COSE474-2024F: Final Project Proposal
Conversation-Based Question Answering: Evaluating NLP Models on Multi-Participant Conversations

Jeongmin Moon

1. Introduction

These days, it is important to provide concise summaries of long texts. Many people often seek summaries of news articles, academic papers and books, often using large language models (LLMs) like ChatGPT to generate those summaries. As a result, Natural Language Processing (NLP) models such as BERT (Devlin et al., 2019) have shown great performance on Question Answering (QA) tasks.

However, most of QA datasets such as Stanford Question Answering Dataset (SQuAD (Rajpurkar et al., 2016)) include document-style input texts. This means NLP models are good at understanding those texts and answering questions based on them. But it is unclear how well the models perform on QA datasets containing different styles of input texts. For example, if conversational texts from a group chat involving multiple participants are given, can the model answer to the question asking about the content of the conversation?

Therefore, it is important to address that uncertainty. By using pre-trained model that shows high performance on document-based QA datasets, I analyzed its capability on conversation-based QA datasets.

2. Related Works

Question Answering (QA) is a key task in NLP that has gained significant attention due to its broad applicability in areas such as information retrieval, customer service, and chat bot system. This section explores existing QA tasks, highlighting diverse approaches and datasets.

Traditional QA Systems Early QA systems usually relied on rule-based approaches, leveraging manually created knowledge bases and linguistic patterns. However, these approaches lacked scalability, as they required extensive domain-specific knowledge and engineering.

Extractive QA The introduction of large-scale datasets, such as SQuAD, marked a turning point for extractive QA. These tasks involve identifying a span of text from a given passage that directly answers the question. Transformer-based models, particularly BERT and its variants like RoBERTa (Liu et al., 2019), have shown outstanding performance in those tasks by leveraging self-attention mechanisms to understand contextual relationships.

Conversational QA Conversational QA focuses on multi-turn interactions including multiple participants where conversational context is maintained across sequential queries. Datasets like CoQA (Reddy et al., 2019) and QuAC (Choi et al., 2018) evaluates conversational capabilities. Transformer-based approaches have shown great performance in understanding conversational context.

3. Method

3.1. Problem Definition

Given a conversational dataset D which includes several conversations between two speakers in chatting system, I created a question-answer pair D_{QA} for each conversations using LLM: $D_{QA} = LLM(D_C)$.

Then, given D and D_{QA}, I created a conversation-based QA dataset D_{conv} and a document-based QA dataset D_{doc} using a rule-based program P: D_{conv}, D_{doc} = $P(D, D_{QA})$.

Then, given D_{conv}, D_{doc} and a pre-trained NLP model M, the performance for each dataset was calculated and compared.

3.2. Conversation-based & Document-based QA

In a QA dataset, a paragraph is typically given to find an answer to the given question. In my case, each conversation is treated as a paragraph. Each conversation in D is structured as follows:

```
Human 1: Hi
Human 2: Any plans for the weekend?
Human 1: my friends are gonna visit me this
     weekend. we might go hiking!
...
```

To generate the conversation-based QA dataset, I simply concatenated each sentence in the conversation. On the other hand, to generated the document-based QA dataset,

I transformed each sentence into a third-person narrative style as shown below:

```
Human 1 said 'Hi!'
Human 2 said 'Any plans for the weekend?'
Human 1 said 'my friends are gonna visit me
     this weekend. we might go hiking!'
...
```

After this transformation, I concatenated the sentences. These concatenated texts were then used to build D_{conv} and D_{doc}, each paired with corresponding question and answer sets.

3.3. Generating Questions & Answers

To create questions and answers based on the conversations from D, I employed OpenAI's GPT-4o model. The generation process focused on ensuring that both questions and answers adhered strictly to the information contained within the provided conversation. One question-answer pair was generated in each conversation, and the methodology ensured that no question or answer introduced information outside the scope of the given conversation.

Question Generation Given a conversation as input, GPT-4o was prompted to generate questions that directly referenced the content of the conversation. These questions were designed to test comprehension of specific details within the conversation. Each question focused on either the explicit facts mentioned or the underlying intent of the conversational participants, ensuring relevance to the provided context.

Answer Generation For each generated question, GPT-4o was tasked with producing concise answers that aligned strictly with the information present in the conversation. To ensure brevity, the answers consisted of only a few words.

One-shot Prompting To improve the generation accuracy, a one-shot prompting technique was applied. In this setup, an example of a conversation, along with its corresponding question and answer, was provided as part of the prompt. This single example served as a guide for GPT-4o, enabling it to better understand the desired structure and constraints for question and answer generation. By including a high-quality example in the prompt, the model was able to produce more accurate and precise QA pairs. The example of my prompt is as follows:

```
Based on the following conversation,
    generate a question and an answer. Each
     question should inquire about one
    specific detail within the conversation
    . The answers must be brief and concise
    , and the information should be
    explicitly contained in the
    conversation. Both question and answer
    should strictly adhere to the given
    conversation. An example is as follows:
{
Conversation:
A: What was your breakfast?
B: I ate sandwich.
A: Cool!

Question:
What did A eat for breakfast?

Answer:
Sandwich
}
Now, I will provide the conversation.
```

4. Experiment

4.1. Dataset

I utilized Human Conversation training data [1] from Kaggle. The dataset consists of multiple conversations between two participants, referred to as Human 1 and Human 2. Each conversation begins with a greeting from Human 1, such as "Human 1: Hi!". Additionally, 'Human 1' and 'Human 2' were replaced with 'Tom' and 'Jerry', based on the assumption that NLP models are more familiar with actual names.

