Joint Contextual Query Rewriting for Virtual Assistant

Anonymous ACL submission

Abstract

Contextual queries are common in multi-turn spoken dialogues with virtual assistants. For example, a user could omit an aforementioned entity using anaphora or nominal ellipsis (entity carry over), or correct a previous recognition error by repeating mistaken words or phrases (correction by repetition). While prior work have touched on these use cases individually, we present a joint query rewriting approach to tackle both. This phonetically aware pointer network model rewrites conversational queries in both use cases into a single context independent query. We compare our joint model with two cascading single task models chained together on a randomly sampled and anonymized virtual assistant dataset. In our experiments, the joint model not only outperforms cascading models by 2.3 points token F1 and 3.6 points exact match accuracy, but also does so while being 1.6 times faster regarding p95 latency.

1 Introduction

Intelligent virtual assistants have been broadly adopted in daily life. To facilitate natural, multi-turn interactions with users, contextual understanding plays a crucial role in modern virtual assistants. Natural conversations are short and context dependent. For example, a user may refer to previously mentioned entities through anaphora or nominal ellipsis. We call this use case **Entity Carry over**, shown in Table 1. When a user issues the query *Where was **Stephen Sondheim** from*, they might expect that the virtual assistant remembers the context, and refer to ***Stephen Sondheim*** with a pronoun in the next query. There are many existing works that tackle similar use cases. (Naik et al., 2018) follows a slot filling paradigm. Entity candidates are preserved in a pool of slot values, and they resolve context dependent queries by carrying-over slots from previous context. (Yang et al., 2019) formulated Chinese nominal ellipsis into zero pronoun (Kong and Ng, 2013), combining both pronoun and nominal ellipsis resolution into anaphora resolution. More recently, **Query Rewriting** (QR) approaches became increasingly popular. Unlike slot filling and anaphora resolution, which requires architectural changes in downstream systems to adopt them, query rewriting turns contextual queries into its context independent counterpart, which can be directly executed by existing QA systems, like in Table 1. This allows the downstream systems to stay stateless, making QR more seamless to integrate. (Quan et al., 2019; Rastogi et al., 2019; Su et al., 2019) formulated query rewriting as a summarization task based on variants of Pointer-Generator Networks (See et al., 2017). (Yu et al., 2020) proposed a few shot learning approach based on GPT-2 (Radford et al., 2019), and (Tseng et al., 2021) showed that joint learning between query rewriting and co-reference resolution benefits both tasks due to their complementary nature.

Entity Carry Over	
Turn 1	Where was ***Stephen Sondheim*** from
Turn 2	Which college did ***he*** go to
Rewrite	Which college did ***Stephen Sondheim*** go to
Correction by Repetition	
Turn 1	Where is ***my sugar*** from
Turn 2	I said ***Meshuggah***
Rewrite	Where is ***Meshuggah*** from

Table 1: Example dialogues for Entity Carry Over and Correction by Repetition. Examples shown are author-created examples based on anonymized and randomly sampled virtual assistant logs. Utterances labeled as "Rewrite" are context independent counterparts for Turn 2 utterances.

Another interesting use case is **Correction by Repetition**. A user may find an error in the recognized text, and attempt to correct the error by repeating the request either fully or partially in a followup. As shown in Table 1, the virtual assistant made a mistake of recognizing ***Meshuggah*** as ***my sugar***. The user then repeated the intended phrase in the second turn. (Litman et al., 2006; Kitaoka et al., 2005; Lopes et al., 2015) proposed various detection methods for correction by repetition. (Nguyen et al., 2021) proposed a Query Rewriting solution based on Pointer Network (Vinyals et al., 2015) that rewrites correction by repetition dialogues into context independent queries.

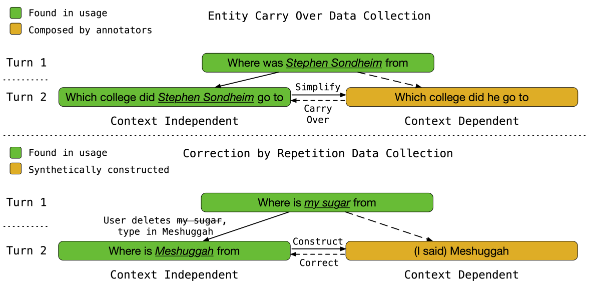
In this work, we propose a Joint Contextual Query Rewrite (JCQR) framework which handles both entity carry over and correction by repetition, and is efficient enough to run on edge devices. We designed a Pointer Network (Vinyals et al., 2015) based model with Phonetic Similarity Attention (PSA), which takes in a concatenated dialog history, and produces a sequence of indices, indicating which of the tokens in the dialog history should be copied over to form the rewritten utterance. The model was evaluated on a large-scale dataset that consists of 350k anonymized QA dialogs randomly sampled from virtual assistant query logs. Our experiments show that our joint model substantially outperforms chained single task models by 2.3 points token F1 and 3.6 points exact match accuracy, which indicates these two tasks can benefit from each other due to their similarities in problem definition and output space. The joint model is also 1.6 times faster as to 95th percentile latency, making it more suitable for edge device deployment.

