OneSug: The Unified End-to-End Generative Framework for E-commerce Query Suggestion

Xian Guo[1], Ben Chen[1]*, Siyuan Wang[1], Ying Yang[1], Mingyue Cheng[2], Chenyi Lei[1], Yuqing Ding[1], Han Li[1]

[1]Kuaishou Technology, Beijing China
[2]University of Science and Technology of China, Hefei, China
{guoxian, chenben03, wangsiyuan12, yangying12, dingyuqing03, lihan08}@kaishou.com;leichenyi@gmail.com;mycheng@ustc.edu.cn

Abstract

Query suggestion plays a crucial role in enhancing user experience in e-commerce search systems by providing relevant query recommendations that align with users' initial input. This module helps users navigate towards personalized preference needs and reduces typing effort, thereby improving search experience. Traditional query suggestion modules usually adopt multi-stage cascading architectures, for making a well trade-off between system response time and business conversion. But they often suffer from inefficiencies and suboptimal performance due to inconsistent optimization objectives across stages. To address these, we propose OneSug, the first end-to-end generative framework for e-commerce query suggestion. OneSug incorporates a prefix2query representation enhancement module to enrich prefixes using semantically and interactively related queries to bridge content and business characteristics, an encoder-decoder generative model that unifies the query suggestion process, and a reward-weighted ranking strategy with behavior-level weights to capture fine-grained user preferences. Extensive evaluations on large-scale industry datasets demonstrate OneSug's ability for effective and efficient query suggestion. Furthermore, OneSug has been successfully deployed for the entire traffic on the e-commerce search engine in Kuaishou platform for over 4 month, with statistically significant improvements in user top click position (-9.33%), CTR (+2.01%), Order (+2.04%) over the online multi-stage strategy, showing great potential in e-commercial conversion.

Code — https://github.com/Edgis/OneSug/blob/main
Datasets —
https://github.com/Edgis/OneSug/blob/main/data.txt
Extended version —
https://github.com/Edgis/OneSug/blob/main/onesug.pdf

Introduction

Query Suggestion is a fundamental module of modern e-commerce search systems, with the aim of enhancing user experience by suggesting new queries related to the user's input. By offering more specific and refined query recommendations, this module assists users in navigating towards their personalized information needs, or providing frequently searched keywords by users with the same interests, to reduce further word typing thus improving the search efficiency. Typically, consider a user looking for a new smartphone and entering "smartphone" as the query, the search engine might display various brands and models of smartphones, and the user might be interested in a specific brand or function. One useful query suggestion module should provide suggestions such as "smartphones best of 2025" or "smartphone value for money ranking", to assist users in narrowing down their search to find the item that best meets their needs.

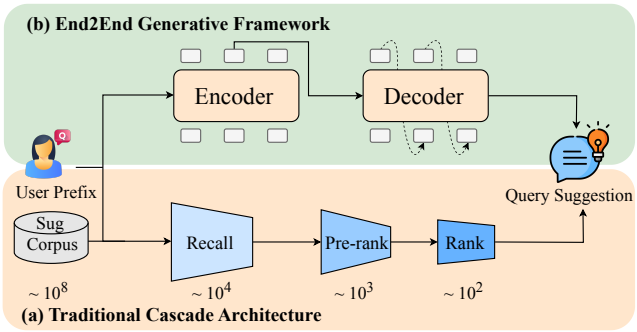

Figure 1: (a) A Traditional Cascade Architecture in Query Suggestion. (b) Our proposed unified end-end generative framework.

Modern query suggestion modules (Ahmad, Chang, and Wang 2018; Chen and Lee 2020) typically leverage past user-system interactions from *query logs* and incorporate popular search trends to refine diverse search intents. To balance efficiency and performance, these systems generally employ a Multi-stage Cascading Architecture (MCA). As illustrated in Figure 1(a), upon prefix entry, the retrieval stage scans the 10^8 candidates down to 10^4. The subsequent pre-ranking stage then processes only the received queries (10^3), before the ranking stage evaluates the top 10^2 candidates.

However, pursuit of the trade-off of system response time and business conversion has resulted in the restricted use of lightweight models in the previous stages (recall, pre-ranking), and complex reasoning can only be carried out in the final ranking stage. This inconsistency makes existing methods have the following limitations: 1) the performance of the previous stage determines the upper bound of the next stage. 2) Heterogeneous modules with different optimization

*Corresponding author.

objectives may lead to sub-optimal performance of the overall cascading framework. 3) Existing methods (Bar-Yossef and Kraus 2011) fail to collect effective queries for unseen prefixes, thus limiting performance in long-tail sessions.

The same problems also occur in the cascading framework of e-commerce/video search, recommendation, and advertising engines (Ko et al. 2022; Xu, He, and Li 2018; Gharibshah and Zhu 2021). To address these, recent works (Deng et al. 2025; Pang et al. 2025) have been devoted to using generative retrieval (GR) methods to replace part or even all of the stages in MCA. Typically, a GR model is trained to directly map items into the ID-based (Rajput et al. 2023) or String-based (Bevilacqua et al. 2022) codes, and then generate the identifiers of candidates in an auto-regressive manner. A typical work among them is OneRec (Deng et al. 2025), which can replace the entire framework through a session-wise generation approach and an iterative preference alignment module, thereby achieving end-to-end video recommendation. Although it is very promising, this paradigm is not suitable for query suggestion. First, video recommendation is a close-vocabulary task, as its inputs and outputs are both certain videos; while for query suggestion, user inputs and corresponding outputs are open. Secondly, suggestion module must consider the relevance between prefixes and queries. Thus a more fine-grained ranking modeling, other than session-wise, is needed.

We propose **OneSug**, an end-to-end generative framework for E-commerce Query Suggestion, which comprises:

1) A **P**refix2query **R**epresentation **E**nhancement (PRE) module. Considering that short prefixes (usually only one word) have ambiguous semantics, we attempt to enhance prefix representation using interactively and semantically related queries. We first finetune a representation model with selected {*prefix*, *queries*} pairs to align the content and business characteristics. Then we adopt the RQ-VAE (Zeghidour et al. 2021) to generate hierarchical quantitative semantic codes so that each prefix can be enhanced by the queries with the nearly same codes. This module can not only alleviate the insufficient semantic representation of short prefixes, but also combine semantic and business characteristics effectively. In addition, the adoption of RQ-VAE can reduce the giant matching computation during inference, further facilitating the practical deployment of generative models.

