

LocalSUG: City-Preference-Enhanced LLM for Query Suggestion in Local-Life Services

Jinwen Chen^{1,2*}, Shiwen Zhang^{1,2*}, Shuai Gong², Zheng Zhang², Yachao Zhao²,
Lingxiang Wang¹, Haibo Zhou², Wei Lin², Hainan Zhang^{1†}

¹Beijing Advanced Innovation Center for Future Blockchain and Privacy Computing

²Meituan

Abstract

In local-life service platforms, query suggestion reduces user effort by generating candidate queries from input prefixes. Traditional multi-stage systems rely heavily on historical popular queries, limiting their ability to capture long-tail and emerging demand. Although LLMs provide strong semantic generalization, their deployment in local-life services faces three challenges: insufficient city-preference awareness, exposure bias in preference optimization, and strict online latency constraints. We propose LocalSUG, an LLM-based query suggestion framework for local-life services. LocalSUG mines city-preference-enhanced candidates from term co-occurrence and injects them into prompts as dynamic references rather than fusing them into model parameters. This allows the model to adapt to changing city preferences, such as merchant openings or closures, while reducing stale or locally invalid suggestions. We further introduce a beam-search-driven GRPO algorithm to align training with inference-time decoding and optimize relevance together with business-oriented rewards. Finally, quality-aware beam acceleration and vocabulary pruning reduce online latency while preserving generation quality. Offline evaluations and large-scale online A/B testing show that LocalSUG improves CTR by +0.35% and reduces the no-result rate by 3.98%, demonstrating its effectiveness in real-world deployment.

1 Introduction

Query suggestion (Mustar et al., 2021) is a critical component of search systems in local-life service platforms (Lan et al., 2025; He et al., 2026a; Wang et al., 2025a), where users seek city-dependent services such as restaurants, hotels, entertainment, and home services, as shown in Figure 1. By predicting candidate queries from user-typed prefixes, query

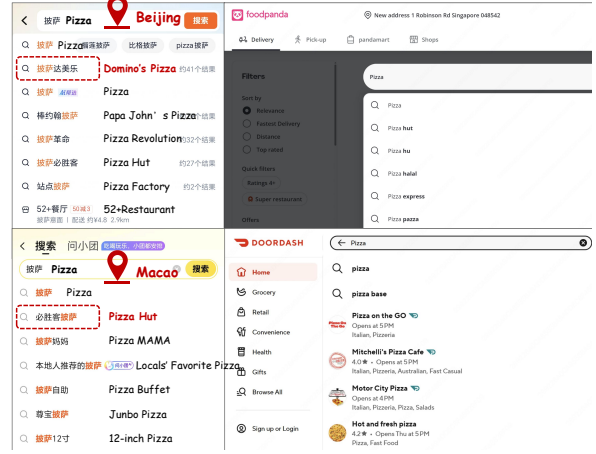


Figure 1: Examples of query suggestion in local-life service platforms.

suggestion reduces typing effort, accelerates search, and guides users toward high-quality intents.

Most production systems (Li et al., 2022; Zhang et al., 2023; Covington et al., 2016; Qin et al., 2022) adopt a Multi-stage Cascading Architecture following a recall–ranking paradigm. While robust and scalable, such systems rely heavily on historical query logs and fixed candidate pools. As a result, they struggle to explore long-tail, compositional, or emerging intents that are insufficiently represented in past data. Recent generative approaches, such as OneSug (Guo et al., 2026), replace retrieval with large language models (LLMs) that directly synthesize suggestions. Owing to their strong semantic generalization ability, LLMs can generate coherent and previously unseen suggestions, offering a promising solution to long-tail sparsity.

However, directly deploying LLMs for query suggestion in local-life platforms introduces three key challenges. **1) Insufficient city-preference awareness:** User intents differ across cities due to merchant availability, regional trends, and local habits. As shown in Figure 1, “pizza” may suggest Domino’s in Beijing but be invalid in cities with-

*Equal contribution.

†Corresponding author: zhanghainan1990@163.com

out available stores. As city preferences change with merchant openings and closures, LLMs without city-specific references may produce locally invalid suggestions, degrading user experience and business performance. **2) Exposure bias in preference optimization:** Existing generative methods use preference optimization (e.g., S-DPO (Chen et al., 2024)) trained at the sequence level on historical logs, but real-world systems generate and rank suggestions list-wise via beam search, creating a mismatch that can harm coherence and ranking consistency. **3) Online inference latency:** Local-life search systems operate under strict real-time latency requirements. High-capacity LLMs provide strong generative quality but incur substantial computational overhead, making naive deployment infeasible for large-scale production environments.

To address these challenges, we propose LocalSUG, an end-to-end generative query suggestion framework tailored for local-life service platforms. First, we design a city-preference-enhanced candidate mining strategy based on term co-occurrence. Instead of directly encoding city preferences into model parameters, LocalSUG injects daily updated city-specific candidates into the prompt as external references, allowing the model to follow current city preferences while avoiding stale memorization. Second, we introduce a Beam-Search-Driven GRPO algorithm that aligns training with inference-time decoding and optimizes sequence-level objectives through a multi-objective reward mechanism balancing relevance and business metrics. Finally, we develop Quality-Aware Accelerated Beam Search (QA-BS) and vocabulary pruning techniques to substantially reduce online latency while preserving generation quality, enabling the large-scale deployment of a 0.6B-parameter LLM in high-concurrency industrial environments.

Extensive experiments on real-world logs show that LocalSUG consistently outperforms strong baselines on HitRate@K, MRR, Diversity, and Quality, while the proposed acceleration strategy achieves a $3.11\times$ decoding speedup. In online A/B tests, LocalSUG reduces the no-result rate by 3.98%, boosts PV CTR by 0.35%, and increases Unique Item Exposure by 7.14%, demonstrating its effectiveness in real-world deployment.

The main contributions are as follows:

- We propose a production-ready city-preference-enhanced query suggestion framework tailored for local-life services,

enabling reliable generation under diverse and evolving city preferences.

- We introduce Beam-Search-Driven GRPO to bridge offline training and real-time list-wise inference, improving ranking stability, diversity, and consistency in production settings.
- We introduce QA-BS and vocabulary pruning techniques to overcome latency bottlenecks, enabling the large-scale deployment of a 0.6B-parameter LLM in high-concurrency industrial environments.
- Extensive offline experiments and online A/B tests verify that LocalSUG significantly reduces no-result cases and improves CTR, demonstrating its superior effectiveness for query suggestion.

2 Methodology

2.1 System Overview

We propose a unified generative framework for local-life services (e.g., food delivery) using **Qwen3-0.6B** (Team, 2025) as the backbone to balance contextual intelligence with strict latency requirements. The overall architecture is illustrated in Figure 2, which consists of three core phases: (a) candidate mining and multi-source context construction; (b) Beam-Search-Driven GRPO training for multi-objective optimization; and (c) quality-aware inference acceleration for online deployment. The model $\mathcal{M}$ processes a concatenated input $\mathcal{X}$:

$$\mathcal{X} = \text{Concat}(p, \mathcal{C}_{\text{cand}}, \mathcal{W}_{\text{hot}}, \mathcal{H}_{\text{beh}}, \mathcal{U}_{\text{prof}}) \quad (1)$$

consisting of: 1) user prefix p ; 2) candidate queries $\mathcal{C}_{\text{cand}} = \{q_1^c, \dots, q_m^c\}$ fetched from a city-preference-enhanced co-occurrence cache; 3) trending hot words $\mathcal{W}_{\text{hot}}$ from internal platform services; 4) behavior history $\mathcal{H}_{\text{beh}}$; and 5) user profile $\mathcal{U}_{\text{prof}}$. The model generates a ranked list $\mathcal{Q} = [q_1, \dots, q_K]$ via beam search with width K . A seed model $\mathcal{M}_t$ is obtained through standard SFT.

