CREAD: Combined Resolution of Ellipses and Anaphora in Dialogues

Bo-Hsiang Tseng[1]*, **Shruti Bhargava**[2], **Jiarui Lu**[2]
Joel Ruben Antony Moniz[2], **Dhivya Piraviperumal**[2], **Lin Li**[2], **Hong Yu**[2]
[1]Engineering Department, University of Cambridge, UK
[2]Apple
bht26@cam.ac.uk
{shruti_bhargava, jiarui_lu,
joelrubenantony_moniz, dhivyaprp, lli9, hong_yu}@apple.com

Abstract

Anaphora and ellipses are two common phenomena in dialogues. Without resolving referring expressions and information omission, dialogue systems may fail to generate consistent and coherent responses. Traditionally, anaphora is resolved by coreference resolution and ellipses by query rewrite. In this work, we propose a novel joint learning framework of modeling coreference resolution and query rewriting for complex, multi-turn dialogue understanding. Given an ongoing dialogue between a user and a dialogue assistant, for the user query, our joint learning model first predicts coreference links between the query and the dialogue context, and then generates a self-contained rewritten user query. To evaluate our model, we annotate a dialogue based coreference resolution dataset, MuDoCo, with rewritten queries. Results show that the performance of query rewrite can be substantially boosted (+2.3% F1) with the aid of coreference modeling. Furthermore, our joint model outperforms the state-of-the-art coreference resolution model (+2% F1) on this dataset.

1 Introduction

In recent years, dialogue systems have attracted growing interest, and been applied to various scenarios, ranging from chatbots to task-oriented dialogues to question answering. Despite rapid progress in dialogue systems, several difficulties remain in the understanding of complex, multi-turn dialogues. Two major problems are anaphora resolution (Clark and Manning, 2016a,b) and ellipsis (Kumar and Joshi, 2016) in follow-up turns. Take the dialogue in Figure 1 as an example: ellipsis happens in `user turn 2` where the user is asking for the capital of *"Costa Rica"* without explicitly mentioning the country again; coreference happens in `user turn 3` where *"the capital"* refers to *"San Jose"*. Without resolving the anaphoric reference and the ellipsis, dialogue systems may fail to generate coherent responses.

(Turn 1)
User: *Who is the president of Costa Rica?*
System: *Carlos Alvarado Quesada is the president of Costa Rica.*

(Turn 2)
User: *And what is the capital?*
[Rewrite: *And what is the capital of Costa Rica?]*
System: *San Jose is the capital of Costa Rica.*

(Turn 3)
User: *What is the population of the capital?*
[Rewrite: *What is the population of San Jose?]*
System (i): *The population of San Jose is 1.03 million.* (California)
System (ii): *The population of San Jose is 342 thousands.* (Costa Rica)

Figure 1: An example of a question-answering dialogue where coreference and ellipsis happen in user query, and the corresponding query rewrite annotation. References to the same entity are highlighted in the same color, and can be resolved by coreference resolution modeling. The two system responses in Turn 3 indicate two possible interpretations of the city *San Jose* by the system.

Query rewrite (Quan et al., 2019) is an approach that converts a context-dependent user query into a self-contained utterance so that it can be understood and executed independent of previous dialogue context. This technique can solve many cases where coreference or ellipsis happens. For instance, "*the capital*" in `user turn 3` is changed to *"San Jose"* in the rewrite. Furthermore, the ellipsis of the country name *"Costa Rica"* in `user turn 2` can be revealed through rewriting. The rewritten utterance improves multi-turn dialogue understanding (Yang et al., 2019) by reducing dependency on the previous turns.

Although query rewrite implicitly resolves coreference resolution, there is information not contained in a rewrite. First, it does not provide a distinct coreference link between mentions across dialogue turns as in the classic coreference resolution task. This is particularly disadvantageous

*Work done while the first author was an intern at Apple.

Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 3390–3406
June 6–11, 2021.

when there is entity ambiguity in the rewritten sentence. For example, in Figure 1, since *"San Jose"* in `Rewrite turn 3` can be either San Jose in Costa Rica or San Jose in California, it is likely that the system ends up with an incorrect response by generating System (i) instead of System (ii) due to the wrong interpretation of San Jose. Second, mention detection, an essential step in coreference resolution (Peng et al., 2015), is not involved in query rewrite. By knowing which span in an utterance is a mention, downstream systems like named entity recognition and intent understanding can perform better (Bikel et al., 2009). Third, if coreference links to dialogue context are available, downstream systems can skip entity linking, which is time-consuming and may introduce noise.

To resolve the above issues, we propose a novel joint learning framework that incorporates the benefits of reference resolution into the query rewrite task. To the best of our knowledge, there does not exist, at the time of writing, an English conversational dataset that couples annotations of both query rewrite and coreference resolution (as links or clusters). This motivates us to collect annotations for query rewrite on a recent dialogue dataset - MuDoCo (Martin et al., 2020), which already has coreference links between user query and dialogue context. Compared to existing query rewrite datasets (Quan et al., 2019; Anantha et al., 2020), rewriting in MuDoCo is much more challenging since it involves reasoning over multiple turns and spans multiple domains.

We design a joint learning model adopting the GPT-2 (Radford et al., 2019) architecture that learns both query rewrite and coreference resolution. Given an ongoing dialogue, our model first predicts the coreference links, if any, between the latest user query and the dialogue context. Then it generates the rewritten query by drawing upon the coreference results. Our experiments show that query rewrite performance can be substantially boosted with the aid of coreference training. In addition, our model outperforms strong baselines for the two individual tasks. Since both the tasks fundamentally solve reference resolution, the joint training facilitates knowledge sharing.

Our contributions can be summarized as follows:

- We present a novel joint learning framework of modeling coreference resolution and query rewrite for multi-turn dialogues.
- Our annotations of query rewrite augment the MuDoCo dataset with query rewrite labels. To the best of our knowledge, our augmented MuDoCo is the first English dialogue dataset with both coreference resolution and query rewrite annotations.
- We propose a novel GPT-2 based model to tackle the two target tasks, and show that joint training with coreference resolution helps in improving the quality of the query rewrites.

