Query Resolution for Conversational Search with Limited Supervision

Voskarides, N.; Li, D.; Ren, P.; Kanoulas, E.; de Rijke, M.

DOI
10.1145/3397271.3401130

Publication date
2020

Document Version
Final published version

Published in
SIGIR '20

License
Article 25fa Dutch Copyright Act (https://www.openaccess.nl/en/in-the-netherlands/you-share-we-take-care)

Citation for published version (APA):
Query Resolution for Conversational Search with Limited Supervision

Nikos Voskarides$^1$  Dan Li$^1$  Pengjie Ren$^1$  Evangelos Kanoulas$^1$  Maarten de Rijke$^{1,2}$

$^1$University of Amsterdam, Amsterdam, The Netherlands  $^2$Ahold Delhaize, Zaandam, The Netherlands
nickvosk@gmail.com, d.li@uva.nl, p.ren@uva.nl, e.kanoulas@uva.nl, m.derijke@uva.nl

ABSTRACT

In this work we focus on multi-turn passage retrieval as a crucial component of conversational search. One of the key challenges in multi-turn passage retrieval comes from the fact that the current turn query is often underspecified due to zero anaphora, topic change, or topic return. Context from the conversational history can be used to arrive at a better expression of the current turn query, defined as the task of query resolution. In this paper, we model the query resolution task as a binary term classification problem: for each term appearing in the previous turns of the conversation decide whether to add it to the current turn query or not. We propose QuReTeC (Query Resolution by Term Classification), a neural query resolution model based on bidirectional transformers. We propose a distant supervision method to automatically generate training data by using query-passage relevance labels. Such labels are often readily available in a collection either as human annotations or inferred from user interactions. We show that QuReTeC outperforms state-of-the-art models, and furthermore, that our distant supervision method can be used to substantially reduce the amount of human-curated data required to train QuReTeC. We incorporate QuReTeC in a multi-turn, multi-stage passage retrieval architecture and demonstrate its effectiveness on the TREC CAsT dataset.

KEYWORDS

Conversational search; Query resolution

1 INTRODUCTION

Conversational AI deals with developing dialogue systems that enable interactive knowledge gathering [17]. A large portion of work in this area has focused on building dialogue systems that are capable of engaging with the user through chit-chat [23] or helping the user complete small well-specified tasks [32]. In order to improve the capability of such systems to engage in complex information seeking conversations [34], researchers have proposed information seeking tasks such as conversational question answering (QA) over simple contexts, such as a single-paragraph text [7, 37]. In contrast to conversational QA over simple contexts, in conversational search, a user aims to interactively find information stored in a large document collection [10].

In this paper, we study multi-turn passage retrieval as an instance of conversational search: given the conversation history (the previous turns) and the current turn query, we aim to retrieve passage-length texts that satisfy the user’s underlying information need [11]. Here, the current turn query may be under-specified and thus, we need to take into account context from the conversation history to arrive at a better expression of the current turn query. Thus, we need to perform query resolution, that is, adding missing context from the conversation history to the current turn query, if needed. An example of an under-specified query can be seen in Table 1, turn #4, for which the gold standard query resolution is: “when was saosin’s first album released?”. In this example, context from all turns #1 (“saosin”), #2 (“band”) and #3 (“first”) have to be taken into account to arrive to the query resolution.

<table>
<thead>
<tr>
<th>Turn</th>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>who formed saosin?</td>
</tr>
<tr>
<td>2</td>
<td>when was the band founded?</td>
</tr>
<tr>
<td>3</td>
<td>what was their first album?</td>
</tr>
<tr>
<td>4</td>
<td>when was the album released?</td>
</tr>
</tbody>
</table>

revised: when was saosin’s first album released?

Relevant passage to turn #4: The original lineup for Saosin, consisting of Burchell, Shekoski, Kennedy and Green, was formed in the summer of 2003. On June 17, the band released their first commercial production, the EP Translating the Name.

Table 1: Excerpt from an example conversational dialog. Co-occurring terms in the conversation history and the relevant passage to the current turn (#4) are shown in bold-face.

Designing automatic query resolution systems is challenging because of phenomena such as zero anaphora, topic change and topic return, which are prominent in information seeking conversations [50]. These phenomena are not easy to capture with standard NLP tools (e.g., coreference resolution). Also, heuristics such as appending (part of) the conversation history to the current turn query are likely to lead to query drift [27]. Recent work has modeled query resolution as a sequence generation task [15, 21, 36]. Another way of implicitly solving query resolution is by query modeling [18, 42, 47], which has been studied and developed under the setup of session-based search [5, 6].
In this paper, we propose to model query resolution for conversational search as a binary term classification task: for each term in the previous turns of the conversation decide whether to add it to the current turn query or not. We propose QuReTeC (Query Resolution by Term Classification), a query resolution model based on bidirectional transformers [43] – more specifically BERT [13]. The model encodes the conversation history and the current turn query and uses a term classification layer to predict a binary label for each term in the conversation history. We integrate QuReTeC in a standard two-step cascade architecture that consists of an initial retrieval step and a reranking step. This is done by using the set of terms predicted as relevant by QuReTeC as query expansion terms.

Training QuReTeC requires binary labels for each term in the conversation history. One way to obtain such labels is to use human-curated gold standard query resolutions [15]. However, these labels might be cumbersome to obtain in practice. On the other hand, researchers and practitioners have been collecting general-purpose passage relevance labels, either by the means of human annotations or by the means of weak signals, e.g., clicks or mouse movements [19]. We propose a distant supervision method to automatically generate training data, on the basis of such passage relevance labels. The key assumption is that passages that are relevant to the current turn share context with the conversation history that is missing from the current turn query. Table 1 illustrates this assumption: the relevant passage to turn #4 shares terms with the conversation history. Thus, we label the terms that co-occur in the relevant passages\(^1\) and the conversation history as relevant for the current turn.