2.2 Prefix-to-Query Candidate Mining

City preferences in local-life services evolve continuously with merchant openings and closures, seasonal trends, and regional consumption shifts. Directly fusing such preferences into model parameters would require frequent retraining and may

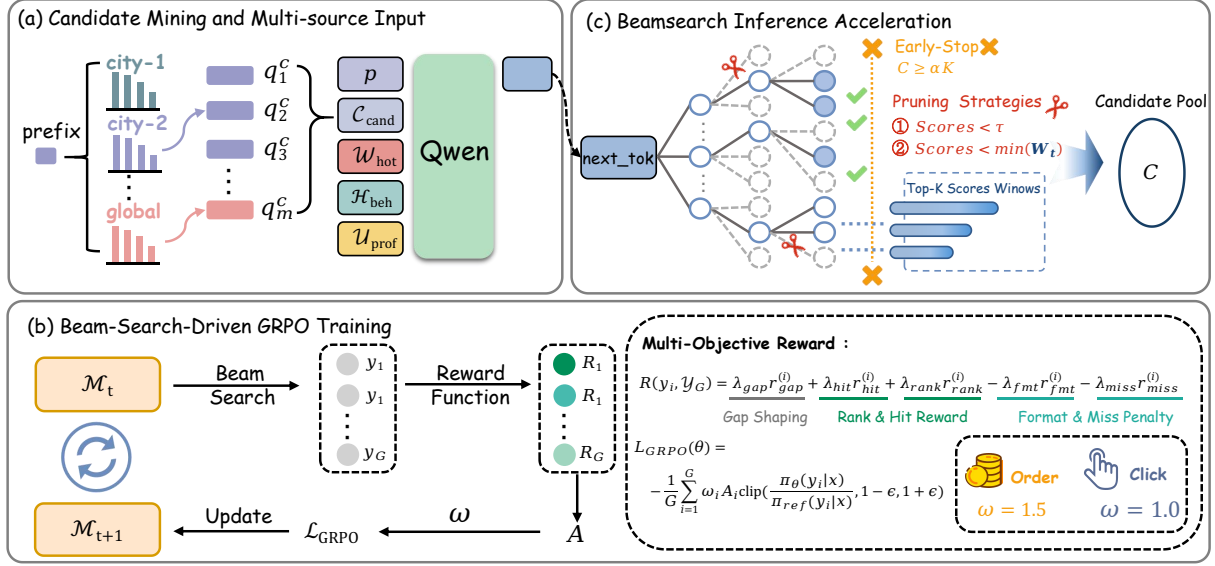


Figure 2: Overview of our proposed generative Query Suggestion framework. (a) **Candidate Mining and Multi-Source Input**: Prefix-to-query candidates are retrieved via city-preference-enhanced and global co-occurrence rules, then concatenated with hot words, user history, and profiles. (b) **Beam-Search-Driven GRPO Training**: The model is optimized using group relative rewards to bridge the gap between training and inference, incorporating hit, rank, and format objectives. (c) **Beam Search Inference Acceleration**: Online performance is enhanced through Quality-Aware Accelerated Beam Search and LLM head pruning.

cause stale memorization. For instance, if all stores of a chain brand close in a city, a parameterized model may still generate that brand as a locally plausible suggestion. We therefore treat city preferences as dynamic external references and inject them into the prompt through candidate queries.

Specifically, we adopt a statistical co-occurrence strategy rather than RQ-VAE (Lee et al., 2022), which introduces online inference overhead and is sensitive to vocabulary distribution shift. We directly extract prefix-query co-occurrence from historical logs. For each prefix p , we maintain two categories of sorted candidate lists based on click counts: **city-specific lists** $\mathcal{L}_p^{\text{city}}$ for each city and a **global list** $\mathcal{L}_p^{\text{global}}$. During prompt construction, we prioritize $\mathcal{L}_p^{\text{city}}$ to reflect current city preferences; if city-specific candidates are insufficient for m slots, we backfill using top queries from $\mathcal{L}_p^{\text{global}}$. This design preserves local precision while ensuring coverage robustness. To capture temporal changes, the co-occurrence candidate pool is updated daily using a sliding window of the past seven days’ logs.

2.3 Beam-Search-Driven GRPO

To bridge the training-inference gap, we employ Beam-Search-Driven Group Relative Policy Optimization (GRPO). Let π_θ denote the policy model and π_{ref} denote the reference model. For each

input x , we sample a group of G outputs $\mathcal{Y}_G = \{y_1, \dots, y_G\}$ based on beam search from the policy π_θ , where $G > K$. The modified GRPO loss is:

$$\mathcal{L}_{GRPO}(\theta) = -\frac{1}{G} \sum_{i=1}^G \omega_i \cdot \mathbb{A}_i \cdot \text{clip} \left(\frac{\pi_\theta(y_i|x)}{\pi_{ref}(y_i|x)}, 1 - \epsilon, 1 + \epsilon \right) \quad (2)$$

where $\mathbb{A}_i = (R(y_i, \mathcal{Y}_G) - \bar{R}) / (\sigma_R + \delta)$ and $\omega_i = \lambda_{order}$ if x leads to a conversion (else 1.0). The unified reward function $R(y_i, \mathcal{Y}_G)$ combines five objectives:

$$R(y_i, \mathcal{Y}_G) = \underbrace{\lambda_{gap} \cdot r_{gap}^{(i)}}_{\text{Gap Shaping}} + \underbrace{\lambda_{hit} \cdot r_{hit}^{(i)}}_{\text{Target Hit}} + \underbrace{\lambda_{rank} \cdot r_{rank}^{(i)}}_{\text{Rank Bonus}} - \underbrace{\lambda_{fmt} \cdot r_{fmt}^{(i)}}_{\text{Format Penalty}} - \underbrace{\lambda_{miss} \cdot r_{miss}^{(i)}}_{\text{Miss Penalty}} \quad (3)$$

Key mechanisms: 1) **Gap Shaping**: We penalize the tail $G - K$ candidates to force high-quality results into the top- K beam. 2) **Rank & Hit**: If ground truth y^* exists in $\mathcal{Y}_G$, we assign a hit bonus and a rank reward $1 / \log_{10}(\text{rank}(y^*) + 1)$, while penalizing non-target candidates ranked higher than y^* . 3) **Format & Miss**: If y^* is missing or format errors (repetition/invalid tokens) occur in the

top- K , we penalize the errors and boost valid candidates from the tail to fill the K slots. The detailed calculation process is described in Appendix A.

2.4 Beam Search Inference Acceleration

To facilitate online deployment, we propose **Quality-Aware Accelerated Beam Search (QA-BS)**, see Algorithm 2 for the detailed inference logic):

- **Quality Verification:** Active beams are pruned if cumulative log-probability $S(y) < \tau$. Completed sequences are accepted only if $S(y) > \min(W_t)$, where W_t are the current top- K scores.
- **Adaptive Termination:** Early exits are triggered if valid candidates saturate capacity ($|C| \geq \alpha K$) or if all active beams fall below τ (Fail-Safe), preventing high-latency tail processing.

