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

Zheng Chen*
zgchen@amazon.com
Amazon Alexa AI
Seattle, WA, USA

Ziyan Jiang*
ziyjiang@amazon.com
Amazon Alexa AI
Seattle, WA, USA

Fan Yang*
ffanyang@amazon.com
Amazon Alexa AI
Seattle, WA, USA

Eunah Cho†
eunahch@amazon.com
Amazon Alexa AI
Seattle, WA, USA

Xing Fan†
fanxing@amazon.com
Amazon Alexa AI
Seattle, WA, USA

Xiaojiang Huang†
xjhuang@amazon.com
Amazon Alexa AI
Seattle, WA, USA

Yanbin Lu†
luyanbin@amazon.com
Amazon Alexa AI
Seattle, WA, USA

Aram Galstyan‡
argalsty@amazon.com
Amazon Alexa AI
Seattle, WA, USA

ABSTRACT

Conversational AI systems like Alexa, Siri, and Google Assistant require an understanding of defective queries to ensure robust conversational functionality and minimize user friction. Such defective queries often stem from user ambiguities, errors, or inaccuracies in automatic speech recognition (ASR) and natural language understanding (NLU).

A *Personalized query rewriting* (personalized QR) system strives to minimize defective queries by considering individual user behavior and preferences. It primarily depends on a user index of past successful interactions with the conversational AI. However, this method faces challenges with *unseen* interactions, i.e., those novel user interactions not covered by the user's historical index. Unseen interactions can constitute half of the total user traffic volume, even when the user index is built from a year's worth of history.

This paper introduces our "*Collaborative Query Rewriting*" approach, aiming at utilizing the underlying topological information to assist in rewriting defective queries arising from unseen user interactions. This approach first builds a "User **F**eedback **I**nteraction **G**raph" (*FIG*) of historical user-entity interactions. We then traverse the multi-hop user affinity to create an additional index, referred to as the "*collaborative user index*".

This paper then further explore the use of Large Language Models (LLMs) in conjunction with graph traversal to further boost the coverage of unseen interactions. We fine-tuned the LLM "Dolly-V2" 7B model on 200K users' affinities, leading to a significant increase in index coverage for future user interactions in comparison to only using graph traversal. This improvement, in turn, not only significantly enhances the performance of query rewriting (QR) for unseen queries, but also facilitates easier balancing between the index size and the coverage, a crucial requirement for the deployment of our approach into production.

CCS CONCEPTS

• **Information systems** → **Information retrieval**;

KEYWORDS

Collaborative Filtering, Large Language Models, Query Rewriting

ACM Reference Format:
Zheng Chen, Ziyan Jiang, Fan Yang, Eunah Cho, Xing Fan, Xiaojiang Huang, Yanbin Lu, and Aram Galstyan. 2018. Graph Meets LLM: A Novel Approach to Collaborative Filtering for Robust Conversational Understanding. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX)*. ACM, New York, NY, USA, 8 pages. https://doi.org/XXXXXXX.XXXXXXX

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.

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

†Team leaders. Authors are alphabetically ordered by last name.

‡Amazon scholar.

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

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 [7, 13, 14, 16]. 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 is intended for the song "abcdefu" by the American singer Gayle.

A *search-based personalized QR* system [7] typically requires a *personalized index* to reflect historical non-defective experiences for each user (see Section 3.4 and Figure 2 for more details). We refer to a personalized index built from the user's own history as the *user history index*. The user history index includes each user's own historical successful queries, rephrases, and rewrites, and related metadata & statistics (e.g. how many times user said a query in the history). 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 degree of match is determined through methods such as elastic search retrieval or a neural similarity model. If a match is found, it is used to rewrite the initial query. The user interactions covered by the user history index are referred to as 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 within a week-long period not being covered by the user history index, even when the index is constructed from a year-long history. We refer to these queries/interactions as "*unseen*". Moreover, we observed a roughly 7% higher defects in the unseen queries/interactions, which highlight the potential opportunity for the unseen interactions.

