Contextual Rephrase Detection for Reducing Friction in Dialogue Systems

Zhuoyi Wang[1*†] **Saurabh Gupta**[2*] **Jie Hao**[2*]
Xing Fan[2] **Dingcheng Li**[2] **Alexander Hanbo Li**[3] **Chenlei Guo**[2]
[1]University of Texas at Dallas
[2]Amazon Alexa AI
[3]Amazon AWS AI
zxw151030@utdallas.edu
{gsaur,jieha,fanxing,lidingch,hanboli,guochenl}@amazon.com

Abstract

For voice assistants like Alexa, Google Assistant and Siri, correctly interpreting users' intentions is of utmost importance. However, users sometimes experience friction with these assistants, caused by errors from different system components or user errors such as slips of the tongue. Users tend to rephrase their query until they get a satisfactory response. Rephrase detection is used to identify the rephrases and has long been treated as a task with pairwise input, which does not fully utilize the contextual information (e.g. users' implicit feedback). To this end, we propose a contextual rephrase detection model **ContReph** to automatically identify rephrases from multi-turn dialogues. We showcase how to leverage the dialogue context and user-agent interaction signals, including user's implicit feedback and the time gap between different turns, which can help significantly outperform the pairwise rephrase detection models.

1 Introduction

Large-scale conversational AI based dialogue systems like Alexa, Siri, and Google Assistant, are getting more and more prevalent in real-world applications to help users across the globe. Natural Language Understanding (NLU) technology is an established component that produces semantic interpretations of a user request. Improving the accuracy of the NLU component is a key consideration for satisfactory end-to-end user experience, especially when the NLU component misinterprets the semantics due to ambiguity or errors that come from the previous component (e.g., Automatic Speech Recognition). For instance, the ASR system may incorrectly recognize *"play jacking the ball"* as *" play jack in the fall"*. These errors accumulate and introduce friction in the dialogue conversation. Fixing these frictions would help users to have a better experience, and engage more with the AI agents.

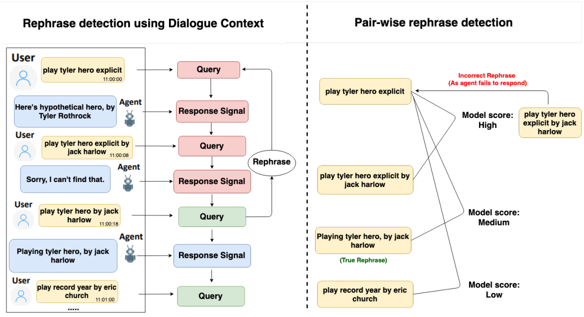

Rephrase detection using Dialogue Context
Pair-wise rephrase detection
User
play tyler hero explicit
11:00:00
Here's hypothetical hero, by Tyler Rothrock
Agent
User
play tyler hero explicit by jack harlow
11:00:08
Agent
Sorry, I can't find that.
User
play tyler hero by jack harlow
11:00:18
Agent
Playing tyler hero, by jack harlow
User
play record year by eric church
11:01:00
.....
Query
Response Signal
Query
Response Signal
Query
Response Signal
Query
Rephrase
play tyler hero explicit
Incorrect Rephrase
(As agent fails to respond)
Model score: High
play tyler hero explicit by jack harlow
play tyler hero explicit by jack harlow
Model score: Medium
Playing tyler hero, by jack harlow
(True Rephrase)
play record year by eric church
Model score: Low

Figure 1: Difference between the contextual rephrase detection and pairwise approach. Pair-wise rephrase detection model computes the similarity score for each pair that appears in the multi-turn dialogue, and selects the maximum score among them for the rephrase prediction. In this case, the pair-wise model without considering context information incorrectly predicts "*play tyler hero explicit by jack harlow*" as the rephrase of user's defective request "*play tyler hero explicit*" since it has highest similarity score.