Our main contributions can be summarized as follows:

1. We model the task of query resolution as a binary term classification task and propose to address it with a neural model based on bidirectional transformers, QuReTeC.

2. We propose a distant supervision approach that can use general-purpose passage relevance data to substantially reduce the amount of human-curated data required to train QuReTeC.

3. We experimentally show that when integrating the QuReTeC model in a multi-stage ranking architecture we significantly outperform baseline models. Also, we conduct extensive ablation studies and analyses to shed light into the workings of our query resolution model and its impact on retrieval performance.

2 RELATED WORK

Conversational search. Early studies on conversational search have focused on characterizing information seeking strategies and building interactive IR systems [3, 4, 9, 30]. Vtyurina et al. [45] investigated human behaviour in conversational systems through a user study and find that existing conversational assistants cannot be effectively used for conversational search with complex information needs. Radlinski and Craswell [35] present a theoretical framework for conversational search, which highlights the need for multi-turn interactions. Dalton et al. [11] organize the Conversational Assistance Track (CAS-T) at TREC 2019. The goal of the track is to establish a concrete and standard collection of data with information needs to make systems directly comparable. They release a multi-turn passage retrieval dataset annotated by experts, which we use to compute our method to the baseline methods.

Query resolution. Query resolution has been studied in the context of dialogue systems. Raghu et al. [36] develop a pipeline model for query resolution in dialogues as text generation. Kumar and Joshi [21] follow up on that work by using a sequence to sequence model combined with a retrieval model. However, both these works rely on templates that are not available in our setting. More related to our work, Elgohary et al. [15] studied query resolution in the context of conversational QA over a single paragraph text. They use a sequence to sequence model augmented with a copy and an attention mechanism and a coverage loss. They annotate part of the QuAC dataset [7] with gold standard query resolutions on which they apply their model and obtain competitive performance. In contrast to all the aforementioned works that model query resolution as text generation, we model query resolution as binary term classification in the conversation history.

Query modeling. Query modeling has been used in session search, where the task is to retrieve documents for a given query by utilizing previous queries and user interactions with the retrieval system [6]. Guan et al. [18] extract substrings from the current and previous turn queries to construct a new query for the current turn. Yang et al. [47] propose a query change model that models both edits between consecutive queries and the ranked list returned by the previous turn query. Van Gysel et al. [42] compare the lexical matching session search approaches and find that naive methods based on term frequency weighing perform on par with specialized session search models. The methods described above are informed by studies of how users reformulate their queries and why [41], which, in principle, is different in nature from conversational search. For instance, in session search users tend to add query terms more than removing query terms, which is not the case in (spoken) conversational search. Another form of query modeling is query expansion. Pseudo-relevance feedback is a query expansion technique that first retrieves a set of documents that are assumed to be relevant to the query, and then selects terms from the retrieved documents that are used to expand the query [1, 22, 29]. Note that pseudo-relevance feedback is fundamentally different from query resolution: in order to revise the query, the former relies on the top-ranked documents, while the latter only relies on the conversation history.

Distant supervision. Distant supervision can be used to obtain large amounts of noisy training data. One of its most successful applications is relation extraction, first proposed by Mintz et al. [26]. They take as input two entities and a relation between them, gather sentences where the two entities co-occur from a large text corpus, and treat those as positive examples for training a relation extraction system. Beyond relation extraction, distant supervision has also been used to automatically generate noisy training data for other tasks such as named entity recognition [49], sentiment classification [39], knowledge graph fact contextualization [44] and dialogue response generation [38]. In our work, we follow the distant supervision paradigm to automatically generate training data for query resolution in conversational search by using query-passage relevance labels.

\(^1\)A relevance passage contains not only the answer to the question but also context and supporting facts that allow the algorithm or the human to reach to this answer.
3 MULTI-TURN PASSAGE RETRIEVAL PIPELINE

In this section we provide formal definitions and describe our multi-turn passage retrieval pipeline. Table 2 lists notation used in this paper.

3.1 Definitions

Multi-turn passage ranking. Let \( q_1, \ldots, q_{i-1}, q_i \) be a sequence of conversational queries that share a common topic \( T \). Let \( q_i \) be the current turn query and \( q_{1:i-1} \) be the conversation history. Given \( q_i \) and \( q_{1:i-1} \), the task is to retrieve a ranked list of passages \( L \) from a passage collection \( D \) that satisfy the user’s information need.\(^2\)

In the multi-turn passage ranking task, the current turn query \( q_i \) is often underspecified due to phenomena such as zero anaphora, topic change, and topic return. Thus, context from the conversation history \( q_{1:i-1} \) must be taken into account to arrive at a better expression of the current turn query \( q_i \). This challenge can be addressed by query resolution.

Query resolution. Given the conversation history \( q_{1:i-1} \) and the current turn query \( q_i \), output a query \( \hat{q}_i \) that includes both the existing information in \( q_i \) and the missing context of \( q_i \) that exists in the conversation history \( q_{1:i-1} \).

3.2 Multi-turn passage retrieval pipeline

Figure 1 illustrates our multi-turn passage retrieval pipeline. We use a two-step cascade ranking architecture [46], which we augment with a query resolution module (Section 4). First, the unsupervised initial retrieval step outputs the initial ranked list \( L_1 \) (Section 3.2.1). Second, the re-ranking step outputs the final ranked list \( L \) (Section 3.2.2). Below we describe the two steps of the cascade ranking architecture.

