Modeling Conversational Structure to improve Conversational Question Answering

Final Report

Yiming Yan
University of Southern California
Los Angeles, CA
yanyimin@usc.edu

Hexiang Hu
University of Southern California
Los Angeles, CA
hexiangh@usc.edu

Abstract

Students often gain knowledge by asking questions via conversation. In a conversation of two people asking and answering questions based on a context, although the questions can be very vague, human can still understand the vague question given the conversation history. In this project, we work on the task Conversational Question Answering (CoQA). We want to understand the question and context better by implicitly and explicitly modeling the structure of dialogue history. We found dialogue history aware context representation is very useful for answer prediction, while the history aware question embedding did not affect the overall performance. Our best model achieved 76.9 F1 score on development set.

1 Introduction

Machine Comprehension (MC) becomes an active field in machine learning recently with the development of large-scale datasets (Rajpurkar et al., 2016; Nguyen et al., 2016). Among many proposed MC tasks, question answering (QA) is one of the most fundamental and straightforward evaluation of MC. To this end, we would like to study QA, and evaluate the role of explicit and implicit structure learning in helping high-level reasoning. Specifically we use the recent conversational QA (CoQA) (Reddy et al., 2018) task as a testbed. In CoQA task, in order to predict the answer, we need to understand the question based on the dialogue history, and find the answer conditioned on the question and the context paragraph. In this project, we hope to discover the usefulness of structure understanding in CoQA. Specifically, we want to learn the structure implicitly and explicitly from dialogue history to understand the question better.

In the domain of this report, we first describe the related works in Section 2 and then introduce the general framework and concrete details of our proposed method in Section 3. We describe the experimental setups, the baseline systems we used and discuss the empirical performance of the CoQA task in Section 4.

2 Related Works

Traditional QA

In the QA task, given contexts \( D = \{D_1, D_2, \cdots, D_N\} \) and a question \( q \), a machine should answer the question as \( a \) based on the contexts, where \( N \) is the number of given contexts. The format of the answer \( a \) could be different, such as multiple choices (Richardson et al., 2013), span of text (Rajpurkar et al., 2016), or free-form answer (Nguyen et al., 2016), etc.

By taking the advantages of self-attention and local convolution (Vaswani et al., 2017), the performance of machine learning algorithms have already on a par with, or even exceeded human performance on some datasets (Seo et al., 2016; Chen et al., 2017; Yu et al., 2018).

We categorize QA modeling methods as abstractive QA and extractive QA. In abstractive QA, the answer texts are generated from vocabulary conditioned on the context paragraph and the question (Chen et al., 2016; Wang and Jiang, 2016; Tan et al., 2017). One problem of answer generation is the ground truth answer can contain rare words that are not in the vocabulary, and the model will never be able to answer the question correctly. (See et al., 2017) proposed pointer generator network (Ptr-Gen Net) inspired by (Vinyals et al., 2015), where the answer generator can also generate words from original context paragraph. Due to the huge search space of possible answers and the difficulty of evaluation for abstractive QA, the
problem is commonly formulated as answer extraction (Seo et al., 2016; Chen et al., 2017; Wang et al., 2018), where the answer is considered as a span of context. In this setting, the problem is then simplified as predicting the start and end position of the answer span.

Conversational QA Students often gain information by asking questions via conversation. Given this fact, some conversational QA tasks have been proposed (Choi et al., 2018; Reddy et al., 2018). Different from the traditional QA tasks, the questions from conversational QA are more likely to be correlated (Yatskar, 2018). For instance, Fig. 1 shows a sample conversation from CoQA dataset, where the questions are highly related to the previous conversation. In this case, structure of dialogue history might be very helpful for question understanding and answer prediction. (Huang et al., 2018) included dialogue flow in context embedding as implicit structure modeling for better text understanding. In this project, we aim to utilize implicit and explicit structure modeling for better question understanding.

3 Methods

The overall approach of our method involves three components: (1) learning word-level representation of the context passage and (2) learning conversation history aware question representation (3) learning prediction model based on passage and question features.

