CGF: Constrained Generation Framework for Query Rewriting in Conversational AI

Jie Hao[1] **Yang Liu**[1] **Xing Fan**[1] **Saurabh Gupta**[2*] **Saleh Soltan**[1]
Rakesh Chada[1] **Pradeep Natarajan**[1] **Chenlei Guo**[1] **Gokhan Tur**[1]
[1]Amazon Alexa AI
[2]Linkedin
{jieha,yangliud,fanxing,ssoltan,rakchada,
natarap,guochenl,gokhatur}@amazon.com
saurabh3949@gmail.com

Abstract

In conversational AI agents, Query Rewriting (QR) plays a crucial role in reducing user frictions and satisfying their daily demands. User frictions are caused by various reasons, such as errors in the conversational AI system, users' accent or their abridged language. In this work, we present a novel Constrained Generation Framework (CGF) for query rewriting at both global and personalized levels. It is based on the encoder-decoder framework, where the encoder takes the query and its previous dialogue turns as the input to form a context-enhanced representation, and the decoder uses constrained decoding to generate the rewrites based on the pre-defined global or personalized constrained decoding space. Extensive offline and online A/B experiments show that the proposed CGF significantly boosts the query rewriting performance.

1 Introduction

Large-scale conversational AI agents such as Alexa, Siri and Google Assistant help millions of users to perform a lot of tasks, such as playing music, controlling light devices at home, etc. In general, such conversational AI agents have multiple components including automatic speech recognition (ASR) and natural language understanding (NLU). ASR is responsible for converting the speech signal of the user's query (e.g., "play Michael Jackson music") to a text transcript. Following this, NLU provides domain/intent classification (e.g., domain: Music, intent: PlayMusic) and entity labelling (e.g., ArtistName: Michael Jackson), which are used to fulfill the user's request.

However, sometimes there are frictions due to speech recognition or NLU errors. For example, ASR errors may lead to an erroneous transcript "*play alien bridges*", when the user actually meant "*play leon bridges*". Due to such errors, the downstream NLU system is affected, capturing a wrong entity "*alien bridges*" for the slot "*ArtistName*". This leads to a bad user experience and the user has to rephrase their query. Additionally current NLU technology has limitations and cannot handle all the user requests. For example, "tv to input three" cannot be properly handled by NLU (the user's intended request is "turn tv to h.d.m.i. three"). To reduce the friction and make the dialog system more robust, query rewriting (QR) (Ponnusamy et al., 2019; Chen et al., 2020) becomes an increasingly important technique in conversational AI agents. In production conversational AI agents, the QR component is often triggered when the system cannot process user requests with a good confidence. For example, if ASR or the named entity recognition (NER) confidence is low, the QR component can be triggered to automatically map a user query to another form, so that the dialog system can successfully take the right action.

Many existing QR systems use search-based pipelines for either global-wise query rewriting (Fan et al., 2021; Chen et al., 2020) or personalized query rewriting (Cho et al., 2021). These systems typically have two steps: retrieval and ranking. Users' historical defect-free interactions with conversational agents are used to construct the global or personalized index. When a new request arrives, the system compares it to those utterances in the index using a retrieval model such as a dual encoder with billion-scale similarity search (e.g., FAISS) (Johnson et al., 2017) and retrieves top N candidates from the index. Then a ranking model is used to rank these candidates with both neural semantic and IR features as input. The system picks the top 1 ranked candidate as the final rewrite. Such a search-based system is widely used in the large scale conversational AI agents since it can effectively control the output because of the use of the index and thus reduce the risky rewrites.

However, there are also limitations in such retrieval-based systems. First, the query and

*Work done when working at Amazon.

rewrite candidate affinity is mainly captured through a vector dot product and lacks token level modeling. Second, a large memory footprint is needed to store dense representations when a large index is used in the retrieval step.