2 Data

Collecting data for conversational use cases that are not supported in the current virtual assistant is a challenging problem: Once a user has tried an unsupported use case and fails, s/he is not likely to use similar queries again. Instead of mining directly for entity carry over and correction by repetition, we took the following approach, illustrated in Fig 1:

- Anonymize data
- Look for existing **improvement opportunities** where user experience can be more convenient if entity carry over and correction by repetition are available
- Annotate / synthetically generate conversational queries given **improvement opportunities** to create desired data points

For entity carry over, the **improvement opportunities** are defined as user queries that have different intents on the same entity in two consecutive turns. The intuition behind this is that a user will be able to omit the redundant entity using anaphora or nom-



🟩 Found in usage
🟨 Composed by annotators
Entity Carry Over Data Collection
Turn 1
Where was *Stephen Sondheim* from
Turn 2
Which college did *Stephen Sondheim* go to
Simplify
Carry Over
Which college did he go to
Context Independent
Context Dependent
🟩 Found in usage
🟨 Synthetically constructed
Correction by Repetition Data Collection
Turn 1
Where is *my sugar* from
User deletes ~~my sugar~~, type in Meshuggah
Turn 2
Where is *Meshuggah* from
Construct
Correct
(I said) Meshuggah
Context Independent
Context Dependent

Figure 1: Illustration of data collection process. Examples shown are author-created examples based on anonymized and randomly sampled virtual assistant logs. In both examples, utterances in green are improvement opportunities found in real-world usage, utterances in yellow are either annotated or synthetically generated, representing our best guess of what the user could do if conversational queries are well-supported. During data collection phase, we follow the solid lines. Given context independent improvement opportunities, conversational queries in yellow are generated. During model training, we follow the dotted lines. The model is provided with context dependent queries, as well as previous context utterances, and is asked to produce context independent queries in the final green box.

inal ellipsis with an oracle virtual assistant that can handle such contextual requests. After these improvement opportunities were collected, we asked annotators to simplify these queries as if they had access to an oracle virtual assistant. An example is shown at the top in Figure 1. Two consecutive turns with an overlapping entity ***Stephen Sondheim*** are found during the data collection phase. Annotators then simplified Turn 2 with a pronoun into *Which college did he go to*. During the model training phase, given Turn 1 context and aforementioned annotation, we trained the model to generate context independent Turn 2 on the left. This dataset contains 150k 2-turn dialogues, with 80%, 10%, and 10% training, validation and test split.

For correction by repetition, **improvement opportunities** are defined as the scenario where the user tapped on the transcribed prompt to edit Turn 1 query into something else in a followup turn. We also ensure that the resulting utterance after edit is one of the ASR hypotheses of Turn 1 so that the issue in Turn 1 is likely to be an ASR transcription error. The intuition is that the virtual assistant should be able to handle users' edit action through voice query in a follow up turn when there was an error with Turn 1 ASR transcription. In addition, we randomly attach common prefixes, such as *I said*, to the context dependent Turn 2. An example is shown at the bottom in Figure 1. During the data

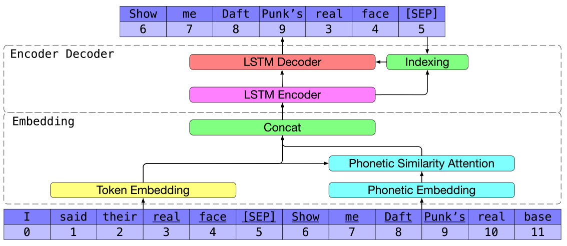


Figure 2: JCQR Architecture. The example shown is an author-created example based on anonymized and randomly sampled virtual assistant logs.

collection phase, after we found that user corrected *my sugar* into *Meshuggah*, we attach a prefix *I said* before *Meshuggah* to form a synthetic context dependent Turn 2 query. During the model training phase, similar to entity carry over, we feed context dependent dialog into the model to generate context independent Turn 2. This dataset contains 200k 2-turn dialogues, with 80%, 10%, and 10% training, validation and test split.

3 Model

To meet our latency requirements on edge devices, we chose LSTM-based Pointer Networks (Vinyals et al., 2015) as our backbone instead of a large pretrained LM due to its lightweight architecture and much reduced search space, as described below. As illustrated in Figure 2, the model generates a sequence of indices pointing to the input token sequence, indicating which token should be copied over in order to form the output. This design choice is backed by the assumption that, all tokens in the rewritten utterance can be found as part of the original dialogue. It's worth noting that while this assumption always holds for our correction by repetition data, it can also cause minor grammatical issues like missing prepositions and incorrect inflection for entity carry over. Fortunately, QA systems are often robust against these kinds of issues, so it is an acceptable tradeoff.