We begin by introducing the context representation model (Huang et al., 2018). Based on this, we augment the base context model with the recurrent module to obtain implicit conversation history aware question representation, for better inference. Next, we perform explicit modeling to aggregate word-level information propagation (Vaswani et al., 2017) over the conversational history, such that a question representation that is capable of learning co-reference and entity relationship is obtained.

3.1 Context Representation Model

Our method is in general belonging to the category of comprehension model (Chen et al., 2017), which leverages the passage context to extract answer span for answering the questions. Here, a model is presented with a passage context \( w_1, ..., w_j \) with \( j \) words as well as a conversation history \( Q \) containing questions \( \{q_1, q_2, ..., q_i\} \). For answering each question, a question aware context representation is build as the cross attended question word representation, in concatenation to the original word embedding \( w_j \) in the passage (same as (Reddy et al., 2018), referred as aligned question embedding there).

Based on this, we further use the context flow proposed by (Huang et al., 2018) for our context representation. The key idea is to build a information flow that connects the context representation of each question turn \( \{q_1, q_2, ..., q_i\} \), which is denoted as \text{flow}. As shown in Figure 2, there are two directions of the information propagation. Each row contains the aforementioned word representations, which is first performed \text{integration} to propagate word information within the context passage. Then a \text{flow} operation is performed to propagate information from context words among different question turns, to leverage conversational information. Here, for both \text{flow} and \text{integration} operation, we used a bidirectional long-short term memory network (BiLSTM), in consistency to (Huang et al., 2018).

As a result, we denote the final context repre-
representation as \( C = \{c_1, \ldots, c_i\} \), which contains contextual word features.

### 3.2 Implicit Question Representation

Now we introduce our question representation networks. We begin by describing a implicit question representation for conversational history, which integrate the word-level information of a question and the conversational history into a vectorized representation.

The details are shown as Figure 3. First of all, the conversation history is represented as word-wise feature vector. We first aggregate the question features by aggregating those word vectors with a bidirectional LSTM on the question level. Then those question vectors are aggregated with a high-level LSTM, in the order of each question turn, to finally obtain the history aware question representation \( q \). This representation is then used for prediction.

### 3.3 Explicit Question Representation

Besides the aforementioned implicit question representation, where the structure of conversation history is first aggregated question wise and then propagated along conversation turns, now we introduce a different variant where word-level information is allowed to propagate from the history to the words in current question.

In particular, we leverage the Transformer (Vaswani et al., 2017) architecture that is widely used in recent machine translation and language modeling tasks. The key idea is that: First, a transformer encoder is individually applied on the conversational history and also the current question; Next, two sets of linear transformations, key transformation and query transformation is used for obtaining attention from query words (current question) to key words (historical words in previous conversations), such that each word in the current question is contextualized to any words in its previous conversational history. Finally, these contextual question word embedding are aggregated together with a self attention layer to generate the question representation \( q \). We refer
3.4 Prediction Model

As mentioned before, our model is de facto a reading comprehension model based on the framework of (Chen et al., 2017). To be concise, aforementioned procedure is applied to extract contextual word features $c_i \in C$ for the passage, as well as the conversation history aware question representation of the current question $q \in Q$. Finally, during the inference process, these two type of features are bilinearly combined to produce the score for current position of the context being a start span ($s$) or an end span ($e$).

\[ P_s(i) \propto \exp (c_i \cdot W_s \cdot q) \]  
\[ P_e(i) \propto \exp (c_i \cdot W_e \cdot q) \]

The predicted span is then used as the answer for the current question.

4 Experiments

In this section, we will describe the experiments setup and compare the performance of our baseline models Seq2Seq, PG-Net, DrQA and the combined pipeline model (Sec. 4.3).

4.1 Setups

We implement all baseline approaches and evaluation metrics based on the original implemented from (Reddy et al., 2018). It contains Seq2Seq and PG-Net models based on OpenNMT toolkit. We also adopted the implementation of DrQA model from (Chen et al., 2017). Similar to SQuAD (Rajpurkar et al., 2016), two evaluation measures, i.e., macro-average F1 score of word overlap, as well as the exact match (EM) score are used. The macro-average F1 is the main evaluation measure.