Previous works (Yuan et al., 2021; Chen et al., 2020; Park et al., 2020) focus on friction reduction in the ASR and NLU components using Query Rewriting (QR) (Grbovic et al., 2015). These approaches reformulate the ASR transcription of user's query, such that it conveys the same meaning/intent, to minimize user dissatisfaction. An important aspect of the QR approaches is to detect user rephrase of a previous query that leads to a satisfactory response. However, these approaches focus only on the pairwise semantic similarity of queries, which does not consider the corresponding user feedback, with proper dialogue context. As shown in Fig. 1, dissatisfied users might provide *implicit feedback*, i.e., they rephrase the previous query (e.g. the first user request *"play tyler hero explicit"* in the left of Fig. 1) multiple times unless the agent does the needful. If we only consider the semantic similarity between different queries from

[*] Equal contribution.

[†] Work done when Zhuoyi Wang was interning at Amazon Alexa AI.

pairwise-based models, the system may choose an unsuitable query *"play tyler hero explicit by jack harlow"* to correct the problematic request. The dialogue context includes additional information like previous turns, the responses of the dialogue agent, and time differences between user queries. By leveraging this context, we can detect the correct rephrase - *"play tyler hero by jack harlow"*, with a much higher probability.

In this paper, we propose an automatic user rephrase detection approach **ContReph**, which leverages implicit user feedback and dialogue context in a multi-turn dialogue setting. ContReph detects if any of the user queries in the dialogue session is rephrased, and then extracts the most probable rephrase span that led to a satisfactory response from the dialogue agent. Specifically, we input the full dialogue session to the model, including agent's responses and capture time gaps between user queries using a novel time-difference encoding scheme. We evaluate the performance of our proposed framework by conducting an extensive set of experiments on production data of a large scale dialogue agent and showcase the effectiveness of our approach against existing methods. Although, in this work, we focus on rephrase detection only, the rephrases identified by our approach can directly be used as query rewrites for reducing friction in dialogue systems.

2 Related Work

2.1 Query Rewriting in Dialogue Systems

Query Rewriting (QR) in dialogue systems aims to correct the ASR interpretation of user's queries to deal with errors across the entire dialogue system pipeline in a single generalized framework. Existing QR approaches tend to apply neural embedding and retrieval based approaches (Yuan et al., 2021; Chen et al., 2020), generation-based approaches (He et al., 2016; Dehghani et al., 2017) and Absorbing Markov Chain (AMC) (Ponnusamy et al., 2020). Chen et al. (2020) apply the language model to pre-train query embeddings on historical user conversation data, Yuan et al. (2021) leverage Graph Neural Networks (Kipf and Welling, 2017) for the same, and then fine-tune on QR training set that consists of (source query, rephrase) pairs. To generate such training set without human annotations, they rely on pairwise rephrase detection models to identify rephrase pairs in historic dialogue sessions. Ponnusamy et al. (2020) propose AMC to identify rephrases within multi-turn dialogues and treat the rephrases directly as rewrites instead of training a neural model. However, the approach is purely statistical and ignores the semantic relevance between the source query and rephrase, which has been proven effective across different datasets and tasks (Conneau and Kiela, 2018; Gao et al., 2021). Our work alleviates this problem by using the BERT model incorporated with dialogue context information.

2.2 Rephrase Detection

Given a pair of sentences P and Q, existing rephrase/paraphrase detection approaches estimate the probability distribution $Pr(y|P, Q)$, where $y = 1$ if P and Q are rephrases, and $y = 0$ otherwise. Typically, these approaches use encoders to embed P and Q, followed by semantic or syntactic similarity measurement. For example, BiMPM (Kim et al., 2019) uses BiLSTM layers for encoding the sentences, and performs a bilateral matching to compute $Pr(y|P, Q)$. Gao et al. (2021) propose SimCSE, which leverages the contrastive learning framework and is shown to produce superior sentence embeddings, from either unlabeled or labeled data. However, the existing approaches are limited by the information they can exploit, especially for dialogue sessions, where a lot of contextual information is available. Hence, we extend these approaches from pairwise to dialogue context level, as described in the next section.