The augmented dataset with our annotations along with the modeling source code are available at `https://github.com/apple/ml-cread`.

2 Related Work

Query Rewrite The most relevant line of research is the adoption of query rewrite in dialogues to tackle anaphora or ellipses. Many prior works employ an LSTM-based seq-to-seq model, which takes the dialogue context and user query as input, and generates the rewritten query. Quan et al. (2019) use the pointer-generator model (See et al., 2017) to rewrite the user query on restaurant-domain task-oriented dialogues. By comparison, query rewrite in MuDoCo dataset is more challenging as it covers 6 domains and the rewriting patterns are more complex and diverse than in the CamRest676 dataset (Wen et al., 2017). Rastogi et al. (2019) introduce an auxiliary objective of copying entity tokens from the delexicalized utterances to augment the learning of pointer network. In Su et al. (2019), two separate attention distributions are learned for the dialogue context and the user query respectively with a control gate. This modified copy mechanism shows improvements over the standard pointer-generator on both LSTM-based models and transformer-based models (Vaswani et al., 2017). Note that in the dataset used in their work, the dialogue context has only 2 utterances; MuDoCo, in contrast, has up to 8 utterances, making it much more challenging for query rewrite.

Coreference Resolution Research on document-based coreference resolution has a long history (a detailed survey can be found in Ng (2010)). Various approaches have been proposed, ranging from learning mention-pair classifiers (Ng and Cardie, 2002; Bengtson and Roth, 2008), latent structured-based models (Fernandes et al., 2012; Björkelund and Kuhn, 2014; Martschat and Strube, 2015) to the more recent neural pipeline based systems

that rely on syntactic parsers (Raghunathan et al., 2010) and clustering algorithms (Clark and Manning, 2016a,b). The first neural end-to-end coreference resolution model was proposed in Lee et al. (2017) and achieved better results without external resources. An improved version was proposed in Lee et al. (2018), which considers higher-order structures by iteratively refining span representations. Recently, powerful pre-trained models have been used to extract representations for these end-to-end models using BERT (Joshi et al., 2019) or SpanBERT (Joshi et al., 2020). Wu et al. (2020) approach the problem in a question answering framework. For each detected mention candidate, the sentence it resides in serves as the query and is used to predict the referent in the passage. Different from these works, we focus on coreference resolution in dialogues with the following main distinctions: 1) the speaker information in dialogues is clear; 2) less descriptive content may cause the pronoun mention to be more ambiguous; and 3) coreference resolution is conducted only between the latest user query and the previous dialogue context — unlike in document-based coreference resolution where a model can look ahead for the resolution, future turns are not available to a dialogue agent. We encourage the reader to refer to Martin et al. (2020) for more details.

Joint Learning In contrast to prior works that focus solely on either query rewrite or coreference resolution, we present a novel joint learning approach to tackle both the tasks using one single model. We hope that this work serves as a first step towards this new, challenging and practical problem for dialogue understanding.

3 Dataset and Task

The MuDoCo dataset contains 7.5k task-oriented multi-turn dialogues across 6 domains. A dialogue has an average of 2.6 turns and a maximum of 5 (a turn includes a user query and a system response). Figure 2 shows an example. For each partial dialogue, the coreference links, if existing, are annotated between the latest user query and its dialogue context. For example, when we consider the partial dialogue preceding up to `user turn 2`, there is a coreference link between the anaphora "*this*" in `user turn 2` and the antecedent "*song*" in `user turn 1`. When an anaphora has multiple antecedents in the context, e.g., "*song*" in `user turn 1` and "*Yellow Submarine*" in `system turn 1`, only one of them is annotated as its referent in the coreference link.

(Turn 1)
User: *Play the Beatles song about submarines.*
System: *Yellow Submarine by the Beatles playing on YouTube Music.*

(Turn 2)
User: *What album is this from?*
[Rewrite: *What album is Yellow Submarine by the Beatles from?*]
System: *Yellow Submarine is from the album Yellow Submarine.*

(Turn 3)
User: *Can I hear the next song on the album?*
[Rewrite: *Can I hear the next song on the Yellow Submarine album?*]
System: *Sure, Only A Northern Song on the album now playing.*

Figure 2: An example from the MuDoCo dataset in the music domain with our query rewrite annotation. Word spans in the same color belong to the same mention cluster.

On top of the existing coreference labels, we annotate the rewrite for each utterance. The goal is to rewrite the query into a self-contained query independent of the dialogue context. 30 annotators are recruited for the data collection. Each of them is shown a partial dialogue, and is asked (1) to decide if the query needs to be rewritten due to coreference or ellipsis; and (2) to provide the rewritten query, when rewriting is required. We notice that there can be various ways of rewriting an utterance. For example, some annotators might include every detail of the rewritten entity, while others might choose a precise term; some might paraphrase the rewritten utterance, while others keep the same expression. To ensure data consistency and high annotation quality, we designed a comprehensive guideline for the annotators to follow and undertook a two-stage collection process: 1) we organized two training sessions with annotators. In each session, 50 representative examples were selected and assigned to each annotator. An author inspected these training results individually and provided feedback to the annotators. 2) 5% of the grading results were manually evaluated by an author for quality assurance. Detailed annotation guidelines can be found in the Appendix.

The joint learning task requires the machine to predict both coreference links and the rewritten query for the latest user query given an ongoing dialogue. The outputs of the two individual tasks complement each other and provide more comprehensive information for dialogue understanding. For instance, the "*Yellow Submarine*" in Figure 2 can be either a song name or an album name. Explicit coreference resolution helps to disambiguate

