Datasets and Benchmarks for Task-Oriented Log Dialogue Ranking Task

Xinnuo Xu\textsuperscript{1}, Yizhe Zhang\textsuperscript{2}, Lars Liden\textsuperscript{2}, Sungjin Lee\textsuperscript{3}\textsuperscript{*}

\textsuperscript{1}The Interaction Lab, Heriot-Watt University, Edinburgh
\textsuperscript{2}Microsoft Research, Redmond, WA, USA
\textsuperscript{3}Amazon Alexa AI, Redmond, WA, USA

xx6@hw.ac.uk, Yizhe.Zhang, Lars.Liden@microsoft.com, sungjinl@amazon.com

Abstract

Although the data-driven approaches of some recent bot building platforms make it possible for a wide range of users to easily create dialogue systems, those platforms don’t offer tools for quickly identifying which log dialogues contain problems. Thus, in this paper, we (1) introduce a new task, log dialogue ranking, where the ranker places problematic dialogues higher (2) provide a collection of human-bot conversations in the restaurant inquiry task labelled with dialogue quality for ranker training and evaluation (3) present a detailed description of the data collection pipeline, which is entirely based on crowd-sourcing (4) finally report a benchmark result of dialogue ranking, which shows the usability of the data and sets a baseline for future studies.

Index Terms: dialogue ranking, dialogue quality, language resource, dialogue system

1. Introduction

Task-oriented dialogue systems provide a natural interface to accomplish various daily-life tasks such as restaurant finding and flight booking. Data-driven approaches offered by common bot building platforms (e.g. Google Dialogflow, Amazon Alexa Skills Kit, Microsoft Bot Framework) make it possible for a wide range of users to easily create dialogue systems with a limited amount of data in their domain of interest.

Figure 1: Operational loop of dialogue system development.

Typically, the development process of a dialogue system based on data-driven approaches \cite{1, 2} goes around an operational loop in Fig 1: (1) The cycle begins with a developer creating a training dataset with seed dialogues. (2) A dialogue system is trained and deployed. (3) Real users interact with the system and generate log dialogues. (4) The developer reviews the logs to identify which log dialogues contain problems. (5) The developer updates the training dataset to fix the problems. (6) The cycle repeats from step 2). Of all steps, (4) is the most significant in slowing down the loop, because of the sheer volume of log dialogues that can be generated and the need to manually inspect each. Thus, it is essential to support tools that help developers quickly identify problematic log dialogues.

To achieve this goal, we introduce a new task, task-oriented log dialogue ranking whose goal is to place problematic dialogues higher in the rank trained on a small number of labelled data. There are prior studies on dialogue quality assessment. \cite{3, 4} investigate automatic ways of dialogue evaluation. \cite{5, 6, 7, 8, 9} focus on dialogue breakdown detection during the dialogue. \cite{10, 11} estimate dialogue quality to improve dialogue policy. However, none has directly dealt with ranking in terms of the quality of task-based dialogues. For automatic error handling, \cite{12, 13, 14} examined different types of errors. Whereas, for our task, it suffices to rank problematic dialogues higher than normal ones so that developers may take appropriate measures based on the goal of their bots, e.g. adding new slots or system actions to handle previously uncovered scenarios.

In the rest of this paper, we present a crowd-sourcing based data collection pipeline using Amazon Mechanical Turk (AMT) in Section 2 and describe the collected data in Section 3. To demonstrate the utility of the data collected, we report a benchmark result for the dialogue ranking task in Section 4 and 5.

2. Data Collection Set-up

To collect log dialogues, we deployed the Pydial restaurant bot\textsuperscript{1} via the Amazon Mechanical Turk (AMT) platform.\textsuperscript{2} The task for the bot is to search for restaurants based on user’s requirements in a multi-turn natural language communication. We ask turkers to find restaurants that satisfy automatically generated requirements, such as food type, location and price range, by chatting with the restaurant bot. Figure 2 shows an example task.

To make the conversation natural and keep users engaged, we encourage turkers to speak in natural utterances by 1) detecting one token user utterance and asking the turker to rephrase the answer with a natural sentence, 2) closing the AMT session and eliminating the conversation if the turker refuses to use natural response for three times. To increase the diversity of the dialogue collection we do not allow any turkers to carry out more than 20 dialogues in total.

At the end of each task, turkers are required to answer a questionnaire including two questions, 1) whether they found restaurants satisfying their requirements, 2) label contextually unnatural turns in the conversation they experienced. The interface of the questionnaire is shown in Figure 3. To control the quality of turker’s judgements we (1) ask turkers to rephrase

\textsuperscript{1}http://www.candial.org/pydial/

\textsuperscript{2}We use the data collection toolkit offered by ParlAI http://www.parl.ai/static/docs/tutorial_mturk.html.
their feedback if their judgements follow incorrect formats, (2) automatically collect unnatural turns in which the bot fails to understand the turner’s utterance and require turers to regener-ate their judgements that miss the seed unnatural turns, (3) close the AMT session and eliminate the conversation if the turker re-fuses to offer qualified judgements for more than three times.

