Graph Meets LLM: A Novel Approach to Collaborative Filtering for Robust Conversational Understanding

Zheng Chen*, **Ziyan Jiang***, **Fan Yang***, **Eunah Cho**,
Xing Fan, **Xiaojiang Huang**, **Yanbin Lu**, **Aram Galstyan**
Amazon
{zgchen, ziyjiang, ffanyang, eunahch, fanxing, xjhuang, luyanbin, argalsty}@amazon.com

Abstract

A *Personalized Query Rewriting* system aims to reduce defective queries to ensure robust conversational functionality by considering individual user behavior and preferences. It's usually structured as a search-based system, maintaining a *user history index* of past successful interactions with the conversational AI. However, this approach encounters challenges when dealing with *unseen interactions*, which refers to new user interactions not covered by the user history index.

This paper introduces our "*Collaborative Query Rewriting*" approach, which utilizes underlying topological information to assist in rewriting defective queries arising from unseen interactions. This approach begins by constructing a "User **F**eedback **I**nteraction **G**raph" (*FIG*) using historical user-entity interactions. Subsequently, we traverse through the graph edges to establish an enhanced user index, referred to as the "*collaborative user index*".

We then delve deeper into the utilization of Large Language Models (LLMs) to assist in graph construction by understanding user preferences, leading to a significant increase in index coverage for unseen interactions. The effectiveness of our proposed approach has been proven through experiments on a large-scale real-world dataset and online A/B experiments.

1 Introduction

Defective queries frequently occur during user interactions with conversational AI systems such as Alexa, Siri or Google Assistant. These are induced by user ambiguities or mistakes, along with errors in automatic speech recognition (ASR) or natural language understanding (NLU). Defective queries impact the robustness of the conversational AI system, as they hinder users from receiving the intended results and often require further clarification. *Query Rewriting* (QR) is a subsystem within the conversational AI that plays a crucial role in reducing defective queries. By automatically refining or correcting these defective queries, QR enhances the overall robustness of the AI system and significantly improves the user experience.

Personalized Query Rewriting (Personalized QR) takes into account individual preferences or unique error patterns identified from a user's historical interactions with the conversational AI. It plays a crucial role in addressing a wide range of user-specific defects, particularly in the torso and tail distribution. For instance, when a user presents a defective query like "play abcdefg", a non-personalized QR system might rewrite it to "play alphabetic song" based on the high overall transition probability from "abcdefg" to "alphabetic song". However, for this particular user, the query was intended for the song "abcdefu" by the American singer Gayle.

A *personalized QR* system is often designed as a search-based approach, which requires a *user history index* to capture historical non-defective user experiences. The user history index includes each user's own historical successful queries, rephrases, rewrites, and related metadata & statistics. During runtime, given a user query (e.g. "play abcdefg"), the system checks if a successful historical query utterance (e.g. "play abcdefu by gale") in the user history index closely matches the current query. The user interactions covered by the user history index are called the "***seen interactions***".

Despite the effectiveness of personalized QR for reducing defects in conversational AI, we have identified the challenge posed by "***unseen interactions***" not covered by the user history index. We have observed that users frequently engage in new experiences, leading to approximately 50% of the queries/interactions not being covered by the user history index. We refer to these queries/interactions as "*unseen*". Moreover, unseen queries/interactions have a defect rate roughly 7% higher. This under-

*Authors contributed equally to this research. Authors alphabetically ordered by last name.

scores the potential benefits of query rewriting for these unseen interactions.

We introduce our approach "*Collaborative Query Rewriting*" (Collaborative QR), designed to overcome the constraints of the user history index. This approach is inspired by our observation that users who interact with similar entities through a conversational AI often make similar queries or experience comparable defects (see Figure 1). The cornerstone of our approach is the "User **F**eedback **I**nteraction **G**raph" (FIG), which captures users' previous interactions with various entities through the conversational AI in a user-entity interaction graph (see Section 3.2). Our key idea is to leverage FIG to form a *collaborative user index* consisting of additional rewrite candidates not found in the user history index.