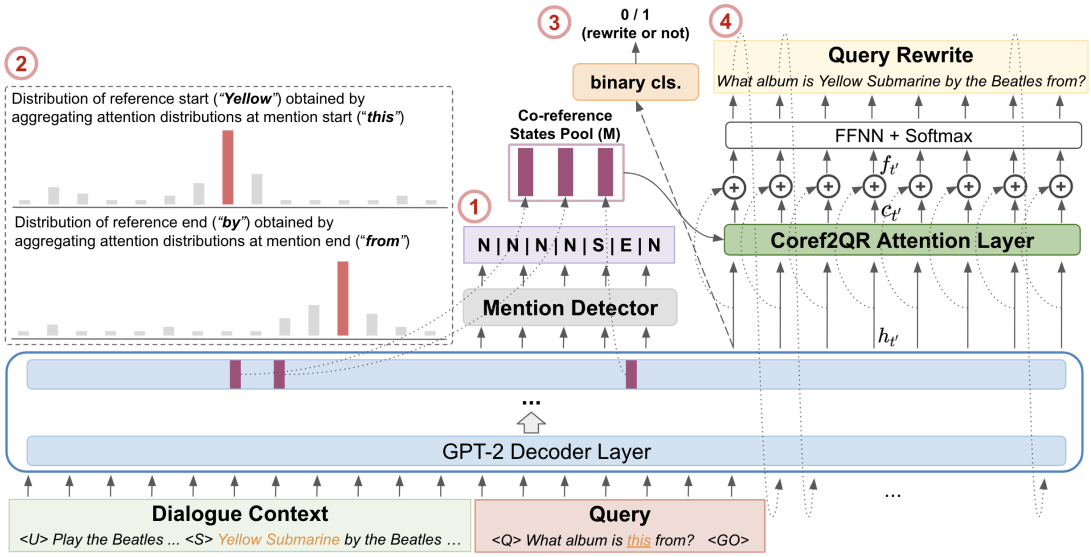

Figure 3: The proposed model for joint learning of coreference resolution and query rewrite, designed using the GPT-2 architecture. Given a dialogue context and a user query, the model first detects the mentions in the query (Step 1); resolves the corresponding reference spans (Step 2); predicts whether the query needs a rewrite or not (Step 3); and, if the model decides to rewrite, generates the rewritten query (Step 4). In this example dialogue, there is a coreference link existing between the mention "*this*" and its referent "*Yellow Submarine*".

between various possibilities by linking entities to previously resolved ones. More importantly, the supervision of coreference resolution can be beneficial to rewriting the anaphora to its antecedent.

4 Modeling

Our proposed model for jointly learning coreference resolution and query rewrite is designed based on the GPT-2 architecture, presented in Figure 3. The input to the model is the concatenation of the dialogue context and the latest user query, where special tokens are used to separate utterances and indicate speaker information. Passing through the standard decoder layers, the hidden state $h_t^l \in \mathbb{R}^d$ and attention score $a_t^{l,j} \in \mathbb{R}^T$ at each position of the input sequence are calculated, where l, j and t denote the index of the decoder layer, that of the attention head, and the input token position respectively; d and T denote the embedding size and the length of the input sequence respectively. Inspired by the end-to-end coreference resolution model (Lee et al., 2017), our model first predicts mentions in the user query and grounds them to their corresponding referent in the dialogue context using attention heads. The model then generates the rewritten query conditioned on the resolved coreference links. The prediction process has four main steps, described in detail below:

Step 1: Mention Detection First, the model detects any possible referring expressions in the user query. Here we use the term `mention` to include all those expressions that require reference resolution (e.g., pronouns or partial entity names). We formulate mention detection as a sequence labeling problem: each token in a query is labelled as one of three classes $\{S, E, N\}$, referring to Start of mention, End of mention and None respectively. This sequence tagger in the mention detector, parameterized by a feed-forward network, takes the hidden states of the query from the last decoder layer as input, and predicts the sequence of class labels. Then the mention spans in the query can be determined by a pair of mention start S and end E tags. For instance, in Figure 3 the label of position "*this*" is class S and that of "*from*" is class E, while the rest of the positions in the query are labelled as class N. We use m_S and m_E to respectively denote the start and end position index of a predicted mention m.

Step 2: Reference Resolution For each detected mention m, the model resolves it to the antecedent (or referent) in the dialogue context by predicting the span boundaries: the position index of the referent start r_S and end r_E. Essentially, the distributions of the boundaries (r_S and r_E) are learned by supervising multiple attention heads associated with the target mention m. In other words, the at-

tention distribution a_{m_S} (the attention score of each position associated with the mention start m_S) is supervised to focus on the referent start r_S. Similarly, attention scores a_{m_E} associated with the mention end m_E are used to learn the boundary of referent end r_E. Concretely:

$$q_{r_S} = \frac{1}{L'J'} \sum_{l}^{L'} \sum_{j}^{J'} a_{m_S}^{l,j} \,, \qquad q_{r_E} = \frac{1}{L'J'} \sum_{l}^{L'} \sum_{j}^{J'} a_{m_E}^{l,j} \,, \tag{1}$$

where q_{r_S} and q_{r_E} are the probability distributions that a given token represents the referent start r_S and end r_E respectively; L' and J' are the specified number of the involved decoder layers and attention heads. We then take the argmax of these boundary distributions to resolve the referent r. Our design of reference resolution effectively leverages the powerful attention mechanism in GPT-2 without adding any extra components for reference resolution.

Step 3: Binary Rewriting Classification The model completes the coreference resolution in steps 1 and 2, after which it starts producing the rewritten query. Unlike existing query rewrite systems that directly generate the rewrite given the input, our model first predicts whether the incoming query requires to be rewritten using a binary classifier. As shown in Figure 3, the classifier, a two-layer feed-forward network followed by a softmax layer, takes as input the hidden state of the first decoding step and predicts a vector with two entries representing the rewrite and no-rewrite classes. Only when the binary prediction is true, i.e., the classifier predicts the class indicating that a rewrite is required, does the model enter Step 4 to generate the rewritten query; otherwise, the input query will be directly copied as the output. We show that a well-learned binary classifier with 93% accuracy functions as a filter that helps the model not only minimize the risk of incorrectly rewriting already self-contained queries, but also allows the rest of the generation process to solely focus on how to rewrite incomplete queries during training.

Step 4: Query Rewrite Generation In this final step, the model runs the generation step based on its binary decision of whether or not to rewrite. Unlike the standard language modeling setup in GPT-2, where the output sequence is generated directly from the last hidden states, we design the `Coref2QR` attention layer that allows information gained during coreference resolution to effectively assist in the query rewrite generation.