In this work, we propose to leverage generation-based models under the Constrained Generation Framework (CGF) for the query rewriting task. Since little work has incorporated the previous context information in query rewriting, although its importance is recognized (Wu et al., 2018), we use the previous dialog context and the user's current request in the encoder. The decoder uses constrained decoding in inference to force the generated rewrite to be in a predefined candidate set. The proposed CGF enables us to mitigate the aforementioned shortcomings from the search-based system since the autoregressive formulation allows the model to directly capture relations between the contextual input and target rewrites and thus effectively cross encode both. Moreover, the memory footprint is greatly reduced because the parameters of our encoder-decoder architecture scale with the vocabulary size, not the index count. Though the neural language generation approaches are known to hallucinate content, our proposed constrained decoding approach with a predefined candidate set makes the generation model faithful to the model input and avoids the potential hallucinations or bad rewrites. We conducted extensive offline experiments for both global and personalized query rewriting to show the effectiveness of the proposed approach. Our online experimental results also demonstrate that the proposed CGF indeed generates rewrites of better quality.

2 Related Work

2.1 Query Rewriting

In dialogue systems, query rewriting benefits dialogue state tracking especially coreference resolution (Rastogi et al., 2019; Vakulenko et al., 2020; Hao et al., 2021), and in general can seamlessly replace the user's utterance in order to remove friction and unsatisfactory experience to users (Ponnusamy et al., 2019; Wang et al., 2021). To do this, Ponnusamy et al. (2019) proposed to reformulate the queries with a Markov Chain. Chen et al. (2020) proposed a retrieval-based model with a pre-training method. Fan et al. (2021) and Cho et al. (2021) leveraged multi-stage search-based systems to perform global and personalized query rewriting. In this work, we propose CGF based on Seq2Seq models to generate a rewrite of the initial user query.

Another thread of work that is related to query rewriting is the Grammatical Error Correction (GEC) task. GEC is the task of correcting different kinds of grammatical errors in text such as spelling, punctuation, and word choice errors. Recently, Seq2Seq based models have become the state-of-the-art approach for GEC (Zhao et al., 2019; Wang et al., 2019; Kaneko et al., 2020). The main difference between GEC and our query rewriting is that GEC is more concerned with grammatical corrections, and we focus on the errors from users, ASR or NLU systems to reduce the friction.

2.2 Constrained Generation

Constrained generation has been applied in many tasks such as machine translation and web search. Hokamp and Liu (2017) introduced grid beam search to allow the inclusion of pre-specified lexical constraints. Mohankumar et al. (2021) applied constrained decoding with a diverse sibling search algorithm for search advertising. To the best of our knowledge, ours is the first work that introduces the constrained decoding into query rewriting for conversational AI agents. Moreover, we extend the approach to personalized rewriting to take full advantage of the constrained generation.

3 CGF for Query Rewriting

As shown in Figure 1, we introduce the sequence-to-sequence (Seq2Seq) model to generate the rewrite, where a bidirectional encoder takes the context and current request as input, and an autoregressive decoder relies on the pre-defined index to perform the constrained decoding in order to generate the target rewrite.

3.1 Context-enhanced Modeling

We adopt the Seq2Seq pre-trained model BART (Lewis et al., 2020). It has the same model architecture as the widely-used Transformer model (Vaswani et al., 2017) and is pre-trained with a denoising way (Devlin et al., 2019). In this work, we flatten the previous dialogue turns (including both user requests and agent responses) and the current user request into a single sequence for the encoder input, as shown in Figure 1, and fine-tune BART.