2) A unified encoder-decoder architecture, which takes the prefix, related queries, user's historical interactive queries, and user profile as inputs, and directly outputs queries the user may be interested in. This unified structure is concise, as it effectively avoids suboptimal final results caused by inconsistent optimization goals at each stage of multi-stage cascading architecture. Thus it can be deployed for end-to-end practical application.

3) A user preference alignment, powered by a **R**eward-**W**eighted **R**anking (RWR) strategy for the generative model. Initially, we categorize user interaction behaviors into six distinct levels and construct nine types of positive and negative sample sequences based on them. Then, we employ Direct Preference Optimization (DPO) (Rafailov et al. 2023) to facilitate the model's learning of preference differences among samples by assigning varying weights according to the level gap. Furthermore, inspired by traditional Click-Through Rate (CTR) models, we extend DPO from contrastive learning with pair-wise data to list-wise ones. Subsequently, a hybrid ranking framework, which integrates list-wise and point-wise approaches, is proposed, aiming to instill ranking ability into the generative model and ensure the accuracy of the generated sequence. Unlike straightforward sampling with only selecting the best and worst samples in Iterative Preference Alignment (Deng et al. 2025), reward-weighted sequence ranking derived from the level gap of user interactive behaviors captures the nuances in user behavior towards different queries more effectively, thus boosting the generative model's capability for concise personalized ranking.

We execute extensive offline evaluations on large-scale industry datasets from the online user search logs, and the significant performance boosts demonstrate the proposed method's effectiveness for e-commerce query suggestion. Online A/B testing also showcased that it can improve the diversity of query suggestions and attract more clicks while lowering the query's top click position, ultimately improving business conversions. To the best of our knowledge, it is the first large-scale industrial solution that can provide effective query suggestions via a unified end-to-end generation framework. Moreover, this method has been deployed for the entire traffic on the e-commerce search engine in Kuaishou platform, with millions of users and serving billions of retrieval PVs, for over 4 month. OneSug yields a 1.82% decrease in the average input length of prefixes, 9.33% in user top click position, 2.01% increase in CTR, 2.04% in Order, and a 1.69% improvement in total revenue, as well as an average reduction of 43.21% for system response time, thereby significantly enhancing e-commerce conversion.

Related Works

Generative Retrieval and Recommendation

Generative Retrieval (GRs) regards large-scale retrieval as sequence-to-sequence generation tasks, has outperformed traditional ANN-based models such as EBR (Huang et al. 2020), and spurs increased exploration in the fields of search and recommendation. Notable contributions in this area include Tiger (Rajput et al. 2023), DSI (Tay et al. 2022), and LC-REC (Zheng et al. 2024).

OneRec (Deng et al. 2025) introduces the first unified framework integrating recall, pre-ranking, and ranking within a single generative model. Through session-wise generation and iterative preference alignment, this approach achieves significant improvements in practical online metrics. In e-commerce search, OneSearch (Chen et al. 2025) provides the first end-to-end generative retrieval framework with superior performance for high-quality recall and ranking. EGA (Zheng et al. 2025) presents a novel unified framework that models the entire advertising pipeline. To tackle inconsistent code generation from varying word distributions, GRAM (Pang et al. 2025) boosts retrieval efficiency and pre-ranking accuracy beyond traditional methods. As shown in Figure 2, Query Suggestion has open-vocabulary inputs and outputs, while recommendation has closed-vocabulary ones. Thus, semantic IDs conveniently model input-output consis-

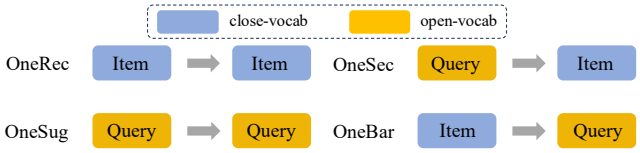

Figure 2: The input and output differences among Recommend, Search/Ads, Query Sug and Bottom Bar.

tency. However, in query suggestion, user expressions vary widely; using semantic IDs for prefixes and queries would cause slight differences to significantly alter the IDs.

Query Suggestion

Previous work on query suggestions uses co-occurrence statistics of prefix and complete query pairs. Of which MPC (Bar-Yossef and Kraus 2011) suggests top queries that start with the user input prefix through *trie*. Other traditional methods adopt user behaviors, term co-occurrence (Huang, Chien, and Oyang 2003), keyword clustering (Sadikov et al. 2010).

K-LAMP (Baek et al. 2024) utilizes LLM as a generative model and constructs an entity-centric knowledge storage, which is then fed to LLM prompt for personalization. TrieNLG (Maurya et al. 2023) integrates popularity signals from trie-based methods with personalization signals from previous session queries.

Despite significant progress in generative models, current approaches are still limited to the recall stage and fail to incorporate ranking signals into the GR models.

Methodology

Preliminary

In this section, we introduce the construction of the end-to-end generative framework for e-commerce query suggestion, from the perspectives of feature engineering. The input of OneSug consists of four parts: 1) user input prefix, denoted as p, which is usually an incomplete query, e.g., "smart" corresponding to a potential purchase intention of "smartphone"; 2) the collected prefix2query sequence, which acts as the augment references for p, denoted as $\mathcal{H}_p = \{q_1^a, q_2^a, \ldots, q_m^a\}$, where q^a represents the query from RQ-VAE and m is the length of the prefix2query sequence; 3) historical search queries $\mathcal{H}_u = \{q_1^h, q_2^h, \ldots, q_n^h\}$, where q^h represents the query that the user searched, and n is the length of the behavior sequence; 4) user profile information $\mathcal{U}$, which is the crowd portrait fitted by the platform. Then OneSug outputs the corresponding query lists $\mathcal{Q}$. OneSug can adopt either encoder-decoder models (e.g. BART (Mustar, Lamprier, and Piwowarski 2020), mT5 (Xue et al. 2020)), or the decoder-only models (e.g. Qwen2.5 (Qwen et al. 2025)) as the foundation. By leveraging the strong instruct-following capabilities and inherent e-commerce knowledge of these generative models, OneSug is expected to generate e-commerce intent queries that meet users' diverse requirements. For the following descriptions, OneSug is denoted as $\mathcal{M}$.