First, all relevant hidden states of mentions and referents predicted in Steps 1 and 2 are assembled to form a memory pool M. Note that it is possible for an example to have more than one coreference link. At each time step t' during the rewrite generation, the Coref2QR attention layer, operating as the standard multi-head attention mechanism, takes $h_{t'}$ as query to attend over the coreference related states M by treating them as keys and values. The resulting attention head $c_{t'}$ is summed with $h_{t'}$ to obtain the feature $f_{t'}$ before the final output token classifier. This design improves information flow between the two tasks, enabling the model to directly utilize information regarding previously resolved coreferents during rewrite generation.

The Coref2QR attention can be applied to any arbitrary decoder layer to facilitate the deeper interaction between rewrite and coreference resolution in the model. Formally, at each decoder layer l, the memory pool M^l stores coreference related states produced at layer l. At the generation step t', the Coref2QR layer takes $h_{t'}^l$ as query to attend over M^l to obtain $c_{t'}^l$. The final feature $f_{t'}$ before the output token classifier is then obtained by $f_{t'} = h_{t'}^L + \frac{1}{L}\sum_l c_{t'}^l$. For simplicity, in Figure 3 we only illustrate the Coref2QR attention for the last decoder layer. Our results and analysis show that this Coref2QR attention design benefits the quality of query rewrite, especially in rewriting an anaphora into its antecedent.

Optimization During training, an input sequence with length T is formed by the concatenation of the dialogue context, the user query and the target query rewrite. Four objectives, corresponding to each step in the model, are used for training. For mention detection, the objective is the cross-entropy between the predicted sequence of mention class p^M and its ground-truth sequence y^M:

$$L^M = \sum_{t=q_S}^{q_E} -\log((\mathbf{y}_t^M)^\top \mathbf{p}_t^M) \,, \tag{2}$$

where q_S and q_E denote the start and end index of the query respectively. $\top$ is the transpose operation.

For each coreference link n, the loss is calculated using the cross-entropy between the predicted

distributions of the antecedent boundaries q^n and the corresponding ground-truth y^{R_n}. The final loss for reference resolution is the sum of losses from the existing coreference links:

$$L^R = \sum_{n=1}^{N} -\log((\mathbf{y}_{r_S}^{R_n})^\top \mathbf{q}_{r_S}^n) - \log((\mathbf{y}_{r_E}^{R_n})^\top \mathbf{q}_{r_E}^n), \quad (3)$$

where N is the number of coreference links in an example. $\mathbf{q}_{r_S}^n$ and $\mathbf{q}_{r_E}^n$ represent the predicted distributions of reference start r_S and reference end r_E respectively. When an example does not contain a coreference link, L^R would be 0.

For query rewrite, the binary classification loss is the two-class cross entropy between the prediction p^B and the binary rewriting label y^B:

$$L^B = -\log((\mathbf{y}^B)^\top \mathbf{p}^B) \quad (4)$$

For generation, as in the standard language modeling task, we use cross-entropy between the predicted sequence p^Q and its ground-truth sequence y^Q:

$$L^Q = \sum_{t'=q_E+1}^{T} -\log((\mathbf{y}_{t'}^Q)^\top \mathbf{p}_{t'}^Q) \quad (5)$$

where t' is the time step in the word sequence of query rewrite. Note that L^Q is 0 for examples that do not need rewrite. The final loss is the sum of all these losses:

$$L = L^M + L^R + L^B + L^Q. \quad (6)$$

5 Experiments

Dataset As discussed in Sec. 3, we conduct all experiments on the MuDoCo dataset and follow the provided data split[1]. Data from 6 domains are aggregated to form train/dev/test sets with 16k, 1.9k and 1.9k examples respectively. Each example contains the dialogue context, the latest user query, and the corresponding coreference resolution and query rewrite annotations. Statistics for each domain are provided in Table 1. Out of all examples that are not the first turn, 64.2% of them contain coreference links and 43.7% of them require query rewrite. This makes the task more challenging, as the model also needs to learn when not to rewrite a query and when to predict no coreference links. Note that not every coreference link requires rewriting, as in the MuDoCo dataset there are coreference annotations where the mention has the exact same word span as its referent.

[1] https://github.com/facebookresearch/mudoco

Domain	Total	Coref.	Rewrite
Calling	10.7k	4.0k (60.5%)	2.2k (33.7%)
Messaging	3.9k	1.7k (69.0%)	1.0k (41.2%)
Music	2.8k	1.4k (77.7%)	1.4k (76.7%)
News	387	156 (66.4%)	189 (76.6%)
Reminders	1.7k	632 (56.4%)	357 (31.9%)
Weather	254	38 (28.6%)	102 (76.6%)
All	19.8k	8.0k (64.2%)	5.3k (43.7%)

Table 1: Total number of examples across six domains in the MuDoCo dataset, number of examples requiring coreference resolution (Coref.) and those that need query rewrite (Rewrite). Percentages are calculated across all follow-up turns (i.e., excluding the first turn).

Setup The GPT-2 decoder layers and word classification layer in our model are initialized with the pre-trained weights from the GPT-2 small model. We fine-tune the model using Adam (Kingma and Ba, 2014) optimizer with learning rate 5e-05 and batch size 15. The criterion for early stopping is the averaged performance of coreference resolution and query rewrite on the development set. Results are obtained as the average of 5 runs.

5.1 Query Rewrite

Evaluation Metrics The standard BLEU-4 (Papineni et al., 2002) between the generated and the target sentences are reported. In addition, to highlight the quality of the rewritten parts in generated sentences, following the post-processing in Quan et al. (2019), we measure an F1 score calculated by comparing machine-generated words with ground truth words for only the ellipsis / co-reference part of user utterances. We also report the percentage of all referents in ground-truth coreference links that were successfully generated in the query rewrite, denoted as reference match (RM). The RM ratio explicitly reflects the quality of coreference resolution in the generated rewritten query.

Baselines The standard seq-to-seq model with attention (seq2seq) and its pointer network (PN; Vinyals et al. (2015)) and pointer-generator network (PG; See et al. (2017)) variants are implemented as baselines. The concatenation of the dialogue context and the query are fed as input, and the output is the target rewrite. The size of the hidden states is 300 and word vectors are initialized with GloVe embeddings (Pennington et al., 2014).