Graph traversal through the FIG is the most straightforward approach for constructing the collaborative user index (see Section 3.3). To enhance the collaborative user index, we delve deeper into *Large Language Models* (LLMs) (see Section 3.4). In this paper, we investigate the potential of integrating a publicly available LLM, "dolly-v2-7b" (Conover et al., 2023)[1], with graph traversal to further improve the coverage of the collaborative user index.

Our key contributions are summarized as the following:

1. To the best of our knowledge, we are the first to propose "Collaborative Query Rewriting", which uses topological user-entity interaction information to reduce query defects. We have validated our approach through experiments on a large-scale real-world dataset and online A/B experiments.
2. We have explored the use of LLMs to learn from the user preference in the FIG graph. Our findings show a marked boost in index coverage for previously unseen user interactions. This led to a notable improvement in the defect reduction trigger rate, while the collaborative user index size cap is reduced from 500 to 200 to save runtime latency and cost.

2 Related Works

Query Rewriting Query Rewriting (QR) in dialogue systems aims to reduce frictions by reformulating the automatic speech recognition component's interpretation of users' queries. Initial efforts (Dehghani et al., 2017; Su et al., 2019) treat QR as a text generation problem.

Some recent studies (Chen et al., 2020b; Fan et al., 2021a; Cho et al., 2021; Naresh et al., 2022) are based on neural retrieval systems. In these retrieval-based frameworks, the rewrite candidate pool is aggregated from users' habitual or historical queries so that the rewrite quality can be tightly controlled. Compared to generation-based systems, retrieval-based systems may sacrifice flexibility and diversity of the rewrites, but in the meanwhile provide more stability which is more important in a runtime production setup.

LLMs for User Preference Learning There has been a surge of recent researches affirming LLM can learn from user affinity and make predictions or recommendations. (Chen, 2023) fine-tunes a LLaMA 7B model to learn from the user affinity of movie-lens dataset and the Amazon beauty dataset, and out-performs the SOTA models on the recommendation task. (Kang et al., 2023) investigates the ability of Large Language Models (LLMs) to understand user preferences and predict user ratings. The study finds that while zero-shot LLMs lag behind traditional recommender models that utilize user interaction data, they can achieve comparable or even superior performance when fine-tuned with a small fraction of the training data. (Cui et al., 2022) proposed a generative pretrained language model that serves as a unified foundation for various tasks in recommender systems, using user behavior data as plain texts and converts tasks into language understanding or generation.

3 Methodology

3.1 Notation and Preliminaries

Users interact with a conversational AI agent by providing an input termed a "*query*". Within the agent, there is a natural language understanding (NLU) component designed to comprehend the details of the given query, such as classifying the query into a domain, extracting and resolving entities from the query. This process is how we capture multi-domain user-entity interactions with the conversational AI.

Definition 1. Let γ be an integer such that $1 \leq \gamma < \infty$. The natural language understanding of a query for our purpose can be understood as a mapping function, $h : Q \rightarrow D \times [E]^{\gamma}$, where Q

[1]This is the best public model available for commercial use at the time of our experiments. Therefore, we chose to experiment with this model to evaluate its performance.