Embedding Two turn queries are first concatenated in reverse chronological order separated by a special token. We used a static word embedding to generate token embeddings $t_i \in \mathbb{R}^D$. Contextualized embedding methods will also apply here should latency allow. We then used Acoustic Neighboring Embedding (Jeon, 2022) to generate phonetic embeddings $p_i \in \mathbb{R}^D$. Acoustic Neighboring Embedding is a biLSTM that takes graphemes as input, and ensures that similar sounding graphemes are close to each other regarding their euclidean distance. Note that any other phonetic similarity preserving embedding method can also be applied here.

Phonetic embeddings are introduced to support correction by repetition, where phonetic similarities between Turn 1 and Turn 2 tokens is a strong signal for this use case. To better utilize these embeddings, we propose Phonetic Similarity Attention (PSA): First, we calculate a pairwise squared euclidean distance matrix D between tokens in Turn 1 and Turn 2. Assume the index of special token "[SEP]" is m, we have

$$d_{ij} = \begin{cases} \|p_i - p_j\|_2^2, & \text{if } (i-m)(j-m) < 0 \\ \infty, & \text{otherwise.} \end{cases} \tag{1}$$

Next, we calculate a phonetic similarity aware attention distribution based on this matrix:

$$psa_{ij} = \frac{\exp\left(-d_{ij}/(2\sigma^2)\right)}{\sum_j \exp\left(-d_{ij}/(2\sigma^2)\right)}, \tag{2}$$

σ is a hyperparameter determining the skewness of this attention distribution. The intuition behind this definition is that, $psa_{ij} \geq psa_{ik}, \forall k$, if token j in turn 2 sounds the most similar to token i in turn 1. With this attention, we calculate a weighted average over all token embeddings in the conversation for the token i, and concatenate it with t_i itself to obtain the final embeddings e_i

$$e_i = (t_i, \sum_j psa_{ij} t_j). \tag{3}$$

Encoder Decoder Similar to pointer networks (Vinyals et al., 2015), embeddings are passed into an encoder LSTM to construct a contextualized encoding, while a decoder LSTM contains a concat attention over encoder encodings, and produces an output distribution across input tokens at each time step. One key difference as compared with standard LSTM decoder lies in the input of the decoder at each time step. Instead of feeding in a previously predicted token, we use the previously predicted index to retrieve the encoder encodings as input for the decoder, adding another connection between the encoder and decoder. In our experiments, this can make a single-layer LSTM encoder decoder's performance comparable to a two-layer LSTM. This allowed us to drive latency down further without accuracy penalties. After a sequence of indices is predicted, we use this sequence to index the input tokens, which gives us the final rewrite output.

Method	ECo		CbR		p95 Latency
	F1	EM	F1	EM	
Cascading					
`E → C`	90.50 ± 0.06	87.02 ± 0.08	91.58 ± 0.05	74.70 ± 0.17	1
`C → E`	90.53 ± 0.07	87.03 ± 0.10	91.68 ± 0.04	75.06 ± 0.13	1
Joint					
`Naive`	90.62 ± 0.07	87.16 ± 0.10	92.33 ± 0.05	76.55 ± 0.13	0.61
`JCQR`	90.65 ± 0.12	87.16 ± 0.15	$\mathbf{92.90 \pm 0.11}$	$\mathbf{78.30 \pm 0.40}$	0.61

Table 2: Experiment results for JCQR. We report 95% confidence interval for token F1 and exact match accuracy generated from 32 independent trials. Statistically significant improvements are marked in bold. 95th percentile latency was measured on the combined test set, treating the latency of the cascading baseline models as 1 unit time

4 Experiments

4.1 Metrics

We compute a bag of words token level F1 metric on the subset of tokens that are present in the target rewrite, but not in the corresponding context dependent query. This metric reflects the model's ability to carry over tokens from previous context. We also calculate an exact string match accuracy (EM) between the model prediction and the target rewrite as a more strict comparison. Metrics for entity carry over (ECo) and correction by repetition (CbR) are measured separately. We also measured the 95 percentile latency for the joint models, using the latency of the cascading baseline models as 1 unit time.

4.2 Baselines

We compare our JCQR model with the following baselines. All encoders and decoders involved are configured as single layer LSTMs with 128 hidden size (bi-LSTM for encoders).

Cascading Single Task Models are two single task models daisy-chained together in a cascading fashion. Each model uses the same model architecture as JCQR, with an additional classifier identifying if an input dialogue falls under its corresponding use case. We consider two configurations: entity carry over fisrt `E` $\rightarrow$ `C`, and correction by repetition first `C` $\rightarrow$ `E`.