4.2 Implementation details

We employed pretrained 300 dimension GloVe (Pennington et al., 2014) embedding are used as word representation, in conjunction to the POS tags and NER tags obtained with SPaCy. For the approaches with Flow, we further employed two sentence representations, i.e., CoVE (McCann et al., 2017) and ELMo (Peters et al., 2018). During training, we use Adam optimizer (Kingma and Ba, 2014).

In the following of this section, we refer to the basic approach as FlowQA, which is similar to the one presented in (Huang et al., 2018). For the models with implicit or explicit modeling conversation history, we refer them as FlowQA + Implicit or FlowQA + Explicit

4.3 Baseline Systems

In this section, we introduce a few baseline models we developed for the task of CoQA (Please refer to the mid-term report or (Reddy et al., 2018) for complete details). Generally, we followed the settings described in (Reddy et al., 2018) and implemented methods of three major categories — conversational models (Sutskever et al., 2014), reading comprehension models (Chen et al., 2017) and their combinations as our baseline approaches. They are listed as what follows:

**Seq2Seq (Sutskever et al., 2014).** Seq2Seq is used to generate free-form answers conditioned on the context and the question. On bidirectional-LSTM are used as the encoder, which sequentially encodes the word features from context passage and question (with 2 previous questions). Attention mechanism from (Bahdanau et al., 2014) is used in this encoder. Then a decoder is further used for generating free-form text answer to the questions.

**PGNet (See et al., 2017).** PGNet is a hybrid between the Seq2Seq baseline and pointer networks (Vinyals et al., 2015). The main idea is that, during the decoding procedure, an additional gating mechanism is used to determine whether to copy words from the context or predict words from the answer vocabulary.

**DrQA (Chen et al., 2017).** DrQA is a reading comprehension model. Different from previous two approach with the abstractive nature, this approach is purely extractive. Similar to our presented approach, this approach follows the three stage prediction framework, except that there is no question representation with history and context flow.

**Pipeline.** Pipeline model is a straight-forward extension to above baselines is to combine them together. In this model, the output of the comprehension model (DrQA) is then used for a PGNet model to generate free-from answers.
4.4 Results and Analysis

The performance on development set are shown in Table 1.

<table>
<thead>
<tr>
<th>Models</th>
<th>F1 score</th>
<th>Exact Match</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Baseline methods</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Seq2Seq</td>
<td>21.5</td>
<td>17.7</td>
</tr>
<tr>
<td>PG-Net</td>
<td>45.2</td>
<td>38.0</td>
</tr>
<tr>
<td>DrQA</td>
<td>53.6</td>
<td>44.3</td>
</tr>
<tr>
<td>Pipeline</td>
<td>65.0</td>
<td>54.9</td>
</tr>
<tr>
<td>DrQA + YNtrick</td>
<td>65.3</td>
<td>55.2</td>
</tr>
<tr>
<td><strong>Our methods</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FlowQA</td>
<td>76.9</td>
<td>68.3</td>
</tr>
<tr>
<td>FlowQA + Implicit</td>
<td>76.9</td>
<td>68.1</td>
</tr>
<tr>
<td>FlowQA + Explicit</td>
<td>76.7</td>
<td>68.1</td>
</tr>
</tbody>
</table>

Based on the performance on development set, seq2seq model performs the worst. By looking into the prediction results, seq2seq model is likely to predict common words appeared in training set such as "yes". PG-Net alleviates this problem by assigning the probability of copying words from the input sequence. Although PG-Net achieves a huge performance gain comparing to seq2seq model, DrQA model performs better than PG-Net. One reason could be that DrQA predicts the start and the end of the answer span instead of generating free-form answers, which makes it easier for the model to achieve higher score. Also, PG-Net memorizes the context and question jointly to generate the answer, while DrQA encodes the context and question separately, so as it can find answers from context given the question embedding. The pipeline model combines DrQA with PG-Net. It firstly extracts the span of text as the evidence of answer, and then generates abstractive answer from the evidence. By adding PG-Net to DrQA, the model has ability of answering more abstractive questions while still being constrained to a small search space. The performance improves a large margin.