Furthermore, statistical analysis of 38 million log entries reveals sparse token frequency. We pruned the LM head to the top 30,000 most frequent tokens, significantly reducing computation for irrelevant tokens with minimal impact on accuracy.

3 Experiments

3.1 Experimental Settings

Datasets We collected 8 consecutive days of real-world exposure and click logs from our local-life service platform. The first 7 days served as the training set (3.5M samples), and the 8th day as the test set (50k samples). For evaluation, we constructed three subsets from the test data: 1) MIX: 10,000 randomly sampled instances; 2) CLICK: 10,000 instances with user clicks; and 3) ORDER: 5222 instances leading to a transaction.

Metrics We evaluate performance using **HitRate@K** (HR@K) and **MRR** (Mean Reciprocal Rank). Additionally, we report **Diversity (DIV)**, defined as the ratio of unique queries to total generated queries, and **Quality (QUA)**, measuring the proportion of generated queries that are free from formatting errors (e.g., garbled text, invalid tokens) and, crucially, contain no *intra-list duplicates* within the top- K outputs for a single input. See Appendix C for further details.

Baselines We compare LocalSUG with four groups of baselines: (1) industrial cascading systems, including MCA and its city-enhanced variant City-MCA; (2) location-aware search baselines adapted from web search, including LMP (Bennett et al., 2011) and SPQR (Huang et al., 2018); (3) generative query suggestion baselines, including OneSug (Guo et al., 2026) and OneSug_{rule}; and (4) LocalSUG variants, including DPO-based training (Rafailov et al., 2023), a Llama backbone, and different GRPO sampling strategies. Detailed descriptions of all baselines are provided in Appendix D.

Implementation Details Experiments were conducted on H20-141G. We utilized **Qwen3-0.6B** (Team, 2025) as the backbone. For SFT, we used 4×H20 GPUs with a global batch size of 2048. For GRPO, the per-device batch size was set to 8, with a group size $G = 16$ (refer to Appendix E for full parameters). Notably, we set the KL penalty coefficient $\beta = 0$, relying solely on the clipping mechanism for training stability.

3.2 Offline Performance

Table 1 presents the offline performance comparison. While MCA and City-MCA maintain high generation quality (QUA), their recommendation utility (HR@12, MRR) and diversity (DIV) are substantially inferior to generative approaches. City-MCA improves MRR over MCA, confirming that city-specific candidate pools benefit ranking, but it remains constrained by fixed historical candidates. LMP and SPQR further demonstrate the value of location-related signals, yet their reliance on explicit metadata or spatial reformulation limits semantic generalization in local-life query suggestion. LocalSUG consistently outperforms the SFT baseline across all metrics, validating that Beam-Search-Driven GRPO effectively bridges the training-inference gap and enhances multi-objective synergy. Notably, LocalSUG significantly surpasses the original OneSug in both HR@12 and MRR. The performance leap of OneSug_{rule} over OneSug indicates that co-occurrence-based candidates provide stronger dynamic references than RQ-VAE-based strategies in latency-sensitive scenarios. The effects of candidate prompts on performance and the extent to which the LLM creates novel suggestions beyond the provided candidates are further analyzed in Appendix G.1. For further performance analysis

Table 1: Offline performance comparison between LocalSUG and various baselines. The best results are highlighted in **bold**, and the second-best results are underlined. For more comparisons across different LocalSUG variants, please refer to Appendix F.

Method	CLICK				ORDER			
	HR@12↑	MRR↑	DIV↑	QUA↑	HR@12↑	MRR↑	DIV↑	QUA↑
MCA	74.24%	40.62%	62.85%	<u>99.94%</u>	76.64%	42.78%	63.80%	<u>99.93%</u>
City-MCA	74.43%	45.93%	61.00%	99.95%	75.77%	47.87%	61.92%	99.96%
LMP	79.67%	40.81%	53.57%	99.75%	79.66%	39.99%	67.81%	99.84%
SPQR	95.95%	70.53%	45.49%	99.83%	96.28%	70.84%	63.75%	99.85%
SFT	96.07%	79.36%	63.67%	90.43%	96.63%	79.39%	61.91%	90.80%
OneSug	73.19%	54.83%	72.18%	99.50%	77.67%	59.26%	70.28%	99.52%
OneSug _{rule}	96.47%	<u>80.69%</u>	68.62%	96.17%	<u>97.07%</u>	<u>81.25%</u>	66.61%	96.54%
LocalSUG	<u>96.36%</u>	81.13%	<u>71.49%</u>	98.55%	97.09%	81.80%	<u>69.31%</u>	98.78%

Table 2: Ablation Study of LocalSUG in Offline Evaluations. The best results are in bold.

Method	CLICK				ORDER			
	HR@12↑	MRR↑	DIV↑	QUA↑	HR@12↑	MRR↑	DIV↑	QUA↑
LocalSUG	96.36%	81.13%	71.49%	98.55%	97.09%	81.80%	69.31%	98.78%
- w/o r_{gap}	95.20%	80.97%	73.31%	97.37%	96.02%	81.66%	72.58%	97.71%
- w/o r_{hit}	95.88%	81.09%	72.45%	97.78%	96.74%	81.74%	71.44%	98.45%
- w/o r_{rank}	96.58%	80.00%	70.73%	98.99%	97.07%	80.59%	68.19%	96.83%
- w/o r_{fmt}	96.03%	79.32%	63.62%	90.36%	96.74%	77.59%	62.91%	89.85%
- w/o r_{miss}	95.62%	81.04%	72.73%	97.49%	96.38%	81.71%	72.02%	97.82%
- w/o ω	96.32%	81.11%	71.62%	98.50%	96.72%	80.63%	71.55%	98.24%

across different LocalSUG variants, please refer to Appendix F.

3.3 Ablation Study

Table 2 presents the ablation results on reward components. Most ablations reduce HR, while all of them reduce MRR. Specifically, r_{rank} and r_{fmt} are critical for ranking capability. The absence of r_{fmt} significantly hampers overall performance, likely because the model tends to overfit on high-confidence but malformed low-quality tokens without format constraints. Furthermore, removing the order-aware weight ω impedes the model’s ability to prioritize high-value conversion samples, resulting in suboptimal performance on the ORDER subset.

3.4 Online Performance

3.4.1 Online A/B Testing

We deployed LocalSUG in production for a one-month A/B test covering 10% of real-world traffic.

Table 3: Online A/B Testing Results of LocalSUG. Significance levels: * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$.

Category	Metric	Relative Change
User Experience	No-Result Rate↓	−3.98%***
Diversity	Unique Item Exposure↑	+7.14%***
	Category Diversity↑	+1.28%***
Efficiency	Session CTR↑	+0.27%***
	PV CTR↑	+0.35%***
User Effort	Average SUG Input Length↓	−0.27%**

Furthermore, we implemented a fully automated pipeline for daily incremental updates, offline evaluation, and deployment. Requiring minimal manual intervention, this pipeline completes a full update cycle in approximately 9 hours using two H20 GPUs, ensuring high responsiveness to evolving user behaviors.

As shown in Table 3, LocalSUG achieves significant improvements. Existing models suffer from data distribution bias and shallow behavioral co-occurrence, which limit diversity and intent capture.

Table 4: Online A/B experiment results across business categories.