3.2.1 Initial retrieval step. In this step we obtain the initial ranked list \( L_1 \) by scoring each passage \( p \) in the passage collection \( D \) with respect to the resolved query \( \hat{q}_i \) using a lexical matching ranking function \( f_1 \). We use query likelihood (QL) with Dirichlet smoothing [51] as \( f_1 \), since it outperformed other ranking functions such as BM25 in preliminary experiments over the TREC CAsT dataset.

3.2.2 Reranking step. In this step, we re-rank the list \( L_1 \) by scoring each passage \( p \in L_1 \) with a ranking function \( f_2 \) to obtain the final ranked list \( L \). To construct \( f_2 \), we use rank fusion and combine the scores obtained by \( f_1 \) (used in initial retrieval step) and a supervised neural ranker \( f_n \). Next, we describe the neural ranker \( f_n \).

Supervised neural ranker. We use BERT [13] as the neural ranker \( f_n \), as it has been shown to achieve state-of-the-art performance in ad-hoc retrieval [25, 33, 48]. Also, BERT has been shown to prefer semantic matches [33], and thereby can be complementary to \( f_1 \), which is a lexical matching method. As is standard when using BERT for pairs of sequences, the input to the model is formatted as \( [\text{CLS}], \hat{q}_i, \text{SEP}, p \)\(^3\), where \( \text{CLS} \) is a special token, \( \hat{q}_i \) is the resolved current turn query, \( p \) is the passage. We add a dropout layer and a linear layer \( L_0 \) on top of the representation of the \( <\text{CLS}> \) token in the last layer, followed by a tanh function to obtain \( f_n \) [25]. We score each passage \( p \in L_1 \) using \( f_n \) to obtain \( L_n \). We fine-tune the pretrained BERT model using pairwise ranking loss on a large-scale single-turn passage ranking dataset [48]. During training we sample as many negative as positive passages per query.

Rank fusion. We design \( f_2 \) such that it combines lexical matching and semantic matching [31]. We use Reciprocal Rank Fusion (RRF) [8] to combine the score obtained by the lexical matching ranking function \( f_1 \), and the semantic matching supervised neural ranker \( f_n \). We choose RRF because of its effectiveness in combining individual rankers in ad-hoc retrieval and because of its simplicity (it has only one hyper-parameter). We define \( f_2 \) as the RRF of \( L_1 \) and \( L_n \) [8]:

\[
f_2(p) = \sum_{L' \in \{L_1, L_n\}} \frac{1}{k + \text{rank}(p, L')}
\]

where \( \text{rank}(p, L') \) is the rank of passage \( p \) in a ranked list \( L' \), and \( k \) is a hyperparameter.\(^4\) We score each passage \( p \) in the initial ranked list \( L_1 \) with \( f_2 \) to obtain the final ranked list \( L \).

Since developing specialized re-rankers for the task at hand is not the focus of this paper, we leave more sophisticated methods for choosing the neural ranker \( f_n \) and for combining multiple rankers as future work. In the next section, we describe our query resolution model, QuReTeC, which is the focus of this paper.

4 QUERY RESOLUTION

In this section we first describe how we model query resolution as term classification (Section 4.1), then present our query resolution model, QuReTeC (Section 4.2), and finally describe how we generate distant supervision labels for the model (Section 4.3).

---

\(^2\)We follow the TREC CAsT setup and only take into account \( q_{1:i-1} \) but not the passages retrieved for \( q_{1:i-1} \).

\(^3\)We set \( k = 60 \) and do not tune it.
4.1 Query resolution as term classification in the conversation history

Previous work has modeled query resolution as a sequence to sequence task [15, 21], where the source sequence is \( q_{1:i} \) and the target sequence is \( q_{1:i}^* \), where \( q_{1:i}^* \) is a gold standard resolution of the current turn query \( q_i \). For instance, the gold standard resolution of turn #4 in Table 1 is: “When was Saosin’s first album released?”

However, since (i) the initial retrieval step of our pipeline (Section 3.2.1) is a term-based model that treats queries as bag of words, and (ii) the supervised neural ranker we use in the re-ranking step (Section 3.2.2) is robust to queries that are not well-formed natural language queries but rather a set of terms to expand the query. Besides, sequence to sequence based models generally need a massive amount of data for training in order to get reasonable performance due to their generation objective [16]. Therefore, we model query resolution as a term classification task: given the conversation history \( q_{1:i-1} \) and the current turn query \( q_i \), output a binary label (relevant or non-relevant) for each term in \( q_{1:i-1} \). Terms in the conversation history \( q_{1:i-1} \) that are tagged as relevant are appended to the current turn query \( q_i \) to form the predicted current turn query resolution \( \hat{q}_i \).

We define the set of relevant resolution terms \( E^*_i(q_i) \) as:

\[
E^*_i(q_i) = \text{terms}(q_i^*) \cap \text{terms}(q_{1:i-1}) \setminus \text{terms}(q_i),
\]

where \( q_i^* \) is a gold standard resolution of the current turn query \( q_i \). Under this formulation, the set of relevant terms \( E^*_i(q_i) \) represents the missing context from the conversation history \( q_{1:i-1} \). For instance, the set of gold standard resolution terms \( E^*_i(q_i) \) for turn #4 in Table 1 is (Saosin, first). Note that \( E^*_i(q_i) \) can be empty if \( q_i = q_i^* \), i.e., the current turn query does not need to be resolved, or if \( \text{terms}(q_{i-1}) \cap \text{terms}(q_{1:i-1}) \) is empty. In our experiments \( \text{terms}(q_i^*) \cap \text{terms}(q_{1:i-1}) = \text{terms}(q_i^*) \), and therefore almost all the gold standard resolution terms can be found in the conversation history.