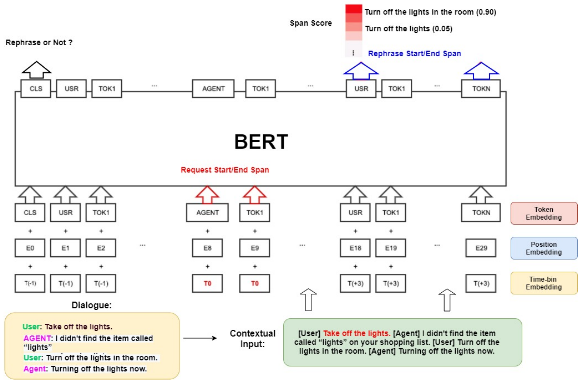
3 Method

3.1 Notations and Problem Definition

We consider a dataset D of M multi-turn dialogue sessions, such that $D = \{S_i\}_{i=1}^{M}$, and every session S is an ordered set of N turns: $S = \{(Q_i, R_i)\}_{i=1}^{N}$. Here i indicates the index of turn, and each turn i consists of a pair (Q_i, R_i), where Q_i is the user's query and R_i is the agent's response to query Q_i. Any two successive turns have a time gap of less than a minute. Given a dialogue session S and a **source turn**, i.e., input pair of query and response (Q_i, R_i), the goal of our model is to predict whether Q_i is rephrased in any of the following turns $(Q_j, R_j)| \, i < j \leq N$. If so, the model should predict the span of Q_j and return null otherwise.

3.2 Model Architecture

Fig. 2 shows the architecture of our model - ContReph. We adopt BERT (Devlin et al., 2019) for



User: Take off the lights.
AGENT: I didn't find the item called "lights"
User: Turn off the lights in the room.
Agent: Turning off the lights now.

[User] Take off the lights. [Agent] I didn't find the item called "lights" on your shopping list. [User] Turn off the lights in the room. [Agent] Turning off the lights now.

Figure 2: A brief description of our model, ContReph, which is fine-tuned on a pre-trained model for predicting the start and end positions of the rephrase part in the contextual input. The bottom of the figure shows the construction of contextual text into model.

encoding dialogue sessions. We flatten the dialogue session into one sequence and feed it to a pre-trained BERT, to compute the dialogue session embedding. We introduce two special tokens: '[USER]' and '[AGENT]', which are used to prefix the user query and agent response in the contextual input, respectively. We cast rephrase detection as a span prediction problem where we predict the probability of start and end span locations on each token's position, using the embedding output of the final BERT layer. We introduce a start vector W_S and an end vector W_E. Assuming the final hidden vector for the i^{th} input token as $T_i \in \mathbb{R}^H$, the probability of token i being the start/end of the rephrase span is computed after applying a softmax on the dot product between T_i and W_S (or W_E) over all of tokens:

$$P_i^S = \frac{\exp^{T_i \cdot W_S}}{\sum_m \exp^{T_m \cdot W_S}}; P_i^E = \frac{\exp^{T_j \cdot W_E}}{\sum_m \exp^{T_m \cdot W_E}}$$

The score of a candidate span from position i to position j is defined as: $s_{ij} = W_S \cdot T_i + W_E \cdot T_j$, where $i < j$. We use $s_{none} = W_S \cdot T_{CLS} + W_E \cdot T_{CLS}$ to represent the score of no-rephrase span. We set threshold τ to decide whether to predict no-rephrase or not. If $max_{j>i}\ s_{ij} > s_{none} + \tau$, then we regard the maximum score span as the rephrase span and null otherwise.

Time difference encoding: In addition to capturing the full dialogue context when making a rephrase prediction, ContReph also considers the time difference between multiple turns. This is an important factor as users are more likely to interrupt the agent and rephrase their query sooner than later, if they don't get the right response. We capture the time differences using **time-bin token embeddings**. Consider a source turn (including a request and a response) $t_{src} = (Q_{src}, R_{src})$, for which we want to detect a rephrase in the session. We refer to its timestamp as ω_{src}. We calculate the time difference $\Delta_i = \omega_i - \omega_{src}$, where ω_i is the timestamp of a turn t_i, for all the turns in the session. $\Delta_i\ \forall i \in [1, n]$ are then mapped to their respective time-bin tokens. These time-bin tokens represent equal sized intervals in Δ's range of [-60, 60] seconds. We then map these tokens to their embeddings. As shown in Fig. 2, the corresponding time-bin token embeddings are added to each token of the turn at the input layer of the model, depending on the turn's bin.

