HierGR: Hierarchical Semantic Representation Enhancement for Generative Retrieval in Food Delivery Search

Fuwei Zhang[1*], Xiaoyu Liu[1*], Xinyu Jia[2], Yingfei Zhang[2], Zenghua Xia[2], Fei Jiang[2], Fuzhen Zhuang[1,3†], Wei Lin[2†], Zhao Zhang[4†]

[1]Institute of Artificial Intelligence, Beihang University, Beijing, China
[2]Meituan, Beijing, China [3]Zhongguancun Laboratory, Beijing, China
[4]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
{zhangfuwei, liuxiaoyv, zhuangfuzhen}@buaa.edu.cn
{jiaxinyu04, zhangyingfei03, xiazenghua, jiangfei05, linwei31}@meituan.com
zhangzhao2021@ict.ac.cn

Abstract

Food delivery search aims to quickly retrieve deliverable items that meet users' needs, typically requiring faster and more accurate query understanding compared to traditional e-commerce search. Generative retrieval (GR), an emerging search paradigm, harnesses the advanced query understanding capabilities of large language models (LLMs) to enhance the retrieval of results for complex and long-tail queries in food delivery search scenarios. However, there are still challenges in deploying GR to online scenarios: 1) the large scale of items; 2) latency constraints unmet by LLM inference in online retrieval; and 3) strong location-based service restrictions on generated items. To explore the application of GR in food delivery search, we optimize both offline training and online deployment, proposing **Hier**archical semantic representation enhancement for **G**enerative **R**etrieval (HierGR). Specifically, for the generation of semantic IDs, we propose an optimization method that refines the residual quantization process to generate hierarchically semantic IDs for items. Additionally, to successfully deploy on Meituan food delivery platform, we utilize the query cache mechanism and integrate the GR model with the online dense retrieval model to fulfill real-world search requirements. Online A/B testing results show that our proposed method increases the number of online orders by 0.68% for complex search intents. The source code is available at https://github.com/zhangfw123/HierGR.

1 Introduction

In food delivery, users expect to quickly find and order meals that can be delivered to their locations (Wang et al., 2022a; Ding et al., 2020). Food delivery search focuses on quickly retrieving deliverable items that match user needs. Compared to traditional e-commerce search (), it requires faster and more accurate query understanding, as food orders are highly time-sensitive and demand real-time availability checks.

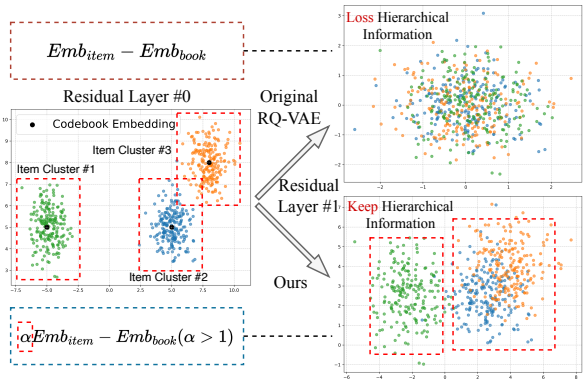

Figure 1: Differences between hierarchical RQ-VAE and origin RQ-VAE.

In recent years, knowledge has played an important role as a bridge across various fields (Tang et al., 2024a; Zhang et al., 2022a,b, 2024a,b,d; Pan et al., 2024; Wang et al., 2024; Li et al., 2025; Kuo et al., 2024; Cheng et al., 2025), including large language models (LLMs). Generative Retrieval (GR) leverages LLMs to generate relevant document identifiers (DocIDs) directly, offering a novel retrieval paradigm. Unlike traditional dense retrieval (DR), GR capitalizes on LLMs' strong semantic understanding, making it more effective for complex, long-tail, and ambiguous queries. This approach shows great potential in applications like e-commerce search (Wu et al., 2024b), document retrieval (Tay et al., 2022; Zhang et al., 2024c), as well as food delivery search.

However, deploying GR on our food delivery search platform still presents significant challenges: (1) **How to design IDs for a large-scale collection of food items?** With hundreds of mil-

*Equal contribution
†Corresponding authors: Fuzhen Zhuang, Wei Lin, and Zhao Zhang

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track), pages 444–455
July 28-30, 2025

lions of items, the deployment faces significant challenges in assigning similar IDs to semantically similar items and distinct IDs to different ones. (2) **How to deploy GR to ensure low latency in online search?** LLMs have high inference latency, making real-time online inference challenging to meet user search latency requirements. (3) **How to ensure that the generated items comply with location-based service (LBS) constraints?** Food delivery search must provide users with items that can be delivered to their locations.

To address these challenges, we explore a series of strategies in both GR model training and online deployment. Specifically, based on the commonly used ID generation method Residual Quantization Variational Autoencoder (RQ-VAE) (Rajput et al., 2023), we propose HierGR, a novel GR method designed to enhance hierarchical semantic representations using a hierarchical RQ-VAE, aiming to reduce semantic loss caused by residual computations. Figure 1 illustrates this clearly: the upper-right subfigure shows that, in the original RQ-VAE, residual representations cluster excessively near the origin after computing next-layer residuals. This clustering causes items from different semantic groups (e.g., three shown types) to overlap, resulting in semantic confusion and identical ID sequences. In contrast, our hierarchical RQ-VAE (lower-right subfigure) preserves more semantic information, ensuring smoother residual computations and clearer hierarchical separation among clusters. This approach maintains distinct clusters (blue and orange points form separate groups, and green points remain clearly isolated) at this residual level. Our method can enhance residual learning on large-scale items.

In the online deployment stage, we conduct a series of optimizations to effectively apply GR in the recall phase of the food delivery search system. First, to ensure that items retrieved by GR satisfy LBS constraints, we reorganize the semantic IDs for GR training. Then, to maintain acceptable online retrieval latency, we introduce a caching mechanism that stores highly exposed queries for online service, achieving a cache hit rate exceeding 95%. Finally, to better integrate with the online system, we combine the prediction scores and results of GR with dense retrieval for ranking, obtaining the final recall results. Here, we summarize our contributions:

- We propose HierGR, a novel GR method designed to enhance hierarchical semantic representations through a hierarchical RQ-VAE, capable of effectively generating semantic IDs for hundreds of millions of online items.
- To successfully deploy the GR model in our system, we implement a series of optimizations that provide valuable insights for industry-wide GR deployment.
- We conduct extensive experiments on the publicly available dataset and online A/B tests, showcasing the effectiveness and potential of applying GR in the food delivery scenario.