4.2 Query resolution model

In this section, we describe our query resolution model, QuReTeC. Figure 2a shows the model architecture of QuReTeC. Each term in the input sequence is first encoded using bidirectional transformers [43] – more specifically BERT [13]. Then, a term classification layer takes each encoded term as input and outputs a score for each term. We use BERT as the encoder since it has been successfully applied in tasks similar to ours, such as named entity recognition and coreference resolution [13, 20, 24]. Next we describe the main parts of QuReTeC in detail, i.e., input sequence, BERT encoder and Term classification layer.

1. Input sequence. The input sequence consists of all the terms in the queries of the previous turns \( q_{1:i-1} \) and the current turn \( q_i \). It is formed as: \([<CLS>, \text{terms}(q_1), ..., \text{terms}(q_{i-1}), <SEP>, \text{terms}(q_i)]\), where \(<CLS>\) and \(<SEP>\) are special tokens. We add a special separator token \(<SEP>\) between the previous turn \( q_{i-1} \) and the current turn \( q_i \) in order to inform the model where the current turn begins. Figure 2b shows an example input sequence and the gold standard term labels.

2. BERT encoder. BERT first represents the input terms with WordPiece embeddings using a 30K vocabulary. After applying multiple transformer blocks, BERT outputs an encoding for each term. We refer the interested reader to the original paper for a detailed description of BERT [15].

3. Term classification layer. The term classification layer is applied on top of the representation of the first sub-token of each term [13]. It consists of a dropout layer, a linear layer and a sigmoid function and outputs a scalar for each term. We mask out the output of \(<CLS>\) and the current turn terms, since we are not interested in predicting a label for those (see Equation (2) for the definition and Figure 2b for an example).

In order to train QuReTeC we need a dataset containing gold standard resolution terms \( E^*_i(q_i) \) for each \( q_i \). The terms in \( E^*_i(q_i) \) are labeled as relevant and the rest of the terms \( \text{terms}(q_{1:i-1}) \setminus E^*_i(q_i) \) as non-relevant. Assuming there exists a gold standard resolution \( q_i^* \) for each \( q_i \), we can derive \( E^*_i(q_i) \) using Equation (2). We use standard binary cross entropy as the loss function.

4.3 Generating distant supervision for query resolution

Recall that the gold standard resolution \( q_i^* \) includes the information in \( q_i \) and the missing context of \( q_i \) that exists in the conversation history \( q_{1:i-1} \). As described above, we can train QuReTeC if we have a gold standard resolution \( q_i^* \) for each \( q_i \). Obtaining such special-purpose gold standard resolutions is cumbersome compared to almost readily available general-purpose passage relevance labels for \( q_i \). We propose a distant supervision method to generate labels to train QuReTeC. Specifically, we simply replace \( q_i^* \) with a relevant passage \( p_{q_i}^* \) in Equation (2) to extract the set of relevant resolution terms \( E^*_i(q_i) \). Table 1 illustrates this idea with an example dialogue and the relevant passage to the current turn query. The gold standard resolution terms extracted with this distant supervision procedure for this example are (Saosin, first, band).

Intuitively, the above procedure is noisy and can result in adding terms to \( E^*_i(q_i) \) that are non-relevant, or adding too few relevant terms to \( E^*_i(q_i) \). Nevertheless, we experimentally show in Section 6.2 that this distant supervision signal can be used to substantially reduce the number of human-curated gold standard resolutions required for training QuReTeC.

The distant supervision method we describe here makes QuReTeC more generally applicable than other supervised methods such as the method in Elgohary et al. [15] that can only be trained with gold standard query resolutions. This is because, apart from manual annotation, query-passage relevance labels can be potentially obtained at scale by using click logs [19], or weak supervision [12].

5 EXPERIMENTAL SETUP

5.1 Research questions

We aim to answer the following research questions:

(RQ1) How does the QuReTeC model perform compared to other state-of-the-art methods?

(RQ2) Can we use distant supervision to reduce the amount of human-curated training data required to train QuReTeC?
(RQ3) How does QuReTeC’s performance vary depending on the turn of the conversation?
For all the research questions listed above we measure performance in both an intrinsic and an extrinsic sense. Intrinsic evaluation measures query resolution performance on term classification. Extrinsic evaluation measures retrieval performance at both the initial retrieval and the reranking steps.

5.2 Datasets
5.2.1 Extrinsic evaluation – retrieval. The TREC CAsT dataset is a multi-turn passage retrieval dataset [11]. It is the only such dataset that is publicly available. Each topic consists of a sequence of queries. The topics are open-domain and diverse in terms of their information need. The topics are curated manually to reflect information-seeking conversational structure patterns. Later turn queries in a topic depend only on the previous turn queries, and not on the returned passages of the previous turns, which is a limitation of this dataset. Nonetheless, the dataset is sufficiently challenging for comparing automatic systems, as we will show in Section 6.1.3. Table 3 shows statistics of the dataset. The original dataset consists of 30 training and 50 evaluation topics. 20 of 50 topics in the evaluation set were annotated for relevance by NIST assessors on a 5-point relevance scale. We use this set as the TREC CAsT test set. The organizers also provided a small set of judgements for the training set, however we do not use it in our pipeline. The passage collection is the union of two passage corpora, the MS MARCO [28] (Bing), and the TREC CAR [14] (Wikipedia passages).