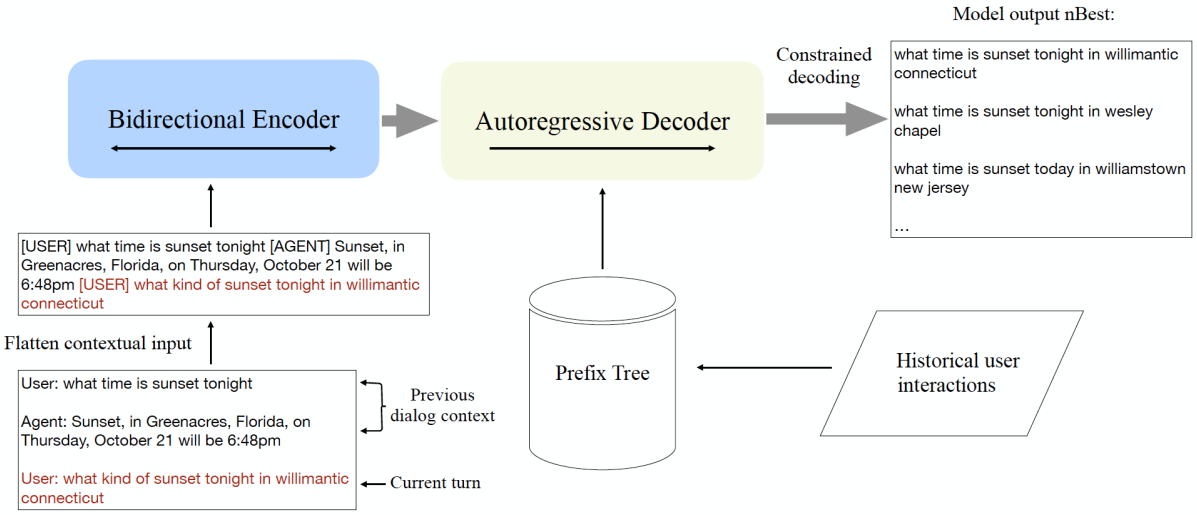


Figure 1: Illustration of the Constrained Generation Framework (CGF) for query rewriting. When a new utterance arrives, the model performs the contextual encoding and constrained decoding and outputs the final rewrites. "Model output nBest" denotes multiple candidates generated using beam search.

Formally, given a context-enhanced request sequence $\mathbf{Q} = \{q_1, ..., q_M\}$, where q_i denotes a token in the sequence, and the corresponding rewrite $\mathbf{R} = \{r_1, ..., r_N\}$. The encoder is responsible for reading the input request and its previous dialogue turns, and the decoder autoregressively generates the rewrites. Given the hidden representations of the context-enhanced request and the rewrite, the conditional probability of the n-th target word $\mathbf{r_n}$ is calculated as following:

$$\mathbf{H}_{Enc} = \text{ENC}_{BART}(\mathbf{Q}^0), \quad (1)$$

$$\mathbf{H}_{Dec} = \text{DEC}_{BART}(\mathbf{R}^0, \mathbf{H}_{Enc}) \quad (2)$$

$$\mathbf{P}(\mathbf{r_n}|\mathbf{R_{<n}}, \mathbf{Q}; \theta) = Softmax(Proj(\mathbf{h_n})) \quad (3)$$

where $\mathbf{h_n}$ is the n-th hidden representation of $\mathbf{H}_{Dec}$. $Proj()$ and $Softmax()$ are two transformation functions in the output layer of the decoder (Vaswani et al., 2017).

3.2 Constrained Decoding

Neural language generation approaches are known to hallucinate content, resulting in generated text that conveys information that does not appear in the input. For example, if a user has a request "*play broadway girls*", the model with free-style generation can generate a rewrite "*play broadway girls by morgan wade*". This is factually wrong since "*morgan wade*" never sings the song "*broadway girls*". This is because general generative models leverage the beam search over the entire vocabulary and thus there is a chance of generating fluent but factually incorrect sentences. Thus, the inability to effectively control the generated text has become one of the biggest obstacles for adopting generative models for query rewriting in conversational AI. In this work, we propose to use constrained decoding in the generative models to reduce the potential bad rewrites.