Model	Prec.	Rec.	F1	BLEU	RM
seq2seq model	38.3	29.6	33.4	81.0	54.7
+ pn (Vinyals et al., 2015)	42.4	34.1	37.6	86.0	61.2
+ pg (See et al., 2017)	41.4	39.5	40.4	86.4	63.2
Our QR-only model	58.9	57.1	57.9	89.8	78.7
Our joint model	**61.0**	**59.5**	**60.2**	**90.2**	**82.0**

Table 2: Query rewrite results in F1 and BLEU. QR-only model is our model variant trained using only objectives of query rewrite.

Results Table 2 shows the query rewrite results. The low F1 score and high BLEU score is because of filtering out the non-rewritten repeated tokens in post-processing when calculating F1. This allows us to better evaluate the quality of rewritten parts and to better differentiate between good and bad generation in our task. We find that our joint model substantially outperforms all LSTM-based seq-to-seq models on all metrics. Although the pointer-generator in LSTMs can effectively copy words from the input to its generation, the powerful transformer architecture with pre-trained weights allows better learning of rewriting patterns.

To fairly investigate the impact of coreference modeling on the generation of query rewrite, we train a variant of our model using only the query rewrite objectives (Eqns. (4) and (5)), denoted as QR-only model. We can see that without coreference resolution, the F1 score drops from 60.2 to 57.9 and the reference match drops from 82.0 to 78.7. This illustrates the improved ability of the joint model to rewrite anaphoric expressions, since the model can leverage its coreference resolution predictions to generate more accurate query rewrites. We present a detailed case study with model predictions in Sec. 5.5.

5.2 Coreference Resolution

Evaluation Metrics The MUC, B^3, and $CEAF_{\phi_4}$ metrics that are widely-used in coreference resolution task are reported. Note that these metrics are calculated based on coreference clusters and we only have ground-truth annotations for coreference links between mentions and referents. To align the links and clusters, during evaluation we post-process both the ground-truth and the model predictions. All the word spans that are identical to the referent in the dialogue context are combined into a cluster so that a link between a mention and a referent can be transformed into a cluster for the standard coreference resolution evaluation.

Baselines To the best of our knowledge, there is no suitable coreference resolution model that is proposed in the same setup for dialogues[2]. We therefore experiment with the state-of-the-art models of document-based coreference resolution, including the end-to-end model (Lee et al., 2017, 2018) using BERT (Joshi et al., 2019) or SpanBERT (Joshi et al., 2020)[3]. Note that these models can only serve as a reference since they are not specifically designed for dialogue-based tasks. Since they require coreference clusters for training, coreference clusters are built from annotated links as in the post-processing step done for evaluation.

Results As seen in Table 3, SpanBERT obtains better results than BERT, which is consistent with the findings in Joshi et al. (2020). This is mainly because SpanBERT is better at capturing span information, which facilitates tasks such as coreference resolution where reasoning about relationships between spans is required. In comparison, our joint learning model achieves competitive and even slightly better results. This indicates that the design of our model leveraging attention heads inside GPT-2 is effective at predicting coreference links in dialogues. To test if the supervision of query rewrite affects the optimization of coreference resolution in joint learning, we train a model variant

[2]The baseline in Martin et al. (2020) is not compared for two reasons: 1) their setups in training/evaluation is different than ours in many ways, e.g., they only consider finished dialogues; 2) their source code is not released.

[3]https://github.com/mandarjoshi90/coref

	MUC			B^3			$CEAF_{\phi_4}$			
	P	R	F1	P	R	F1	P	R	F1	Avg. F1
c2f-coref + BERT (Joshi et al., 2019)	72.2	66.7	69.3	74.5	67.9	71.0	77.7	72.6	75.1	71.8
c2f-coref + SpanBERT (Joshi et al., 2020)	71.7	**71.4**	71.5	73.5	**72.5**	73.0	77.8	74.9	76.3	73.6
Our coref-only model	**78.8**	69.4	**73.8**	**79.6**	71.3	**75.2**	80.7	75.1	77.8	**75.6**
Our joint model	78.3	69.4	73.6	79.5	71.2	75.1	**81.1**	**75.1**	**78.0**	**75.6**

Table 3: Coreference resolution results. "Our coref-only model" is our model variant trained only using the objectives of coreference resolution.

	Prec.	Rec.	F1	BLEU	RM
complete model	**61.0**	**59.5**	**60.2**	**90.2**	**82.0**
- coref2qr attention	55.5	59.3	57.3	89.3	80.6
- coref. modeling	58.9	57.1	57.9	89.8	78.7
- binary head	54.6	54.4	54.3	88.9	78.9

Table 4: Ablation study of our joint learning model on query rewrite performance.

Model	Calling		Messaging		Music		All	
	coref.	elp.	coref.	elp.	coref.	elp.	coref.	elp.
seq2seq+pg	56.0	36.2	63.6	36.2	45.5	38.2	52.0	34.5
QR-only	75.4	51.4	77.6	**66.5**	59.8	45.5	69.0	49.3
Joint	**78.3**	**52.0**	**81.3**	64.3	**63.1**	**51.1**	**72.1**	**50.9**

Table 5: Query rewrite performance (F1) over three main domains and All test set with respect to two types of rewriting: coreference (coref.) and ellipses (elp.).

using only the objectives for coreference resolution (Eqns. (2) and (3)), denoted as `coref-only` model. It is observed that the results of the coref-only model are very close to that of the joint model, showing that the addition of coreference resolution in joint learning is beneficial to query rewrite without sacrificing the performance of the former.

5.3 Ablation Study

Here, we investigate how the different components in our joint model contribute to the performance of query rewrite. We remove one component at a time and examine the performance of query rewrite. As shown in Table 4, without the designed coref2qr attention layer, the performance degrades with a drop of 2.9% F1 and 1.4% RM rate. By further removing the supervision of coreference modeling from our joint learning model, the model is solely optimized towards the objectives of query rewrite and produces worse results compared to the complete model. These results indicate that through joint learning, the model's ability of generating the rewritten query improves, including its ability to rewrite the anaphora with its antecedent, by leveraging the information from coreference resolution modeling. In addition, the binary head plays an essential role in our model. The accuracy of this binary classifier is 93.9%. Without the binary head, the performance drop can be up to 5.9% F1 (60.2 -> 54.3). This shows that with the binary classification, the model is able to focus on rewriting the input query without worrying about whether to rewrite or not.