Has-Rephrase	No-Rephrase
(Source turn) [USER] Open the blinds [AGENT] I didn't find a device named blinds	[USER] How far is twin lakes iowa from here ? [AGENT] Sorry, I couldn't find what you're looking for.
[USER] Open the blind [AGENT] I can't find blind.	[USER] How far is fort dodge iowa from here ? [AGENT] Fort Dodge is 351.8 mi away by car
[USER] Open the right blind. (Rephrase) [AGENT] Done.	[USER] How far is storm lake iowa from here ? [AGENT] Storm Lake is 413.1 mi away by car.

Table 1: Examples of dialogue sessions.

4 Experiments

4.1 Data

Machine-Annotated set: We sample multi-turn dialogue sessions between users and a large scale conversational AI agent from anonymous historic interactions. We use an existing model based on Absorbing Markov Chain (AMC) (Ponnusamy et al., 2020) to discover rephrase turns in these sessions, and only keep the instances where the AMC model is highly confident in predicting a rephrase (if the session has one) and a no-rephrase. Based on this, we divide the dataset into two types: **Has-Rephrase** and **No-Rephrase**, respectively. We split this dataset into train, validation and test sets, with the statistics shown in Table 2. Since this dataset is labeled using a model, we refer to it as **Machine-Annotated set**. We use the training split to fine-tune our model ContReph and other baselines (Section 4.3). An example of this dataset with labels is shown in Table 1.

Human-Annotated set: For a more comprehensive evaluation of ContReph with other baselines, we construct another test set where the rephrases are identified by human annotators. We sample historic sessions and keep only those sessions where AMC model predicted a no-rephrase, but human annotators labeled rephrases with high confidence. We refer to this one as **Human test set**. This is a

Machine-Annotated set						**Human set**
Train		Validation		Test		Test
Has-Rephrase	No-Rephrase	Has-Rephrase	No-Rephrase	Has-Rephrase	No-Rephrase	Has-Rephrase
317931	282069	50897	49103	407544	394051	5463

Table 2: No. of instances (dialogue sessions) in different datasets.

Approach	**Machine-Annotated test set**						**Human test set**	
	Has-Rephrase(%)		All(%)		No-Rephrase(%)		Has-Rephrase (%)	
	EM	TR	EM	TR	EM	TR	EM	TR
Existing Work (Pairwise-based approach)								
SimCSE-unsup	[10,15]	[15,20]	[50,55]	[10,15]	[90,95]	[5,10]	[10,15]	[10,15]
SimCSE-sup	46.26	48.57	24.31	23.91	1.59	-1.6	28.92	30.08
BERT-NSP	67.09	73.80	37.1	34.54	6.08	-6.08	29.56	30.87
DPR	9.03	10.21	2.50	7.29	-4.26	4.26	18.28	19.23
Our work (Context-based approach)								
ContReph w/o time	76.16	77.93	41.50	36.14	**6.75**	**-6.77**	56.08	57.04
ContReph	**78.25**	**79.73**	**43.05**	**37.40**	6.36	-6.35	**57.45**	**58.26**

Table 3: Evaluation performance from different approaches on both Machine-Annotated and Human test set. All the numbers are absolute differences with respect to the baseline: “SimCSE-unsup”. For example, “46.26” denotes “SimCSE-unsup metric + 46.26”. “[10,15]” denotes the exact score falls into between 10 to 15. The “TR” on No-Rephrase dataset denotes the false trigger rate (lower is better).

more challenging test set as our AMC model failed to predict the rephrases in this set. Moreover, its domain distribution is different from training set as it is sampled from a different time period.

4.2 Evaluation Metrics

We use the following evaluation metrics:

Exact Match (EM): For a Has-Rephrase instance, this score is 1 if the predicted span exactly matches the labeled rephrase, and is 0 otherwise. For a No-Rephrase instance, the score is 1 if the model predicts a null span, and 0 otherwise.

Trigger Rate (TR): Trigger Rate is the fraction of instances on which the model makes a non-null prediction.

4.3 Baselines and Experimental Setup

To evaluate the rephrase detection performance, we compare our method with a few baselines which are pairwise-based. **BERT-NSP** (Devlin et al., 2019): we fine-tune BERT with the same training split, then predict if the input pairs are rephrases. **DPR** (Karpukhin et al., 2020): we follow a recent retrieval model DPR to train a dual BERT model with positive rephrase pairs and in-batch negatives. **SimCSE** (Gao et al., 2021): we fully use the training data to train both unsupervised (SimCSE-unsup) and supervised (SimCSE-sup) models.