Beam search has been widely used in Seq2Seq models during inference to improve the search quality. The standard beam search consists of selecting the top B hypotheses with the highest log probability $S(\hat{r}_t, \hat{r}_{<t}|Q) = S(\hat{r}_{<t}|Q) + logP(\hat{r}_t|\hat{r}_{<t}, Q)$ at each time step t, where $\hat{r}_t$ denotes the token in the generated hypothesis. Allowing to generate any token from the vocabulary at every decoding step might lead the model to generate output strings that are not valid (i.e., bad rewrite). Hence, we resort to constrained beam search, forcing to only decode valid rewrites from a predefined candidate set. We define our constraint in terms of a prefix tree T, where nodes are tokens from the vocabulary. For each node $t \in T$, its children indicate all the allowed continuations from the prefix, which is defined as traversing the trie from the root to t. More formally, when decoding the token r_t at time step t, the constrained probability distribution is calculated as:

$$\widetilde{P} = \begin{cases} P(\hat{r}_t = r|\hat{r}_{<t}, Q), & \text{if } r \in \text{suffix}_T(\hat{r}_{<t}) \\ 0, & \text{otherwise} \end{cases}$$

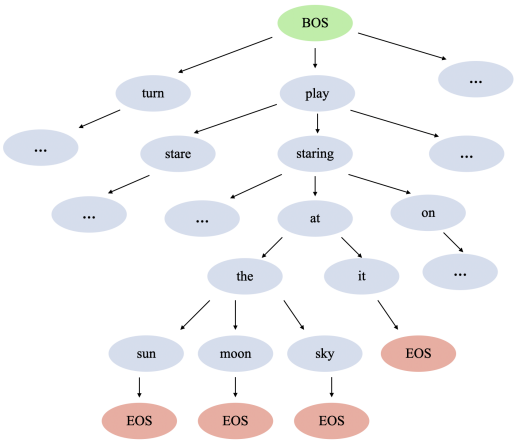


Figure 2: A snapshot of the utterance trie we construct based on the global index. When the model has generated a sequence "[BOS] *play staring at*" during the decoding process, in the next step, using the pre-defined trie, the model is only allowed to generate either "*the*" or "*it*". Then, if the model generates "*the*" next, it is only allowed to generate one of the three words "*sun*", "*moon*" or "*sky*" in the step after it.

where we remove all the tokens r that are not a suffix of the already generated sequence $\hat{r}_{<t}$ in the trie. In this way, we can ensure that the model is only allowed to generate the rewrites from the predefined candidates set.[1]

In the trie shown in Figure 2, each path from the root node to the leaf node (e.g., [BOS] $\rightarrow$ *play* $\rightarrow$ *staring* $\rightarrow$ *at* $\rightarrow$ *it* $\rightarrow$ [EOS]) represents an utterance that we allow the model to generate. "[BOS]" is the special token indicating the beginning of a sequence. Similarly, "[EOS]" denotes the end of a sequence.

3.3 Global and Personalized Query Rewriting

Constrained generation with the predefined decoding space can not only reduce the risks, but also offer flexibility to conduct rewrite with utterance sets predefined at different granularities. In this section, we introduce how to conduct the global and personalized query rewriting with CGF.

Global Query Rewriting Global query rewriting means that the rewrite for a request is applicable for all the users. For example, for a query "*tv to input three*", the ideal rewrite for this query is "*turn tv to h. d. m. i. three*", which is applicable to all the users who might say this request. In the proposed CGF, we pre-define the global constrained decoding space to include all the rewrite candidates that the model is allowed to generate. To achieve this, inspired by the approach to construct the global index in Fan et al. (2021), we build the global trie that provides rewrite candidates extracted from all the users' interactions. The global trie is generated from the aggregated, anonymized historical interactions between the users and the agent within a period of time (e.g., 30 days). In addition, after collecting all the user historical interactions, we rely on a defect detection model (Gupta et al., 2021) to filter out the defective utterances. Note that since constrained decoding with the trie doesn't need to store dense vectors of the index, we can reduce the memory footprint greatly and thus potentially enlarge the trie comparing to the index of search-based models in real online systems.