We introduce our approach, "*Collaborative Query Rewriting*" (Collaborative QR), which aims to address the challenge of index coverage. This approach is inspired by our observation that users who interact with similar entities through a conversational AI system 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. Each interaction edge in the FIG corresponds to historical successful queries/rephrases made by users, successful rewrites generated by the QR system, and feedback signals from users(see Section 3.1). Our key idea is to leverage the FIG to form a *collaborative user index* consisting of additional rewrite candidates not in the user history index.

Graph traversal through the FIG is the most straightforward approach for constructing the collaborative user index. We search user affinity in the FIG along "user→entity→user→entity→..." paths, and employ rules to filter, rank, and select top historical non-defect query utterances in the user affinity. Table 1 shows a few coverage statistics that support this method. We built the FIG from one-year user history, and evaluate one-week future user interactions. We find a considerable 40% of the defective unseen interactions can be covered within the 5-hop affinity.

While graph traversal offers notable benefits, it also introduces certain challenges. In a production setting, traversal is limited to a maximum of 3 hops due to computational limitations. Even more critical is the issue of index size. A search-based system requires careful index size management as an overly large index could not only harm the retrieval precision, but also increase the runtime system's latency. However, for the collaborative user index generated through graph traversal, it requires a fairly large index size cap of 500 for each user to achieve satisfactory coverage of unseen interactions.[1]. Therefore, improving the ranking of the collaborative user index and reducing its size is an important problem to address for collaborative QR.

To enhance the collaborative user index, we look into *Large Language Models* (LLMs)[3, 15, 18]. LLMs have shown remarkable adaptability in various NLP tasks. However, to the best of our knowledge, there are no existing publications on leveraging LLMs to enhance the user index for query defect reduction and improve the robustness of conversational AI systems. In this paper, we explore the possibility of applying a publicly available LLM "Dolly-V2" 7B[12] in combination with graph traversal to further improve the coverage of the collaborative user index. The increased coverage would also facilitate balancing between coverage and index size cap[2]. The basic idea is straightforward - prompting the LLM with items the user has interacted with in the past, and ask the LLM to generate other items the user might be interested in the future. The method leverages the latent relational knowledge already embodied in the LLM, and breaks the limitation by the physical user affinity in the FIG graph (see the example in 4.2.3).

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.
(2) We show that we can achieve competitive QR performance for unseen user interactions compared to defect reduction on "seen" interactions. This is accomplished by creating a collaborative user index through graph traversal, with an index size cap for the collaborative user index at 500. We have validated our approach through production A/B testing, where it has shown significant improvements.
(3) We have explored the use of LLMs to learn from the user affinity in the FIG graph. After fine-tuning the "Dolly-V2" 7B model on 200K user affinity learning examples, we saw a significant increase in the index coverage for 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.

[1]The user history index only needs a much smaller size cap of 100 for each user. User history index only stores user's own historical interactions, and the cap at 100 is enough to for the index store all user historical interactions for the majority of users.

[2]Note that user index can be pre-computed offline, therefore LLM inference latency is not impacting the runtime system efficiency. The approach is feasible as long as the collaborative user index can be pre-computed within reasonable time and cost, say 2 weeks with 64 A-100 GPUs

n-Hop Affinity	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 in The Affinity	<100	~600	~3K	~20K	~100K

Table 1: Unseen user interaction coverage by the collaborative user index enriched by up to 5-hop user affinity in FIG. FIG is built from one-year user history and the evaluation is done on one-week interactions in the future.

2 RELATED WORKS

Many previous works have been proposed for robust conversational understanding through QR. For non-personalized QR, [2] examined paraphrase retrieval for reducing defects in question answering tasks. [20] suggested a Markov chain model for extracting users' reformulation patterns that can help identify rewrite pairs. [6] proposed a query encoder pre-trained with a large number of user rephrases and then fine-tuned for search-based QR task utilizing the query encoder. For personalized QR, [16] compared a retrieval model with a pointer-generator network with hierarchical attention for performing personalized rewrites within the smart home domain, [9] extended search-based QR in spoken dialogue systems by combining a global layer for generic, non-personalized rewrites and a personalized layer. [7] proposed a personalized index combining different types of user affinities, and introduced a re-ranking layer for the retrieval results.