5.2.2 Intrinsic evaluation – query resolution. The original QuAC dataset [7] contains dialogues on a single Wikipedia article section regarding people (e.g., early life of a singer). Each dialogue contains up to 12 questions and their corresponding answer spans in the section. It was constructed by asking two crowdworkers (a student and a teacher) to perform an interactive dialogue about a specific topic. Elgohary et al. [15] crowdsourced question resolutions for a subset of the original QuAC dataset [7]. All the questions in the dev and test splits of [15] have gold standard resolutions. We use the dev split for early stopping when training QuReTeC and evaluate on the test set. When training with gold supervision (gold standard query resolutions), we use the train split from [15], which is a subset of the train split of [7]; all the questions therein have gold standard resolutions. Since QuAC is not a passage retrieval collection, in order to obtain distant supervision labels (Section 4.3), we use a window of 50 characters around the answer span to extract passage-length texts, and we treat the extracted passage as the relevant passage. When training with distant labels, we use the part of the train split of [7] that does not have gold standard resolutions.

The TREC CAsT dataset [11] also contains gold standard query resolutions for its test set. However, it is too small to train a supervised query resolution model, and we only use it as a complementary test set.

The two query resolution datasets described above have three main differences. First, the conversations in QuAC are centered around a single Wikipedia article section about people whereas the conversations in CAsT are centered around an arbitrary topic. Second, the answers of the QuAC questions are spans in the Wikipedia section whereas the CAsT queries have relevant passages that originate from different Web resources besides Wikipedia. Third, later turns in QuAC do depend on the answers in previous turns, while in CAsT they do not (Section 3.1). Interestingly, in Section 6.1 we demonstrate that despite these differences, training QuReTeC on QuAC generalizes well to the CAsT dataset.

Table 4 provides statistics for the two datasets.5 First, we observe that the QuAC dataset is much larger than CAsT. Also, QuAC has a larger number of terms on average than CAsT (~97 vs ~40) and a larger negative-positive ratio (~20:1 vs ~40:1). This is because in QuAC the answers to the previous turns are included in the conversation history whereas in CAsT they are not. For this reason, we expect query resolution on QuAC to be more challenging than on CAsT.

5.3 Evaluation metrics
5.3.1 Extrinsic evaluation – retrieval. We report NDCG@3 (the official TREC CAsT evaluation metric), Recall, MAP, and MRR at rank 1000. We also provide performance metrics averaged per turn to show how retrieval performance varies across turns.

We report on statistical significance with a paired two-tailed t-test. We depict a significant increase for $p < 0.01$ as $\uparrow$.

5.3.2 Intrinsic evaluation – query resolution. We report on Micro-Precision (P), Micro-Recall (R) and Micro-F1 (F1), i.e., metrics calculated per query and then averaged across all turns and topics. We ignore queries that are the first turn of the conversation when calculating the mean, since we do not predict term labels for those.

5 Note that the first turn in each topic does not need query resolution because there is no conversation history at that point and thus the query resolution CAsT test has 20 (the number of topics) fewer queries than in Table 3.
In order to perform intrinsic and extrinsic evaluation by comparing against Table 4: Query resolution datasets statistics. In the Split column, we indicate the where the positive term labels originate from: either gold (gold standard resolutions) or distant (Section 4.3).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Split</th>
<th>#Queries</th>
<th>#Terms (per query)</th>
</tr>
</thead>
<tbody>
<tr>
<td>QuAC</td>
<td>Train (gold)</td>
<td>20,181</td>
<td>97.96 ± 61.02</td>
</tr>
<tr>
<td></td>
<td>Train (distant)</td>
<td>31,538</td>
<td>99.78 ± 62.36</td>
</tr>
<tr>
<td></td>
<td>Dev (gold)</td>
<td>2,196</td>
<td>95.49 ± 58.79</td>
</tr>
<tr>
<td></td>
<td>Test (gold)</td>
<td>3,373</td>
<td>96.96 ± 59.24</td>
</tr>
<tr>
<td>CAS(T)</td>
<td>Test (gold)</td>
<td>153</td>
<td>39.97 ± 17.97</td>
</tr>
</tbody>
</table>

### 5.4 Baselines

We perform intrinsic and extrinsic evaluation by comparing against a number of query resolution baselines. Next, we provide a detailed description of each baseline:

- **Original** This method uses the original form of the query. We explore different variations for constructing \( q_i \): (1) current turn only (cur), (2) current turn expanded by the previous turn (cur+prev), (3) current turn expanded by the first turn (cur+first), and (4) all turns.

- **RM3** [1] A state-of-the-art unsupervised pseudo-relevance feedback model. RM3 first performs retrieval and treats the top-\( n \) ranked passages as relevant. Then, it estimates a query language model based on the top-\( n \) results, and finally adds the top-\( k \) terms to the original query. As with Original, we report on different variations for constructing the query: cur, cur+prev, cur+first and all turns. In order to apply RM3 for query resolution we append the top-\( k \) terms to the original query \( q_i \) to obtain \( \hat{q}_i \).

- **NeuralCoref** [7] A coreference resolution method designed for chatbots. It uses a rule-based system for mention detection and a feed-forward neural network that predicts coreference scores. We perform coreference resolution on the conversation history \( q_{t:i−1} \) and the current turn query \( q_i \). The output \( \hat{q}_i \) consists of \( q_i \) and the predicted terms in \( q_{t:i−1} \) where terms in \( q_i \) refer to.