Prefix2Query Representation Enhancement

Prefix-Query Alignment Considering its strong generality across various Chinese NLP tasks, we utilize BGE (Xiao et al. 2023) as our initial representation model. However, its limited knowledge of the e-commerce domain necessitates integrating BGE with the real user-prefix-query interactions. Additionally, the fundamental divergence between prefixes and queries leads to misaligned representation spaces, obstructing further consistent optimization. Inspired by QARM (Luo et al. 2024), we aim to align representations and infuse latent retrieval knowledge to ensure that BGE can reflect real business characteristics. Specifically, we generate high-quality prefix2query and query2query pairs using existing retrieval models like ItemCF (Sarwar et al. 2001) and Swing (Yang et al. 2020), then select the semantically relevant pairs to enhance the performance. Using these high-quality data pairs, the aligned BGE model is trained as follows:

$$\begin{aligned} E_{\text{trigger}} &= \text{BGE}\left(T_{\text{trigger}}\right), \\ E_{\text{target}} &= \text{BGE}\left(T_{\text{target}}\right), \\ \mathcal{L}_{\text{align}} &= \text{Batch-Contrastive}\left(E_{\text{trigger}}, E_{\text{target}}\right). \end{aligned} \tag{1}$$

where T_{trigger} and T_{target} denote the original text of the trigger and target query, E_{trigger} and E_{target} represent their embeddings generated from learnable BGE, and $\mathcal{L}_{\text{align}}$ is our alignment training loss.

In the e-commerce query suggestion scenario, the prefixes entered by users are often relatively short (usually only one word), which leads to the ambiguity of prefix semantic representation. To enrich these prefixes with semantically and interactively related queries, we propose the prefix representation enhancement through prefix-query co-occurrence. Specifically, for a given prefix p , we obtain an augmented prefix embedding $e_p^* \in \mathbb{R}^d$ by using highly-related queries via the aligned BGE model:

$$\begin{aligned} \bar{e_q^c} &= \frac{1}{k}\sum_{i=1}^{k} e_{q_i}^c, \\ e_p^* &= (1-w)\cdot e_p + w \cdot \bar{e_q^c} \quad \text{where } w \in (0,1), \end{aligned} \tag{2}$$

where e_p is the original prefix embedding, $e_{q_i}^c$ is the embedding of the i-th query co-occurred with prefix p, $\bar{e_q^c}$ is the mean pooling result of these query embeddings, and e_p^* is the augmented prefix embedding adjusted by weight w. Here, w for OneSug training is set to 0.5.

In addition, to provide references for ambiguous prefixes, we also build a high-quality query pool. The embeddings of the prefix and the high-quality query via the aligned BGE model will be passed to the next stage.

Hierarchical Quantitative Semantic ID Generator In this stage, Semantic IDs are generated using RQ-VAE (Zeghidour et al. 2021), with the following objective:

$$\begin{aligned} \mathcal{L}(x) &:= \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{rqvae}}, \\ \mathcal{L}_{\text{recon}} &:= \|x - \widehat{x}\|^2, \\ \mathcal{L}_{\text{rqvae}} &:= \sum_{d=0}^{m-1} \|\text{sg}[r_i] - e_{c_i}\|^2 + \beta\|r_i - \text{sg}[e_{c_i}]\|^2, \end{aligned} \tag{3}$$

where x, $\widehat{x}$ is the input of the DNN encoder and the output of the DNN decoder, respectively. $sg[\cdot]$ is the stop-gradient operation, r_i is the i-th residual embedding, and e_{c_i} is the index of the closest centroid embedding.

Clustering-based Search After extracting the prefix's semantic ID via RQ-VAE, we identify top-k most relevant