2 Related Work

2.1 Sparse & Dense Retrieval

The search process for food delivery is similar to traditional search scenarios, currently relying primarily on sparse and dense retrieval methods for recall, such as BM25 (Robertson et al., 2009), DPR (Karpukhin et al., 2020), ANCE (Xiong et al.), ColBERT (Khattab and Zaharia, 2020), etc. Recent advancements in GR have introduced a variety of new methods.

2.2 Generative Retrieval

DSI (Tay et al., 2022) is the first model to transform documents into unique document ID for GR. SE-DSI (Tang et al., 2023) extends DSI (Tay et al., 2022), which incorporates semantic learning techniques. SEAL (Bevilacqua et al., 2022) proposes autoregressive search engines that generate substrings as DocIDs, while NOVO (Wang et al., 2023) focuses on creating learnable document identifiers. RIPOR (Zeng et al., 2024), on the other hand, emphasizes scalability in GR. GenRRL (Zhou et al., 2023) integrates reinforcement learning to enhance relevance feedback, whereas LTRGR (Li et al., 2024) optimizes GR models by leveraging the ranking task. GDR (Yuan et al., 2024) addresses challenges related to memory efficiency in generative dense retrieval. Furthermore, Wu et al. introduced multi-vector dense retrieval. SEATER (Si et al., 2023) constructs a balanced K-ary tree using Constrained K-means and introduces an alignment loss to better capture token relationships. Hi-gen (Wu et al., 2024b) employs category information for clustering through K-means. GenRet (Sun et al., 2024) adopts an

Encoder-Decoder framework to sequentially generate ID tokens, demonstrating a step-by-step retrieval process. Additionally, GR^2 (Tang et al., 2024b) incorporates multi-graded relevance into the training of GR. These approaches collectively showcase the diverse strategies being developed to advance GR systems.

However, most of the aforementioned methods are not directly applicable for online deployment due to their high complexity. This paper primarily explores and validates the feasibility of implementing GR in the food delivery search scenario, successfully deploying the system and yielding significant benefits.

3 Method & Deployment Pipeline

Figure 2 illustrates the framework for **offline training** and **online deployment**.

3.1 Offline Training

During the offline training phase, we address the critical challenge of generating IDs for hundreds of millions of standardized product units (SPUs). While RQ-VAE (Rajput et al., 2023) provides learnable semantic encoding, its residual quantization inherently causes representation collapse, particularly diluting hierarchical semantics at scale. To resolve this, we propose HierGR with multi-level quantization layers that explicitly preserve semantic granularity. For training GR, we leverage LLMs like Qwen2.5 (Yang et al., 2024) through full fine-tuning.

3.1.1 Hierarchical RQ-VAE

Generally, the hierarchical RQ-VAE process consists of the following three phases:

SPU Encoding. Given an SPU i, we extract its semantic embedding $\boldsymbol{e_i}$ from the online semantic embeddings.

Hierarchical Residual Quantization (RQ). The core concept of hierarchical RQ is to retain part of the residual information from the previous level when computing the residual for the next level, effectively mitigating representation collapse. Appendix B includes a simple analysis demonstrating how our method reduces the semantic loss of residuals. Specifically, hierarchical RQ encodes the SPU embedding $\boldsymbol{e_i}$ into a low-dimensional representation using a deep neural network (DNN) encoder E:

$$\boldsymbol{z} = E(\boldsymbol{e_i}). \tag{1}$$

Next, $\boldsymbol{r}_0 = \boldsymbol{z}$ is used as the residual embedding at the first level of RQ. At each level l, a codebook $\mathcal{C}^l = \{\boldsymbol{c}_k^l\}_{k=1}^K$ is provided for quantization, where $\boldsymbol{c}_k^l$ represents the k-th codebook embedding at level l, and K denotes the codebook size. The l-th level of residual $\boldsymbol{r}_l (l = 0, 1, 2, \dots)$ is then used to find the index of the nearest embedding in $\mathcal{C}^l$, given by $c_l = \arg\min_k \|\boldsymbol{r}_l - \boldsymbol{c}_k^l\|_2$. After that, the residual is iteratively updated as:

$$\boldsymbol{r}_{l+1} = \alpha_l \cdot \boldsymbol{r}_l - \boldsymbol{c}_{c_l}^l, \tag{2}$$

where $\alpha_l > 1$ determines the proportion of residual preserved for the next level $l + 1$.

This procedure yields a semantic ID tuple $(c_0, \dots, c_{m-1})$ corresponding to the indices of the nearest codebook embeddings at each level, where m denotes the maximum level depth.

Reconstruction & Training. In the final stage of hierarchical RQ, we need to reconstruct the SPU embedding after quantization. Since we preserve portions of the residual at each level, the reconstructed representation can be written as follows:

$$\hat{\boldsymbol{z}} = \sum_{l=0}^{m-1} \left[\boldsymbol{c}_{c_l}^l + (1 - \alpha) \cdot \boldsymbol{r}_l \right]. \tag{3}$$

Here m is the layer number of RQ. Then, the quantized embedding $\hat{\boldsymbol{z}}$ is fed into a DNN decoder D to reconstruct the input $\boldsymbol{e_i}$ via a reconstruction loss $\mathcal{L}_{\text{recon}} = \|\boldsymbol{e_i} - D(\hat{\boldsymbol{z}})\|_2^2$. Finally, the optimization objective combines reconstruction loss $\mathcal{L}_{\text{recon}}$ with the residual quantization loss $\mathcal{L}_{\text{rq}}$:

$$\begin{aligned} \mathcal{L}_{\text{training}} &= \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{rq}}, \\ \mathcal{L}_{\text{rq}} &= \sum_{l=0}^{m-1} \left(\|\text{sg}[\boldsymbol{r}_l] - \boldsymbol{c}_{c_l}^l\|_2^2 + \beta \|\boldsymbol{r}_l - \text{sg}[\boldsymbol{c}_{c_l}^l]\|_2^2 \right), \end{aligned} \tag{4}$$

where $\text{sg}[\cdot]$ denotes the stop-gradient operation, which prevents gradient updates for the quantized embeddings during backpropagation. The first term in $\mathcal{L}_{\text{rq}}$ ensures that the codebook vectors $\boldsymbol{c}_{c_l}^l$ are close to the corresponding residuals $\boldsymbol{r}_l$. The second term, weighted by the hyperparameter β, constrains the residuals to remain close to the selected codebook entries.