Business Category	PV CTR	Session CTR	Excl.-Orig. Session CTR	Avg. L2 Cat.	UV CTR
Life Event	+3.15%	+2.74%	+3.97%	+3.50%	+1.75%
Beauty & Medical	+0.32%	+0.62%	+0.93%	+3.74%	+0.68%
Transportation	+19.09%	+15.17%	+18.93%	+8.93%	+13.86%
Parenting & Education	+3.81%	+2.94%	+3.28%	+1.86%	+1.94%
Entertainment	+0.79%	+0.65%	+0.76%	+3.36%	+0.38%
Medicine	+2.69%	+1.36%	+1.90%	+0.91%	+1.33%
Merchandise	+3.16%	+2.43%	+3.01%	+1.75%	+1.65%
Travel	+1.29%	+1.06%	+1.64%	+2.05%	+1.27%
Unknown	-0.90%	+0.26%	+0.33%	+4.42%	+0.08%
Movie	+0.63%	+0.73%	+1.42%	+4.00%	+0.32%
Hotel & Homestay	+1.48%	+0.34%	+0.15%	+2.15%	+1.19%
Non-transaction	+6.03%	+4.80%	+10.50%	+5.28%	+3.98%
Dining	-0.56%	+0.12%	+0.30%	+3.34%	+0.30%

LocalSUG mitigates these limitations by expanding the query space and uncovering latent long-tail demand. The online results show that the no-result rate decreases by 3.98%. System diversity also improves: Unique Item Exposure and Category Diversity increase by 7.14% and 1.28%, respectively. By better capturing user intent, LocalSUG improves semantic matching. Consequently, Session CTR and PV CTR increase by 0.27% and 0.35%, respectively, while the average SUG input length decreases by 0.27%. These results demonstrate that users can find relevant results with fewer interactions, thereby improving the overall search experience.

3.4.2 Performance across Business Categories

To further evaluate the generalizability of our method, we conduct a fine-grained analysis across different business categories. Table 4 reports the relative improvements over the baseline.

Our method yields positive results across nearly all business categories. Notably, the Transportation and Non-transaction categories exhibit the most substantial improvements, with PV CTR increasing by +19.09% and +6.03%, respectively. This indicates that our approach generalizes well across diverse service domains.

3.4.3 Performance across City Tiers

To examine whether our method benefits cities of different development levels, we conduct a comparative analysis across four city tiers (S, A, B, C), where S-tier represents the most developed cities

Table 5: Online A/B experiment results across city tiers.

City Tier	PV CTR	Session CTR	Excl.-Orig. Session CTR	Avg. L2 Cat.
S-tier	+0.01%	+0.05%	+0.14%	+1.35%
A-tier	0.00%	+0.15%	+0.13%	+1.27%
B-tier	+0.35%	0.00%	-0.02%	+1.29%
C-tier	+0.63%	+0.48%	+0.72%	+1.60%

and C-tier the least. Table 5 reports the relative improvements of our method over the baseline.

Our method shows overall positive gains across all city tiers, with stronger advantages in less-developed markets (C-tier). This suggests that the approach is particularly effective in less-developed markets, where query diversity and category coverage yield greater marginal value.

3.5 Analysis

3.5.1 Efficiency of QA-BS and Pruning

As shown in Figure 3, QA-BS significantly accelerates inference, increasing the decoding speed from 1.41 to 3.77 samples/s. With vocabulary pruning, the speed is further improved to 4.38 samples/s, resulting in an overall $3.11\times$ speedup relative to standard beam search. Meanwhile, the acceleration introduces only negligible degradation in HitRate and MRR, with diversity and generation quality remaining stable, demonstrating that the proposed acceleration strategy effectively improves efficiency without compromising model performance.

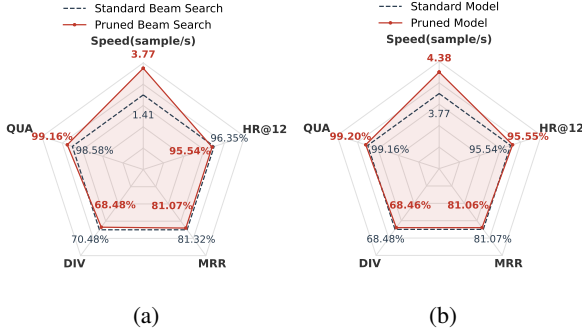


Figure 3: Impact of (a) QA-BS and (b) vocabulary pruning on inference latency and model performance.

Table 6: Human Ratings of Generated Queries.

	Bad	Fair	Good
Control Group	2.38%	0.91%	96.71%
LocalSUG	1.55%	0.50%	97.95%

3.5.2 Evaluation of User Experience

To strictly measure user satisfaction, we conducted a human evaluation involving three professional annotators located in different cities across the country. The evaluation was performed on an internal annotation platform, where annotators were blind to the source of suggestions (online system or our model) and rated only perceived relevance and diversity. Each sample was independently rated by three annotators, and only unanimous results were retained, yielding a final set of 2,000 samples per method (onlineMCA and LocalSUG). The suggestion queries were rated as Bad, Fair, or Good. As detailed in Table 6, LocalSUG achieved a higher proportion of ‘Good’ ratings and a lower rate of ‘Bad’ and ‘Fair’ outputs compared to onlineMCA, thereby verifying its effectiveness in enhancing user experience.

4 Related Work

Traditional recommendation methods heavily rely on feature interaction design and engineering, optimizing performance through meticulously constructed model architectures (Boldi et al., 2008; Cheng et al., 2016; Guo et al., 2017; Naumov et al., 2019; Zhou et al., 2018, 2019; Fan et al., 2019). Recent research has introduced generative approaches in specific components such as recall and ranking, breaking through the expressive limitations of traditional models (Rajput et al., 2023; Ren et al., 2024; Yang et al., 2025; Wang et al., 2025b; Kong et al.,

2025; Shi et al., 2025; Chen et al., 2025; He et al., 2026b; Han et al., 2025).

End-to-end generative recommendation is replacing traditional pipelines with unified models. Following the pioneering framework design (Deng et al., 2025), recent studies have further extended this paradigm through advanced architectures (Qiu et al., 2025; Zheng et al., 2025) and system-level optimizations for better quality and practicality (Zhou et al., 2025; Agarwal et al., 2025; Li et al., 2025). Additionally, researchers have adapted generative models to specific tasks, addressing explainability, local services, and semantic integration (Xing et al., 2026; Wei et al., 2026; Wang et al., 2025b).

5 Conclusion

In this paper, we propose LocalSUG, an end-to-end generative framework for low-latency query suggestion in local-life services. LocalSUG injects term co-occurrence-based city-specific candidates into prompts rather than model parameters, enabling adaptation to evolving city preferences while reducing stale or locally invalid suggestions. To mitigate exposure bias, we introduce Beam-Search-Driven GRPO with multi-objective rewards, and further combine Quality-Aware Accelerated Beam Search with vocabulary pruning for efficient online deployment. Extensive offline evaluations and online A/B testing demonstrate LocalSUG’s effectiveness in improving query suggestion, breaking feedback loops, and satisfying long-tail demands in industrial ecosystems.

Limitations

Despite its effectiveness, LocalSUG has limitations that motivate future work. First, city-preference-enhanced candidate mining depends on historical logs. Although daily co-occurrence candidates capture evolving preferences, they may not respond quickly to newly opened merchants, temporary closures, or sudden trend shifts. We therefore plan to integrate dense semantic retrieval, real-time merchant status, and external knowledge to address cold-start and rapidly changing scenarios. Second, GRPO relies on manual reward shaping, with coefficients balancing diversity and business metrics. Future work will explore automated or dynamic weight adjustment to achieve a more robust Pareto-optimal balance across different service categories.