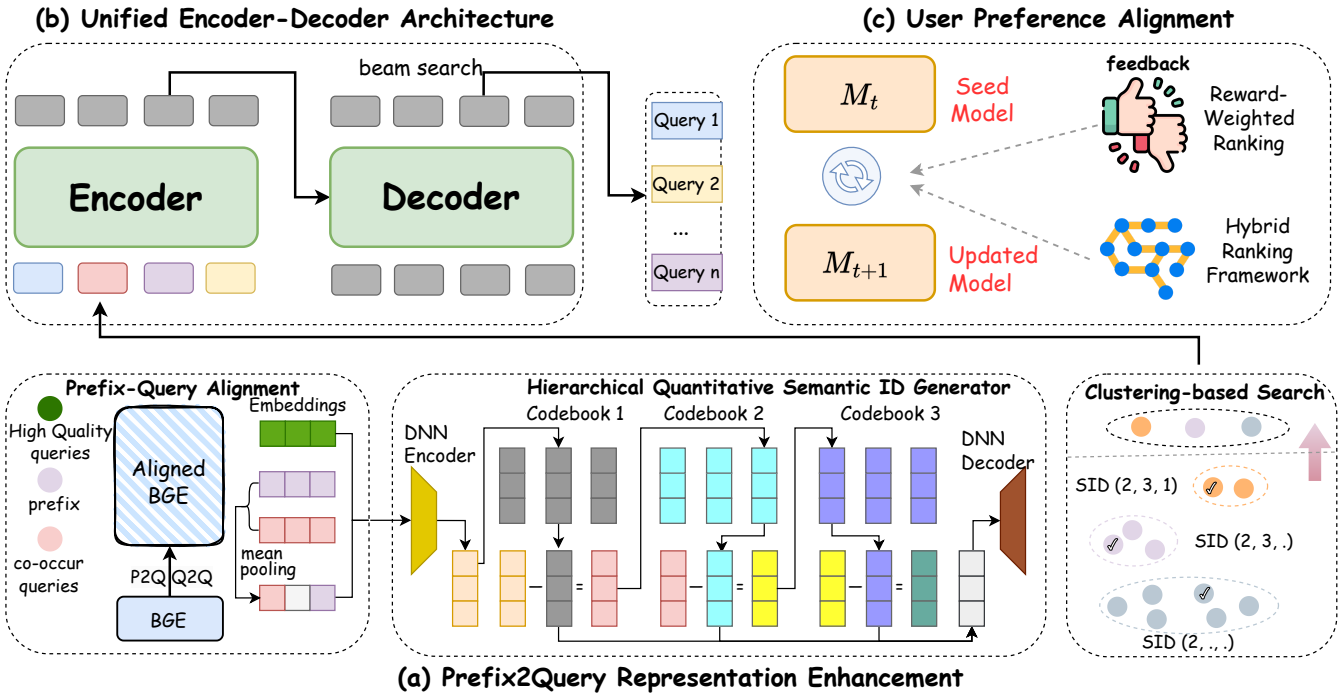

Figure 3: The OneSug framework comprises: (a) Prefix2Query Representation Enhancement, provides reference augment for ambiguous prefixes. (b) Unified Encoder-Decoder Architecture, generating queries autoregressively. (c) User Preference Alignment, which instills ranking ability into DPO training with the assistance of reward-weighted ranking (RWR) strategy.

queries through a clustering-based search. The fine-to-coarse hierarchical search operates in two stages: 1) Retrieving high-quality queries with exact Semantic ID matches in the fine-grained layer. 2) Incorporating queries with partial Semantic ID matches in subsequent coarse-grained layers. Within each coarse-grained layer, cluster ordering is determined by centroid distance between target and candidate clusters. Additionally, the top-k queries will be further screened based on diversity and relevance. This module addresses insufficient short prefix semantics while combining semantic and conversion features. RQ-VAE further reduces inference matching complexity, facilitating the deployment of the GR model.

Unified Encoder-Decoder Architecture

Instead of the online multi-stage cascading architecture, OneSug directly outputs the queries that users are most interested in based on the generative architecture using beam search. The output of OneSug model $\mathcal{M}$ is formalized as:

$$Q := \mathcal{M}(p, \mathcal{H}_p, \mathcal{H}_u, \mathcal{U}), \tag{4}$$

As illustrated in Figure 3(b), our model adheres to the transformer-based (Mustar, Lamprier, and Piwowarski 2020) architecture, comprising an encoder that models user historical interactions and a decoder dedicated to query generation. Specifically, the encoder utilizes stacked multi-head self-attention and feed-forward layers to process user input prefix p, prefix2query sequence $\mathcal{H}_p$, historical search queries $\mathcal{H}_u$, and user profile information $\mathcal{U}$, thus producing the encoded historical interaction features $\mathcal{H}$. It can be indicated as $\mathcal{H} = \text{Encoder}(p, \mathcal{H}_p, \mathcal{H}_u, \mathcal{U})$. The decoder takes the target query's tokens as the inputs and generates the query candidates auto-regressively. For the unified training, we insert a start token $t_{[\text{CLS}]}$ at the first place, and a separate token $t_{[\text{SEP}]}$ between adjacent elements to form the input to the encoder.

$$x_u = \{t_{[\text{CLS}]}, p, t_{[\text{SEP}]}, \mathcal{H}_p, t_{[\text{SEP}]}, \mathcal{H}_u, t_{[\text{SEP}]}, \mathcal{U}\}. \tag{5}$$

We utilize the cross-entropy loss for next-token prediction on the target query.After a certain amount of training steps on the generation task, we obtain the seed model $\mathcal{M}_t$.

User Preference Alignment

Reward-Weighted Ranking Inspired by RLHF's (Ouyang et al. 2022) success in NLP, we apply it to our LM-based search system to incorporate diverse user preferences for ranking. Traditional alignment methods (Rafailov et al. 2023) heavily rely on manually labeled samples and training-based reward models. However, these approaches face two key challenges: sensitivity to the quality of human annotations and significant training complexity introduced by the reward model. While OneRec (Deng et al. 2025) tackles the challenge of the sparsity of user-item interaction by proposing a personalized multi-task reward model, our solution instead utilizes feedback from online search systems as a more accessible source of reward signals. Specifically, we categorize the user interactive behaviors in the search system into six distinct levels, as detailed in Table 1. Then we assign the weighted reward score $r(x_u, q) = \lambda \cdot e^{pi}$ to each level, where λ is the base weight (set to [2.0, 1.5, 1.0, 0.5, 0.2, 0.0] for the six levels) to reflect the distribution of different samples, and e^{pi} is the ratio of each interactive query in the same level. So that the more times a <prefix, query>pair appears at the same level, the greater reward score is assigned. As a side

note, the reason why we did not use a CTR-based model as the reward model like OneRec(Deng et al. 2025) is not only that it requires hundreds or thousands of features to train a high-quality rank model, but also that it is not conducive to the optimization of the subsequent model's update, due to the high probability of online data deviation.

We take the positive samples among *Order*, *Item Click*, and *Click*, and negative samples among *Show*, *Not Show*, and *Rand*, then construct nine types of sampled <*positive*, *negative*>pairs(e.g. <*Order*, *Show*>). For each pair, the user's preference difference rw_Δ is computed as:

$$rw_\Delta = \frac{1.0}{r(x_u, q_w) - r(x_u, q_l)}, \tag{6}$$

where q_w is the chosen sample, q_l is the rejected sample. Bigger rw_Δ values encourage the model to distinguish the nuances of user interactive behaviors, and also the co-occurrence behavior of prefixes and queries.