There are a few prior works leveraging graph for query rewriting. [1] studied query rewriting for sponsored search using the historical click graph to identify rewrites that are similar to the user's query based on SimRank, where the goal is to identify more relevant ads to display to users. [21] created a customer-query interaction graph and applied it to non-personalized QR through, first pre-training query embeddings using different graph representation learning methods Deepwalk, ConvKB, and GraphSAGE, and then fine-tuning on the QR task.

There has been a surge of recent researches affirming LLM can learn from user affinity and make predictions or recommendations. [5] fine-tunes a LLaMA 7B model to learn from the user affinity of movie-lens dataset and the Amazon beauty dataset, and outperforms the SOTA models on the recommendation task. [11] 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. [8] 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.

In comparison with the prior works, our "Collaborative Query Rewriting" work distinguishes itself by designing a user-entity interaction graph for personalized QR, leveraging user affinity to enrich user index. Through graph traversal, we build our production collaborative QR system that achieves a significant higher trigger, while maintaining competitive precision and runtime latency. Through LLM affinity learning, we demonstrate that collaborative QR can still further substantially enhance defect reduction.

3 METHODOLOGY

3.1 User Feedback Interaction Graph

A conversational AI system requires user profiles and historical behavioral data for its personalization features. *Graph* emerges as a natural structure to represent user historical interactions with various entities through the conversational AI. 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[10] 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"[3]) 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[10], barge-in rate (the probability that the user interrupted the agent's response to this query), termination-rate (the probability that the user stopped the agent's response to this query).

3.2 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 3.1) and integrate them into our