Acknowledgments

This work was funded by the National Natural Science Foundation of China (NSFC) under Grants No. 62406013 and No. U25B2070, the Beijing Advanced Innovation Center Funds for Future Blockchain and Privacy Computing, and the Fundamental Research Funds for the Central Universities.

References

- Prabhat Agarwal, Anirudhan Badrinath, Laksh Bhasin, Jaewon Yang, Jiajing Xu, and Charles Rosenberg. 2025. [Autoregressive generative retrieval for industrial-scale recommendations at pinterest](#). In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, CIKM '25, page 6861–6862, New York, NY, USA. Association for Computing Machinery.
- Paul N. Bennett, Filip Radlinski, Ryen W. White, and Emine Yilmaz. 2011. [Inferring and using location metadata to personalize web search](#). In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, page 135–144, New York, NY, USA. Association for Computing Machinery.
- Paolo Boldi, Francesco Bonchi, Carlos Castillo, Debora Donato, Aristides Gionis, and Sebastiano Vigna. 2008. [The query-flow graph: model and applications](#). In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, page 609–618, New York, NY, USA. Association for Computing Machinery.
- Xiangyi Chen, Kousik Rajesh, Matthew Lawhon, Zelun Wang, Hanyu Li, Haomiao Li, Saurabh Vishwas Joshi, Pong Eksombatchai, Jaewon Yang, Yi-Ping Hsu, Jiajing Xu, and Charles Rosenberg. 2025. [Pinfm: Foundation model for user activity sequences at a billion-scale visual discovery platform](#). In *Proceedings of the Nineteenth ACM Conference on Recommender Systems*, RecSys '25, page 381–390, New York, NY, USA. Association for Computing Machinery.
- Yuxin Chen, Junfei Tan, An Zhang, Zhengyi Yang, Leheng Sheng, Enzhi Zhang, Xiang Wang, and Tat-Seng Chua. 2024. On softmax direct preference optimization for recommendation. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA. Curran Associates Inc.
- Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. [Wide & deep learning for recommender systems](#). In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, DLRS 2016, page 7–10, New York, NY, USA. Association for Computing Machinery.
- Paul Covington, Jay Adams, and Emre Sargin. 2016. [Deep neural networks for youtube recommendations](#). In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, page 191–198, New York, NY, USA. Association for Computing Machinery.
- Jiaxin Deng, Shiyao Wang, Kuo Cai, Lejian Ren, Qigen Hu, Weifeng Ding, Qiang Luo, and Guorui Zhou. 2025. [Onerec: Unifying retrieve and rank with generative recommender and iterative preference alignment](#). Preprint, arXiv:2502.18965.
- Miao Fan, Jiacheng Guo, Shuai Zhu, Shuo Miao, Mingming Sun, and Ping Li. 2019. [Mobius: Towards the next generation of query-ad matching in baidu's sponsored search](#). In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 2509–2517, New York, NY, USA. Association for Computing Machinery.
- Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. Deepfm: a factorization-machine based neural network for ctr prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 1725–1731. AAAI Press.
- Xian Guo, Ben Chen, Siyuan Wang, Ying Yang, Mingyue Cheng, Chenyi Lei, Yuqing Ding, and Han Li. 2026. [Onesug: the unified end-to-end generative framework for e-commerce query suggestion](#). In *Proceedings of the Fortieth AAAI Conference on Artificial Intelligence and Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence and Sixteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'26/IAAI'26/EAAI'26. AAAI Press.
- 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. 2025. [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, New York, NY, USA. Association for Computing Machinery.
- Hang He, Chuhuai Yue, Chengqi Dong, Mingxue Tian, Hao Chen, Zhenfeng Liu, Jiajun Chai, Xiaohan Wang, Yufei Zhang, Qun Liao, Guojun Yin, Wei Lin, Chengcheng Wan, Haiying Sun, and Ting Su. 2026a. [Localsearchbench: Benchmarking agentic search in real-world local life services](#). In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, KDD '26, page 8989–9000, New York, NY, USA. Association for Computing Machinery.
- Ruining He, Lukasz Heldt, Lichan Hong, Raghunandan Keshavan, Shifan Mao, Nikhil Mehta, Zhengyang Su, Alicia Tsai, Yueqi Wang, Shao-Chuan Wang,

- Xinyang Yi, Lexi Baugher, Baykal Cakici, Ed Chi, Cristos Goodrow, Ningren Han, He Ma, Romer Rosales, Abby Van Soest, and 4 others. 2026b. [Plum: Adapting pre-trained language models for industrial-scale generative recommendations](#). In *Proceedings of the ACM Web Conference 2026*, WWW '26, page 8093–8104, New York, NY, USA. Association for Computing Machinery.
- Zhipeng Huang, Yuqiu Qian, and Nikos Mamoulis. 2018. [Location-aware query reformulation for search engines](#). *Geoinformatica*, 22(4):869–893.
- Lingwei Kong, Lu Wang, Changping Peng, Zhangang Lin, Ching Law, and Jingping Shao. 2025. [Generative click-through rate prediction with applications to search advertising](#). *Preprint*, arXiv:2507.11246.
- Xiaochong Lan, Jie Feng, Jiahuan Lei, Xinlei Shi, and Yong Li. 2025. [Benchmarking and advancing large language models for local life services](#). In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, KDD '25, page 4566–4577, New York, NY, USA. Association for Computing Machinery.
- Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. 2022. [Autoregressive image generation using residual quantization](#). In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11513–11522.
- Xiang Li, Xiaojiang Zhou, Yao Xiao, Peihao Huang, Dayao Chen, Sheng Chen, and Yunsen Xian. 2022. [Autofas: Automatic feature and architecture selection for pre-ranking system](#). In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 3241–3249, New York, NY, USA. Association for Computing Machinery.
- Yue Li, Idris Jafarov, Kanan Zeynalov, and Chaocen Tang. 2025. [Clear-rec: A contrastive long-term memory-enhanced, explainable, and adaptive recommendation framework](#). In *Proceedings of the 2nd International Conference on Artificial Intelligence of Things and Computing*, AITC '25, page 236–244, New York, NY, USA. Association for Computing Machinery.
- Agnès Mustar, Sylvain Lamprier, and Benjamin Piwowarski. 2021. [On the study of transformers for query suggestion](#). *ACM Trans. Inf. Syst.*, 40(1).
- Maxim Naumov, Dheevatsa Mudigere, Hao-Jun Michael Shi, Jianyu Huang, Narayanan Sundaraman, Jongsoo Park, Xiaodong Wang, Udit Gupta, Carole-Jean Wu, Alisson G. Azzolini, Dmytro Dzhulgakov, Andrey Mallevich, Ilia Cherniavskii, Yinghai Lu, Raghuraman Krishnamoorthi, Ansha Yu, Volodymyr Kondratenko, Stephanie Pereira, Xianjie Chen, and 5 others. 2019. [Deep learning recommendation model for personalization and recommendation systems](#). *Preprint*, arXiv:1906.00091.
- Jiarui Qin, Jiachen Zhu, Bo Chen, Zhirong Liu, Weiqi Liu, Ruiming Tang, Rui Zhang, Yong Yu, and Weinan Zhang. 2022. [Rankflow: Joint optimization of multi-stage cascade ranking systems as flows](#). In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 814–824, New York, NY, USA. Association for Computing Machinery.
- Junyan Qiu, Ze Wang, Fan Zhang, Zuowu Zheng, Jile Zhu, Jiangke Fan, Teng Zhang, Haitao Wang, Yongkang Wang, and Xingxing Wang. 2025. [Ega-v1: Unifying online advertising with end-to-end learning](#). *Preprint*, arXiv:2505.19755.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. [Direct preference optimization: Your language model is secretly a reward model](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 53728–53741. Curran Associates, Inc.
- Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Keshavan, Trung Vu, Lukasz Heidt, Lichan Hong, Yi Tay, Vinh Q. Tran, Jonah Samost, Maciej Kula, Ed H. Chi, and Maheswaran Sathiamoorthy. 2023. [Recommender systems with generative retrieval](#). In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.
- Yuxin Ren, Qiya Yang, Yichun Wu, Wei Xu, Yalong Wang, and Zhiqiang Zhang. 2024. [Non-autoregressive generative models for reranking recommendation](#). In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 5625–5634, New York, NY, USA. Association for Computing Machinery.
- Teng Shi, Jun Xu, Xiao Zhang, Xiaoxue Zang, Kai Zheng, Yang Song, and Enyun Yu. 2025. [Gensar: Unifying balanced search and recommendation with generative retrieval](#). In *Proceedings of the Nineteenth ACM Conference on Recommender Systems*, RecSys '25, page 124–134, New York, NY, USA. Association for Computing Machinery.
- Qwen Team. 2025. [Qwen3 technical report](#). *Preprint*, arXiv:2505.09388.
- Guoquan Wang, Qiang Luo, Weisong Hu, Pengfei Yao, Wencong Zeng, Guorui Zhou, and Kun Gai. 2025a. [Fim: Frequency-aware multi-view interest modeling for local-life service recommendation](#). In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '25, page 1748–1757, New York, NY, USA. Association for Computing Machinery.
- Yifan Wang, Weinan Gan, Longtao Xiao, Jieming Zhu, Heng Chang, Haozhao Wang, Rui Zhang, Zhenhua Dong, Ruiming Tang, and Ruixuan Li.