Hybrid Ranking Framework Relying solely on pair-wise comparisons, Direct Preference Optimization (DPO) (Rafailov et al. 2023) struggles to learn absolute likelihood across different samples, thus leading to an absence of ranking capacity. With the assistance of RWR, the DPO obtains a certain ranking ability by dynamically adjusting the sample weights. For generalization, we also propose a target reward margin $\delta > 0$ to ensure the reward of the chosen sample exceeds the rejected one, at least δ. The pair-wise DPO loss can be optimized as follows:

$$\begin{aligned}&\mathcal{L}_{\text{pair-wise}}(\pi_\theta; \pi_{\text{ref}}) \\ &= -\mathbb{E}\Bigg[\log\sigma\Big(rw_\Delta\big(max(0, \hat{r}_\theta(x_u, q_w) - \hat{r}_\theta(x_u, q_l) - \delta)\big)\Big) \\ &\qquad + \alpha \log \pi_\theta(q_w|x_u)\Bigg],\end{aligned} \tag{7}$$

where

$$\hat{r}_\theta(x_u, q_{w/l}) = \beta \log \frac{\pi_\theta(q_{w/l}|x_u)}{\pi_{\text{ref}}(q_{w/l}|x_u)}, \tag{8}$$

$\hat{r}_\theta(x_u, q_w)$ and $\hat{r}_\theta(x_u, q_l)$ are the rewards implicitly defined by the language model π_θ and reference model π_{ref}. To avoid the model solely catering to the reward model at the expense of generation quality, we introduce $\log \pi_\theta(q_w|x_u)$, which is known as the negative log-likelihood (NLL) loss (Ouyang et al. 2022). Furthermore, rw_Δ is introduced to assign dynamic weights to different samples.

Conventional DPO is built with the Bradley-Terry (Bradley and Terry 1952) preference model, such a training paradigm fails to fully leverage user preference data and overlooks the minor differences of various negative samples for the same prefix, thereby impeding the alignment of LMs with user preferences. Inspired by S-DPO (Chen et al. 2024), we generalize the traditional Plackett-Luce(PL) (Plackett 1975) preference model, which is designed for full relative rankings, to accommodate partial rankings, a more natural fit for recommendation tasks. In detail, during the preference learning stage, rather than constructing a single <*positive*, *negative*>pairs, we pair input tokens with both positive and multiple negatives to build a more comprehensive preference dataset. Similarly, we devised a margin loss $\mathcal{L}_{\text{margin}}$ (Boser, Guyon, and Vapnik 1992) for list-wise modeling. By requiring a clear separation larger than δ, the model becomes less sensitive to minor variations or label noise in the training data. The list-wise DPO loss can be optimized as follows:

Feedback	Description
Order	User buys items through the specific query
Item Click	User clicks items through the specific query
Click	User clicks the specific query
Show	Specific query is shown in the panel
Not Show	Query is NOT shown but exists in recall
Rand	Random query in ranking candidates

Table 1: Online Feedback of Query Suggestion.

$$\begin{aligned}&\mathcal{L}_{\text{list-wise}}(\pi_\theta; \pi_{\text{ref}}) \\ &= -\mathbb{E}\left[\log\sigma\left(\log \sum_{q_l \in Q_l} \exp\left(rw_\Delta \mathcal{L}_{\text{margin}}\right)\right) + \alpha \log \pi_\theta\left(q_w|x_u\right)\right],\end{aligned} \tag{9}$$

where

$$\mathcal{L}_{\text{margin}} = \max\left(0, \hat{r}_\theta(x_u, q_w) - \hat{r}_\theta(x_u, q_l) - \delta\right) \tag{10}$$

Q_l is a set of negative samples. By combining list-wise preference alignment and the log-likelihood of the chosen sample, we created a new hybrid paradigm for generative ranking. Notably, when the number of candidates N is 2, which means there is only one negative item, $\mathcal{L}_{\text{list-wise}}$ reduces to $\mathcal{L}_{\text{pair-wise}}$. The proof is provided in the extended version (Guo et al. 2025).

Experiment

Experimental Settings

Datasets We extracted user interactive pairs from Kuaishou's online e-commerce logs between February 2025 and March 2025. It contains about 100 million PVs, and all the following offline and ablation experiments were conducted on the full or part of this data. The collections spanned 32 days, with the first 30 days used for model training and the last 2 days used as the test set. We will release an anonymized real-user dataset publicly upon corporate approval.

Evaluation Metrics Since OneSug is designed to make up for the limitations of traditional cascading architectures, here we take into account the recall and ranking performance. We employed HitRate@K and Mean Reciprocal Ranking (MRR) as the evaluation metrics, which are widely used in search and recommendation systems. All data presented were the average values for all tests.

Baseline Methods We compared OneSug with the following series of representative architectures.

- Multi-stage Cascading Architecture (MCA): BGE for recall, DCN (Wang et al. 2021) for pre-ranking, and DIN (Zhou et al. 2018) for ranking.
- Online Multi-stage Cascading Architecture (onlineMCA): the results of the online system, which include the recall, preranking, and ranking stages.
- Generative Retrieval Architecture (GRA): encoder-decoder architecture (BART (Mustar, Lamprier, and Piwowarski 2020), mT5 (Xue et al. 2020)) and the decoder-only architecture (Qwen2.5 (Qwen et al. 2025)).

Method	Click		Order	
	HR@16	MRR	HR@16	MRR
MCA	73.89%	39.95%	80.71%	44.03%
onlineMCA	78.61%	45.97%	84.55%	51.85%
GRA_{SFT}	73.16%	40.06%	79.25%	44.28%
GRA_{DPO}	75.50%	41.19%	81.68%	45.30%
OneSug_{Bart-B}	82.14%	50.55%	87.40%	56.34%
OneSug_{Bart-L}	82.84%	51.27%	88.12%	56.80%
OneSug_{mT5-S}	82.01%	50.40%	87.26%	55.87%
OneSug_{mT5-B}	83.63%	53.01%	88.19%	57.63%
$\text{OneSug}_{Q-0.5B}$	85.58%	55.34%	90.13%	60.00%
$\text{OneSug}_{Q-1.5B}$	89.60%	60.49%	94.95%	63.48%
OneSug_{Q-3B}	**93.37%**	**66.31%**	**95.13%**	**67.40%**

Table 2: Offline performances of our proposed OneSug on the industry dataset. The best results are in bold, and the results of onlineMCA are underlined.

Implementation Details The selected BGE version is *bge-base-zh-v1.5*. Considering that the online query suggestion system only displays 16 query candidates for each prefix at a time, the beam search size is set to 32. The batch size for SFT and DPO is set to 512 and 128. For RQ-VAE in the PRE module, the block number L of encoder and decoder is 3, the number of codebook layers $C = 4$, and the codebook size W of each layer is 512. We take 10 related queries for prefix representation enhancement and 10 historical clicked queries in user logs as context. For brevity, $\text{OneSug}_{Qwen2.5\text{-}xB}$ is abbreviated as $\text{OneSug}_{Q\text{-}xB}$ throughout the paper.