Using the trained hierarchical RQ-VAE, we generate semantic IDs by identifying the nearest codebook index at each level for every SPU. For instance, if SPU i is assigned the index tuple (1, 3, 2), its semantic ID is represented as

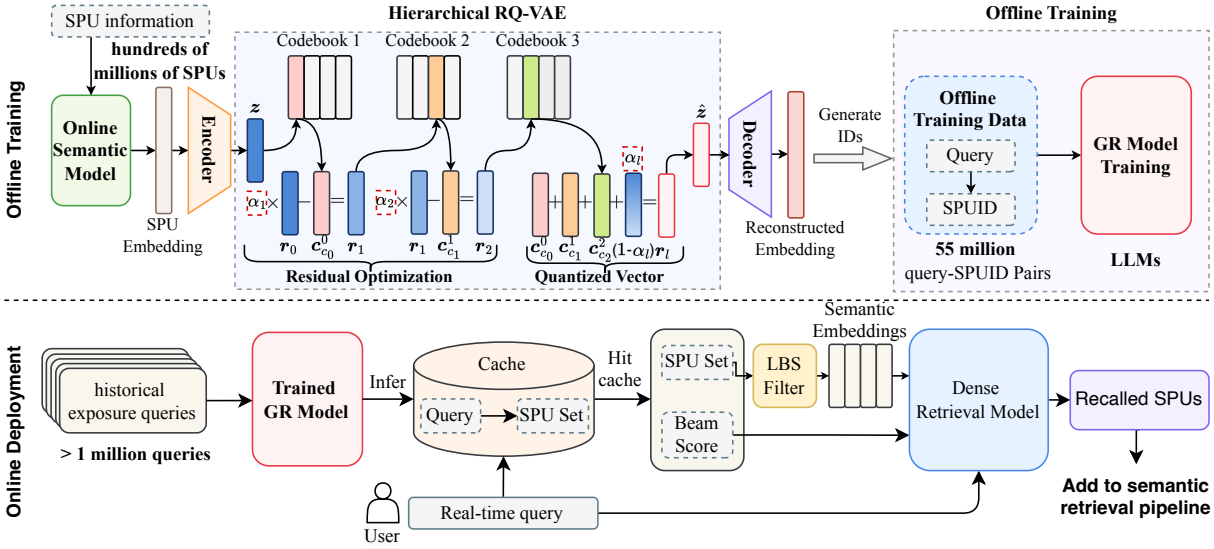

Figure 2: Overall framework of our proposed method, including offline training and online deployment.

"<a_1><b_3><c_2>". Each level-specific component (e.g., "<a_1>" or "<b_3>") serves as a new token for LLM to learn.

3.1.2 Training GR Model

Training Data Collection. We employ the following steps to collect training data.

(1) Query-based conversion attribution: We first track user behavior sequences that occur after users perform search queries. Specifically, we monitor the actions users take after entering queries, such as clicks, views, and final purchases. When a user successfully converts (e.g., completes a purchase or another desired action), we attribute this successful conversion back to the original query that initiated the interaction. This approach helps us clearly link queries with their relevant converted Standard Product Units (SPUs).

(2) Relevance-Based Filtering: Next, we perform an offline analysis to evaluate the relevance of the converted SPUs associated with each query. We carefully examine each SPU to ensure it appropriately matches the user's original search intent. Any SPUs determined to be irrelevant or poorly aligned with user intent are excluded from the dataset to maintain data quality and accuracy.

(3) Prioritized data collection: Finally, we collect data based on prioritized conversion performance. For each individual query, we rank all relevant SPUs according to their total number of successful conversions. We then select only the top 50 SPUs with the highest conversion counts for each query. These top-performing query-SPU pairs form our refined, high-quality training dataset, effectively focusing the dataset on the most successful and relevant items.

To address locality-based service (LBS) constraints, we truncate semantic IDs to transform SPU generation into SPU cluster generation. From sequences like "<a_1><b_3><c_2>", we remove the trailing portion (e.g. "<c_2>") to obtain "<a_1><b_3>" as a cluster ID for SPUs sharing this prefix. This approach enables GR to generate SPU collections that the online system can filter based on delivery area availability.

Training Process. We construct query-clusterID pairs (e.g., "cake→<a_2><b_3>") for training. To balance query understanding capability and training efficiency, we employ Qwen 2.5-1.5B (Yang et al., 2024) with full-parameter fine-tuning using a sequence-to-sequence (seq2seq) paradigm.

3.2 Deployment on Food Delivery Search Platform

The online deployment consists of two key components: **Query Caching** and **Hybrid GR-DR for SPU Recall**.

3.2.1 Query Caching

For online deployment, we implement a query caching mechanism to meet real-time latency requirements. In the food delivery search context, due to high query repetition rates, caching retrieved results achieves over 95% hit rate. We selected the top 1 million queries for caching based on 30-day exposure frequency. During GR model inference, we use a beam size of 100 to return 100 semantic IDs simultaneously, preserving scores

for each ID to facilitate integration with the online dense retrieval (DR) model.