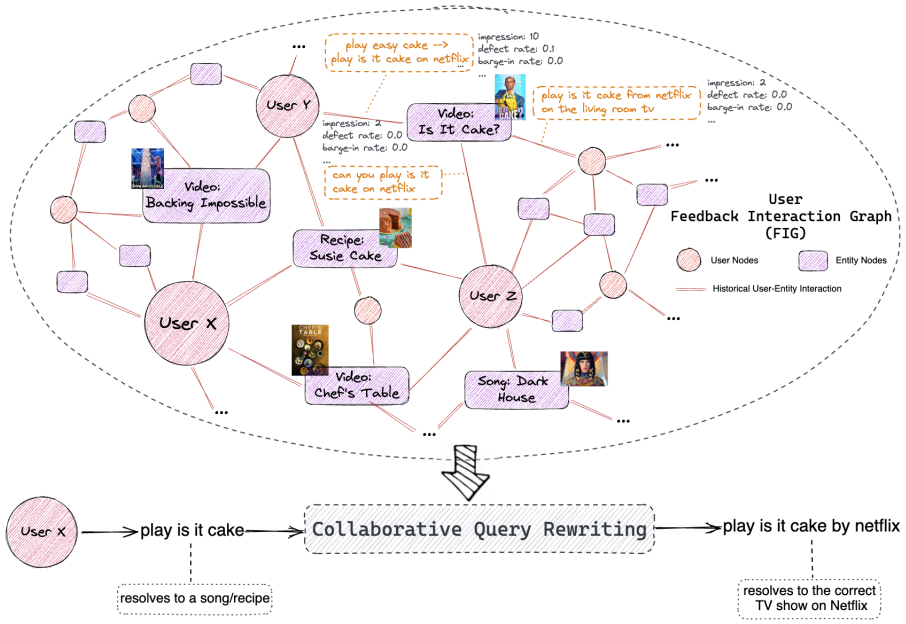

Figure 1: High-level illustration of the FIG and its application in collaborative QR. User X and Y interacted with the same or similar baking videos, which indicates their similarity. There was a successful historical rewrite "play easy cake → play is it cake on netflix" for user Y, which effectively resolved entity to the correct one. When user X encounters a defective query such as "play is it cake", the historical rewrite from user Y ("play easy cake → play is it cake on Netflix") can be considered as a rewrite candidate and utilized to correct it as "play is it cake by Netflix".

refers to the query space, D refers to the domain space and E refers to the entity space. The entity space, $E := E_T \times E_V$, may further be decomposed into the entity type space E_T and the entity value space E_V. All spaces are defined over Unicode strings.

As an example, given a query string q ="*Play The Real Slim Shady*", the corresponding NLU hypothesis is $h(q)$ =*(Music, [(SongName, The Real Slim Shady)])* where the domain is *Music* and the entity value is *"The Real Slim Shady"* with the entity type of *"SongName"*.

3.2 User Feedback Interaction Graph

Graph emerges as a natural structure to represent user historical interactions with various entities through the conversational AI (Markowitz et al., 2023). These entities can span diverse categories like songs, videos, books, and more. We extract non-defective user-entity interactions from the raw conversational AI logs and integrate them into a user-entity interaction graph. Different nodes of the graph represent different users and entities, while the edges encapsulate the information related to their interactions. This interaction information encompasses the user's queries as well as associated feedback signals (e.g. impression, defect rate, barge-in rate, termination-rate). We refer to this graph as "*User Feedback Interaction Graph*" (FIG), where the term "feedback" emphasizes that the graph incorporates explicit and implicit feedback from users.

Figure 1 offers a high-level depiction of the FIG and its application in collaborative QR. It includes user nodes (such as "User X", "User Y", "User Z") and entity nodes (like "Video: Is It Cake" and "Recipe: Susie Cake"). The user queries encapsulated in the edges represent *non-defective* interactions between the user and the entity. Here, "non-defective" refers to user-entity interactions where the defect rate (Gupta et al., 2021) falls below a certain threshold. These queries might consist of the user's original input utterances (for example, "play is it cake from netflix on the living room tv"[2]) or a pair of utterances if a rewrite for the original input was successful in the past (such as "play easy cake" being revised to "play is it cake on netflix"). Feedback signals, also encapsulated in the edges, include various elements such as impression (representing the past frequency of the query), defect rate (Gupta et al., 2021), barge-in rate (the probability that the user interrupted the agent's response to this query), termination-rate (the probability that the

[2]All queries are in lower case for this paper.

user stopped the agent's response to this query).

3.3 Collaborative User Index Through Graph Traversal

We leverage the FIG to build a collaborative user index through graph traversal. The intuition is that users who have interacted with the same entities in the past are likely similar, and could also exhibit similar interactions in the future. As we traverse the FIG, we collect the interaction information encapsulated within these edges (e.g. historical queries for this interaction, with their associated feedback signals, see Section 3.2) and integrate them into our collaborative user index. Within this process, user queries are considered potential candidates for rewriting, while feedback signals can serve as ranking features (see Section 3.5). Currently, we limit our consideration up to a 3-hop traversal (that is, paths such as "User X $\rightarrow$ Entity A $\rightarrow$ User Y $\rightarrow$ Entity B") due to computational resource constraints. The collaborative user index is pre-constructed offline to reduce runtime latency and is periodically refreshed. We also implement heuristic rules to surface more promising candidates while controlling the size of the collaborative user index (see Appendix A.1).