Offline Performance

For comprehensive evaluations, we took MCA and onlineMCA as the baselines. As depicted in Table 2, MCA achieves much lower Hitrate and MRR metrics than the online system, as many high-quality queries are eliminated in the initial recall and pre-ranking stages, thus lowering the upper bound of the next stage. OnlineMCA adopts multi-recall and complex (pre-)ranking strategies with hundreds of features, which can alleviate this problem to some extent. However, this operation will amplify the inference cost, significantly increase the system's response time, and degrade the user experience.

As for the GR models, GRA_{SFT}, GRA_{DPO} were trained with the input paradigm of <*prefix*, *historical sequence*, *user profile*>. DPO adopted the interactive queries as the positive and sampled the candidates with the lowest CTR score in the same PV as the negative. We can find DPO can enhance the preference modeling and get a higher performance. But GRA_{DPO} is still inferior to the online system, as it cannot deeply explore the semantic richness of prefixes and effectively distinguish the differentiated preferences contained in different user behavior levels. These limitations reduce the effectiveness of GR when applied to industry online engines.

OneSug series achieves the best overall performance. As an example, OneSug_{Bart-B} achieves improvements of 3.19% on HR@16 and 3.54% on MRR over onlineMCA. While for the larger model OneSug_{Q-3B}, the improvements can be remarkable at 12.67% and 17.95%. We also conduct further studies on the effects of different pre-trained architectures and model sizes. For encoder-decoder models, Bart and mT5 of similar size perform similarly, and larger models have better performance. It is more prominent in the decoder-only model, i.e, as the model performance significantly improves when the model size increases from 0.5B to 3B.

Method	IPL	TCP	CTR	Order
OneSug_{Pair}	-1.99%	-9.02%	+1.78%	+1.97%
OneSug_{List}	-1.82%	-9.33%	+2.01%	+2.04%

Table 3: Online results for A/B testing.

Method	Full recall	Page good	Query good
OneSug_{Pair}	+6.72%	+9.44%	+20.49%
OneSug_{List}	+8.48%	+11.02%	+22.51%
$\text{OneSug}_{Q-0.5B}$	+11.25%	+18.35%	+32.50%

Table 4: Manual evaluation results for online experience.

Online A/B Testing

To verify OneSug's online effectiveness, we compared it with onlineMCA in the e-commerce search engine of the Kuaishou platform through rigorous online A/B tests. It takes the short prefix the user entered as input and outputs the query candidates, where a query with a higher score would be listed in a more forward exposure position. We assessed the impact of each method on 1) the average input length of the prefix (IPL), 2) user top click position (TCP), 3) click-through rate (CTR), 4) average order volume (Order).

As indicated in Table 3, OneSug_{List} yields a 1.82% decrease in the average IPL, 9.33% in TCP, 2.01% increase in CTR, and finally 2.04% in Order, thereby significantly enhancing industry revenue. The OneSug_{Pair} model demonstrates weaker performance compared to its list-wise counterpart. While the larger $\text{OneSug}_{Q-0.5B}$ model shows more substantial performance gains in offline evaluations. However, the intensive computational demands limit their practical application in real-time search scenarios with stringent latency requirements. Ultimately, OneSug_{List} has been successfully deployed for the entire traffic in Kuaishou for over 4 month, which serves hundreds of millions of users generating billions of PVs.

We also conducted additional manual evaluations. The metrics are 1) page good rate - an evaluation indicator for the overall user experience, encompassing suggestion diversity, safety, relevance, and novelty dimensions. 2) full recall rate, an evaluation indicator for query coverage.3) query good rate - for each query, The results in Table 4 show that OneSug_{List} improves full recall by 8.48%, page good rate by 11.02%, and query good rate by 22.51%.

We estimated the average response time of the OneSug model, compared to the online system. As shown in Figure 4, OneSug can replace the multi-retrieval, (pre-)ranking stage,

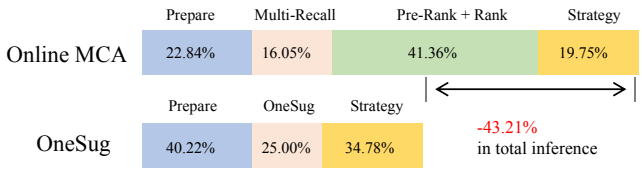

Figure 4: onlineMCA vs OneSug: System Response Time.

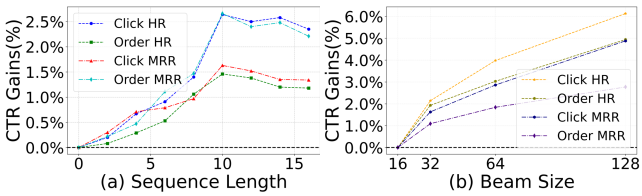

Figure 5: The ablation study of related query sequence length and beam size in inference.

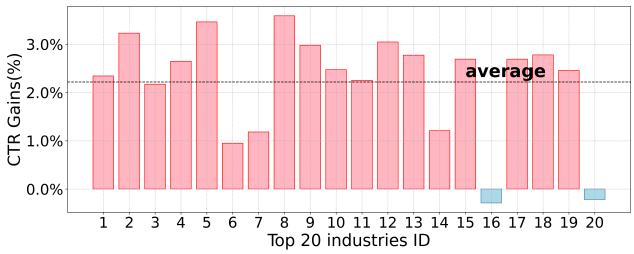

Figure 6: The CTR relative gains for top 20 industries.

and finally make an average reduction of 43.21%.