4.2. Model

I utilized a variation of BERT-large model that was fine-tuned on the SQuAD dataset [2] from Hugging Face. This model was initially pre-trained on an English corpus and subsequently fine-tuned on the SQuAD dataset.

4.3. Evaluation

To evaluate the model's performance on the QA task, it is necessary to compare the ground-truth answers with the model's predictions. While comparing single-word answers by checking for exact matches is sufficient, the answers in D_{QA} often consist of multiple words. As a result, a direct comparison is not ideal, as answers composed of different words can still convey similar meanings. Therefore, I employed two approaches to compare the answers: F1 score and text embeddings.

4.3.1. F1 SCORE

F1 score measures the similarity between two answers by calculating the harmonic mean of precision and recall. Precision represents the proportion of correctly predicted words among all predicted words, while recall indicates the pro-

[1] https://www.kaggle.com/datasets/projjal1/human-conversation-training-data

[2] https://huggingface.co/google-bert/bert-large-uncased-whole-word-masking-finetuned-squad

portion of correctly predicted words among all ground-truth words.

4.3.2. TEXT EMBEDDINGS

I utilized Sentence Transformer from Hugging Face, specifically all-MiniLM-L6-v2 [3], to encode both the prediction and the answer into embedding vectors. Their similarity was then calculated using cosine similarity.

5. Results

5.1. Quantitative Results

Dataset Style	F1 Score	Text Embeddings
Conversation-based	0.669	0.834
Document-based	0.674	0.837

Table 1. Performance comparison on D_{conv} and D_{doc}

Table 1 presents the result of QA tasks on D_{conv} and D_{doc}. Each was evaluated using two evaluation methods: F1 score and text embeddings (cosine similarity).

There was no significant difference between D_{conv} and D_{doc}. While the performance on D_{doc} was slightly higher than that on D_{conv}, which is consistent with the initial assumption that NLP models are good at QA tasks on document-based dataset, the difference was negligible.

5.2. Qualitative Results

Below are some examples of QA pairs along with their predicted answers.

```
Q: Where is Tom taking a skydiving lesson?
A: seville
Predicted answer(conv): seville, spain
Predicted answer(doc): seville, spain
```

In this case, even if the predicted answer entail the ground-truth answer, the exact words may differ. As a result, the F1 score is low, whereas their text embeddings show high similarity.

```
Q: Where is Tom planning to go around
    Christmas?
A: disneyland
Predicted answer(conv): disneyland trip
    around christmas? jerry : oh wow, that
    sounds like so much fun! ...
Predicted answer(doc): disneyland trip
    around christmas?'jerry said'oh wow,
    that sounds like so much fun! ...
```

[3]https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

In this case, while the model successfully identified the start position of the answer in the given paragraph, it failed to determine the correct end position. Consequently, the model predicted an overly long answer.

6. Discussion

According to the initial assumption, it was anticipated that NLP models would perform better on QA tasks with a document-style dataset compared to a conversation-style dataset, as they are typically pre-trained on document-style texts like Wikipedia. However, the experimental results showed minimal difference, indicating that the model's ability to predict answer from a given paragraph and question remained consistent regardless of whether the paragraph was in a conversation style (e.g., Tom: Hi!) or a third-person narrative style(e.g., Tom said 'Hi!').

This is because, even if the model might be more familiar with the third-person narrative style, there is no significant difference between the two styles. The key words from the original sentences remain unchanged after transforming the text into document-style. As a result, the model is still able to understand the paragraph and extract an answer from it.

In Table 1, the performance was significantly higher when the text embedding method was used. This is because text embeddings calculate the semantic similarity between two texts more accurately than the F1 score method, which relies on exact word matches. For instance, when the ground-truth answer is 'australia' and the predicted answer is 'australia and new zealand', the F1 score is 0.4, whereas the cosine similarity between their text embedding vectors is 0.71, effectively capturing the true semantic similarity.

6.1. Limitations & Future Works

In my approach, only simple rules were applied when transforming conversation-style text into document-style text. This may have caused the text to not be fully expressed in a proper document format. For a more precise transformation, a more sophisticated approach, such as replacing pronouns like 'I' or 'You' with the corresponding names, would be necessary.

The conversations in D consisted of only two speakers. By using a dataset with conversations involving three or more participants, the task would become more challenging for the model to infer answers, allowing us to observe potential performance differences between D_{conv} and D_{doc}.

I utilized an LLM to generate question-answer pairs from D. However, as LLMs are not perfect, it is important to acknowledge the potential for errors or noise in the generated data.

References

Choi, E., He, H., Iyyer, M., Yatskar, M., tau Yih, W., Choi, Y., Liang, P., and Zettlemoyer, L. Quac : Question answering in context, 2018. URL https://arxiv.org/abs/1808.07036.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach, 2019. URL https://arxiv.org/abs/1907.11692.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. Squad: 100,000+ questions for machine comprehension of text, 2016. URL https://arxiv.org/abs/1606.05250.

Reddy, S., Chen, D., and Manning, C. D. Coqa: A conversational question answering challenge, 2019. URL https://arxiv.org/abs/1808.07042.