- **BiLSTM-copy** [15] A neural sequence to sequence model for query resolution. It uses a BiLSTM encoder and decoder augmented with attention and copy mechanisms and also a coverage loss [40]. It initializes the input embeddings with pretrained GloVe embeddings. Given \( q_{t:i−1} \) and \( \hat{q}_i \), it outputs \( \hat{q}_i \). It was optimized on the QuAC gold standard resolutions.

#### 5.4.1 Intrinsic evaluation – query resolution.

In order to perform intrinsic evaluation on the aforementioned baselines, we take the query resolution they output (\( \hat{q}_i \)) and apply Equation (2) by replacing \( q_i^* \) with \( \hat{q}_i \) to obtain the set of predicted resolution terms.

### 5.4.2 Extrinsic evaluation – initial retrieval.

Here, apart from the aforementioned baselines, we also use the following baselines:

- **Nugget** [18]. Extracts substrings from the current and previous turn queries to build a new query for the current turn.

- **QCM** [47]. Models the edits between consecutive queries and the results list returned by the previous turn query to construct a new query for the current turn.

- **Oracle** Performs initial retrieval using the gold standard resolution query. Released by the TRECCAsT organizers.

### 5.4.3 Extrinsic evaluation – reranking.

Since developing specialized rerankers for multi-turn passage retrieval is not the focus of this paper, we evaluate the reranking step using ablation studies. For reference, we also report on the performance of the top-ranked TRECCAsT 2019 systems [11]:

- **TREC-top-auto** Uses an automatic system for query resolution and BERT-large for reranking.

- **TREC-top-manual** Uses the gold standard query resolution and BERT-large for reranking.

### 5.5 Implementation & hyperparameters

**Multi-turn passage retrieval** We index the TREC CAS(T) collections using Anserini with stopword removal and stemming. In the initial retrieval step (section 3.2.1) we retrieve the top 1000 passages using QL with Dirichlet smoothing (we set \( \beta = 2500 \)). We use the default value for the fusion parameter \( k = 60 \) [8] in Eq. (1). In the reranking step (section 3.2.2) we use a PyTorch implementation of BERT for retrieval [25]. We use the bert base uncased pretrained BERT model. We fine-tune the BERT reranker with MSMARCO passage ranking dataset [2]. We train on 100K randomly sampled training triples from its training set and evaluate on 100 randomly sampled queries of its development set. We use the Adam optimizer with a learning rate of 0.001 except for the BERT layers for which we use a learning rate of 3e-6. We apply dropout with a probability of 0.2 on the output linear layer. We apply early stopping on the development set with a patience of 2 epochs based on MRR.

**Query resolution** We use the bert large uncased model. We implement QuReTeC on top of HuggingFace’s PyTorch implementation of BERT. [11] We use the Adam optimizer and tune the learning rate in the range [2e-5, 3e-5, 3e-6]. We use a batch size of 4 and do gradient clipping with the value of 1. We apply dropout on the term classification layer and the BERT layers in the range [0.1, 0.2, 0.3, 0.4]. We optimize for F1 on the QuAC dev (gold) set.

---

6 Note that given the very small size of the TRECCasT training set we do not compare to more sophisticated yet data-hungry pseudo-relevance feedback models such as [29].
7 https://medium.com/huggingface/state-of-the-art-neural-coreference-resolution-for-chatbots-33023656de90
8 https://nlp.stanford.edu/projects/glove/
9 We use the nugget version that does not depend on anchors text since they are not available in our setting.
10 https://github.com/castorini/anserini
11 https://github.com/huggingface/transformers
Table 5: Intrinsic evaluation for query resolution on the QuAC test set. Cur, prev, first and all refer to using the current, previous, first or all turns respectively.

<table>
<thead>
<tr>
<th>Method</th>
<th>P</th>
<th>R</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original (cur+prev)</td>
<td>22.3</td>
<td>46.4</td>
<td>30.1</td>
</tr>
<tr>
<td>Original (cur+first)</td>
<td>41.1</td>
<td>49.5</td>
<td>44.9</td>
</tr>
<tr>
<td>Original (all)</td>
<td>12.3</td>
<td>100.0</td>
<td>21.9</td>
</tr>
<tr>
<td>NeuralCoref</td>
<td>65.5</td>
<td>30.0</td>
<td>41.2</td>
</tr>
<tr>
<td>BiLSTM-copy</td>
<td>67.0</td>
<td>53.2</td>
<td>59.3</td>
</tr>
<tr>
<td>QuReTeC</td>
<td>71.5</td>
<td>66.1</td>
<td>68.7</td>
</tr>
</tbody>
</table>

Table 6: Intrinsic evaluation for query resolution on the TREC CASt test set. Cur, prev, first and all refer to using the current, previous, first, or all turns respectively.

<table>
<thead>
<tr>
<th>Method</th>
<th>P</th>
<th>R</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original (cur+prev)</td>
<td>32.5</td>
<td>43.9</td>
<td>37.4</td>
</tr>
<tr>
<td>Original (cur+first)</td>
<td>43.0</td>
<td>74.0</td>
<td>54.4</td>
</tr>
<tr>
<td>Original (all)</td>
<td>18.6</td>
<td>100.0</td>
<td>31.4</td>
</tr>
<tr>
<td>RM3 (cur)</td>
<td>35.8</td>
<td>8.3</td>
<td>13.5</td>
</tr>
<tr>
<td>RM3 (cur+prev)</td>
<td>34.6</td>
<td>32.5</td>
<td>33.5</td>
</tr>
<tr>
<td>RM3 (cur+first)</td>
<td>40.9</td>
<td>32.9</td>
<td>36.5</td>
</tr>
<tr>
<td>RM3 (all)</td>
<td>41.5</td>
<td>38.8</td>
<td>40.1</td>
</tr>
<tr>
<td>NeuralCoref</td>
<td>83.0</td>
<td>28.7</td>
<td>42.7</td>
</tr>
<tr>
<td>BiLSTM-copy</td>
<td>51.5</td>
<td>36.0</td>
<td>42.4</td>
</tr>
<tr>
<td>QuReTeC</td>
<td>77.2</td>
<td>79.9</td>
<td>78.5</td>
</tr>
</tbody>
</table>