- 2025b. [Act-with-think: Chunk auto-regressive modeling for generative recommendation](#). *Preprint*, arXiv:2506.23643.
- Zhipeng Wei, Kuo Cai, Junda She, Jie Chen, Minghao Chen, Yang Zeng, Qiang Luo, Wencong Zeng, Ruiming Tang, Kun Gai, and Guorui Zhou. 2026. [Oneloc: Geo-aware generative recommender systems for local life service](#). In *Proceedings of the Nineteenth ACM International Conference on Web Search and Data Mining*, WSDM '26, page 735–744, New York, NY, USA. Association for Computing Machinery.
- Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muenighoff, Defu Lian, and Jian-Yun Nie. 2024. [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, New York, NY, USA. Association for Computing Machinery.
- Haibo Xing, Hao Deng, Yucheng Mao, Lingyu Mu, Jinxin Hu, Yi Xu, Hao Zhang, Jiahao Wang, Shizhun Wang, Yu Zhang, Xiaoyi Zeng, and Jing Zhang. 2026. [Reg4rec: Reasoning-enhanced generative model for large-scale recommendation systems](#). In *2026 IEEE 42nd International Conference on Data Engineering (ICDE)*, pages 3799–3812.
- Yuhao Yang, Zhi Ji, Zhaopeng Li, YI LI, Zhonglin Mo, Yue Ding, Kai Chen, Zijian Zhang, Jie Li, shuanglong li, and LIU LIN. 2025. [Sparse meets dense: Unified generative recommendations with cascaded sparse-dense representations](#). In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.
- Zhixuan Zhang, Yuheng Huang, Dan Ou, Sen Li, Longbin Li, Qingwen Liu, and Xiaoyi Zeng. 2023. [Rethinking the role of pre-ranking in large-scale e-commerce searching system](#). *Preprint*, arXiv:2305.13647.
- Zuowu Zheng, Ze Wang, Fan Yang, Jiangke Fan, Teng Zhang, Yongkang Wang, and Xingxing Wang. 2025. [Ega-v2: An end-to-end generative framework for industrial advertising](#). *Preprint*, arXiv:2505.17549.
- Guorui Zhou, Hengrui Hu, Hongtao Cheng, Huanjie Wang, Jiaxin Deng, Jinghao Zhang, Kuo Cai, Lejian Ren, Lu Ren, Liao Yu, Pengfei Zheng, Qiang Luo, Qianqian Wang, Qigen Hu, Rui Huang, Ruiming Tang, Shiyao Wang, Shujie Yang, Tao Wu, and 56 others. 2025. [Onerec-v2 technical report](#). *Preprint*, arXiv:2508.20900.
- Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. [Deep interest evolution network for click-through rate prediction](#). In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press.
- Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. [Deep interest network for click-through rate prediction](#). In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, page 1059–1068, New York, NY, USA. Association for Computing Machinery.

Algorithm 1: Reward Calculation for Beam-Search-Driven GRPO

In : Outputs $\mathcal{Y}_G$, Truth y^* , Beam K , Scores $\mathcal{S}$, Coeffs $\lambda_{\{gap,fmt,rank,hit,miss\}}$

Out : Rewards $\mathcal{R} = \{R_1, \dots, R_G\}$

- 1 Initialize $\mathcal{R} \leftarrow \mathbf{0}$, $cnt_{bad} \leftarrow 0$
- 2 Sort indices I by $\mathcal{S}$ desc; get $rank_i$ for each y_i
- 3 Gap term $v_{gap} = \lambda_{gap} \cdot 1.0$
- 4 Tail penalty $v_{tail} = (K \cdot v_{gap}) / (G - K)$
- 5 **foreach** $i \in \{1, \dots, G\}$ **do** $R_i += v_{gap}$ if $rank_i \leq K$, otherwise $R_i -= v_{tail}$
- 6 **foreach** $i \in \{1 \dots G\}$ **do**
 - 7 **if** y_i is invalid **then**
 - 8 $R_i -= \lambda_{fmt}$;
 - 9 **if** $rank_i \leq K$ **then** $cnt_{bad} ++$
- 10 Find the best rank of y^* :
 $rank^* = \min\{rank_i \mid y_i = y^*\}$, if it exists
- 11 **if** $rank^*$ is not None **then**
 - 12 $idx \leftarrow$ index of best match
 - 13 **Bonus** $B = \lambda_{rank} / \log_{10}(rank^* + 1)$
 - 14 **if** $rank^* \leq K$ **then**
 - 15 $B += \lambda_{hit}$
 - 16 **else**
 - 17 $B += \lambda_{hit} + v_{tail}$
 - 18 $R_{idx} += B$
 - 19 **foreach** j where $rank_j < rank^*$ **do**
 - 20 $R_j -= \lambda_{rank} / \log_{10}(rank_j + 1)$
 - 21 **foreach** j where $rank_j > K \wedge y_j$ is valid $\wedge cnt_{bad} > 0$ **do**
 - 22 $R_j = \max(R_j, 1)$
 - 23 $cnt_{bad} - 1$
 - 24 **else**
 - 25 **foreach** i where $rank_i \leq K/2$ **do**
 - 26 $R_i = \min(R_i, -\lambda_{miss})$
 - 27 **foreach** i where y_i is valid **do**
 - 28 $R_i = \max(R_i, 1)$
 - 29 **return** $\mathcal{R}$

A The Algorithm of Reward Calculation

Algorithm 1 illustrates the complete reward calculation pipeline.

B The Algorithm of Quality-Aware Accelerated Beam Search

Algorithm 2 illustrates the complete process of the Quality-Aware Accelerated Beam Search.

C Metric Definitions

We formally define the evaluation metrics used in our experiments. Let $\mathcal{T}$ be the test set of size N . For each input i , the model generates a ranked list of K queries, denoted as $\mathcal{Q}_i = [q_{i,1}, q_{i,2}, \dots, q_{i,K}]$.