For **ContReph**, we choose the official pre-trained BERT-base model[1] and fine-tune on it. Models are selected by early stopping on validation set. More implementation details and hyper-parameters can be found in Appendix.

4.4 Results

In Table 3[2], we show evaluation of our approach against other baselines. ContReph consistently achieves better performance on machine and human-annotation test sets. It is better than the state-of-the-art pairwise BERT-NSP method on human test set by 27.89% on EM score, and also improves overall EM score by almost 6% on machine-annotated set. This clearly shows the benefits of capturing dialogue context. Moreover, removing time difference encoding from ContReph leads to a drop of 1.55% and 1.37% in EM score on machine and human-annotation test sets, respectively. This proves that capturing time difference between turns can further improve rephrase detection. We notice that human test set is more challenging due to different domain distribution, and hence EM scores for it are much lower, compared to machine-annotated set. BERT-NSP achieves the best results amongst the baselines, which highlights the benefits of utilizing transformer’s self-attention mechanism across the queries: it encodes the two queries as a single sequence with a separator, while other baselines encode the queries independently with BERT and then apply a similarity function. Note that ContReph utilizes the self-attention mechanism across *all turns* of the dialogue.

[1] https://github.com/google-research/bert

[2] Due to business reasons, the rough range of “SimCSE-unsup” performance and absolute difference are indicated in the table. All numbers are statistically significant.

5 Conclusion and Future Work

In this paper, we presented a novel approach for detecting user rephrases in multi-turn dialogue systems. Users tend to rephrase their queries until they get the desired response from AI agents. Our system can detect these rephrases with a high accuracy using the dialogue context and significantly outperforms other approaches that consider queries in a pair-wise manner only. The output of our model is a crucial step towards building self-learning mechanisms in dialogue agents to fix issues with minimal human intervention.

For future work, we plan to leverage contrastive learning strategies as a post-training step, which could help us obtain better query representations, before we do fine-tuning for rephrase detection. We also want to deploy the detected rephrases as query rewrites to gauge how much we can improve the UX of a real world dialogue system.

References

Zheng Chen, Xing Fan, and Yuan Ling. 2020. Pre-training for query rewriting in a spoken language understanding system. In *ICASSP*.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An evaluation toolkit for universal sentence representations. In *European Language Resources Association (ELRA)*.

Mostafa Dehghani, Sascha Rothe, Enrique Alfonseca, and Pascal Fleury. 2017. Learning to attend, copy, and generate for session-based query suggestion. In *CIKM*.

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

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

Mihajlo Grbovic, Nemanja Djuric, Vladan Radosavljevic, Fabrizio Silvestri, and Narayan Bhamidipati. 2015. Context-and content-aware embeddings for query rewriting in sponsored search. In *SIGIR*.

Yunlong He, Jiliang Tang, Hua Ouyang, Changsung Kang, Dawei Yin, and Yi Chang. 2016. Learning to rewrite queries. In *CIKM*.

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*.

Seonhoon Kim, Inho Kang, and Nojun Kwak. 2019. Semantic sentence matching with densely-connected recurrent and co-attentive information. In *AAAI*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

Sunghyun Park, Han Li, Ameen Patel, Sidharth Mudgal, Sungjin Lee, Young-Bum Kim, Spyros Matsoukas, and Ruhi Sarikaya. 2020. A scalable framework for learning from implicit user feedback to improve natural language understanding in large-scale conversational ai systems. *arXiv preprint arXiv:2010.12251*.

Pragaash Ponnusamy, Alireza Roshan Ghias, Chenlei Guo, and Ruhi Sarikaya. 2020. Feedback-based self-learning in large-scale conversational ai agents. In *AAAI*.

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*.