**Baselines** For RM3, we tune the following parameters: \( n \in \{3, 5, 10, 20, 30\} \) and \( k \in \{5, 10\} \) and set the original query weight to the default value of 0.8. For Nugget, we set \( k_{snippet} = 10 \) and tune \( \delta \in \{0.95, 0.97, 0.99\} \). For QCM, we tune \( \alpha \in \{1.0, 2.2, 3.0\} \), \( \beta \in \{1.6, 1.8, 2.0\} \), \( \epsilon \in \{0.06, 0.07, 0.08\} \) and \( \delta \in \{0.2, 0.4, 0.6\} \). For both Nugget and QCM we use Van Gysel et al. [42]'s implementation. For fair comparison, we retrieve over the whole collection rather than just reranking the top-1000 results. The aforementioned methods are tuned on the small annotated training set of TREC CASt. For query resolution, we tune the greedyness parameter of NeuralCoref in the range \( \{0.5, 0.75\} \). We use the model of BiLSTM-copy released by [15], as it was optimized specifically for QuAC with gold standard resolutions.

**Preprocessing** We apply lowercase, lemmatization and stopword removal to \( q_i, q_{1:i-1} \) and \( q_i \) using Spacy.\(^2\) before calculating term overlap in Equation 2.

6 RESULTS & DISCUSSION

In this section we present and discuss our experimental results.

6.1 Query resolution for multi-turn retrieval

In this subsection we answer (RQ1): we study how QuReTeC performs compared to other state-of-the-art methods when evaluated on term classification (Section 6.1.1), when incorporated in the initial retrieval step (Section 6.1.2) and in the reranking step (Section 6.1.3).

6.1.1 Intrinsic evaluation. In this experiment we evaluate query resolution as a term classification task.\(^{13}\) Table 5 shows the query resolution results on the QuAC dataset. We observe that QuReTeC outperforms all the variations of Original and the NeuralCoref by a large margin in terms of F1, precision and recall – except for Original (all) that has perfect recall but at the cost of very poor precision. Also, QuReTeC substantially outperforms BiLSTM-copy on all metrics. Note that BiLSTM-copy was optimized on the same training set as QuReTeC (see Section 5.5). This shows that QuReTeC is more effective in finding missing contextual information from previous turns.

Table 6 shows the query resolution results on the CASt dataset. Generally, we observe similar patterns in terms of overall performance as in Table 5. Interestingly, we observe that QuReTeC generalizes very well to the CASt dataset (even though it was only trained on QuAC) and outperforms all the baselines in terms of F1 by a large margin. In contrast, BiLSTM-copy fails to generalize and performs worse than Original (cur+first) in terms of F1. NeuralCoref has higher precision but much lower recall compared to QuReTeC.

6.1.2 Query resolution for initial retrieval. In this experiment, we evaluate query resolution when incorporated in the initial retrieval step (Section 3.2.1). We compare QuReTeC to the baseline methods in terms of initial retrieval performance. Table 7 shows the results. First, we observe that QuReTeC outperforms all the baselines by a large margin on all metrics. Also, interestingly, QuReTeC achieves performance close to the one achieved by the Oracle performance (gold standard resolutions). Note that there is still plenty of room for improvement even when using Oracle, which indicates that

\(^{13}\)Note that the performance of Original (cur) is zero by definition when using the current turn only (see Eq. 2). Thus, we do not include it in Tables 5 and 6. Also, RM3 is not applicable in Table 5 since QuAC is not a retrieval dataset.

Table 7: Initial retrieval performance on the TREC CASt test set for different query resolution methods. The retrieval model is fixed (same as in Section 3.2.1). Significance is tested against RM3 (cur+first) since it has the best NDCG@3 among the baselines.

<table>
<thead>
<tr>
<th>Method</th>
<th>Recall</th>
<th>MAP</th>
<th>MRR</th>
<th>NDCG@3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original (cur)</td>
<td>0.438</td>
<td>0.129</td>
<td>0.310</td>
<td>0.155</td>
</tr>
<tr>
<td>Original (cur+prev)</td>
<td>0.572</td>
<td>0.181</td>
<td>0.475</td>
<td>0.235</td>
</tr>
<tr>
<td>Original (cur+first)</td>
<td>0.655</td>
<td>0.214</td>
<td>0.561</td>
<td>0.282</td>
</tr>
<tr>
<td>Original (all)</td>
<td>0.694</td>
<td>0.190</td>
<td>0.552</td>
<td>0.256</td>
</tr>
<tr>
<td>RM3 (cur)</td>
<td>0.440</td>
<td>0.140</td>
<td>0.320</td>
<td>0.158</td>
</tr>
<tr>
<td>RM3 (cur+prev)</td>
<td>0.575</td>
<td>0.200</td>
<td>0.482</td>
<td>0.254</td>
</tr>
<tr>
<td>RM3 (cur+first)</td>
<td>0.656</td>
<td>0.225</td>
<td>0.551</td>
<td>0.300</td>
</tr>
<tr>
<td>RM3 (all)</td>
<td>0.666</td>
<td>0.195</td>
<td>0.544</td>
<td>0.266</td>
</tr>
<tr>
<td>Nugget</td>
<td>0.426</td>
<td>0.101</td>
<td>0.334</td>
<td>0.145</td>
</tr>
<tr>
<td>QCM</td>
<td>0.392</td>
<td>0.091</td>
<td>0.317</td>
<td>0.127</td>
</tr>
<tr>
<td>NeuralCoref</td>
<td>0.565</td>
<td>0.176</td>
<td>0.423</td>
<td>0.212</td>
</tr>
<tr>
<td>BiLSTM-copy</td>
<td>0.552</td>
<td>0.171</td>
<td>0.403</td>
<td>0.205</td>
</tr>
<tr>
<td>QuReTeC</td>
<td>0.754(^\ast)</td>
<td>0.272(^\ast)</td>
<td>0.637(^\ast)</td>
<td>0.341(^\ast)</td>
</tr>
<tr>
<td>Oracle</td>
<td>0.785</td>
<td>0.309</td>
<td>0.660</td>
<td>0.361</td>
</tr>
</tbody>
</table>