3.2.2 Hybrid GR-DR for SPU Recall

Due to the limited ranking capability of GR, we propose a hybrid recall method named Hybrid GR-DR, integrating GR with our online DR model. Specifically, when a user's query hits the cache, we first retrieve the corresponding cluster IDs from the cache and map them back to their associated SPUs. The SPU set corresponding to the k-th cluster ID is denoted as $S_{ID_k} = \{spu_1, spu_2, \dots\}$. Next, we gather all SPUs relevant to the user's geographic location, forming a local SPU set S_{local}, which is then intersected with the SPU set obtained from GR. This intersection yields the final candidate set that satisfies the location-based service (LBS) constraints:

$$S = S_{local} \cap (S_{ID_1} \cup S_{ID_2} \cup \cdots \cup S_{ID_N}). \quad (5)$$

Here, N represents the number of cluster IDs related to the user's query. Then, we obtain the query embedding $\boldsymbol{q}$ and SPU embeddings $\boldsymbol{s}_1, \dots, \boldsymbol{s}_{|S|}$ using the encoding module of DR, where $|S|$ is the number of SPUs in S. Finally, we derive the final ranking scores for each SPU by combining the cosine similarity scores from DR with the beam scores from GR, as shown below:

$$\text{score}(\text{spu}_i) = \text{beam\_score}(spu_i) \cdot \cos(\boldsymbol{q}, \boldsymbol{s}_i), \quad (6)$$

where $\text{score}(\text{spu}_i)$ denotes the ranking score of the i-th SPU in S, and $\text{beam\_score}(\text{spu}_i)$ represents the beam score of the cluster ID to which spu_i belongs. By incorporating the beam scores, we ensure that highly relevant SPUs generated from GR are ranked higher.

Finally, we integrate the sorted SPUs into the online recall pipeline, delivering the final recall results to users.

4 Experiment

4.1 Experimental Setup

Datasets. For offline evaluation, we conduct experiments on the widely used MSMARCO dataset (Nguyen et al., 2016), derived from web search queries and corresponding passages, following the same settings as LTRGR (Li et al., 2024). For online deployment, we train HierGR on 55 million query-clusterID pairs and compare it to the fully deployed model in the recall stage.

Table 1: Experimental Results on MSMARCO dataset.

Model	R@10	R@20	R@100	MRR@10
BM25 (2009)	28.6	47.5	66.2	18.4
SEAL (2022)	19.8	35.3	57.2	12.7
NCI (2022b)	-	-	-	9.1
DSI (2022)	-	-	-	19.8
MINDER (2023)	29.5	53.5	<u>78.7</u>	18.6
LTRGR (2024)	<u>40.2</u>	**64.5**	**85.2**	<u>25.5</u>
HierGR	**47.9**	<u>63.9</u>	74.6	**37.9**
HierGR w/o optim	39.6	56.3	67.8	30.1

Evaluation Metric. For MSMARCO dataset, we employ the RECALL and MRR metrics, including RECALL@5,20,100 (R@5,20,100), and MRR@10. For online evaluation, we track efficiency metrics: 1) UV_CXR: order rate among search users, 2) PV_CXR: order-to-exposure ratio, 3) OPTU: orders per 1,000 users, and 4) AOP: average order position. Appendix A.1 presents details of these metrics.

Baselines. We compare our method against baselines including BM25 (Robertson et al., 2009), SEAL (Bevilacqua et al., 2022), DSI (Tay et al., 2022), NCI (Wang et al., 2022b), MINDER (Li et al., 2023), and LTRGR (Li et al., 2024). For online evaluation, we compare directly with the **fully deployed online recall model**, reporting incremental improvements across various metrics.

Implementation Details. For the MSMARCO dataset, we use BERT (Devlin et al., 2019) for representation generation and T5-base (Raffel et al., 2020) as the backbone. RQ-VAE is configured with 4 layers, each containing 256 codebooks with an embedding dimension of 32. It is trained using the AdamW optimizer with a learning rate of 0.001 for 300 epochs. The model is further trained for 100 epochs with a learning rate of 0.0005, and the α values for residual optimization are set to [1.1, 1.05, 1.0, 1.0]. For online deployment, the semantic vectors of SPU employed by the online DR model are used as input to the Hierarchical RQ-VAE. Qwen2.5-1.5B is used as the base model. To address LBS constraints, we use the same training parameters as those used for MSMARCO but utilize only the first two layers of semantic IDs generated by the RQ-VAE for GR training. The α values for optimizing residual calculations across the layers are set to [1.05, 1.01, 1.0, 1.0]. The GR model is fine-tuned on 55 million data samples over 5 epochs. All experiments

Table 2: Parameter analysis of α on MSMARCO.

Values of α	Type	R@10	R@20	R@100	MRR@10
[1.2, 1.1, 1.05, 1.0]	Decreasing	38.7	54.5	64.9	29.5
[1.1, 1.05, 1.0, 1.0]	Decreasing	**47.9**	**63.9**	**74.6**	**37.9**
[1.05, 1.0, 1.0, 1.0]	Decreasing	45.3	61.3	70.3	34.1
[1.2, 1.2, 1.2, 1.2]	Fixed	37.4	52.3	65.8	28.7
[1.1, 1.1, 1.1, 1.1]	Fixed	46.3	62.2	70.9	34.9
[1.05, 1.05, 1.05, 1.05]	Fixed	44.2	59.5	68.9	33.5

are conducted on a computing platform equipped with eight A100 80G GPUs.

4.2 Experimental Results on MSMARCO

Table 1 presents the results in percentage (%) on MSMARCO dataset. **Bold** and underlined font represent the best and second-best results.

Overall Performance. HierGR significantly outperforms the state-of-the-art model LTRGR (Li et al., 2024) on MSMARCO in terms of R@10 and MRR@10, indicating that HierGR produces more accurate results at higher ranks. HierGR performs slightly worse than LTRGR on R@100. We hypothesize that this discrepancy arises because LTRGR utilizes a multi-view text-based identifier for GR, incorporating diverse textual information such as titles, pseudo-queries, and substrings. By directly generating text, LTRGR can leverage extensive beam search, resulting in improved recall at lower ranks (e.g., R@100), albeit at the expense of precision among top-ranked results.

Table 3: Online A/B testing results (relative improvement) on well-known food delivery platform. **Overall** is the total performance on all search intents. The second group represents the performance on different search intents.