After considering both runtime system latency constraints and the coverage of unseen interactions, we have settled on a size cap of 500 for the collaborative user index.

3.4 Collaborative User Index Enhanced By LLMs

Large language models have showcased remarkable capability in deducing user preferences and predicting future behavior by analyzing historical interactions (Chen, 2023; Wang et al., 2023). Before the graph traversal step, we employ a large language model for link prediction between the user nodes and the entity nodes. We have chosen the "dolly-v2-7b" model (Conover et al., 2023) and apply instruction tuning, a proven effective method in recent developments of large language models (LLMs) (Taori et al., 2023; Wei et al., 2021; Ouyang et al., 2022). At present, our exploration is centered on the Music/Video domains, which account for approximately 80% of the total user traffic volume.

To perform fine-tuning, we utilize the user's historical interacted entities as training input, which corresponds to the user's 1-hop connected nodes in the FIG. The training labels for the model consist of the entities that the same user interacted with during the subsequent month following the training input. Here are the examples of the training data:

> **Instruction**: Recommend ten other movies based on the user's watching history.
> **Input**: The user watched movies "Pink Floyd - The Wall", "Canadian Bacon", "G.I. Jane", "Across the Universe", ..., "Down by Law".
> **Label**: "Almost Famous", "Full Metal Jacket", "The Hurt Locker", ...

> **Instruction**: Recommend ten other songs based on the user's listening history.
> **Input**: The user listened to songs "Jolene by Dolly Patron", "I Walk the Line by Johnny Cash", "Ring of Fire by Johnny Cash", ..., "Take Me Home, Country Roads by John Denver".
> **Label**: "Fancy by Reba McEntire", "Sweet Dreams by Patsy Cline", "Coat of Many Colors", ...

At the inference stage, the LLM can infer potential edges between a user node and entity nodes that are not currently connected to the user node in the FIG. Then these predicted potential edges are utilized in the graph traversal through paths like "User X $\rightarrow$ Predicted Entity A $\rightarrow$ User Y". We collect rewrite candidates and their associated features on the "Predicted Entity A $\rightarrow$ User Y" edges to enrich the User X's collaborative index.

3.5 Search-Based Collaborative QR System

Our collaborative QR system adopts a search-based approach similar to previous QR systems (Fan et al., 2021b; Cho et al., 2021; Naresh et al., 2022; Cai et al., 2023), follows a two-stage design consisting of a retrieval module (L1) and a ranking module (L2), as illustrated in Figure 2. The collaborative user index serves as both the search space and ranking feature store.

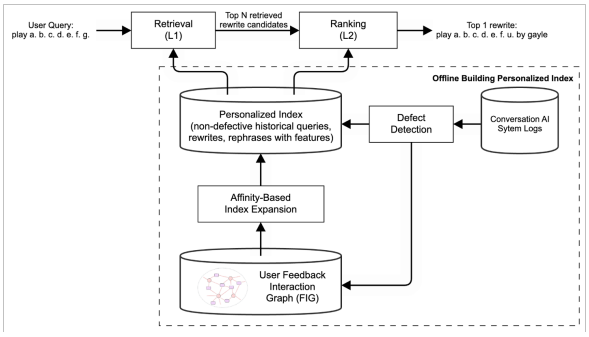

Figure 2: The high-level workflow of our search-based collaborative query rewriting system. It consists of a retrieval module (L1) and a ranking module (L2). A personalized index (collaborative user index) is created based on the FIG to serve as both the search space and ranking feature store.

#	System	Opportunity Test Set (Seen Interactions)		Opportunity Test Set (Unseen Interactions)		Guardrail Test Set
		Precision	Trigger Rt.	Precision	Trigger Rt.	False Trigger Rt.
1	**Personalized QR(Baseline)**	82.0%	79.5%	N/A	N/A	10.4%
2	**Collaborative QR**	78.3%	**82.4**%	74.5%	4.77%	12.5%
3	**+ L1 Encoder More Transformer layers**	80.2%	80.9%	76.5%	4.82%	8.6%
4	**+ L2 affinity/guardrail-based features**	**85.2**%	81.5%	**83.1**%	**5.01**%	**2.1**%

Table 1: Evaluation results on the offline test sets (see Section 4.1.1 for dataset details). Comparison between 1 and 2 shows Collaborative QR enables rewrites for user *unseen interactions*. However, the expanded user index in Collaborative QR degrades the rewrite quality (lower precision on the *seen interactions* and higher false trigger on the *guardrail test set*). To mitigate, we introduce more transformer layers and add the affinity & guardrail features (see Appendix A.2). With these updates, the Collaborative QR system is able to outperforms the existing Personalized QR system for seen interactions and the guardrail test set.