Personalized Query Rewriting A crucial nature of query rewriting is that often it needs to reflect personal preference or personalized error types to recover from the defect (Cho et al., 2021). For example, for the same defective request "*turn on the moon*", the intended request for user A may be "*turn on the moonlight sonata*", whereas user B might want to "*turn on the moon lamp*". Thus, the global query rewriting described above can not handle such cases. It is necessary to have a personalized query rewriting system to fill this gap. The vanilla Seq2Seq models are not able to perform personalized generation naturally. In contrast, our proposed CGF can allow the generation-based models to perform personalized query rewriting by using a personalized constrained decoding space for each user. For a request coming from a specific user, the model is only allowed to generate a rewrite from the pre-defined personalized decoding space. We follow Cho et al. (2021) to build the constrained decoding space for each user, leveraging their individual interaction history. The utterances included in the constrained decoding space (i.e., trie) reflect satisfied experiences for each user within the past 30 days. In this work, we utilize the model trained with the global training data and apply the personalized trie on it for personalized rewriting.

[1] Note that we mask the probabilities of the invalid tokens and do not re-normalize the probability over the vocabulary. We found it is more effective this way.

4 Offline Experiments

4.1 Data

We train our proposed method with weak-labeled data annotated by a model (**Machine-Annotated**). Specifically, we first leverage a defect detection model (Gupta et al., 2021) to find two consecutive deidentified user utterances, where the first turn was defect, but the second turn was successful. Then, we further filter out consecutive utterances with a time gap larger than 35 seconds and edit distance larger than 5. For evaluation, we curated human-annotated test data (**Human-Annotated**). For both global and personalized test sets, we make sure the target rewrites are in the global/personalized constrained decoding space. Table 1 gives the statistics of the data set. Note that all the data has been de-identified.

Data Type	Machine		Human
	Train	Valid	Test
Global QR	6.5m	0.4m	6k
Personalized QR	6.5m	0.4m	5k

Table 1: Statistics of the query rewriting data sets. "Machine" denotes the Machine-Annotated data. "Human" denotes the Human-Annotated data.

4.2 Model Setup

In this work, we fine-tune the pre-trained BART model (Lewis et al., 2020). We compare our proposed model with several baselines. For global query rewriting task, we have two baselines: 1) **DPR** (Karpukhin et al., 2020): we follow a recent retrieval model DPR to train a dual BERT model. 2) **UFS-QR** (Fan et al., 2021): we implement the search-based approach **UFS-QR** that contains a retrieval layer and ranking layer. For personalized query rewriting, we have **Personalized UFS-QR** (Cho et al., 2021) and **DPR** as the baselines. **Personalized UFS-QR** extends the UFS-QR by incorporating the personalized features into the ranking model and index construction. In addition, we also compare with the CGF model that uses the global trie. More details of model training fro the CGF and baselines training can be found in Appendix A.1. We follow Fan et al. (2021) to build the global trie, which contains 27M unique utterances, and Cho et al. (2021) to build the personalized trie. On memory (disk space) footprint, the global trie we built is 856M, in contrast, the built FAISS index is 36G for UFS-QR and 89G for DPR with the same utterances.

System	Precision	Trigger Rate
DPR	0.0	0.0
UFS-QR	+10.59%	-13.22%
CGF	+36.62%	+275.23%
Ablations		
CGF w/o CE	+34.43%	+272.34%
CGF w/o CD	+34.97%	+274.02%
CGF w/o both	+32.74%	+266.17%

Table 2: Global query rewriting evaluation. We compare our proposed CGF with the existing search-based query rewriting systems on human annotated test sets. "CE" denotes context-enhanced encoding. "CD" denotes the constrained decoding. All the numbers are relative differences with respect to the baseline: "DPR".

4.3 Evaluation Metrics

For evaluation, we use utterance level precision and trigger rate. Precision denotes how often the triggered rewrite matches the correct rewrite. The trigger rate is the fraction of instances for which the model makes a prediction with the final beam score above a predefined threshold[2]. We set the threshold to -0.2 for our proposed CGF models.

4.4 Global Query Rewriting Results

Table 2 shows the CGF main results with ablations on the two human-annotated test sets. CGF with context-enhanced encoding and constrained decoding achieves the best performance on precision and trigger rate on the two test sets. Our approach outperforms the search-based UFS-QR system and retrieval system DPR by more than 14% and 21% on precision respectively. Moreover, the proposed approach can confidently trigger more cases.