3. Dataset Description
From the collected dialogues, we label successful dialogues without any contextually unnatural turns as 0 (normal dialogue), and the rest as 1 (problematic dialogue). In general, we collected 1470 normal dialogues and 1896 problematic dialogues, with average dialogue length as 5.87 turns, 8.28 turns respectively. Examples are shown in Table 1.

Fig 4, 5, 6 show the distribution of the number of sentences, number of tokens in user utterances, utterance repetition rate of the problematic and normal dialogues respectively. The repetition rate is calculated per dialogue by 1−Nn/N, where Nn and N denote the number of unique sentences and the number of sentences in a dialogue respectively. It is obvious that normal dialogues are more efficient (shorter dialogue length and lower repetition rate) with higher user engagement (longer user utterances). We randomly split the corpus into train/validation/test sets (normal dialogues 1321/100/200; problematic dialogues 1029/100/200).

4. Benchmarks
A dialogue ranker aims to assign higher scores to problematic dialogues than normal ones so that developers may quickly identify problematic dialogues in the ranked list of log dialogues. Formally, given a dialogue D, where,

\[ D = \{S_1, U_1, S_2, \ldots , U_{n-1}, S_n\} \]

a dialogue ranker \( \psi \) produces a score of D being problematic where \( S_i \) and \( U_i \) are the system and user utterance in \( i^{th} \) turn, respectively. 3To train the dialogue ranker \( \psi \), we formulate the ranking task as binary classification where problematic and normal dialogues correspond to positive and negative classes, respectively. We optimize the cross-entropy objective:

\[ L_{\text{sent}} = \frac{1}{N} \sum_{i=1}^{N} [y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)] \]

where \( \hat{y}_i = 1/\left(1+\exp(-\psi(D_i))\right) \) and \( y_i = 1 \) for positive class and 0 otherwise. We apply two document classification models for the dialogue ranking task.

4.1. BiLSTM+MLP:
Fig 7 shows the structure of the BiLSTM+MLP. We first use a bidirectional Long short-term memory (Bi-LSTM) [15] to encode utterances in dialogue D:

\[ D_{\text{emb}} = \{S_{i\text{emb}}, U_{i\text{emb}}, S_{i+1\text{emb}}, \ldots , U_{n-1\text{emb}}, S_n\mb} \]

where \( S_{i\text{emb}} = \text{Bi-LSTM}(S_i) \) and \( U_{i\text{emb}} = \text{Bi-LSTM}(U_i) \). After that, we apply a Feed-Forward Network (FFN) followed by a sigmoid activation to each pair of system-user utterances to measure the consistency of adjacency pairs:

\[ A_i = \text{sigmoid}(FFN_{\text{pred}}([X_{i\text{emb}}, Y_{i\text{emb}}])) \]

where \( (X_{i\text{emb}}, Y_{i\text{emb}}) \) is either \( (S_{i\text{emb}}, U_{i\text{emb}}) \) or \( (U_{i\text{emb}}, S_{i+1\text{emb}}) \). Finally, the ranker \( \psi \) produces a ranking score for the dialogue based on the consistency scores:

\[ \psi(D) = \text{sigmoid}(FFN_{\text{pred}}([A_1, \ldots , A_{n-1}])) \]

Note that the FFNs in Eq 1 and Eq 2 are using two different sets of parameters.

4.2. HierAttn:
Inspired by [16], we adopt Hierarchical Attention (HierAttn) for the dialogue ranking. Fig 8 shows the structure for the HierAttn. We first use a Bi-LSTM to encode each utterance. Then the embedding for an utterance is calculated as a weighted sum of the contextual word representations. For example, the embedding \( S_{i\text{emb}} \) for the system utterance in the \( i^{th} \) turn is:

\[ u_{it} = \text{tanh}(W_{u} h_{it} + b_{u}) \]

\[ \alpha_{it} = \frac{\exp(u_{it}^\top a_w)}{\sum_{j} \exp(u_{ij}^\top a_w)} \]

\[ S_{i\text{emb}} = \sum_{t} \alpha_{it} h_{it} \]

where \( h_{it} = [\tilde{h}_{it}, \tilde{h}_{it}^\top] \) is the word annotations, a concatenation of the hidden states of the Bi-LSTM. \( W_u \) and \( b_u \) are parameters for the non-linear transition. Word context vector \( a_w \) is randomly initialized and jointly learned during training.

We introduce the same structure with different parameters to map the utterance embeddings to a document embedding:

\[ u_i = \text{tanh}(W_s h_i + b_s) \]

\[ \alpha_i = \frac{\exp(u_i^\top a_s)}{\sum_{j} \exp(u_j^\top a_s)} \]

\[ v = \sum_{i} \alpha_i h_i \]

One turn consists of a pair of system and user utterances.
where the document embedding \(v\) is the weighted sum of the contextual utterance representations that is a concatenation of the hidden states of the sentence-level Bi-LSTM. \(W_s\) and \(b_s\) are parameters for the non-linear transition and \(u_d\) is the document context vector. \(\psi(D)\) is calculated as \(\text{sigmoid}(\text{FFN}(v))\).