5.4 Analysis

In this section we analyze query rewrite performance on two different types of rewriting, coreference (coref.) and ellipses (elp.). The F1 score over three main domains and all test sets are reported in Table 5. The seq2seq+pg model is the baseline seq2seq model with pointer-generator; QR-only model is our model variant but trained without coreference modeling. The overall trend shows that 1) when the dialogue contains coreferences, the joint learning model is more capable of rewriting the query by leveraging its coreference predictions; 2) when coreferences are not present but the query still needs rewriting on account of information omission, the joint model can still perform competitively with the QR-only model.

5.5 Case Study

We demonstrate several examples of query rewrites generated by different models to provide more insights into the task and into the benefits of joint learning. The coreference links predicted by the joint learning model are appended after its generated rewrite. Two examples that require coreference resolution in query rewrite are shown in Table 6. In the left dialogue, *"the song"* in the user query refers back to *"Talking to the Moon"* mentioned in the first user turn. Both seq2seq+pg and QR-only model fail to generate the correct reference in the rewrite, probably because of the high

Dialogue Context	usr: When was Talking to the Moon by Bruno Mars released? sys: On April 12, 2011. usr: Who produced the song? sys: The Smeezingtons and Bhasker.	usr: I want to send a message. sys: Who do you want to send a message to? usr: To Ariana. sys: Ariana Smith or Ariana Taylor?
User Query	Could you play the song for me?	The second one.
Rewrite Label	Could you play **Talking to the Moon** for me?	**Ariana Taylor**.
seq2seq+pg	Could you play **the moon** for me? ✗	**Ariana Smith**. ✗
QR-only	Could you play the song for me? ✗	**Ariana Smith**. ✗
Joint	Could you play **Talking to the Moon** for me? (song -> Talking to the Moon) ✓	**Ariana Taylor**. (one -> Ariana Taylor) ✓

Table 6: Two coreference examples from test set with rewrites generated by three models: 1. seq-to-seq model with pointer-generator (seq2seq+pg); 2. Our QR-only model; 3. Our joint learning model. The rewritten parts are highlighted in bold. The coreference links predicted by 3. are presented as (mention -> antecedent).

Dialogue Context	usr: What's the temperature like in Richmond today? sys: The temperature is going to be a warm 85%, but there is a chance of rain.
User Query	What are the chances of rain today?
Rewrite Label	What are the chances of rain today **in Richmond**?
seq2seq+pg	What are the chances of rain today? ✗
QR-only	What are the chances of rain today **in Richmond**? ✓
Joint	What are the chances of rain today **in Richmond**? ✓

Table 7: An example with ellipsis from the test set. Rewrites generated by three different models are shown.

complexity of a long dialogue. The joint learning model not only correctly predicts the coreference link pointing from the mention to its referent in the first turn, but also generates a rewrite perfectly consistent with its coreference prediction. A similar trend can be observed in the right example. The first two models cannot identify which *"Ariana"* to generate, while our model is able to rewrite with the correct one with the aid of the correct coreference resolution. While our model does well on most of the test cases, there are situations where the joint model fails to predict correctly. A representative failure example is provided in Appendix A.2.

Table 7 shows an ellipsis example. The implicit location in the user query can be recovered through rewriting by both GPT-2 based models, while the LSTM-based model tends to keep the query. This indicates that 1) even with the pointer-generator's ability to copy source text, the seq2seq model is not capable enough of handling the difficult information omission rewrite; 2) the joint learning model still performs well on ellipses, while substantially benefiting in coreference cases.

6 Conclusion and Future Work

We propose a novel joint learning framework for coreference resolution and query rewrite in dialogues. Modeling coreference resolution not only complements the missing information in query rewrite, but is also beneficial to rewriting anaphoric expressions. Our joint learning model can predict coreference links between the user query and dialogue context, and generate the rewritten query. We show that with the aid of coreference resolution, the performance of query rewrite can be substantially boosted. Furthermore, our model produces competitive results in coreference resolution when compared to state-of-the-art BERT-based systems. We hope that the presented joint learning task with the release of our query rewrite annotations on the MuDoCo dataset provides a promising research direction in multi-turn dialogue understanding.

One restriction of our model is that by virtue of the model being designed to predict the boundaries of a reference, our model is only able to handle cases involving continuous spans of words. In addition, the influence of query rewrite on coreference resolution is limited due to the nature of the information flow in our current model design. Future work will focus on these perspectives.

Acknowledgements

The authors would like to thank Hadas Kotek for her help with the data annotation guidelines and the organization of the grading project. The authors would also like to thank Barry-John Theobald, Stephen Pulman, Jason Williams and Murat Akbacak for discussions and feedback, and the anonymous reviewers for their helpful feedback.

References

Raviteja Anantha, Svitlana Vakulenko, Zhucheng Tu, Shayne Longpre, Stephen Pulman, and Srinivas Chappidi. 2020. Open-domain question answering goes conversational via question rewriting. *arXiv preprint arXiv:2010.04898*.

Eric Bengtson and Dan Roth. 2008. Understanding the value of features for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 294–303.

Daniel M Bikel, Vittorio Castelli, Radu Florian, and Ding-jung Han. 2009. Entity linking and slot filling through statistical processing and inference rules. In *TAC*.

Anders Björkelund and Jonas Kuhn. 2014. Learning structured perceptrons for coreference resolution with latent antecedents and non-local features. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 47–57.

Kevin Clark and Christopher D Manning. 2016a. Deep reinforcement learning for mention-ranking coreference models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2256–2262.

Kevin Clark and Christopher D Manning. 2016b. Improving coreference resolution by learning entity-level distributed representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 643–653.