The retrieval module in our collaborative QR system aims to retrieve a set of relevant rewrite candidates from the index. The goal is to maximize recall with low latency and computational cost. Our production system uses a Transformer-based model as the utterance encoder, by taking a similar approach as (Chen et al., 2020a)[3]. The learning objective of the retrieval module is to project the embeddings of the input query and that of target rewrite closely.

After retrieving potential rewrite candidates, the ranking module leverages a gradient boosting ranker model to select the most suitable rewrite. The current ranker incorporates various aforementioned feedback signals as features calculated at both the global level and the user level. For example, the user level impression feature counts the number of times the query appears in the user history, while the global level impression feature indicates its occurrence across histories of all users.

The collaborative user index expands search space. While this expansion adds new opportunities, it also introduces more noise. Table 1 row #2 shows the QR quality is notably harmed by the increased search space. To mitigate, we adopt a strategy of increasing the size of the encoder by stacking more Transformer layers in the retrieval module. We further incorporate *guardrail features* and *graph-based features* to address false triggers and ensure system precision (see Appendix A.2).

4 Experiments

In this section, we first demonstrate the effectiveness of collaborative QR, that the collaborative user index built through graph traversal can boost defect reduction, and we are able to achieve competitive precision performance with the much enlarged index after applying techniques mentioned in Section 3.5. After that, we demonstrate the potential of applying LLMs to significantly further boost collaborative user index coverage even with a smaller index size, and thereby further improve the defect reduction ability.

4.1 Collaborative QR With Graph Traversal

4.1.1 Data

The offline evaluation of our graph-traversal based collaborative QR system includes two *opportunity test sets* and one *guardrail test set*. As mentioned in Section 3.3, the user index size is limited to 500.

- The opportunity test sets are weakly-labeled data, similar to previous works (Fan et al., 2021b; Cho et al., 2021). We begin by identifying pairs of consecutive user utterances, where the first turn is defective but the second turn is successful. We utilize a defect detection model (Gupta et al., 2021) to determine whether an utterance is defective or not. To minimize potential noise in the data and identify pairs where the second utterance is indeed a rephrase of the first utterance, we apply additional filters such as edit-distance and ASR n-best filters. Finally, the second utterance in the pair will be used as the rewrite label for the first utterance. We create two opportunity test sets: 1) *Seen Interactions*: the rewrite label exists in the user's own history; 2) *Unseen Interactions*: the rewrite label is not found in the user's history.

- The guardrail test set consists of historically successful user query utterances. The QR system should not trigger rewrite for any test case in the guardrail test set.

[3]We stack Transformer layers as the L1 encoder model that can run within the latency budget.

4.1.2 Evaluation metrics

For opportunity test sets, we use precision and trigger rate as metrics. Precision measures how often the triggered rewrite's NLU result matches the rewrite label's NLU result. Trigger rate represents the ratio between rewrite-triggered test cases and all test cases. The QR component is triggered when the prediction score of the top 1 rewrite is above an empirically chosen threshold.

For the guardrail test set, we utilize the false trigger rate as the metric. This rate also represents the ratio between the number of test cases that trigger a rewrite and the total number of test cases. However, in the guardrail test set, these cases should not be triggered. Therefore a lower false trigger rate is indicative of a better QR system.