Formulating CoQA task as extractive QA sets an upperbound to the model performance. According to (Yatskar, 2018), 21.4% answers in CoQA dataset are Yes/No questions. However, if the words "yes" and "no" have not appeared in the context, our model will never be able to predict it correctly. Here we apply a simple trick called YNtrick by predicting answer types before span prediction. Specifically, besides span of text type of answers, we take "yes", "no" and "unknown" as three extra classes of answers, and consider it as a simple classification problem. From Table 1, we see that applying this simple trick boosts the DrQA performance almost 10%.

By considering dialogue flow in context embedding (FlowQA), the performance further increases around 10% (76.9% F1 and 68.3% EM), which proves our hypothesis that dialogue history is crucial in question-aware context understanding and answer prediction. However surprisingly, neither implicit nor explicit structure modeling helps increasing the performance. One possible reason is that context embedding has already includes enough question information for answer generation. Hence improving question understanding does not help the overall performance.

In Table 2, we analyzed the performance in different answer types, where "others" means all the answers other than "yes", "no" and "unknown". All the four methods in Table 2 answer "yes" questions very well, but perform worse on "no" questions. In our observation, the dataset is biased of having more "yes" samples. And lack of enough training samples might be the reason of bad performance for "no"s.

The dataset contains articles from different domains as shown in Table 3. We analyze the domain specific performance on Table 3.

Overall, from the experiments, we find that currently extractive method with simple YNtrick outperforms abstractive method by a large margin. Incorporating dialogue history in context embedding is very important in question-aware context understanding. Modeling dialogue structure in question understanding does not help in FlowQA, because the context embedding already carries enough information for the dialogue flow and complex question embedding is no longer needed.

5 Conclusion

In this project, we compared the baseline methods and current state-of-the-art method in conversational QA task. And evaluated the usefulness of implicit and explicit dialogue structure modeling in question understanding. From our observation, flowQA performs well by including dialogue information in context embedding. Building on top of flowQA, implicit and explicit dialogue history aware question embedding does not affect the
Table 2: FlowQA results analysis

<table>
<thead>
<tr>
<th>Answer Type</th>
<th>DrQA</th>
<th>FlowQA</th>
<th>Implicit + FlowQA</th>
<th>Explicit + FlowQA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>81.8</td>
<td>87.5</td>
<td>87.5</td>
<td>86.6</td>
</tr>
<tr>
<td>No</td>
<td>48.6</td>
<td>65.0</td>
<td>68.1</td>
<td>69.3</td>
</tr>
<tr>
<td>Others</td>
<td>66.2</td>
<td>77.1</td>
<td>66.0</td>
<td>76.6</td>
</tr>
<tr>
<td>Overall</td>
<td>65.3</td>
<td>76.9</td>
<td>68.1</td>
<td>68.1</td>
</tr>
</tbody>
</table>

Table 3: FlowQA domain results analysis

<table>
<thead>
<tr>
<th>Domain</th>
<th>DrQA</th>
<th>FlowQA</th>
<th>Implicit + FlowQA</th>
<th>Explicit + FlowQA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Children stories</td>
<td>–</td>
<td>75.4</td>
<td>76.5</td>
<td>75.7</td>
</tr>
<tr>
<td>Literature</td>
<td>–</td>
<td>73.0</td>
<td>73.7</td>
<td>72.9</td>
</tr>
<tr>
<td>Mid-high school</td>
<td>–</td>
<td>75.1</td>
<td>74.5</td>
<td>75.0</td>
</tr>
<tr>
<td>News</td>
<td>–</td>
<td>79.7</td>
<td>79.5</td>
<td>78.9</td>
</tr>
<tr>
<td>Wikipedia</td>
<td>–</td>
<td>80.9</td>
<td>80.4</td>
<td>81.1</td>
</tr>
</tbody>
</table>

overall performance due to the informativeness of context representation.

References