Search Intent	UV_CXR↑	PV_CXR↑	OPTU↑	AOP↓
Overall	+0.10%	+0.29%	+0.11%	-0.43%
FOOD	+0.07%	+0.13%	+0.04%	-0.22%
POI	+0.16%	+0.26%	+0.15%	**-1.48%**
COMPLEX	**+0.59%**	**+1.12%**	**+0.68%**	-0.28%

Ablation Study. HierGR w/o optim presents the results without applying hierarchical RQ-VAE. As shown, the performance drops across all metrics, indicating that our simple optimization effectively enhances the quality of semantic IDs, thereby improving the effectiveness of GR.

4.3 Parameter Analysis

We conducted hyperparameter experiments on the weight of the proportion of residual preserved α in the hierarchical RQ-VAE. Since our RQ has 4 layers, there are 4 α values for all layers, which we present as a list from α_0 to α_3. Table 2 presents the results. From Table 2, we can draw the following conclusions: 1) Having α decrease as the RQ level increases yields better performance, as semantic loss occurs with greater magnitude in the first two layers; 2) Excessively large or small values of α negatively impact the quality of the IDs, leading to a poor performance of GR.

4.4 Online A/B Testing Results

Table 3 reports the results of our online A/B tests (two weeks). The **FOOD** intent represents user queries seeking physical food items, while the **POI** intent corresponds to queries targeting store searches. The **COMPLEX** intent encompasses more sophisticated queries, such as broad-category food searches, long-tail queries, and natural language questions (e.g., "What should I eat for fitness?"). Caching 1 million high-frequency queries can handle 95% of online search requests. Table 3 clearly shows that the GR model achieves notable improvements across various efficiency metrics, indicating its superior performance. In particular, for the **COMPLEX** intent, UV_CXR increases by 0.59%, PV_CXR increases by 1.12%, and OPTU improves by 0.68%. These results highlight the ability of GR to effectively address diverse user queries, demonstrating stronger generalization capabilities. Furthermore, across all intents, the AOP metric—a key indicator of user experience—decreases, leading to improved ranking quality by positioning relevant items higher in the results. This enables users to locate and order desired food more efficiently.

We also present additional online metrics, as

Table 4: Statistics of other online metrics.

Metric	Value
Average number of SPUs retrieved that meet the LBS constraint	174.9 (MAX:200)
Average number of additional SPUs retrieved compared to online semantic recall	22.8 (MAX:168)

shown in Table 4. The "Average number of SPUs retrieved that meet the LBS constraint" indicates the number of SPUs that satisfy the LBS constraints retrieved by the GR model, with an average of 174.9 and a maximum of 200 (we allocated a retrieval quota of 200 for the GR Model). This demonstrates that the GR model can provide sufficient results to meet online requirements. Furthermore, the "Average number of additional SPUs retrieved compared to online semantic recall" shows the additional items beyond those found by the online semantic recall method, indicating that our model can provide extra SPUs that semantic models cannot retrieve.

Table 5: Collision rate↓ on different datasets.

Model	MSMARCO	online SPUs
HierGR	**3.10%**	**41.57%**
HierGR w/o optim	3.62%	47.64%

4.5 Collision Rate Analysis

Table 5 reports the collision rates of semantic IDs on both the MSMARCO dataset and the online SPUs. The results demonstrate that HierGR, through the optimization of RQ-VAE, effectively mitigates the collision rate of IDs. The high collision rate observed in the online setting can be attributed to the presence of hundreds of millions of SPUs, underscoring the challenges associated with large-scale online deployment. During deployment, all conflicting SPUs sharing the same semantic ID are grouped into a single cluster.

4.6 Case Studies

Table 6 presents the results inferred by the GR model we deployed. As can be observed, for queries like "bread" which cover a wide variety of types, GR can deeply understand and generate different kinds of bread, enhancing diversity. For knowledge-based queries like 'What should I eat for fitness and weight loss?", GR is capable of understanding people's intentions and providing foods related to weight loss.

Table 6: Case studies of online GR results are presented, with the names of the retrieved foods simplified for clarity in display.

Query	GR results
面包 bread	吐司、三明治、可颂、甜甜圈、法棍 Toast, Sandwich, Croissant, Donut, Baguette
健身减肥该吃什么 What should I eat for fitness and weight loss?	鸡胸肉、荞麦面、牛肉沙拉、水煮鸡蛋、法式香煎三文鱼、低卡虾仁西兰花 Chicken breast, Soba noodles, Steak salad bowl, Hard-boiled eggs, Pan-seared salmon, Shrimp & broccoli stir-fry

5 Conclusion

In this paper, we identify challenges that GR faces in practical industrial deployments. To address these challenges, we conduct a series of explorations on both offline training and online deployment for food delivery search. We propose HierGR, which utilizes hierarchical RQ-VAE to reduce ID collision rates during the ID learning process. For online deployment, we analyze the unique characteristics of food delivery search and develop a comprehensive deployment strategy. Additionally, we construct a large-scale domain-specific dataset to effectively train our online GR model for food delivery search. Experimental results on the public benchmark demonstrate the effectiveness of HierGR. Most significantly, online A/B testing shows our deployed GR model achieves a 0.68% increase in the number of orders per thousand users (OPTU) for complex intent search, indicating that the deployed GR model has significant potential for food delivery search.

Acknowledgements

This work was supported by the National Key Research and Development Program of China under Grant No. 2024YFF0729003, the National Natural Science Foundation of China under Grant Nos. 62176014, 62276015, 62206266, the Fundamental Research Funds for the Central Universities.

References

Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Scott Yih, Sebastian Riedel, and Fabio Petroni. 2022. Autoregressive search engines: Generating substrings as document identifiers. *Advances in Neural Information Processing Systems*, 35:31668–31683.

Jiehan Cheng, Zhicheng Dou, Yutao Zhu, and Xiaoxi Li. 2025. Descriptive and discriminative document identifiers for generative retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 11518–11526.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

Xuetao Ding, Runfeng Zhang, Zhen Mao, Ke Xing, Fangxiao Du, Xingyu Liu, Guoxing Wei, Feifan Yin, Renqing He, and Zhizhao Sun. 2020. Delivery scope: A new way of restaurant retrieval for on-demand food delivery service. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3026–3034.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48.