Naive Joint Model trains both tasks in one model. Instead of having a Phonetic Similarity Attention like JCQR, this model simply concatenates token embeddings t_i and phonetic embeddings p_i to obtain the final embeddings $\hat{e}_i$

4.3 Results

Our experimental results are shown in Table 2. Comparing the cascading single task models with the joint models on entity carry over task, while joint models do have slightly better mean token F1 score and exact match accuracy, the difference is not statistically significant. The advantage starts to show in correction by repetition, where we see as much as 2.3 points F1 score improvement and 3.6 points exact match accuracy gain. This improvement is mainly attributed to the output space similarity between these two tasks. While entity carry over and correction by repetition each have their own unique use case patterns, their target rewritten utterances live in a common pool of context independent queries. Multitask learning allows the decoder to be sufficiently trained, which is especially helpful for correction by repetition since the number of tokens that need to be copied from previous contexts are much greater in CbR than ECo. Joint models are also 1.6 times faster as to 95th percentile latency, which is expected considering the cascading nature of baselines.

Comparing between two joint model variants, JCQR shows 0.6 points F1 improvement and 1.8 points exact match improvement comparing to naive model, thanks to Phonetic Similarity Attention. PSA emphasizes on similar sounding tokens between turns, which is crucial to correction by repetition. Considering it only improves consumption of phonetic information, we don't expect a difference in entity carry over metrics.

5 Conclusion

We proposed a generalizable joint contextual query rewriting framework using Phonetic Similarity Attention (PSA) for entity carry over and correction by repetition. The joint model substantially outperforms cascading single task models, while being more efficient and edge device friendly. We showcased the accuracy and latency benefits of joint learning for query rewrite, and we hope this framework could benefit future related research.

References

Woojay Jeon. 2022. Acoustic neighbor embeddings.

Norihide Kitaoka, Naoko Kakutani, and Seiichi Nakagawa. 2005. Detection and recognition of correction utterances on misrecognition of spoken dialog system. *Syst. Comput. Japan*, 36(11):24–33.

Fang Kong and Hwee Tou Ng. 2013. Exploiting zero pronouns to improve chinese coreference resolution. In *EMNLP*, pages 278–288.

Diane Litman, Marc Swerts, and Julia Hirschberg. 2006. Characterizing and predicting corrections in spoken dialogue systems. *Computational Linguistics*, 32(3):417–438.

José Lopes, Giampiero Salvi, Gabriel Skantze, Alberto Abad, Joakim Gustafson, Fernando Batista, Raveesh Meena, and Isabel Trancoso. 2015. Detecting repetitions in spoken dialogue systems using phonetic distances. In *Sixteenth Annual Conference of the International Speech Communication Association*.

Chetan Naik, Arpit Gupta, Hancheng Ge, Lambert Mathias, and Ruhi Sarikaya. 2018. Contextual slot carryover for disparate schemas. *CoRR*, abs/1806.01773.

Hoang Long Nguyen, Vincent Renkens, Joris Pelemans, Srividya Pranavi Potharaju, Anil Kumar Nalamalapu, and Murat Akbacak. 2021. User-Initiated Repetition-Based Recovery in Multi-Utterance Dialogue Systems. In *Proc. Interspeech 2021*, pages 226–230.

Jun Quan, Deyi Xiong, Bonnie Webber, and Changjian Hu. 2019. Gecor: An end-to-end generative ellipsis and co-reference resolution model for task-oriented dialogue. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4539–4549.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Pushpendre Rastogi, Arpit Gupta, Tongfei Chen, and Mathias Lambert. 2019. Scaling multi-domain dialogue state tracking via query reformulation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, pages 97–105, Minneapolis, Minnesota. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Hui Su, Xiaoyu Shen, Rongzhi Zhang, Fei Sun, Pengwei Hu, Cheng Niu, and Jie Zhou. 2019. Improving multi-turn dialogue modelling with utterance ReWriter. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 22–31, Florence, Italy. Association for Computational Linguistics.

Bo-Hsiang Tseng, Shruti Bhargava, Jiarui Lu, Joel Ruben Antony Moniz, Dhivya Piraviperumal, Lin Li, and Hong Yu. 2021. CREAD: combined resolution of ellipses and anaphora in dialogues. *CoRR*, abs/2105.09914.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Wei Yang, Rui Qiao, Haocheng Qin, Amy Sun, Luchen Tan, Kun Xiong, and Ming Li. 2019. End-to-end neural context reconstruction in Chinese dialogue. In *Proceedings of the First Workshop on NLP for Conversational AI*, pages 68–76, Florence, Italy. Association for Computational Linguistics.

Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul N. Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020. Few-shot generative conversational query rewriting. *CoRR*, abs/2006.05009.