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

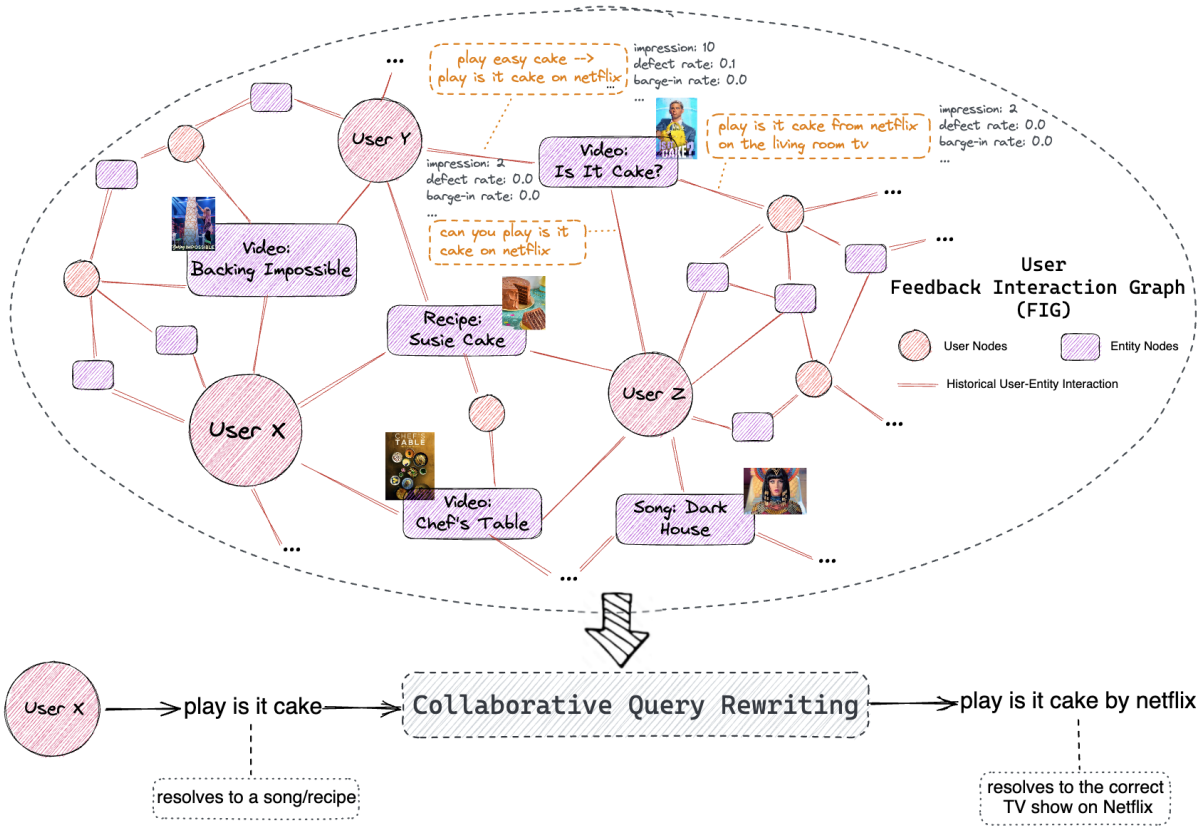

Figure 1: High-level illustration of the FIG and its application in collaborative QR. User X and Y interacted with the same or similar baking videos, which indicates their similarity. There was a successful historical rewrite "play easy cake → play is it cake on netflix" for user Y, which effectively 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".

collaborative user index. Within this process, user queries are considered potential candidates for rewriting, while feedback signals can serve as ranking features (see Section 3.4). Currently, we limit our consideration to a 3-hop traversal (that is, paths such as "User X → Entity A → User Y → 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).

After jointly considering runtime system constraints and unseen interaction coverage, we currently choose 500 as the collaborative user index size cap.

3.3 Collaborative User Index Enhanced By LLM

Large language models have showcased remarkable capability in deducing user preferences and predicting future behavior by analyzing historical interactions[5]. 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" model and apply instruction-based fine-tuning, a proven effective method in recent developments of large language models (LLMs)[17, 19]. Currently, our exploration is focused on the Music/Video domains (they are dominant domains taking about 80% of total user traffic volume), where we aim to leverage the factual knowledge stored within the LLM's parameters.

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 10 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 → Predicted Entity A → User Y". We collect rewrite candidates and their associated features on the "Predicted Entity A → User Y" edges to enrich the User X's collaborative index.

3.4 Search-Based Collaborative QR System

Our collaborative search-based QR system, similar to previous search-based QR systems[4, 7, 9], follows a two-stage design consisting of a retrieval module (L1) and ranking module (L2), as illustrated in Figure 2. The *personalized index* serves as both the search space and ranking feature store. In our system, the personalized index includes the user history index plus the collaborative user index, as described by Section 3.2 and Section 3.3.

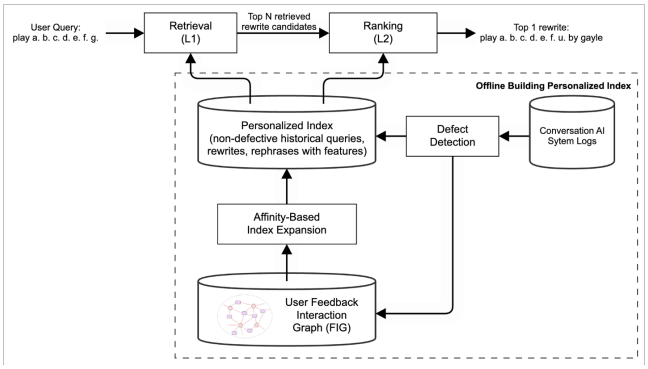

Figure 2: The high-level workflow of our search-based collaborative query rewriting system.

The retrieval module in our collaborative QR system aims to retrieve a set of relevant rewrite candidates from the personalized 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 [6][4]. The learning objective of the retrieval module is to project the embedding of 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. These features are calculated at 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's history, while the global level impression feature indicates its occurrence across all users' histories.

The collaborative user index expands search space. While this expansion introduces more opportunities, it also introduces more noise. Table 2 row #2 shows the QR quality is significantly harmed by the increased search space. To resolve this issue, 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 B).