Tzu-Lin Kuo, Tzu-Wei Chiu, Tzung-Sheng Lin, Sheng-Yang Wu, Chao-Wei Huang, and Yun-Nung Chen. 2024. A survey of generative information retrieval. *arXiv preprint arXiv:2406.01197*.

Xiaoxi Li, Jiajie Jin, Yujia Zhou, Yuyao Zhang, Peitian Zhang, Yutao Zhu, and Zhicheng Dou. 2025. From matching to generation: A survey on generative information retrieval. *ACM Transactions on Information Systems*, 43(3):1–62.

Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. 2023. Multiview identifiers enhanced generative retrieval. In *61st Annual Meeting of the Association for Computational Linguistics, ACL 2023*, pages 6636–6648. Association for Computational Linguistics (ACL).

Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. 2024. Learning to rank in generative retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 8716–8723.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human-generated machine reading comprehension dataset.

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. 2024. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3580–3599.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

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

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Zihua Si, Zhongxiang Sun, Jiale Chen, Guozhang Chen, Xiaoxue Zang, Kai Zheng, Yang Song, Xiao Zhang, Jun Xu, and Kun Gai. 2023. Generative retrieval with semantic tree-structured item identifiers via contrastive learning. *arXiv preprint arXiv:2309.13375*.

Weiwei Sun, Lingyong Yan, Zheng Chen, Shuaiqiang Wang, Haichao Zhu, Pengjie Ren, Zhumin Chen, Dawei Yin, Maarten Rijke, and Zhaochun Ren. 2024. Learning to tokenize for generative retrieval. *Advances in Neural Information Processing Systems*, 36.

Junfei Tang, Ran Song, Yuxin Huang, Shengxiang Gao, and Zhengtao Yu. 2024a. Semantic-aware entity alignment for low resource language knowledge graph. *Frontiers of Computer Science*, 18(4):184319.

Yubao Tang, Ruqing Zhang, Jiafeng Guo, Jiangui Chen, Zuowei Zhu, Shuaiqiang Wang, Dawei Yin, and Xueqi Cheng. 2023. Semantic-enhanced differentiable search index inspired by learning strategies. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4904–4913.

Yubao Tang, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Wei Chen, and Xueqi Cheng. 2024b. Generative retrieval meets multi-graded relevance. In

The Thirty-eighth Annual Conference on Neural Information Processing Systems.

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

Wenjie Wang, Honghui Bao, Xinyu Lin, Jizhi Zhang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. 2024. Learnable item tokenization for generative recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 2400–2409.

Xing Wang, Ling Wang, Shengyao Wang, Jize Pan, Hao Ren, and Jie Zheng. 2022a. Recommending-and-grabbing: A crowdsourcing-based order allocation pattern for on-demand food delivery. *IEEE Transactions on Intelligent Transportation Systems*, 24(1):838–853.

Yujing Wang, Yingyan Hou, Haonan Wang, Ziming Miao, Shibin Wu, Qi Chen, Yuqing Xia, Chengmin Chi, Guoshuai Zhao, Zheng Liu, et al. 2022b. A neural corpus indexer for document retrieval. *Advances in Neural Information Processing Systems*, 35:25600–25614.

Zihan Wang, Yujia Zhou, Yiteng Tu, and Zhicheng Dou. 2023. Novo: learnable and interpretable document identifiers for model-based ir. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 2656–2665.

Shiguang Wu, Wenda Wei, Mengqi Zhang, Zhumin Chen, Jun Ma, Zhaochun Ren, Maarten de Rijke, and Pengjie Ren. 2024a. Generative retrieval as multi-vector dense retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1828–1838.

Yanjing Wu, Yinfu Feng, Jian Wang, Wenji Zhou, Yunan Ye, Rong Xiao, and Jun Xiao. 2024b. Hi-gen: Generative retrieval for large-scale personalized e-commerce search. *arXiv preprint arXiv:2404.15675*.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Peiwen Yuan, Xinglin Wang, Shaoxiong Feng, Boyuan Pan, Yiwei Li, Heda Wang, Xupeng Miao, and Kan Li. 2024. Generative dense retrieval: Memory can be a burden. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2835–2845.

Hansi Zeng, Chen Luo, Bowen Jin, Sheikh Muhammad Sarwar, Tianxin Wei, and Hamed Zamani. 2024. Scalable and effective generative information retrieval. In *Proceedings of the ACM on Web Conference 2024*, pages 1441–1452.

Fuwei Zhang, Zhao Zhang, Xiang Ao, Dehong Gao, Fuzhen Zhuang, Yi Wei, and Qing He. 2022a. Mind the gap: Cross-lingual information retrieval with hierarchical knowledge enhancement. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 4345–4353.

Fuwei Zhang, Zhao Zhang, Xiang Ao, Fuzhen Zhuang, Yongjun Xu, and Qing He. 2022b. Along the time: Timeline-traced embedding for temporal knowledge graph completion. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 2529–2538.

Fuwei Zhang, Zhao Zhang, Fuzhen Zhuang, Zhiqiang Zhang, Jun Zhou, and Deqing Wang. 2024a. Multi-view temporal knowledge graph reasoning. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 4263–4267.

Fuwei Zhang, Zhao Zhang, Fuzhen Zhuang, Yu Zhao, Deqing Wang, and Hongwei Zheng. 2024b. Temporal knowledge graph reasoning with dynamic memory enhancement. *IEEE Transactions on Knowledge and Data Engineering*, 36(11):7115–7128.

Hailin Zhang, Yujing Wang, Qi Chen, Ruiheng Chang, Ting Zhang, Ziming Miao, Yingyan Hou, Yang Ding, Xupeng Miao, Haonan Wang, et al. 2024c. Model-enhanced vector index. *Advances in Neural Information Processing Systems*, 36.

Miao Zhang, Tingting He, and Ming Dong. 2024d. Meta-path reasoning of knowledge graph for commonsense question answering. *Frontiers of Computer Science*, 18(1):181303.