A Appendix

A.1 Implementation Details

Baselines For the BERT-NSP baseline, similar with the BERT next sentence prediction task (Devlin et al., 2019), we fine-tune the BERT model with a binary classification objective. We break down all the Has-Rephrase and No-Rephrase sessions into (query, rephrase) pairs with a 0-1 label (whether the rephrase is true or not). For the DPR model, we follow the DPR (Karpukhin et al., 2020) training scheme which compares all pairs of questions and passages in a batch. We only use the positive rephrase pairs extracted from Has-Rephrase sessions, and use the cosine similarity scores for cross entropy. The most recent baseline is SimCSE (Gao et al., 2021), which is a simple contrastive learning framework but greatly advances the sentence embeddings. For the unsupervised setting (SimCSE-unsup), we extract all the queries from both Has-Rephrase and No-Rephrase sessions as the unlabelled training data, and follow Gao et al. (2021) to take an input sentence and predict itself with a contrastive objective, with only standard dropout used as noise. For the supervised setting (SimCSE-sup), we extract the positive rephrase pairs from Has-Rephrase sessions, and use the other queries from the same session as the hard negatives. Moreover, in order to fully use the training data and make a fair comparison, we also use the source query and itself as the positive pairs from No-Rephrase session, with only standard dropout used as noise. Other queries from the same session were used as hard negatives. For the other model configurations and related hyper-parameters, we are consistent with the original works (Devlin et al., 2019; Karpukhin et al., 2020; Gao et al., 2021). We set the threshold to 0.70 for BERT-NSP, 0.75 for DPR and 0.85 for the SimCSE models.

Our models We set a mini-batch size of 64 and use Adam optimizer for optimization during the fine-tuning for 10 epochs. We set an initial learning rate of $4 \cdot 10^{-5}$. We select the threshold τ for no-rephrase span prediction on the validation set, following the same approach as Devlin et al. (2019), and use this value on all the test sets. All the experiments are performed with Nvidia V100 GPU.

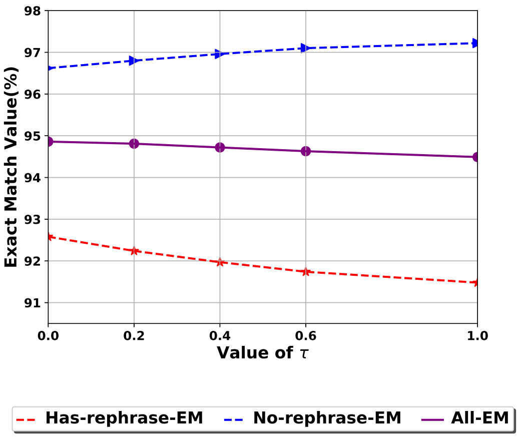


Figure 3: Validation set performance for Has-rephrase, No-rephrase and overall EM score with different τ values.

A.2 Analysis

Effect of τ This is the major parameter that we tune for determining whether the prediction should be null or not (as described in Sec. 3.2). We change τ in the range of 0.0 to 1.0, and evaluate the model performance on validation set. Fig. 3 shows the effect of changing τ on the Exact Match score for Has-Rephrase, No-Rephrase and all the dialogue sessions together. As we increase τ, the model predicts a null rephrase span more often and hence gets better performance on the No-Rephrase set, and vice-versa. To balance this trade-off, we choose the value of τ that maximizes the EM score on "All" validation set, i.e. looking at Has-Rephrase and No-Rephrase together. We also ensure that our data splits are balanced, i.e., contain almost equal fraction of Has-Rephrase and No-Rephrase cases (See Table 2).

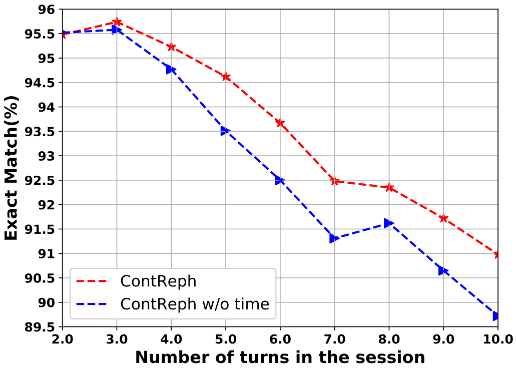


Figure 4: EM score on machine-annotated test set with varying number of turns in the dialogue.