Table 2 also lists the ablation study results for the global query rewriting task using CGF. "w/o both" denotes the CGF without context-enhanced encoding and constrained decoding, in which the model takes only the query as the encoder input and conduct the unconstrained generation. In particular, although we see that the overall performance of the "w/o CD" model is not bad, it still suffers from hallucinations. Examples with factually incorrect generation can be found in Appendix A.3 Table 5. It is clear that using context-enhanced encoding and

[2]The final beam score is formalized as $log(P_\theta(y|x)) = \Pi_{i=1}^{N} p_\theta(y_i|y_{<i}, x)$, where θ is the model parameters and x is the model input.

System	Precision	Trigger Rate
CGF (Global trie)	0.0	0.0
DPR (Personalized index)	+16.03%	-51.85%
Personalized UFS-QR	+16.61%	+15.69%
CGF (Personalized trie)	+19.33%	+17.68%

Table 3: Personalized query rewriting evaluation. All the numbers are relative differences with respect to the baseline: "CGF (Global trie)".

constrained decoding proved useful. Combining them together is better, resulting in higher precision and at the same time higher trigger rate.

4.5 Personalized Query Rewriting Results

Results for the personalized query rewriting on the Human-Annotated test set using our proposed CGF are in Table 3. We use the same trained model as the global query rewriting task. The only difference is that during inference, the constrained decoding space is changed to the personalized one based on each user's historical interactions and thus varies across users. As can be seen, CGF outperforms the CGF global model (i.e., Global trie). Also, it outperforms search-based Personalized UFS-QR and DPR respectively by 2.7% and 3.3% on precision, with a higher trigger rate.

5 Online Experiments

5.1 Deployment

In the online system, we run the global and personalized CGF models in parallel. When both the global and personalized components return a rewrite candidate, we prioritize the results from the personalized model over the global one to support any possible personalization of QR. No rewrite will be output from the system if neither model manages to generate a rewrite.

5.2 Online Results

To investigate the effectiveness of the introduced techniques, we leverage the proposed model CGF to generate the rewrites and deploy them into the online environment. We compare it with the no-CGF rewrites within the English speaking users environment. The data was collected for more than one week over a significant percentage of traffic via the A/B testing framework. We use one primary metric to evaluate the performance of our proposed CGF approach during A/B: *Defect Rate*. It denotes the total number of rewritten utterances that are defective divided by the total number of rewritten utterances. We leverage the defect detection model proposed by Gupta et al. (2021) to measure if an utterance is defective.

From A/B results, we observed significant[3] relative reduction of defect rate: **28.97%** and 1 million of new rewrites generated by the proposed approach per week. Table 4 shows the cases where the original requests had unsatisfying responses from the agent and after the rewrite, the friction was removed with satisfying responses. For example, due to an ASR error, the agent response to the original request "how old is tommy in it" cannot fulfill the user's need. Even without context information, i.e., when the request is the first turn, the CGF can successfully rewrite it, yielding the right response from the agent. More online examples can be found in Table 4.

5.3 Limitations

Trie Coverage Although we have used 27M unique utterances in the global trie and 30 day's non-defective turns for each user for constrained decoding, the proposed system cannot handle the cold start (e.g., a recent released song) or out-of-trie rewrite cases (the rewrite for the request is not in the trie). To mitigate this, we plan to update the global trie weekly and personalized trie daily. Also we will work on constraining a part of utterance generation (i.e., entity) instead of the entire utterance to enlarge the decoding space.

Latency Generation-based models always suffer from the latency issue due to its autoregressive generation process. In the CGF deployment, we changed the model to ONNX[4] version and speed up the inference by 30.6%. However, CGF is still 1.5 times slower than the search-based system (Fan et al., 2021). Considering this, we will explore the non-autoregressive approach and related model optimization approaches like distillation, pruning, etc.