Ablation Study

Here the ablation study were conducted on OneSug$_{Bart-B}$. As shown in Table 5, the combination of prefix2query representation enhancement (PRE) and the reward-weighted ranking (RWR) module can improve the performance of the GR module with an average of 8.57% in HR@16, and 11.28% in MRR. We observed a marked improvement with the incremental application of margin loss and rw_Δ, showing the importance of assigning varying weights to RWR module. These all highlight that reward-weighted ranking derived from the level gap of user interactive behaviors can distinguish the nuances in user intent towards different items. The removal of the PRE module resulted in noticeable drops for Hitrate and MRR (-3.68% and -2.30%), demonstrating that interactive and semantically related queries are crucial for enhancing the prefix representation. This indicates that diverse negative samples help the model quickly learn the differences among various user interactive behavior levels, thereby achieving more concise and effective ranking results.

As depicted in Figure 5(a), longer sequences does not necessarily mean better, because too long user history sequences will introduce more disturbances. While the performance of OneSug continues to improve as the beam size increases, from 16 to 128, as plotted in Figure 5(b). These can induce that moderately longer sequences and bigger beam sizes may lead to more accurate predictions.

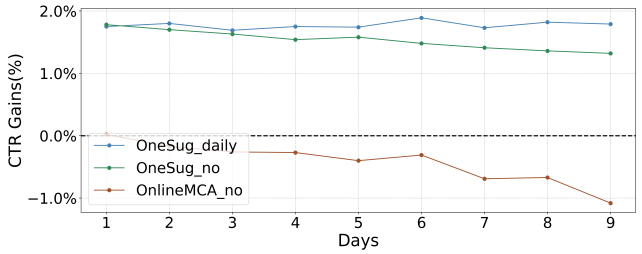

Figure 7: The CTR gains for models with updates or not.

Method	Click		Order	
	HR@16	MRR	HR@16	MRR
OneSug$_{List}$	**82.14%**	**50.55%**	**87.40%**	**56.34%**
OneSug$_{Pair}$	79.39%	47.42%	85.12%	53.01%
– w/o margin	78.81%	46.89%	84.62%	52.57%
– w/o rw_Δ	77.90%	44.41%	84.17%	49.18%
– w/o RWR	77.28%	42.28%	82.48%	46.66%

Table 5: Ablation study of OneSug in offline evaluations.

Method	Top	Middle	Long-tail
OneSug$_{Pair}$	+0.97%	+1.03%	+3.26%
OneSug$_{List}$	+1.15%	+1.32%	+3.59%

Table 6: Online CTR relative gains for prefix categories.

Further Analysis

We discuss questions about the deployment of OneSug.

1) **What are the main aspects of the online gains for the OneSug model?** Here we drilled down from the industries dimension and prefix popularity dimension. As shown in Figure 6, we computed the CTR relative gains for 20 industries. 18 of 20 can get the increases, with an average of 2.22%. Three industries were negatively affected, but these values are not significant. As for the prefix popularity dimension, we divided all prefixes into three categories: top (PV number daily larger than 1,000), middle (larger than 100 and less than 1,000), and long-tail (less than 100). The CTR relative gains for each were listed in Table 6. Prefixes of all categories are enhanced with either OneSug models, and the gains of the long-tail category are much larger than other categories.

2) **Does the OneSug model need to be updated regularly?** Models in the (pre-)ranking module of traditional multi-stage cascading architecture often needed to be updated regularly. As plotted in Figure 7, here we tested the robustness of onlineMCA and OneSug with no-daily updates (noted with "*_no*"). The baseline is a daily-updated onlineMCA. We can see that both OneSug_no and onlineMCA_no are decreased with the day increasing, but the decrease of the OneSug model is much smaller than that of onlineMCA, at -0.6% compared to -1.1%. We only use the data from the past three days to update the user preference alignment stage and find that we can maintain effective iterations of the model (noted as "*_daily*") with low costs. =Additional discussion is provided in the in the extended version (Guo et al. 2025)

Conclusion

In this paper, we present OneSug, an end-to-end generative framework for e-commerce query suggestion that effectively overcomes the limitations of traditional multi-stage cascading architecture. Extensive offline and online evaluations confirm OneSug's effectiveness in boosting query diversity, click-through rates, and business conversions. Furthermore, OneSug has been successfully deployed for the entire traffic on the e-commerce search engine of the Kuaishou platform for over 4 months. Its successful deployment underscores its practical applicability and potential to significantly enhance industry revenue.

References

Ahmad, W. U.; Chang, K.-W.; and Wang, H. 2018. Multi-task learning for document ranking and query suggestion. In *International conference on learning representations*.

Baek, J.; Chandrasekaran, N.; Cucerzan, S.; Herring, A.; and Jauhar, S. K. 2024. Knowledge-augmented large language models for personalized contextual query suggestion. In *Proceedings of the ACM Web Conference 2024*, 3355–3366.

Bar-Yossef, Z.; and Kraus, N. 2011. Context-sensitive query autocompletion. In *Proceedings of the 20th international conference on World wide web*, 107–116.

Bevilacqua, M.; Ottaviano, G.; Lewis, P.; Yih, S.; Riedel, S.; and Petroni, F. 2022. Autoregressive search engines: Generating substrings as document identifiers. *Advances in Neural Information Processing Systems*, 35: 31668–31683.

Boser, B. E.; Guyon, I. M.; and Vapnik, V. N. 1992. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, 144–152.

Bradley, R. A.; and Terry, M. E. 1952. Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika*, 39(3/4): 324–345.

Chen, B.; Guo, X.; Wang, S.; Liang, Z.; Lv, Y.; Ma, Y.; Xiao, X.; Xue, B.; Zhang, X.; Yang, Y.; et al. 2025. Onesearch: A preliminary exploration of the unified end-to-end generative framework for e-commerce search. *arXiv preprint arXiv:2509.03236*.

Chen, R.-C.; and Lee, C.-J. 2020. Incorporating behavioral hypotheses for query generation. *arXiv preprint arXiv:2010.02667*.

Chen, Y.; Tan, J.; Zhang, A.; Yang, Z.; Sheng, L.; Zhang, E.; Wang, X.; and Chua, T.-S. 2024. On Softmax Direct Preference Optimization for Recommendation. ArXiv:2406.09215 [cs].

Deng, J.; Wang, S.; Cai, K.; Ren, L.; Hu, Q.; Ding, W.; Luo, Q.; and Zhou, G. 2025. Onerec: Unifying retrieve and rank with generative recommender and iterative preference alignment. *arXiv preprint arXiv:2502.18965*.