4.1.3 Offline Evaluation Results

Table 1 shows the performance of collaborative QR on the test sets. We use a personalized QR system (Cho et al., 2021) as the baseline. As indicated by #1 and #2, collaborative QR is capable of enabling rewrites on the "unseen interactions" test set, which the baseline system couldn't handle at all. However, the precision drops significantly on the "seen interactions" test set (78.3% compared to 82.0%) due to the much larger search space of the collaborative user index, leading to a higher rate of false triggers. To mitigate this performance degradation, as discussed in Section 3.5, we introduce a larger utterance encoder to the L1 retrieval and incorporate guardrail features to the L2 ranking. Following these improvements, as shown by #4, we achieve better precision performance (85.2% compared to 82.0%). Furthermore, we notice a substantial reduction in the false trigger rate on the guardrail test set (10.4% reduced to 2.1%).

4.1.4 Online Evaluation Results

We have deployed our collaborative QR system and evaluated its online performance. Overall it introduces **23%** additional personalized defect removal. In the A/B experiment, we observed a statistically significant **19%** relative reduction of the defect rate with a p-value < 0.0001.

4.2 Collaborative QR Enhanced By LLMs

4.2.1 Collaborative user index coverage

Table 2 shows the coverage of unseen interactions by collaborative user indexes constructed using different methods. We evaluate the coverage on two dominant domains Video and Music (∼80% of traffic volume). Notably, the Dolly-V2 enhanced index with a size cap of 200 significantly outperforms graph-traversal based index with a size cap of 500.

	Video			Music		
Index Construction Method	**100**	**200**	**500**	**100**	**200**	**500**
Graph Traversal Only	1.8%	3.8%	6.3%	1.1%	2.7%	5.4%
+Dolly-V2 Link Prediction (not fine-tuned)	2.5%	5.3%	N/A	1.4%	3.6%	N/A
+Dolly-V2 Link Prediction (fine-tuned)	**10.8%**	**24.5%**	N/A	**8.5%**	**18.4%**	N/A

Table 2: Comparison of unseen user interaction coverage by collaborative user indexes constructed by different methods, with index size cap 100, 200 and 500.

4.2.2 Offline Evaluation Results

Table 3 shows QR performance for using collaborative user indexes constructed by different methods, with collaborative user index size reduced to 200 to save runtime latency and cost. LLM-enhanced collaborative user index achieves significantly higher trigger rate with comparable precision due to its much higher coverage.

4.2.3 An example of LLM-driven rewrite trigger

A user likes playing musicals, and has historically listened to musicals such as "My Fair Lady", "The Sound of Music", "Hamilton", etc. The user interacted another musical "Guys and Dolls" in the next week. This musical was not in the 3-hop user affinity in FIG (but in the 5-hop affinity) and hence not captured by the graph traversal. The LLM infers this potential future affinity. As a result, we covered a defective query with ASR error "could you gods dolls" and rewrite it as "play guys and dolls". To get some insight, we asked the fine-tuned Dolly-V2 7B model the following question.

> User previously listened to "My Fair Lady", "The Sound of Music", "Hamilton", ... Why would you recommend "Guys and Dolls"?

It responded

> "Guys and Dolls" is a classic well-known musical in the same genre.

This example serves to illustrate that there exists an inherent knowledge graph within the parameters of the LLM. By observing user affinity, the LLM could utilize this internal knowledge to infer user preferences that may extend beyond the physical topology of the user-entity interaction graph.

Index Construction Method	Video			Music		
	p@1	trigger	false trigger	p@1	trigger	false trigger
Graph Traversal Only	**81.5**%	3.7%	2.7%	79.7%	2.2%	1.8%
+Dolly-V2 Link Prediction (fine-tuned)	81.3%	**19.6**%	**2.2**%	**81.5**%	**15.4**%	**1.7**%

Table 3: Comparison of the QR performance for unseen user interactions, using collaborative user indexes built using different methods. Collaborative index size is reduced from 500 to 200 in comparison to graph-traversal enhanced index.

5 Conclusion

In this paper, we initially highlight the potential advantages of query rewriting in addressing users' unseen interactions. We then propose the "Collaborative Query Rewriting" approach that aims to reduce defective queries arising from unseen interactions. Performance degradation due to an enlarged index was rectified by implementing additional transformer layers for the L1 retrieval model and incorporating guardrail and graph features in the L2 ranking model.

Furthermore, we investigated the potential of an LLM in enhancing the collaborative QR approach. We found great potential for an LLM to significantly improve the coverage of the collaborative user index that can lead to a significant 5 to 6 times more reduction of query defects.