6 Conclusion

In this work, we propose CGF, a novel paradigm for query rewriting: generate target rewrite autoregressively with context-enhanced encoding and constrained decoding. CGF is a general framework for different query rewriting purposes where one can

[3] p-value<0.0001

[4] https://github.com/onnx/onnx

Original request (w/o context): USER: how old is tommy in it **CGF rewrite:** how old is tommyinnit	**Agent response (before rewrite):** Here's what I found from the web: A 24-year-old man whose... **Agent response (after rewrite):** TommyInnit is 17 years old.	**Defect Reduction:** 50.20%
Original request (w/o context): USER: play night talk by drake **CGF rewrite:** play knife talk by drake	**Agent response (before rewrite):** I couldn't find night talk **Agent response (after rewrite):** Here's Knife Talk , by Drake (feat. 21 Savage, Project Pat) , on Amazon Music.	**Defect Reduction:** 41.52%
Original request (w/ context): USER: play little yancy AGENT: Lil' Fancy from Apple Music.y USER: play little yankees praise part **CGF rewrite:** play little yancy praise party	**Agent response (before rewrite):** Sorry, I'm having trouble finding the song. **Agent response (after rewrite):** Here is little yancy praise party.	**Defect Reduction:** 22.71%

Table 4: Online Performance of CGF with Qualitative Examples

freely define the decoding space (e.g., global, personalized or domain-specific space). Both offline and online experiments show that our approach consistently and significantly improves query rewriting performance.

References

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)*, pages 7969–7973. IEEE.

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*, pages 179–188.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Xing Fan, Eunah Cho, Xiaojiang Huang, and Chenlei Guo. 2021. Search based self-learning query rewrite system in conversational ai. In *2nd International Workshop on Data-Efficient Machine Learning (DeMaL)*.

Saurabh Gupta, Xing Fan, Derek Liu, Benjamin Yao, Yuan Ling, Kun Zhou, Tuan-Hung Pham, and Chenlei Guo. 2021. Robertaiq: An efficient framework for automatic interaction quality estimation of dialogue systems. In *2nd International Workshop on Data-Efficient Machine Learning (DeMaL)*.

Jie Hao, Linfeng Song, Liwei Wang, Kun Xu, Zhaopeng Tu, and Dong Yu. 2021. RAST: Domain-robust dialogue rewriting as sequence tagging. In *EMNLP*.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. *ACL*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*.

Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2020. Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction. In *ACL*.

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 *EMNLP*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*.

Akash Kumar Mohankumar, Nikit Begwani, and Amit Singh. 2021. Diversity driven query rewriting in search advertising. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3423–3431.

Pragaash Ponnusamy, Alireza Roshan Ghias, Chenlei Guo, and Ruhi Sarikaya. 2019. Feedback-based self-learning in large-scale conversational ai agents. *arXiv preprint arXiv:1911.02557*.

Pushpendre Rastogi, Arpit Gupta, Tongfei Chen, and Lambert Mathias. 2019. Scaling multi-domain dialogue state tracking via query reformulation. In *NAACL*.

Svitlana Vakulenko, Shayne Longpre, Zhucheng Tu, and Raviteja Anantha. 2020. Question rewriting for conversational question answering. *arXiv preprint arXiv:2004.14652*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *NIPS*.

Liang Wang, Wei Zhao, Ruoyu Jia, Sujian Li, and Jingming Liu. 2019. Denoising based sequence-to-sequence pre-training for text generation. *arXiv preprint arXiv:1908.08206*.

Zhuoyi Wang, Saurabh Gupta, Jie Hao, Xing Fan, Dingcheng Li, Alexander Hanbo Li, and Chenlei Guo. 2021. Contextual rephrase detection for reducing friction in dialogue systems. In *EMNLP*.

Xianchao Wu, Ander Martinez, and Momo Klyen. 2018. Dialog generation using multi-turn reasoning neural networks. In *NAACL*.

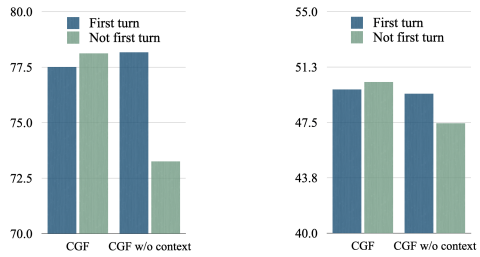
Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. In *NAACL*.