HitRate@K (HR@K) and MRR HR@K calculates the percentage of cases where the ground truth Y_i is present in the generated list $\mathcal{Q}_i$. MRR computes the mean reciprocal rank of the first correct answer.

$$HR@K = \frac{1}{N} \sum_{i=1}^N \mathbb{I}(Y_i \in \mathcal{Q}_i), \quad (4)$$

$$MRR = \frac{1}{N} \sum_{i=1}^N \frac{\mathbb{I}(Y_i \in \mathcal{Q}_i)}{\text{rank}_i}, \quad (5)$$

where rank_i denotes the position of Y_i in $\mathcal{Q}_i$.

Diversity (DIV) DIV measures the richness of the generated vocabulary. It is the ratio of unique query strings across the entire dataset to the total number of generated queries:

$$DIV = \frac{|\text{Unique}(\bigcup_{i=1}^N \mathcal{Q}_i)|}{N \times K}. \quad (6)$$

Quality (QUA) This metric evaluates the validity of individual generated queries at the query level, aligning with the *Format Penalty* defined in our GRPO objective. We first define a normalization function $\text{norm}(\cdot)$ (e.g., lowercasing and removing punctuation). Let $\tilde{q}_{i,j} = \text{norm}(q_{i,j})$.

A generated query $q_{i,j}$ is considered effective only if it satisfies two conditions: (1) $\mathcal{F}(q_{i,j})$: The raw query is free from structural errors (e.g., **invalid tokens**, garbled text). (2) $\mathcal{U}(q_{i,j})$: The normalized query is unique relative to higher-ranked queries in the same list (i.e., $\tilde{q}_{i,j} \notin \{\tilde{q}_{i,1}, \dots, \tilde{q}_{i,j-1}\}$).

The metric is calculated as:

$$QUA = \frac{1}{N \times K} \sum_{i=1}^N \sum_{j=1}^K \mathbb{I}(\mathcal{F}(q_{i,j}) \wedge \tilde{q}_{i,j} \notin \{\tilde{q}_{i,m} \mid m < j\}). \quad (7)$$

Algorithm 2: Quality-Aware Accelerated Beam Search

Input: Input x , Beam Width K , Max Tokens T , Absolute Threshold τ , Saturation Coeff α , Min Results R_{min} , Search Width K_{search} , Top-K Window Size K_{win}

Output: Set of generated sequences Y

```

1  $B \leftarrow \{(\text{tokenize}(x), 0.0)\}$ 
2  $C \leftarrow \emptyset$ 
3 for  $t \leftarrow 1$  to  $T$  do
4   if  $B$  is empty then break
5    $Candidates \leftarrow \emptyset$ 
6    $C_{temp} \leftarrow \emptyset$ 
7    $S_{win} \leftarrow \text{InitList}(-\infty, \text{size} = K_{win})$ 
8   foreach  $(seq, score) \in B$  do
9      $P(v|seq) \leftarrow \text{Model}(seq)$ 
10    foreach token  $v \in \text{TopK}(P)$  do
11       $s_{new} \leftarrow score + \log P(v|seq)$ 
12       $seq_{new} \leftarrow seq + v$ 
13       $\text{UpdateSortedWindow}(S_{win}, s_{new})$ 
14      if  $v \in \text{StopIDs}$  then
15        if  $s_{new} > \tau$  then
16          Add  $(seq_{new}, s_{new})$  to  $C_{temp}$ 
17        end
18      else
19        Add  $(seq_{new}, s_{new})$  to  $Candidates$ 
20      end
21    end
22  foreach  $(c, s) \in C_{temp}$  do
23    if  $s > \min(S_{win})$  and  $s \geq \tau$  then
24      Add  $(c, s)$  to  $C$ 
25    end
26  end
27  if  $|C| \geq \alpha \cdot K$  then break
28   $S_{max\_cand} \leftarrow \max_{(c,s) \in Candidates} s$ 
29  if  $S_{max\_cand} < \tau$  and  $|C| \geq R_{min}$  then
30    break
31  end
32   $\mathcal{A} \leftarrow \{(c, s) \in Candidates \mid s \geq \tau\}$ 
33   $B \leftarrow \text{TopK}(\mathcal{A}, K_{search})$ 
34 end
35  $Y \leftarrow \text{Top-}K(C \cup B)$  sorted by score
36 return  $Y$ 

```

This formulation ensures that intra-list duplicates and malformed outputs are strictly penalized.

Table 7: Hyperparameter settings utilized in the proposed framework.

Param	Value	Param	Value
m	10	n	10
K	12	G	16
ϵ	0.1	δ	1e-4
λ_{gap}	1.0	λ_{hit}	1.0
λ_{rank}	2.0	λ_{fmt}	4.0
λ_{miss}	1.0	λ_{order}	1.5
τ	-15	α	1.8
R_{min}	4	K_{win}	15
T	15	K_{search}	12
lr	2e-6	β	0

Online Metrics

- **No-Result Rate:** The frequency of requests that yield no results.
- **Unique Item Exposure:** The average number of de-duplicated exposed items per user, measuring the breadth of item discovery.
- **Category Diversity:** The average number of distinct L2 categories exposed per user, capturing the diversity of recommendation coverage.
- **PV CTR:** The ratio of total clicks to total recommendation impressions across all page views.
- **Session CTR:** The proportion of sessions containing at least one click, reflecting overall user engagement at the session level.
- **Average SUG Input Length:** The average input character length per user.

D Baseline Details

We compare LocalSUG with the following baselines.

MCA MCA is the standard industrial multi-stage cascading architecture. It follows a recall-ranking pipeline, where BGE¹ (Xiao et al., 2024) is used for candidate retrieval and DIN (Zhou et al., 2018) is used for ranking.

¹<https://huggingface.co/BAAI/bge-base-zh-v1.5>

Table 8: Offline performance comparison between LocalSUG and various variants. The best results are highlighted in **bold**, and the second-best results are underlined.

Method	CLICK				ORDER			
	HR@12↑	MRR↑	DIV↑	QUA↑	HR@12↑	MRR↑	DIV↑	QUA↑
LocalSUG _{T=0.8}	96.45%	80.30%	67.34%	95.14%	97.07%	80.24%	64.93%	94.85%
LocalSUG _{T=1.0}	96.52%	80.47%	68.69%	96.96%	97.09%	80.58%	66.71%	97.25%
LocalSUG _{T=1.2}	<u>96.50%</u>	79.47%	68.26%	96.68%	97.09%	79.77%	66.16%	96.73%
LocalSUG _{DPO}	95.69%	<u>80.97%</u>	72.83%	<u>97.47%</u>	96.38%	81.45%	70.47%	<u>97.74%</u>
LocalSUG _{Llama}	95.68%	80.73%	69.90%	95.78%	96.65%	<u>81.65%</u>	68.07%	96.43%
LocalSUG	96.36%	81.13%	<u>71.49%</u>	98.55%	97.09%	81.80%	<u>69.31%</u>	98.78%

City-MCA City-MCA is a city-enhanced variant of MCA. Compared with MCA, it uses city-specific candidate pools during recall and ranking, allowing the traditional cascading system to exploit city-level preference signals.

LMP LMP is adapted from location metadata-based personalization in web search (Bennett et al., 2011). It incorporates user location metadata as an additional personalization signal for query suggestion, enabling location-conditioned ranking over candidate queries.

