **profile** and **behavioral history** (recent searches & recently clicked products).
3. Prefer keywords from the candidate list; supplementation is permitted.
4. Output at most **5 keywords**; each keyword can have a maximum of 10 Chinese characters; comma-separated; each keyword must be independently retrievable.

Example:
Input:
Query: autumn-winter outfit
User profile: female, aged 18–23
Recent searches: ["autumn-winter outfit", "autumn-winter coat women", "autumn-winter trousers men", "hoodie", "fruits"]
Recent clicks: ["spicy hotpot instant noodles 437 g", "classic rice noodles 360 g × 8 bags"]
Candidates: ["wool overcoat", "down jacket", "woolen coat", "thick hoodie", "knitwear", "windbreaker", "thermal underwear"]
Output:
wool overcoat women, down jacket women, woolen coat women, hooded hoodie women, windbreaker women

Now, based on the user's query and behavioral history, extract or supplement keywords from the candidate list that the user is likely interested in.
Input:
Query: `{}`
User profile: `{}`
Recent searches: `{}`
Recent clicks: `{}`
Candidate keywords: `{}`

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction explicitly state three contributions (thought-augmented query understanding, reasoning-internalized self-distillation, and behavior-feedback preference alignment), each of which is substantiated by corresponding methodology sections (§2.1–2.3), offline ablations (§3.1–3.2), and large-scale online A/B tests (§3.3). Limitations are discussed in §4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: §4 (Limitations) explicitly discusses three limitations: (1) evaluation is conducted on a single commercial e-commerce platform; (2) the thought-augmented module depends on a capable teacher LLM (Qwen3-32B), which may not be available in all deployments; (3) online A/B tests measure only short-term business metrics, leaving long-horizon effects to future work.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: This paper does not include formal theorems or proofs. The methodology and appendix sections present algorithmic formulations and loss functions (e.g., Eq. 1–14) rather than theoretical results requiring formal proof.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The implementation details in §3 provides all key hyperparameters, including pretrained backbones (BART-B, GPT-2, Qwen3-0.6B), beam search size (512), batch sizes, distillation temperature (τ=1.0), KL weight (α_{KL}=0.1), R-Drop coefficient (α_{R}=0.5), FGM magnitude (ϵ=0.6), focal loss parameters, and TPMA-GRPO reward values. The code is provided in the supplemental material.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The implementation code is provided in the supplemental material. The training data consists of proprietary click and purchase logs from a commercial mall search platform and cannot be publicly released; however, the code and detailed hyperparameter descriptions enable reproduction on similarly structured datasets.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Training and test details are provided in §3: the training set covers three months of clicked and purchased $\langle$query, item$\rangle$ pairs; the test set is drawn from the most recent day's logs ($\sim$30K page views, 30K clicks, 7,229 orders). All hyperparameters are reported in the implementation details paragraph of §3, and evaluation metrics (HR@10, MRR@10) are defined therein.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Online A/B test results (§3.3 are reported with statistical significance ($p < 0.05$). Offline results (§3.1–3.2) are averaged over multiple runs with different random seeds, ensuring stability of the reported numbers.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: Detailed compute resource information (GPU type, memory, training time) involves proprietary internal infrastructure and cannot be disclosed. The paper report model sizes and dataset scale, which provide a rough indication of the computational requirements.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This work involves optimization of an e-commerce search system using anonymized user behavior logs collected under the platform's data usage policy. No sensitive personal data is exposed, and the research does not involve any activity that conflicts with the NeurIPS Code of Ethics.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: The paper does not include a dedicated broader impacts discussion. On the positive side, the proposed framework improves search relevance, reduces information bubbles, and enhances cold-item exposure, benefiting both users and merchants. Potential negative impacts include the risk of reinforcing consumption biases through personalized recommendations and concerns around user data privacy in commercial search systems. We will add a broader impacts section in the final version.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This paper presents a generative search framework for e-commerce, not a general-purpose pretrained language model or scraped dataset. The released code does not pose high misuse risk.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All pretrained models and methods used in this paper, including BART, GPT-2, Qwen3, BGE, CLIP, UniECS, R-Drop, FGM, GRPO, and listwise DPO, are properly cited with their original papers. These models are used in accordance with their respective open-source licenses.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The implementation code released in the supplemental material is accompanied by documentation covering the training pipeline, and usage instructions for the proposed components. The proprietary training data cannot be released, but a few data examples are disclosed and the code is self-contained for use with similarly structured datasets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

Justification: The manual evaluation (Appendix G.2) involves trained human annotators rating 3,200 query-item pairs across three axes (page good rate, item quality, query-item relevance). The evaluation criteria and rating dimensions are described in Appendix G.2. Annotators are employees compensated according to standard platform labor policies.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The manual evaluation task involves rating e-commerce search results for quality and relevance, which poses minimal risk to participants. No sensitive personal information was collected from annotators, and no IRB approval is required for this type of low-risk annotation task.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

Answer: [Yes]

Justification: The use of Qwen3-32B for offline keyword-based CoT generation is a core and original component of the thought-augmented query understanding module (§2.1). Its role, prompt design, and output format are described in detail in §2.1 and Appendix H.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.