Yujia Zhou, Zhicheng Dou, and Ji-Rong Wen. 2023. Enhancing generative retrieval with reinforcement learning from relevance feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12481–12490.

A More Details of our Experiments

In this section, we provide more experimental details.

A.1 Evaluation Metrics

Here, we describe the calculation method of the metrics. Note that PV represents a merchant.

- **RECALL:** The proportion of relevant items successfully retrieved over the total number of relevant items, calculated as:

$$\text{RECALL} = \frac{\sum \text{Retrieved Relevant Items}}{\sum \text{Total Relevant Items}} \tag{7}$$

- **MRR:** The Mean Reciprocal Rank, which is the average of the reciprocal ranks of the first relevant item in all queries, calculated as:

$$\text{MRR} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{\text{Rank}_i} \tag{8}$$

- **PV_CXR:** The ratio of food delivery order page views to exposure page views, calculated as:

$$\text{PV\_CXR} = \frac{\sum \text{Order PV}}{\sum \text{Exposure PV}} \tag{9}$$

- **UV_CXR:** The ratio of unique ordering users to users exposed, calculated as:

$$\text{UV\_CXR} = \frac{\sum \text{Distinct(Order Users)}}{\sum \text{Distinct(Exposure Users)}} \tag{10}$$

- **AOP**: The average exposure position of successful orders, calculated as:

$$\text{AOP} = \frac{\sum \text{Exposure Positions}}{\sum \text{Order PV}} \tag{11}$$

- **OPTU**: The number of successful orders per thousand search users, calculated as:

$$\text{OPTU} = \left(\frac{\sum \text{Order}}{\sum \text{Search Users}} \right) \times 1000 \tag{12}$$

A.2 Baselines

Here, we will provide detailed descriptions of our baselines.

- **BM25** (Robertson et al., 2009): BM25 is a classic sparse retrieval model that enhances term-document matching by leveraging term frequency and inverse document frequency, effectively improving information retrieval.
- **DSI** (Tay et al., 2022): DSI employs a hierarchical k-means clustering approach to organize document representations, constructing the DocID by combining category indices from multiple layers.
- **NCI** (Wang et al., 2022b): NCI utilizes neural network architectures to enhance document retrieval performance.
- **MINDER** (Li et al., 2023): MINDER generates text-based IDs from multi-view information to improve retrieval effectiveness.
- **LTRGR** (Li et al., 2024): LTRGR optimizes pre-trained GR models by incorporating an auxiliary ranking task.

For the online baseline, we compare our model against the fully deployed and stable product retrieval model that is already running in production, which includes but is not limited to query rewriting, semantic retrieval, personalized retrieval, and other components. Our GR model is integrated into the existing retrieval pipeline to evaluate its effectiveness.

B Simple Analysis of the Effectiveness of HierGR

We demonstrate that for two distinct embeddings that are very closed to their codebook embeddings at level k, our proposed hierarchical RQ-VAE preserves more of their semantic differences in subsequent quantization levels compared to the vanilla RQ-VAE.

Suppose there are two different items A and B. Let $\boldsymbol{r}_A^l$ and $\boldsymbol{r}_B^l$ be two distinct embeddings with semantic differences at level l. If $\boldsymbol{r}_A^l$ and $\boldsymbol{r}_B^l$ are very close to their respective codebooks $\boldsymbol{q}^l(A)$ and $\boldsymbol{q}^l(B)$, then the next level residuals in vanilla RQ-VAE are computed as follows:

$$\boldsymbol{r}_A^{l+1} = \boldsymbol{r}_A^l - \boldsymbol{q}^l(A) \approx \mathbf{0}, \tag{13}$$

$$\boldsymbol{r}_B^{l+1} = \boldsymbol{r}_B^l - \boldsymbol{q}^l(B) \approx \mathbf{0}. \tag{14}$$

At this point, the computation method of vanilla RQ-VAE will cause the residuals of item A and

Table 7: Expanded analysis: case studies of online GR results.