A Appendix

A.1 Model Set up

For CGF, we use a batch size of 2048 tokens, dropout rate of 0.1 and adam optimizer. The learning rate is 3E-5 and linearly warms up over the first 5% steps, then decreases proportionally to the inverse square root of the step number. All the models are trained on eight NVIDIA Tesla V100 GPU.

For the DPR baseline on global rewriting task, we follow Karpukhin et al. (2020) to train the dual BERT model with machine-annotated training set and in-batch negatives. During the inference, for each user request, we use FAISS (Johnson et al., 2019) search and return top K (K=1 in this work) relevant rewrites from a global index, which contains 27M unique requests as same as global trie. For the persoanlized rewriting task, similarly, the DPR will return a rewrite from the user's index, which contains 30 day's non-defective user histroical utterances as same as personalized trie.



(a) Precision (b) Trigger Rate

Figure 3: Global query rewriting evaluation on the first turn and not first turn subsets. "CGF w/o context" denotes the CGF without context-enhanced encoding.

A.2 Effect of Context-enhanced Modeling

We study the effect of the context-enhanced modeling in this subsection. As in some of test cases, there are not previous context available and the model will rewrite the first turn of the multi-turn dialogue session. We investigate if the proposed model is robust and effective for both of such cases. Thus, we split the global test set into the with previous context ("First turn") and without previous context cases ("Not first turn").

As shown in the Figure 3, the CGF gets significant improvement for both precision and trigger rate on the "Not first turn" test set comparing to CGF without context-enhanced encoding, which demonstrates the effect of the context information during the model training. Moreover, on the "First turn" test set, surprisingly, when there is no previous context for the CGF model, the performance only decreases slightly. This suggests that the model is good at generalization and robust for various test cases in the actual scenario.

A.3 Case Study

We present several representative cases so that we can further understand the effect of the context-enhanced encoding and constrained decoding in CGF. As shown in Table 5, the first example illustrates the cases when CGF w/o context-enhanced encoding gives a rewrite that changes the semantic meanings of the source request ("what kind of" -> "what is") and is not faithful. However, with consideration of previous context information, the CGF is able to understand the user intent and provide the accurate rewrite. The second case corresponds to the situation of carrying over an correct entity from context and replacing the wrong entity in the current utterance, while as shown in the table, this is not hard for our context-enhanced encoding

Dialog & Reference	CGF	CGF w/o CD	CGF w/o CE
USER: what time is sunset tonight AGENT: sunset, in greenacres, florida, on thursday, october 21 will be 6:48pm USER: what kind of sunset tonight in willimantic connecticut Reference: what time is sunset tonight in willimantic connecticut	what time is sunset tonight in willimantic connecticut	what time is sunset tonight in willimantic connecticut	what is sunset tonight in willimantic connecticut
USER: play little yancy AGENT: Lil' Fancy from Apple Music.y USER: play little yankees praise part Reference: play little yancy praise party	play little yancy praise party	play little yancy praise party	play little yankees praise party music
USER: play in jesus name by katie nicole Reference: play in jesus name by katy nichole	play in jesus name by katy nichole	play in jesus name by kayla nicole	play in jesus name by katy nichole

Table 5: Rewrite examples from offline experiments. In the dialog session, the last turn from the user is the current request which is needed to be rewritten by the model. "CGF w/o CD" denotes the model CGF without constrained decoding, "CGF w/o CE" denotes the CGF without context-enhanced encoding.

models. However, without considering the context information, the model sometimes fails. The third case shows that without constrained decoding, the CGF has a factual inconsistency generation ("kayla nicole" is an artist but never sung "in jesus name"). This is a common situation for generation-based models, especially on unseen data samples. Conversely, this situation rarely happens with constrained decoding, as the generation is based on the predefined constrained decoding space and we will never have such factual inconsistency generation.