SPQR SPQR is adapted from spatial proximity query reformulation (Huang et al., 2018). It uses location-aware reformulation signals to enhance query suggestions, emphasizing spatially relevant query alternatives.

SFT SFT denotes the backbone model fine-tuned with standard supervised learning on query suggestion logs, without preference optimization.

OneSug OneSug is a reproduction of the state-of-the-art end-to-end generative query suggestion framework (Guo et al., 2026). It directly generates ranked suggestion lists with a unified generative model.

OneSug_{rule} OneSug_{rule} replaces the original OneSug candidate construction strategy with our co-occurrence rule-based candidates, allowing us to isolate the effect of candidate quality.

LocalSUG Variants We further evaluate several LocalSUG variants. LocalSUG_{DPO} replaces Beam-Search-Driven GRPO with DPO (Rafailov et al., 2023). LocalSUG_{Llama} replaces Qwen3-0.6B with Llama-3.2-1B-Instruct². We also test

different sampling strategies during GRPO exploration while keeping deterministic beam search for inference.

E Hyperparameter Settings

Table 7 presents the key hyperparameters used for both training and inference.

F Comparison with LocalSUG variants

As shown in Table 8, we compare LocalSUG with several variants. The observation that substituting beam search with random sampling substantially diminishes MRR and diversity underlines the advantages of training-inference sampling consistency. Moreover, we find that vanilla DPO training results in a clear performance drop in Hit Rate and MRR. Notably, our framework maintains competitive performance with a Llama backbone, highlighting its generalizability across various large language models.

G Additional Analysis

G.1 Impact of Context and Decoding Width

We analyze the sensitivity of the model to candidate count (m) and beam width (K) using the MIX dataset. As shown in Figure 4a, the number of candidates has a minimal impact on inference speed, as input length is dominated by user history. While the model collapses without candidates, providing merely $m = 2$ candidates remarkably recovers 90.2% of the optimal HitRate and 96.1% of MRR, demonstrating robustness to scenarios with limited candidate availability. We further analyze the source of generated suggestions. We find that 41% of the generated queries do not exactly appear in the provided candidate set, indicating that the model does not simply copy all

²<https://huggingface.co/meta-llama/Llama-3.2-1B-Instruct>

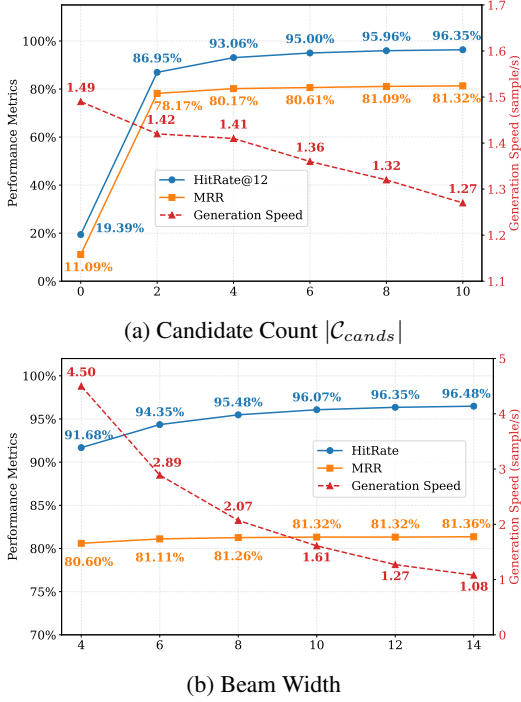


Figure 4: Sensitivity analysis of retrieval candidates and beam search width on efficiency and effectiveness.

prompt candidates as valid outputs. Instead, it selectively uses candidates as city-preference references and further composes new suggestions based on user-side context. For example, if a user frequently consumes rice noodles and has recently visited MixC, the model may generate a new query such as “MixC rice noodles”, which is absent from the candidate set but consistent with both the user’s historical preference and recent behavior. In our logs, such a generated query was eventually clicked by the user, demonstrating LocalSUG’s ability to create personalized and context-aware suggestions beyond candidate copying. Figure 4b highlights the latency-performance Pareto frontier. Reducing K significantly boosts speed with marginal performance loss; specifically, $K = 6$ retains 97.9% of the performance of $K = 12$ while achieving a $2.28\times$ speedup.

G.2 Robustness to Group Size G

In our Beam-Search-Driven GRPO, the group size G determines the ratio of negative samples during training. We investigated the impact of varying G from 14 to 20 on the MIX dataset. As illustrated in Figure 5, aside from minor fluctuations in Diversity (DIV), performance across all key metrics remains highly stable. This indicates that our method is robust to the hyperparameter G , maintaining effective-

Table 9: Ablation results (mean $\pm$ std) over 3 random seeds on the MIX dataset.

Model	HR@12	MRR
LocalSUG	96.12% $\pm$ 0.03%	81.40% $\pm$ 0.02%
w/o r_{gap}	95.54% $\pm$ 0.04%	81.21% $\pm$ 0.11%
w/o r_{hit}	95.79% $\pm$ 0.11%	81.30% $\pm$ 0.03%
w/o r_{rank}	96.53% $\pm$ 0.00%	80.13% $\pm$ 0.11%
w/o r_{fmt}	95.83% $\pm$ 0.03%	76.20% $\pm$ 0.83%
w/o r_{miss}	95.90% $\pm$ 0.02%	81.36% $\pm$ 0.01%
w/o ω	96.14% $\pm$ 0.03%	81.29% $\pm$ 0.08%

Model	DIV	QUA
LocalSUG	72.05% $\pm$ 0.04%	98.30% $\pm$ 0.06%
w/o r_{gap}	72.31% $\pm$ 0.02%	97.66% $\pm$ 0.05%
w/o r_{hit}	71.65% $\pm$ 0.31%	97.86% $\pm$ 0.27%
w/o r_{rank}	70.00% $\pm$ 0.07%	99.00% $\pm$ 0.02%
w/o r_{fmt}	62.47% $\pm$ 0.18%	89.04% $\pm$ 0.09%
w/o r_{miss}	71.58% $\pm$ 0.02%	97.93% $\pm$ 0.03%
w/o ω	71.72% $\pm$ 0.50%	97.76% $\pm$ 0.47%

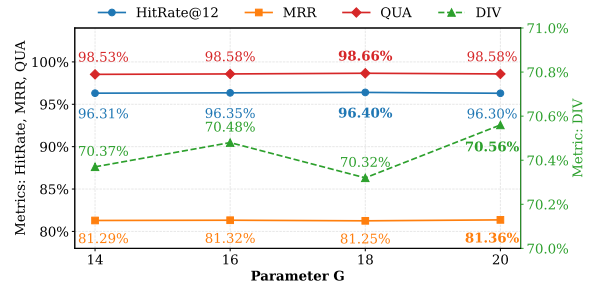


Figure 5: Performance stability across different GRPO group sizes (G).

tiveness without requiring extensive tuning of the group size.

G.3 Ablation with Multiple Random Seeds

To address the concern that minor metric differences in our original ablation study (Table 2) might be attributed to training noise, we re-conducted all ablation experiments using three different random seeds and report the mean and standard deviation on the MIX dataset. As shown in Table 9, the consistently low standard deviations across all variants indicate stable training, ruling out randomness as a factor. The results reaffirm that the format and rank rewards are the most critical components: removing the format reward (w/o r_{fmt}) leads to severe degradation in MRR (-5.20%) and QUA (-9.26%), while removing the rank reward (w/o r_{rank}) causes the largest MRR drop (-1.27%).