[4]We stack Transformer layers as the L1 encoder model that can run within the latency budget. The runtime system has strict latency requirement and cannot yet host a larger model like BERT.

4 EXPERIMENTS

In this section, we first demonstrate the validity our essential idea of collaborative QR, that the collaborative user index built through graph traversal can already boost defect reduction, and we are able to achieve competitive precision performance with the much enlarged index after applying techniques mentioned in Section 3.4.

After that, we demonstrate the potential of applying LLM to significantly further boost collaborative user index coverage even with a smaller index size cap, and thereby introduce 5 to 6 times more defect reduction.

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.*

- The opportunity test sets are weak-labeled data similar to previous works[7, 9]. 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[10] 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.

4.1.2 Evaluation metrics. For opportunity test sets, we use precision and trigger rate as metrics. Precision measures how often the triggered top 1 rewrite's NLU hypothesis matches the rewrite label's NLU hypothesis. 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 query rewriting system.

4.1.3 Offline Evaluation Results. Table 2 shows the performance of collaborative QR on the test sets. As indicated by #1 and #2, collaborative QR is capable of enabling rewrites on the "unseen interactions" test set. However, the precision performance drops significantly (74.5% 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

#	Ranking	Opportunity Test Set (Seen Interactions)		Opportunity Test Set (Unseen Interactions)		Guardrail Test Set
		p@1	Trigger Rt.	p@1	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 guardrail/graph-based features**	**85.2%**	81.5%	**83.1%**	**5.01%**	**2.1%**

Table 2: Collaborative QR evaluation results (L1 retrieval + L2 ranking) on the test sets. The results verify our essential idea of Collaborative QR, and affirm competitive rewrite quality can be achieved for the unseen interactions, even when the collaborative user index size cap is much larger.

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

	Video			Music		
Index Construction Method	**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 4: Comparison of the QR performance for unseen user interactions, using collaborative user indexes built using different methods. Collaborative index size is limited to 200.

discussed in Section 3.4, we introduce a larger utterance encoder to the L1 retrieval and incorporate guardrail features to the L2 ranking. Following these improvements, as shown by #3 and #4, we achieve competitive precision performance (83.1% 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 deployed our collaborative QR system and evaluated its online performance. Overall it significantly introduces **23%** additional personalized defect removal. In the A/B experiment, we observed significant **19%** relative reduction of defect rate with p-value<0.0001.

4.2 Collaborative QR Enhanced By LLMs

4.2.1 Collaborative user index coverage. Table 3 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 fine-tuned Dolly V2 enhanced collaborative user index significant exceeds the performance of the graph traversal only method. The Dolly V2 enhanced index with index size 200 significantly outperforms graph-traversal based index with size cap 500.

4.2.2 Offline Evaluation Results. Table 4 shows QR performance for using collaborative user indexes constructed by different methods, with collaborative user index size limited to 200. LLM-enhanced collaborative user index achieves significantly higher trigger rate 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 particularly 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.

5 CONCLUSION

In this paper, we first identify a large volume of unseen user interactions occur every week with higher rate of defective queries. We then propose the "Collaborative Query Rewriting" approach that aims to reduce these defects specifically in 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. As a future course of action, we aim to experiment techniques such as distillation, teacher models, etc. to further optimize performance.

ACKNOWLEDGMENTS

REFERENCES

[1] Ioannis Antonellis, Hector Garcia-Molina, and Chi-Chao Chang. 2008. Simrank++ query rewriting through link analysis of the clickgraph (poster). In *Proceedings of the 17th international conference on World Wide Web*. 1177–1178.

[2] Daniele Bonadiman, Anjishnu Kumar, and Arpit Mittal. 2019. Large scale question paraphrase retrieval with smoothed deep metric learning. *arXiv preprint arXiv:1905.12786* (2019).

[3] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. *ArXiv* abs/2005.14165 (2020).

[4] 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.

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

[6] Zheng Chen, Xing Fan, and Yuan Ling. 2020. 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.

[7] 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.