Eraldo Fernandes, Cicero dos Santos, and Ruy Luiz Milidiú. 2012. Latent structure perceptron with feature induction for unrestricted coreference resolution. In *Joint Conference on EMNLP and CoNLL-Shared Task*, pages 41–48.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Mandar Joshi, Omer Levy, Luke Zettlemoyer, and Daniel S Weld. 2019. Bert for coreference resolution: Baselines and analysis. 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 5807–5812.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Vineet Kumar and Sachindra Joshi. 2016. Non-sentential question resolution using sequence to sequence learning. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2022–2031, Osaka, Japan. The COLING 2016 Organizing Committee.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 687–692.

Scott Martin, Shivani Poddar, and Kartikeya Upasani. 2020. MuDoCo: Corpus for multidomain coreference resolution and referring expression generation. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 104–111, Marseille, France. European Language Resources Association.

Sebastian Martschat and Michael Strube. 2015. Latent structures for coreference resolution. *Transactions of the Association for Computational Linguistics*, 3:405–418.

Vincent Ng. 2010. Supervised noun phrase coreference research: The first fifteen years. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 1396–1411.

Vincent Ng and Claire Cardie. 2002. Identifying anaphoric and non-anaphoric noun phrases to improve coreference resolution. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Haoruo Peng, Kai-Wei Chang, and Dan Roth. 2015. A joint framework for coreference resolution and mention head detection. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 12–21, Beijing, China. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

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, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nathanael Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher D Manning. 2010. A multi-pass sieve for coreference resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 492–501.

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.

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.

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.

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 *Advances in neural information processing systems*, pages 5998–6008.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in neural information processing systems*, pages 2692–2700.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gasic, Lina M Rojas Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449.

Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. 2020. CorefQA: Coreference resolution as query-based span prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6953–6963, Online. Association for Computational Linguistics.

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.

A Appendices

A.1 Training details

The average run time for training our joint learning model is 6 hours using GTX 1080 Ti. Based on the GPT-2 architecture, our model has 148M parameters. For the attention heads used for predicting the referent in Equation 1, hyper-parameter boundaries for L' and J' are: $1 \leq L' \leq 12$ and $1 \leq J' \leq 12$. The best performance is obtained when only using the last two decode layers ($L' = 2$) with 3 attention heads used in each layer ($J' = 3$). Hyper-parameters are tuned based on the averaged performance of query rewrite and coreference resolution on the development set.

A.2 Sample Model-Generated Failure Cases

We find that our joint model makes mistakes when the coreference signal is ambiguous and there is complex dialogue context (e.g., having multiple person names in an utterance). In a representative example (Table 8), even though the joint model predicts one of the coreference links correctly (*one -> call*) and generates the corresponding rewritten span (*call from Sana and Erica*), it fails to infer that the pronoun *her* refers to *Deirdre*, and simply ignores the corresponding rewrite. This is likely because there are many female names that the pronoun *her* can refer to in this utterance, and these types of complex cases are too infrequent in the training corpus for the model to learn well.

A.3 Query Rewrite Annotation Guideline

The annotation guidelines for collecting query rewrites on the MuDoCo dataset are provided in the following pages. Note that we annotate the rewrite label for every utterance, including the system response, even though they are not used in our experiments.

Dialogue Context	usr: Answer the call. sys: Its Sana and Erica, however, Deirdre on the other line and said its an emergency.
User Query	Very well, cancel the first one and put her through.
Rewrite Label	Very well, cancel the **call with Sana and Erica** and put **Deirdre** through. (one -> call, her -> Deirdre)
seq2seq+pg	Very well, cancel the first **message** and put her through
QR-only	Very well, cancel **Deirdre** and put her through.
Joint	Very well, cancel the **call from Sana and Erica** through. (one -> call, her -> Sana)

Table 8: A complex dialogue example where all systems fail to rewrite correctly. Ground-truth and prediction of coreference are appended correspondingly. Rewritten parts are highlighted in bold.

MuDoCo Grading Guideline

Overview

In this project, you will be given a conversation between a user and a virtual assistant. Certain parts of the utterances (both user and assistant) might require previous context to be fully understood. **The goal of the project is to make *minimal* changes to the current turn so that it can be understood independently, without needing access to the prior context**. This may mean one (or more) of several things:

- Replacing a **pronoun** with its full referent
- Spelling out the content of an **elided phrase**
- Adding elements mentioned in the prior context that are now a part of the understood **Common Ground** of the conversation.

Here is a simple example to get us started. This is a multi-turn conversation that contains a referring expression "that week". If we only had access to the "current" turn, we wouldn't be able to resolve the meaning of "that week"; also, we would incorrectly resolve the intended location of the question to the speaker's location instead of to Sao Paulo, as the previous context makes clear is actually intended.

```
User:      Will next week be a good time to vacation in Sao Paulo?
Assistant: There will be thunderstorms and a lot of rain.
User:      What will the temperature be like that week?              <-- current
```

Our goal in this project is to rewrite **those parts of the current turn that require context**. Here, we can replace "that week" with "next week", and insert "in Sao Paulo" to allow the correct location to be inferred from the request.

```
User:      Will next week be a good time to vacation in Sao Paulo ?
Assistant: There will be thunderstorms and a lot of rain .
User:      What will the temperature be like next week in Sao Paulo?  <-- current
```

Notice that we do not otherwise alter the rest of the utterance, beyond these minimal changes.

Case by Case Guidelines

We introduce our reference / ellipsis resolution guideline in the following categories:

1. **Do rewrite ellipsis as well as references**
2. **Do not paraphrase or summarize but do use the phrasing that appeared in the context**
3. **A special case exception: calls, messages, reminders**
4. **Multiple references**
5. **Data errors**

To formally define what an appropriate "resolution" of a reference / ellipsis is, we introduce the concept of **minimal changes**, which has the following properties we will illustrate with examples below:

1. You may add information that was explicitly uttered in one of the previous turns of the conversation.
2. You should **not** add more information than necessary to make the utterance understandable without additional context, even if such information is available.

3. Special case exception: references to calls, messages, and reminders don't need additional resolution beyond reference to "the call", "the message", or "the reminder".