Limitations

The collaborative user index size limit is a major production concern and blocker as it is the main latency factor. The index built by graph traversal needs a large limit 500 to ensure sufficient coverage. We find this leads to higher timeout ratio during runtime. The LLM-based approach can build an collaborative index of much higher coverage at an even smaller index size limit 200 and improves runtime latency, but it does demand much higher index building cost. As a future course of action, we aim to experiment techniques such as distillation, teacher models, etc. to optimize the LLM-based index building cost.

Ethics Statement

Our team places the utmost importance on maintaining customer confidentiality and privacy. All customer data utilized in our research and development processes are anonymized to ensure privacy. Furthermore, all experiments are conducted in isolated environments to provide an additional layer of data security. The runtime system operates within a fully encrypted environment, adding another layer of protection for customer data. Lastly, the design principles of our system adhere to the commitment of unbiased data usage and AI algorithms, emphasizing fair and equitable treatment of all user interactions.

References

Jinglun Cai, Mingda Li, Ziyan Jiang, Eunah Cho, Zheng Chen, Yang Liu, Xing Fan, and Chenlei Guo. 2023. KG-ECO: Knowledge Graph Enhanced Entity Correction For Query Rewriting. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 1–5.

Zheng Chen. 2023. PALR: Personalization Aware LLMs for Recommendation. *arXiv preprint arXiv:2305.07622* (2023).

Zheng Chen, Xing Fan, and Yuan Ling. 2020a. Pre-training for query rewriting in a spoken language understanding system. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 7969–7973.

Zheng Chen, Xing Fan, Yuan Ling, Lambert Mathias, and Chenlei Guo. 2020b. Pre-Training for Query Rewriting in A Spoken Language Understanding System. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings* (2020). http://dx.doi.org/10.1109/ICASSP40776.2020.9053531

Eunah Cho, Ziyan Jiang, Jie Hao, Zheng Chen, Saurabh Gupta, Xing Fan, and Chenlei Guo. 2021. Personalized search-based query rewrite system for conversational ai. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*. 179–188.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. *Free Dolly: Introducing the World's First Truly Open Instruction-Tuned LLM*.

Zeyu Cui, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. 2022. M6-Rec: Generative Pretrained Language Models are Open-Ended Recommender Systems. *arXiv preprint arXiv:2205.08084* (2022).

Mostafa Dehghani, Sascha Rothe, Enrique Alfonseca, and Pascal Fleury. 2017. Learning to attend, copy, and generate for session-based query suggestion. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1747–1756.

Xing Fan, Eunah Cho, Xiaojiang Huang, and Chenlei Guo. 2021a. Search based Self-Learning Query Rewrite System in Conversational AI. *In 2nd International Workshop on Data-Efficient Machine Learning (DeMaL)* (2021).

Xing Fan, Eunah Cho, Xiaojiang Huang, and Edward Guo. 2021b. Search based self-learning query rewrite system in conversational ai. (2021).

Saurabh Gupta, Xing Fan, Derek Liu, Benjamin Yao, Yuan Ling, Kun Zhou, Tuan-Hung Pham, and Edward Guo. 2021. Robertaiq: An efficient framework for automatic interaction quality estimation of dialogue systems. (2021).

Wang-Cheng Kang, Jianmo Ni, Nikhil Mehta, Maheswaran Sathiamoorthy, Lichan Hong, Ed Chi, and Derek Zhiyuan Cheng. 2023. Do LLMs Understand User Preferences? Evaluating LLMs On User Rating Prediction. *arXiv preprint arXiv:2305.06474* (2023).

Elan Markowitz, Ziyan Jiang, Fan Yang, Xing Fan, Tony Chen, Greg Ver Steeg, and Aram Galstyan. 2023. Multi-Task Knowledge Enhancement for Zero-Shot and Multi-Domain Recommendation in an AI Assistant Application. *arXiv preprint arXiv:2306.06302* (2023).

Niranjan Uma Naresh, Ziyan Jiang, Ankit, Sungjin Lee, Jie Hao, Xing Fan, and Chenlei Guo. 2022. PENTATRON: PErsonalized coNText-Aware Transformer for Retrieval-based cOnversational uNderstanding. In *Conference on Empirical Methods in Natural Language Processing*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems* 35 (2022), 27730–27744.