Query	GR results
breakfast (English Query)	全麦番茄辣松贝果（无油无糖）、番茄肉松贝果、法式香蒜司康、北海道吐司、金枪鱼可颂三文治、草莓甜甜圈、法式传统法棍、原味碱水棒、酥皮菠萝包、港式黄油菠萝包、蒜香奶酪爆浆面包、法式香蒜面包、蜂蜜黄油吐司片、.....
breakfast	Whole Wheat Tomato & Spicy Pork Floss Bagel (Oil-free & Sugar-free), Tomato & Pork Floss Bagel, French Garlic Scone, Hokkaido Milk Toast, Tuna Croissant Sandwich, Strawberry Donut, Traditional French Baguette, Original Pretzel Stick, Crispy Pineapple Bun, Hong Kong Style Butter Pineapple Bun, French Garlic Bread, Honey Butter Toast Slices, ...
早餐	鲜磨豆浆、牛肉包子、东北玉米、招牌蒸饺、奶黄包、五香卤鸡蛋、茶叶蛋、香酥油条 1 根、鸡蛋青菜粥、小米南瓜粥、胡辣汤、鸡蛋肠粉、.....
breakfast	Freshly Ground Soy Milk, Beef Baozi, Northeastern Chinese Sweet Corn, House Specialty Steamed Dumplings, Custard Buns, Five-Spice Braised Egg, Tea-Marinated Egg, Crispy Fried Breadstick (1 piece), Egg and Vegetable Congee, Millet and Pumpkin Porridge, Spicy Hot Soup, Egg Cheung Fun (Rice Noodle Roll), ...
高蛋白低脂饮食	虾仁蔬菜减脂沙拉、香草鸡胸肉糙米饭、鲜虾仁健身套餐、厚蛋牛肉三明治、香煎牛排蔬菜能量碗、优质嫩煎牛排菠菜卷、香煎黑椒鸡胸杂粮拌饭、香煎蟹柳滑蛋、香煎鸡胸肉荞麦面、.....
High-protein and low-fat diet	Shrimp & Vegetable Fat-loss Salad, Herbed Chicken Breast with Brown Rice, Fitness Shrimp Meal Set, Beef & Thick Omelette Sandwich, Pan-fried Steak Veggie Power Bowl, Premium Pan-fried Steak Spinach Wrap, Black Pepper Chicken Breast Multigrain Rice, Pan-fried Crab Stick with Scrambled Egg, Pan-fried Chicken Breast Buckwheat Noodles, ...
儿童适合吃什么	虾仁拌意大利弯管面儿童套餐、儿童金枪鱼拌饭（沙拉酱无拌饭酱）、番茄炒鸡蛋（小份）、番茄牛肉儿童餐、宝宝串串香、儿童牛排 + 意面 + 煎蛋、营养蒸蛋、儿童餐-虾仁咖喱炒饭套餐、宝宝卤肉饭、.....
What foods are suitable for children?	Shrimp with Elbow Macaroni Kids Meal, Tuna Rice Bowl for Kids (Salad Dressing Only), Stir-Fried Tomato and Egg (Small Portion), Beef and Tomato Kids Meal, Baby-Friendly Skewers (Assorted Mini Sticks), Kids Steak Meal with Pasta and Fried Egg, Nutritious Steamed Egg Custard, Kids Shrimp Curry Fried Rice Set, Baby-Style Braised Pork Rice, ...
高热量的小吃	超值至尊 pizza 经典 9 寸、香辣鸡腿堡鸡肉卷套餐、美式芝加哥鸡排热狗、韩式炸鸡火鸡面、大份薯条、香辣大鸡排、炭烤大牛肉串 10 串、奥尔良烤鸡肉披萨 7 英寸、鸡米花 Popcorn Chicken、.....
Calorie-dense snacks	9-inch Classic Supreme Pizza (Value Deal), Spicy Chicken Thigh Burger & Chicken Wrap Combo, American Chicago-Style Chicken Cutlet Hot Dog, Korean Fried Chicken with Spicy Buldak Noodles, Large French Fries, Spicy Jumbo Chicken Cutlet, Charcoal-Grilled Beef Skewers (10 pieces), 7-inch New Orleans-Style Grilled Chicken Pizza, Popcorn Chicken (Crispy Bite-Sized Chicken), ...

item B at the next level to be close to the zero vector. This will affect the subsequent residual calculations, making the IDs of the two items identical in future processes, thus losing hierarchical semantic information.

However, if we calculate the residual by our proposed hierarchical RQ-VAE, the next level residuals are computed as follows:

$$\boldsymbol{r}_A^{l+1} = \alpha_l \cdot \boldsymbol{r}_A^l - \boldsymbol{q}^l(A) \approx (\alpha_l - 1) \cdot \boldsymbol{r}_A^l, \quad (15)$$

$$\boldsymbol{r}_B^{l+1} = \alpha_l \cdot \boldsymbol{r}_B^l - \boldsymbol{q}^l(B) \approx (\alpha_l - 1) \cdot \boldsymbol{r}_B^l. \quad (16)$$

Here, we still retain some of the higher-level semantic information to prevent the representation modeling of two different items from causing them to retain a certain level of hierarchical information.

Similarly, for items A and B, if they are very close to the same codebook vector $\boldsymbol{q}^l(C)$, we can also demonstrate that the next-level residuals are nearly identical in vanilla RQ-VAE:

$$\boldsymbol{r}_A^{l+1} = \boldsymbol{r}_A^l - \boldsymbol{q}^l(C), \quad (17)$$

$$\boldsymbol{r}_B^{l+1} = \boldsymbol{r}_B^l - \boldsymbol{q}^l(C). \quad (18)$$

If $\boldsymbol{r}_A^l \approx \boldsymbol{r}_B^l \approx \boldsymbol{q}^l(C)$, then $\boldsymbol{r}_A^{l+1} \approx \boldsymbol{r}_B^{l+1} \approx \mathbf{0}$, causing the same issue of losing semantic differences.

In contrast, our hierarchical RQ-VAE computes:

$$\boldsymbol{r}_A^{l+1} = \alpha_l \cdot \boldsymbol{r}_A^l - \boldsymbol{q}^l(C), \quad (19)$$

$$\boldsymbol{r}_B^{l+1} = \alpha_l \cdot \boldsymbol{r}_B^l - \boldsymbol{q}^l(C). \quad (20)$$

Even when $\boldsymbol{r}_A^l \approx \boldsymbol{r}_B^l \approx \boldsymbol{q}^l(C)$, the subtle differences between $\boldsymbol{r}_A^l$ and $\boldsymbol{r}_B^l$ are amplified by the factor α_l, allowing these semantic differences to propagate to subsequent quantization levels.

This amplification mechanism ensures that our hierarchical RQ-VAE maintains finer semantic distinctions throughout the quantization hierarchy, resulting in more expressive and discriminative representations compared to original RQ-VAE.

C More Analysis of Online Search Results

C.1 Case Studies for Search Results

Additionally, we provide more case studies, as shown in Table 7. Interestingly, the GR model can return relevant results of different types based on queries in different languages, such as the English query "breakfast" and the Chinese query ‘‘早餐". For the query "breakfast", it returned many bread-based food items; for the Chinese query ‘‘早餐", it returned numerous breakfast foods that align with Chinese dietary habits. Furthermore, the latter cases demonstrate that the GR model exhibits strong capabilities in understanding various knowledge domains of queries.

D Limitations and Future Works

Our current deployment method is based on caching, which covers 95% of online requests. However, 5% of online search queries still remain uncovered. Additionally, although we have optimized the collision of semantic IDs during the encoding process and generated better semantic IDs, the quality of these IDs cannot be directly evaluated during the RQ-VAE training phase and still needs to be assessed based on the retrieval effectiveness of the final GR model. In addition, our GR model is currently unable to generate personalized outputs based on users' specific needs. For example, if a user wants the fastest possible delivery, the model cannot identify and generate SPUs with quicker delivery options. These limitations require further exploration.

In future work, we will explore how to make GR cover more online queries, as well as how to enable GR to directly generate SPUs that satisfy LBS constraints.