Do rewrite ellipsis as well as references

Example 1:

```
User:       Can we expect rain anytime soon?
Assistant:  Rain is possible early next week.
User:       How much chance of rain will there be?              <-- current
```

As we can see, the intended time of the current utterance is understood to be "early next week", since that is the time that is added to the common ground in a previous turn in the conversation as relevant to this weather inquiry. However, if we don't have this information, we would naturally interpret the time of the current turn as "now'". Consequently, we make a minimal change to the current turn by inserting this time information:

```
Desired:    How much chance of rain will there be next week?
Undersired: skip
```

Example 2:

```
User:       It's freezing! Is it going to snow?
Assistant:  There's a 65 percent chance of snow tonight
User:       What about this weekend?                            <-- current
```

In this case, the user elided the entire clause "is it going to snow", which we should recover:

```
Desired:    Is it going to snow this weekend?
Undersired: skip
```

Example 3:

We may also see cases of noun phrases with some form of ellipsis. These are not implicit arguments but rather explicitly mentioned, but have some *partially* missing information. In such a case we want to fill those out according to our rules to ensure that they are recoverable based on the current turn alone:

```
User:      Play Somewhere Over the Rainbow
Assistant: OK, playing Somewhere Over the Rainbow by Judy Garland.
User:      Can I hear the version by Ariana Grande?
```

The relevant NP here is "the version", which we want to spell out explicitly as "the version of Somewhere Over the Rainbow ":

```
Desired:    Can I hear the version of Somewhere Over the Rainbow by Ariana Grande?
Undersired: Skip
```

Do not paraphrase or summarize but do use the phrasing that appeared in the context

Do not rephrase / summarize. Instead, stick to the phrasing that appears in the context as much as possible. We give several examples of this below.

Example 1:

```
User:       What is the latest news on the Brewers ?
Assistant:  There are two recent stories . One is about their lose last night , and th
User:       What happened to him ?
Assistant:  He stepped on his Jesus Aguilar foot .
User:       Then what happened ?
Assistant:  The Brewers bench cleared because they were mad at him .
User:       Can you send more about this to my screen ?     <-- current
```

In this case, the desired behavior is to simply resolve "this" into "this story", rather than providing a paraphrase that collects information from all of the previous turns to create a phrase that wasn't actually uttered in the conversation at all. There is a single story ongoing in this context, so there's not need for a fuller description of this entity:

```
Desired:    Can you send more about this story to my screen ?
Undersired: Can you send more about The Brewers bench cleared because they were mad at
```

Example 2:

```
User:       Can you play Moonlight Sonata by Beethoven?
Assistant:  Yes, you have the live recording of Moonlight Sonata.
User:       I dont want that version. Can you search for the best version on the recor
Assistant:  Yes, the best version is from 1914 and can stream from YouTube to your pla
User:       Okay, stream and save to my playlist under Relaxing.
```

There is an implied argument here (roughly, stream *it* and save *it* ...), which we can recover from the context. Here, we choose the simplest form we can use to complete the utterance, and do not add information across turns.

```
Desired:    Okay, stream the Moonlight Sonata by Beethoven and save it to my playlist
Undersired: Okay, stream the 1914 version of Moonlight Sonata by Beethoven from YouTub
```

A special case exception: Calls, messages, reminders

For these entities, you don't need to resolve nominal ellipsis or write out its content in full, just call these entities "the call", "the message", "the reminder". This is an exception to the rules above – there is no need to provide additional information even if it present in the exchange.

Example 1:

```
User:       Who is calling me right now?
Assistant:  Marty and Brad are calling you via conference call.
User:       Answer the call but change it to voice only.          <-- current
```

Reference is already made to “the call”, and there is only one call referenced in this exchange, so nothing needs to be done here:

```
Desired:    Skip
Undersired: Answer the conference call from Marty and Brad but change it to voice only
```

Example 2:

```
User:       I need to send a message.
Assistant:  OK, to who?
User:       To my yoga class.
Assistant:  OK, what do you want to say?
User:       Let them know that class is on for tomorrow.
Assistant:  OK, I have sent your yoga class the message.            <-- current
```

Likewise, there is just one message being discussed here, so there is no need for resolution or added information.

```
Desired:    Skip
Undersired: OK, I have sent your yoga class the message that class is on for tomorrow.
```

Example 3:

```
User:       Is there a reminder for this afternoon?
Assistant:  There is one reminder for your sons school.
User:       What does it say?                                   <-- current
```

Here we have a pronoun that needs to be resolved to a noun phrase. The simple noun phrase “the reminder” is all that’s needed, please don’t add information about the content of the reminder, since this reminder isn’t ambiguous with any other reminder in the context.

```
Desired:    What does the reminder say?
Undersired: What does the reminder for my sons school say?
```

Multiple references

If there are multiple references towards the same entity, you only need to resolve it in the first one. If there is a reference towards an entity in the utterance itself, you don’t need to resolve it at all since it’s already fully interpretable from that turn alone.

Example 1:

```
User:       Send a message to Stephen Bradley.
Assistant:  Okay, what is the message?
User:       Ask him if he wants in on the office lottery pool.        <-- current
```

The current turn contains two pronouns referring to the same individuals. Once the first pronoun is resolved, it provides a reference for the second pronoun within the same sentence. Therefore, we only need to resolve that first

instance (in fact, resolving the second pronoun leads to an unnatural paraphrase that should always be avoided):

```
Desired:    Ask Stephen Bradley if he wants in on the office lottery pool.
Undesired:  Ask Stephen Bradley if Stephen Bradley wants in on the office lottery pool
```

Example 2:

```
User:      Set a reminder.
Assistant: Okay, what for?
User:      Remind me to call my daughter for her wedding.              <-- current
```

The pronoun “her” receives a referent within the current turn (namely, “my daughter“), so there is no need to do anything here. You should skip, and not spell out the referent of the pronoun.

```
Desired:   Skip
Undesired: Remind me to call my daughter for my daughters wedding.
```

Data errors

Some times the request itself could be faulty, here’s a couple of cases::

Example 1:
When there is ambiguity in the reference / nominal ellipsis, you should skip

```
User:      Who called me while I was in class?
Assistant: You have a missed call from James and also Sam
User:      Okay, call him back and set my status to online.            <-- current
```

Here since there are two entities, “James” and “Sam”, both common male names, it is impossible to determine its referent. You should skip in this case

```
Desired:   Skip
Undesired: Okay, call James back and set my status to online.
```