5. Experiments and results

We test the two dialogue rankers on the 400-instance balanced test set. To get stable and convincing results, we randomly sample balanced training sets (1000 problematic dialogues and 1000 normal dialogues) for 10 times and train dialogue rankers on each set. Then we report average performance for each ranker over 10 models.

For both models, the vocabulary size is 150 and the dimension of word embeddings is 64. All bidirectional RNNs are one layer LSTM of size 64. Adam optimizer [17] is used to update the parameters with a learning rate 0.001. BiLSTM+MLP is using the publicly available AllenNLP\(^4\) framework [18]. HierAttn is implemented on Keras\(^5\).

We use ranking metrics Precision at \(k\) (P@K) for evaluation, which corresponds to the number of problematic dialogues in the top \(k\) ranked options.

An alternative avenue can be to leverage off-the-shelf dialogue quality classifiers which are trained on open-domain corpora such as dialogue breakdown detection challenge systems in DSTC6 [19].\(^6\) In fact, if off-the-shelf models work

---

\(^{4}\)https://allennlp.org

\(^{5}\)The HierAttn is based on code https://github.com/richliiao/textClassifier

\(^{6}\)https://github.com/dbd-challenge/dbdc3
One normal example in the labelled corpus

SYS: hello, welcome to the restaurant finding system. how may i help you?
USER: hi i’m looking for a restaurant.
SYS: what part of town do you have in mind?
USER: the east part of town please.
SYS: what kind of food would you like?
USER: i’d like chinese food please.
SYS: yu garden is a nice place, it serves Chinese food and it is in the east part of town.
USER: what is the phone number?
SYS: yu garden is a nice place their phone number is 01223248882.
USER: what is the address?
SYS: yu garden is a nice place their address is 529 newmarket road fen ditton.
USER: thank you
SYS: thank you, goodbye.

One problematic example in the labelled corpus

SYS: hello, welcome to the restaurant finding system. how may i help you?
USER: hello i would really like to eat some cheap indian tonight
SYS: what part of town do you have in mind?
USER: i would really like to eat in the centre of town
SYS: i am sorry but there is no place, in the expensive price range, serving corsica food and in the area of centre.
USER: i would really like so indian thats cheap in the centre of town
SYS: the gandhi is a nice place, it is in the cheap price range, it is in the centre of town and it serves indian food.
USER: thats great, have you got a contact number?
SYS: what part of town do you have in mind?
USER: the centre of town
SYS: eifes restaurant is a nice place, it is in the centre of town, it is in the moderate price range and it serves turkish food.
USER: i want to eat at the gandhi have you got the address?
SYS: the gandhi is a nice place their address is 72 regent street city centre.
USER: thank you, have you got a phone number for the gandhi?
SYS: the gandhi is a nice place their phone number is 0122335942.
USER: thanks
SYS: the gandhi is a nice place, it serves mexican food, it is in the expensive price range and it is in the south part of town.
USER: quit
SYS: thank you, goodbye.

Table 1: One normal and one problematic examples in the labelled corpus.

<table>
<thead>
<tr>
<th>Precision@</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBDC3</td>
<td>.700</td>
<td>.800</td>
<td>.800</td>
<td>.825</td>
</tr>
<tr>
<td>BiLSTM+MLP</td>
<td>1.00</td>
<td>.975</td>
<td>.953</td>
<td>.937</td>
</tr>
<tr>
<td>HierAttn</td>
<td>1.00</td>
<td>.985</td>
<td>.943</td>
<td>.937</td>
</tr>
</tbody>
</table>

Table 2: Evaluation result: precision@k, corresponds to the number of problematic dialogues in the top k ranked options.

5.1. Results and Discussion

• DBDC3 vs. Supervised models: Table 2 shows that the ranker that is trained on existing open-domain corpora does not work as well as the models specifically trained on a task-oriented corpus, indicating that the use of domain-relevant data is crucial in obtaining high performance.

• BiLSTM+MLP vs. HierAttn: Table 2 shows that except for the P@30, HierAttn is either same with or outperforms BiLSTM+MLP. Both models achieve over 90% precision in top ranked dialogues (top50). It proves that rankers trained on the small dialogue set collected by the pipeline described in the Section 2 are helpful for developers to discriminate problematic dialogues in practical. However, the accuracy drops when k is over 100. It shows the challenges of the task-oriented dialogue ranking task.

6. Conclusions

In this paper, we introduced a new log dialogue ranking task and described an entirely crowd-sourcing based data collection pipeline. The initial benchmark results show the utility of the collected corpus in training a ranker with high performance. To facilitate further research on the log dialogue ranking task, we release the collected data and the software at https://github.com/XinnuoXu/DRank. Our future work includes exploring unsupervised data augmentation techniques to effectively reduce the cost of data collection.
7. References