Number of turns in Dialogue Sessions We show the performance variation with the number

Type	Dialog	Source query	Reference	Model Prediction
Correct	USER: Put on man song AGENT: This Old Man , by Cedarmont Kid. USER: Put on the but i man song. AGENT: Sorry, I didn't get that. USER: Put on a man song. AGENT: Here's a playlist you might like: Latin Hits USER: Put on Iron Man song. AGENT: Here's Iron Man , by Black Sabbath	USER: Put on man song.	USER: Put on Iron Man song.	USER: Put on Iron Man song.
False trigger	USER: Play twenty one by polo g. AGENT: 21 by polo g from Spotify. USER: Play me something. AGENT: Here's a station you might like: Platinum Pop. USER: Play baby something by polo g. AGENT: Be something by polo g from Spotify.	USER: Play twenty one by polo g	NULL	USER: Play baby something by polo g.
No trigger	USER: What time is it now ? AGENT: It's 8:01 PM. USER: Play simon time. AGENT: From Time (feat. Jhene Aiko) by Drake. USER: Play sign of the times. AGENT: Sign of the Times by Harry Styles.	USER: Play simon time.	USER: Play sign of the times.	NULL
Wrong match	USER: Alarm ring AGENT: Hmm, I don't know that. USER: Alarm rain. AGENT: ... USER: Arm ring AGENT: Ring is armed in stay mode, Side window was bypassed.	USER: Alarm ring.	USER: Arm ring.	USER: Alarm rain.
	USER: Play can't go on by Chicago AGENT: I can't find can't go on by Chicago USER: Play on my own by Chicago AGENT: Here's My Own Best Friend. USER: Play will you still love me by Chicago AGENT: Will You Still Love Me? , by Chicago	USER: Play can't go on by Chicago	USER: Play will you still love me by Chicago	USER: Play on my own by Chicago

Table 4: Case study for various types of correct and incorrect predictions from our ContReph model.

Dialog	Source query	Reference	ContReph w/o Time	ContReph
USER: Turn off appointment. AGENT: Which event would you like to delete? USER: Nothing AGENT: Ok, canceled. USER: Turn off table AGENT: Sorry, I didn't get that. USER: Turn off apartment AGENT: OK. USER: Turn off alarm AGENT: ...	Turn off appointment.	Turn off apartment	Turn off alarm	Turn off apartment
USER: Play indie kid music. AGENT: The Scotts by the scotts, Travis Scott. USER: Play indie music AGENT: Playing Indie music from Apple Music. USER: Play pride by kendrick lamar AGENT: PRIDE, by Kendrick Lamar.	Play indie kid music.	Play indie music	Play pride by kendrick lamar	Play indie music

Table 5: Case study for ContReph vs ContReph w/o time.

of turns in the dialogue sessions in Fig. 4. The number of turns has a significant effect on the EM score, even if we balance the length distribution of dialogues during training. This result shows that for the sessions with more turns, the context can be unrelated to the current request, and this unrelated context can impact the accuracy negatively. Interestingly, capturing the time difference between the turns helps here, especially for the longer sessions. With the temporal information, the model can automatically decide which context is irrelevant and can thus ignore it.

A.3 Case Study

We show four scenarios in Table 4, where the first one is a correct prediction and other three are failure cases: 1) False-trigger, where the model predicts that current query/request should be rephrased, but actually the dialogue does not contain a rephrase; 2) No-trigger, where the model judges the request need not be rephrased, but actually the dialogue has a rephrase; and finally, 3) wrong match, which means the model predicts a wrong span. False triggering usually happens if the user issues similar back-to-back queries with very small time gaps in between. Wrong match mostly happens if there are multiple successful rephrases in the session.

We also show comparison between the predictions of ContReph w/o Time and ContReph in Table 5. In the first scenario, the user says “turn off alarm” 20 seconds after “turn off apartment”. The model without time tends to pick the last successful query as rephrase, whereas ContReph is aware of the fact that “turn off alarm” happened long after “turn off apartment”, and hence picks the latter as the rephrase.

The second scenario is similar where the user listens to Kendrick Lamar, 45 seconds after listening to Indie Music. Hence, the request “Play pride by kendrick lamar” is not a rephrase, but just another song that the user listened to. ContReph, being aware of the temporal information, picked the right rephrase again, which is “Play indie music”.