[8] 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).

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

[10] 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).

[11] 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).

[12] Databricks Labs. 2021. Dolly: A Tool for Data Generation and Benchmarking. https://github.com/databrickslabs/dolly Available at: https://github.com/databrickslabs/dolly.

[13] Sen Li, Fuyu Lv, Taiwei Jin, Guiyang Li, Yukun Zheng, Tao Zhuang, Qingwen Liu, Xiaoyi Zeng, James Kwok, and Qianli Ma. 2022. Query Rewriting in TaoBao Search. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 3262–3271.

[14] 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*.

[15] OpenAI. 2023. GPT-4 Technical Report. *ArXiv* abs/2303.08774 (2023).

[16] Alireza Roshan-Ghias, Clint Solomon Mathialagan, Pragaash Ponnusamy, Lambert Mathias, and Chenlei Guo. 2020. Personalized query rewriting in conversational ai agents. *arXiv preprint arXiv:2011.04748* (2020).

[17] 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.

[18] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aur'elien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models. *ArXiv* abs/2302.13971 (2023).

[19] 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).

[20] Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020. Few-shot generative conversational query rewriting. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 1933–1936.

[21] Siyang Yuan, Saurabh Gupta, Xing Fan, Derek Liu, Yang Liu, and Chenlei Guo. 2021. Graph enhanced query rewriting for spoken language understanding system. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 7997–8001.

A RULES FOR CONSTRAINT USER AFFINITY

The following rules are applied to user affinity when building the collaborative user index as described in Section 3.2

- We only consider entities such as songs, albums, artists, books, videos and shopping items for 3-hop affinity. In addition, there must be three unique 2-hop "user X→entity A→user Y" paths from user X to user Y to make sure there is a reasonable level of connections between them, otherwise we do not consider Y's affinity for expanding X's index. This index expansion increases coverage of user X's unseen defective interactions at both the entity level and the query level.
- We in addition consider entities such as genres, apps, cities, states, device/routine/contact names for only 2-hop affinity. These entities are intuitively less personalized and it is easier to introduce high-degree nodes along the paths with them. Therefore we use this constraint to make sure we only leverage rewrite candidates from the common affinity of X and Y for these types of entities. This expansion still increases query-level coverage of user X's unseen defective interactions even though we do not introduce new entities into X's index.

B AFFINITY FEATURES & GUARDRAIL FEATURES FOR RANKING

After retrieving potential rewrite candidates, the ranking module leverages a gradient boosting ranker model to select the most suitable rewrite. The current ranker has considered impression and various aforementioned defect signals as features. These features are calculated for query utterances and entities at the global level and the user level. For instance, the user-level entity impression feature counts the number of times the entity appears in the user history. The same training data used for the retrieval model, but with the addition of features, is used for training the model.

We first introduce an additional level of features - *multi-hop affinity features*; for example, the entity impression feature at the affinity level counts the occurrences of the entity in the user's constrained affinity as described Section 3.2. In addition, we consider the following affinity features.

- The number of hops between the user and the rewrite candidate in the affinity, which can be 1,2,3 in our case.
- User X & Y similarity features, such as the number of unique paths between the two users, the sum of impressions of these paths, the degree difference between X and Y, and the Jaccard distance between X and Y's neighborhood. All rewrites candidates along the "user X→...→user Y→..." paths have the same feature values for all user X & Y similarity features.

We also introduce *guardrail features* to counteract false trigger, and especially prevent the entity-swap error. A bad entity-swap is the typical type of error due to the enlarged index. One user queried

"play songs by pink", which indeed meant an artist named "Pink". However, "Pink" was not present in the user history. Our enlarged index did cover the artist "Pink" associated with a rewrite candidate "play a million dreams by pink", but it also introduced another artist "Pink Floyd" with a rewrite candidate "play songs from pink floyd". The latter is falsely triggered due to its higher retrieval score from the current search-based QR system. We found query entity signals (e.g. impression, defect rate), and the similarities between query/rewrite entities are most critical to prevent the entity-swap error.