Hui Su, Xiaoyu Shen, Rongzhi Zhang, Fei Sun, Pengwei Hu, Cheng Niu, and Jie Zhou. 2019. Improving multi-turn dialogue modelling with utterance rewriter. *arXiv preprint arXiv:1906.07004* (2019).

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca.

Yancheng Wang, Ziyan Jiang, Zheng Chen, Fan Yang, Yingxue Zhou, Eunah Cho, Xing Fan, Xiaojiang Huang, Yanbin Lu, and Yingzhen Yang. 2023. Recmind: Large language model powered agent for recommendation. *arXiv preprint arXiv:2308.14296* (2023).

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2021. Finetuned Language Models Are Zero-Shot Learners. *ArXiv* abs/2109.01652 (2021).

A Appendix

A.1 Heuristic Rules For Graph Traversal

To identify more promising rewrite candidates while maintaining the size of the collaborative user index, we apply the following heuristic rules during the construction of the collaborative user index via graph traversal:

- We only consider a traversal path when the two user nodes on this path have at least 3 common neighbors in the graph. For example, given a traversal path "User X → Entity A → User Y → Entity B", two user nodes "User X" and "User Y" must share at least 3 common entity neighbors. This ensures that the two users indeed exhibit similar behaviors to each other.
- For various entity types, we consider different maximum traversal path lengths. Specifically, for entities that might encapsulate more personalized information, like songs, albums, artists, books, videos, and shopping items, we set the maximum path length to 3, following the pattern: "User X → Entity A → User Y → Entity B". For entities that may represent less personalized information, like genres, apps, cities, and states, we establish a maximum path length of 2, which follows the pattern: "User X → Entity A → User Y".

A.2 Graph-Based Features & Guardrail Features for Ranking

During the ranking stage, we further incorporate guardrail features and graph-based features to address false triggers and ensure system precision.

Graph-based features refer to statistical features derived from the FIG graph:

- **User-Entity Nodes Distance Feature**: This feature represents the distance between the user node and the entity node associated with the rewrite candidate. In our context, possible distances are 1, 2, or 3. For instance, considering the collaborative index for "User X" and a path "User X → Entity A → User Y → Entity B", if the rewrite candidate stems from the edge "User Y → Entity B", the distance is set to 3.
- **User-User Nodes Relation Features**: These features depict the relationship between two user nodes linked with the rewrite candidate. For example, when a rewrite candidate is derived through the path "User X → Entity A

N-Hop Traversal	1	2	3	4	5
% **Unseen Interactions Covered**	0%	10%	20%	26%	31%
% **Defective Unseen Interactions Covered**	0%	12%	24%	32%	40%
Avg. # of Rewrite Candidates	<100	∼600	∼3K	∼20K	∼100K

Table 4: The coverage of unseen user interactions by the collaborative user index, which is constructed by up to 5-hop graph traversal within FIG. This FIG is derived from a user history spanning one year, and the assessment is based on interactions from a subsequent week.

→ User Y → Entity B", the two user nodes in consideration are "User X" and "User Y". The features encompass the number of shared neighbors between the two user nodes, the difference in their degrees, and their Jaccard similarity score, derived from their respective neighbors in the graph.

Guardrail features are used to prevent the entity-swap error. An entity-swap is a common error that arises due to the expanded index, where the accurate entity in the original query gets substituted by a similar one. For instance, when a user queries "play songs by pink", intending to refer to the artist "Pink", the collaborative user index might mistakenly suggest a rewrite like "play songs from Pink Floyd" and introduce a different artist "Pink Floyd". Empirically, we've found that certain guardrail features, such as entity impression, entity defect rate, and the similarities between the entities in the original query and the rewrite candidate, are highly effective in preventing entity-swap errors.

A.3 Unseen User Interaction Coverage Through Different Hops Of Graph Traversal

Table 4 indicates that a significant 40% of the defective unseen interactions can be addressed within the 5-hop graph traversal as detailed in Section 3.3. This observation is a main motivation for this work. However, in production, we opt for a maximum 3-hop graph traversal. This choice arises from concerns over computational expenses and the potential for increased noise resulting from an expanded user index.