\(^\ast\)Note that the performance of Original (cur) is zero by definition when using the current turn only (see Eq. 2). Thus, we do not include it in Tables 5 and 6. Also, RM3 is not applicable in Table 5 since QuAC is not a retrieval dataset.
6.1.3 Query resolution for reranking. In this experiment, we study the effect of QuReTeC when incorporated in the reranking step (Section 3.2.2). We keep the initial ranker fixed for all QuReTeC models. Table 8 shows the results. First, we see that BERT-base improves over the initial retrieval model that uses QuReTeC for query resolution on the top positions (second line). Second, when we fuse the ranked list retrieved by BERT-base and the ranked list retrieval by the initial retrieval ranker using RRF, we significantly outperform BERT-base on all metrics (third line). This shows that the two rankers can be effectively combined with RRF, which is a very simple fusion method that only has one parameter which we do not tune. We also see that our best model outperforms TREC-top-auto on all metrics. Furthermore, by comparing RRF (line 3) to Oracle (line 4) we see that there is still plenty of room for improvement for reranking, which is a clear direction for future work. This also shows that the TREC CAsT dataset is sufficiently challenging for comparing automatic systems. Note that TREC-top-manual uses the gold standard query resolutions and is thereby not directly comparable with the rest of the methods.

6.2 Distant supervision for query resolution

In this section we answer (RQ2): Can we use distant supervision to reduce the amount of human-curated query resolution data required to train QuReTeC? Figure 3 shows the query resolution performance when training QuReTeC under different settings (see figure caption for a more detailed description). For QuReTeC (distant full & gold partial) we first pretrain QuReTeC on distant and then resume training with different fractions of gold. First, we see that QuReTeC performs competitively with BiLSTM-copy even when it does not use any gold resolutions (distant full). More importantly, when only trained on distant, QuReTeC performs remarkably well in the low data regime. In fact, it outperforms BiLSTM-copy (trained on gold) even when using a surprisingly low number of gold standard query resolutions (200, which is ~1% of gold). Last, we see that as we add more labelled data, the effect of distant supervision becomes smaller. This is expected and is also the case for the model trained on QuAC train (gold).

In conclusion, our distant supervision method can be applied on different encoders, we performed an additional experiment where we replaced BERT with a simple BiLSTM as the encoder in QuReTeC. Similarly to the previous experiment, we observed a substantial increase in F1 when retraining with 2K gold standard resolutions (+12 F1) over when only using gold resolutions.

In conclusion, our distant supervision method can be used to substantially decrease the amount of human-curated training data required to train QuReTeC. This is especially important in low resource scenarios (e.g. new domains or languages), where large-scale human-curated training data might not be readily available.

6.3 Analysis

In this section we perform analysis on QuReTeC when trained with gold standard supervision.

6.3.1 Query resolution performance per turn. Here we answer (RQ3) by analyzing the robustness of QuReTeC at later conversation turns.
in resolving “him” → [“Alexander”, “Graham” “Bell”] but misses the connection between “this” and “dehusking machine”.

Extrinsic – initial retrieval Table 10 shows an example from the CAS-T test set where QuReTeC succeeds and RM3 (cur-first), the best performing baseline for initial retrieval, fails. First, note that a topic change happens at Q7 (the topic changes from general real-time databases to Firebase DB). We observe that QuReTeC predicts the correct terms, and a relevant passage is retrieved at the top position. In contrast, RM3 (cur-first) fails to detect this topic change and therefore an irrelevant passage is retrieved at the top position that is about real-time databases on mobile apps but not about Firebase DB.

7 CONCLUSION

In this paper, we studied the task of query resolution for conversational search. We proposed to model query resolution as a binary term classification task: whether to add terms from the conversation history to the current turn query. We proposed QuReTeC, a neural query resolution model based on bidirectional transformers. We proposed a distant supervision method to gather training data for QuReTeC. We found that QuReTeC significantly outperforms multiple baselines of different nature and is robust across conversation turns. Also, we found that our distant supervision method can substantially reduce the required amount of gold standard query resolutions required for training QuReTeC, using only query-passage relevance labels. This result is especially important in low resource scenarios, where gold standard query resolutions might not be readily available.

As for future work, we aim to develop specialized rankers for both the initial retrieval and the reranking steps that incorporate QuReTeC in a more sophisticated way. Also, we want to study how to effectively combine QuReTeC with text generation query resolution methods as well as pseudo-relevance feedback methods. Finally, we aim to explore weak supervision signals for training QuReTeC [12].