Gharibshah, Z.; and Zhu, X. 2021. User response prediction in online advertising. *aCM Computing Surveys (CSUR)*, 54(3): 1–43.

Guo, X.; Chen, B.; Wang, S.; Yang, Y.; Lei, C.; Ding, Y.; and Li, H. 2025. OneSug: The Unified End-to-End Generative Framework for E-commerce Query Suggestion. *arXiv preprint arXiv:2506.06913*.

Huang, C.-K.; Chien, L.-F.; and Oyang, Y.-J. 2003. Relevant term suggestion in interactive web search based on contextual information in query session logs. *Journal of the American Society for Information Science and Technology*, 54(7): 638–649.

Huang, J.-T.; Sharma, A.; Sun, S.; Xia, L.; Zhang, D.; Pronin, P.; Padmanabhan, J.; Ottaviano, G.; and Yang, L. 2020. Embedding-based retrieval in facebook search. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2553–2561.

Ko, H.; Lee, S.; Park, Y.; and Choi, A. 2022. A survey of recommendation systems: recommendation models, techniques, and application fields. *Electronics*, 11(1): 141.

Luo, X.; Cao, J.; Sun, T.; Yu, J.; Huang, R.; Yuan, W.; Lin, H.; Zheng, Y.; Wang, S.; Hu, Q.; et al. 2024. QARM: Quantitative Alignment Multi-Modal Recommendation at Kuaishou. *arXiv preprint arXiv:2411.11739*.

Maurya, K. K.; Desarkar, M. S.; Gupta, M.; and Agrawal, P. 2023. trie-nlg: trie context augmentation to improve personalized query auto-completion for short and unseen prefixes. *Data Mining and Knowledge Discovery*, 37(6): 2306–2329.

Mustar, A.; Lamprier, S.; and Piwowarski, B. 2020. Using BERT and BART for query suggestion. In *Joint Conference of the Information Retrieval Communities in Europe*, volume 2621. CEUR-WS. org.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744.

Pang, M.; Yuan, C.; He, X.; Fang, Z.; Xie, D.; Qu, F.; Jiang, X.; Peng, C.; Lin, Z.; Luo, Z.; et al. 2025. Generative Retrieval and Alignment Model: A New Paradigm for E-commerce Retrieval. In *Companion Proceedings of the ACM on Web Conference 2025*, 413–421.

Plackett, R. L. 1975. The analysis of permutations. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 24(2): 193–202.

Qwen; :; Yang, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Li, C.; Liu, D.; Huang, F.; Wei, H.; Lin, H.; Yang, J.; Tu, J.; Zhang, J.; Yang, J.; Yang, J.; Zhou, J.; Lin, J.; Dang, K.; Lu, K.; Bao, K.; Yang, K.; Yu, L.; Li, M.; Xue, M.; Zhang, P.; Zhu, Q.; Men, R.; Lin, R.; Li, T.; Tang, T.; Xia, T.; Ren, X.; Ren, X.; Fan, Y.; Su, Y.; Zhang, Y.; Wan, Y.; Liu, Y.; Cui, Z.; Zhang, Z.; and Qiu, Z. 2025. Qwen2.5 Technical Report. arXiv:2412.15115.

Rafailov, R.; Sharma, A.; Mitchell, E.; Manning, C. D.; Ermon, S.; and Finn, C. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36: 53728–53741.

Rajput, S.; Mehta, N.; Singh, A.; Hulikal Keshavan, R.; Vu, T.; Heldt, L.; Hong, L.; Tay, Y.; Tran, V.; Samost, J.; et al. 2023. Recommender systems with generative retrieval. *Advances in Neural Information Processing Systems*, 36: 10299–10315.

Sadikov, E.; Madhavan, J.; Wang, L.; and Halevy, A. 2010. Clustering query refinements by user intent. In *Proceedings of the 19th international conference on World wide web*, 841–850.

Sarwar, B.; Karypis, G.; Konstan, J.; and Riedl, J. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, 285–295.

Tay, Y.; Tran, V.; Dehghani, M.; Ni, J.; Bahri, D.; Mehta, H.; Qin, Z.; Hui, K.; Zhao, Z.; Gupta, J.; et al. 2022. Transformer memory as a differentiable search index. *Advances in Neural Information Processing Systems*, 35: 21831–21843.

Wang, R.; Shivanna, R.; Cheng, D.; Jain, S.; Lin, D.; Hong, L.; and Chi, E. 2021. Dcn v2: Improved deep & cross network and practical lessons for web-scale learning to rank systems. In *Proceedings of the web conference 2021*, 1785–1797.

Xiao, S.; Liu, Z.; Zhang, P.; and Muennighoff, N. 2023. C-Pack: Packaged Resources To Advance General Chinese Embedding. arXiv:2309.07597.

Xu, J.; He, X.; and Li, H. 2018. Deep learning for matching in search and recommendation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 1365–1368.

Xue, L.; Constant, N.; Roberts, A.; Kale, M.; Al-Rfou, R.; Siddhant, A.; Barua, A.; and Raffel, C. 2020. mT5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

Yang, X.; Zhu, Y.; Zhang, Y.; Wang, X.; and Yuan, Q. 2020. Large scale product graph construction for recommendation in e-commerce. *arXiv preprint arXiv:2010.05525*.

Zeghidour, N.; Luebs, A.; Omran, A.; Skoglund, J.; and Tagliasacchi, M. 2021. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30: 495–507.

Zheng, B.; Hou, Y.; Lu, H.; Chen, Y.; Zhao, W. X.; Chen, M.; and Wen, J.-R. 2024. Adapting large language models by integrating collaborative semantics for recommendation. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, 1435–1448. IEEE.

Zheng, Z.; Wang, Z.; Yang, F.; Fan, J.; Zhang, T.; and Wang, X. 2025. EGA: A Unified End-to-End Generative Framework for Industrial Advertising Systems. *arXiv preprint arXiv:2505.17549*.

Zhou, G.; Zhu, X.; Song, C.; Fan, Y.; Zhu, H.; Ma, X.; Yan, Y.; Jin, J.; Li, H.; and Gai, K. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 1059–1068.