NLP Final Project Report

Ji-In Kwak
Institute for Software Research
Carnegie Mellon University
Pittsburgh, PA 15213
jiink@andrew.cmu.edu

Kawon Lee
Institute for Software Research
Carnegie Mellon University
Pittsburgh, PA 15213
kawonl@andrew.cmu.edu

Sehee Kim
Institute for Software Research
Carnegie Mellon University
Pittsburgh, PA 15213
seheek@andrew.cmu.edu

Younkyung Jwa
Institute for Software Research
Carnegie Mellon University
Pittsburgh, PA 15213
younkyuj@andrew.cmu.edu

Abstract

Question answering and question generation are important tasks in the NLP field. We constructed a toy dataset similar to SQuAD dataset based on various types of documents from Wikipedia. After preprocessing the context, our trained retriever reduced the search scope to paragraphs related to the question. In Question generation, we convert the pronouns into clear nouns. And using paraphrasing, the meaning of the sentence was changed clearly and then the QG process was carried out. As the language model, QA used the DistilBERT model and QG used Prophetnet. Through this, QA developed into a model that makes higher EM and F1 in a faster time than the baseline model. QG generated answerable questions with correct grammar and fluency.

1 Introduction

In today's world, a considerable amount of machines in our surroundings seem to have incorporated software that allows people to interact with them through conversation. If this function is to be used continuously in the future, it is necessary for it to have competent Question Answering (QA) and Question Generation (QG) systems. We attempted to create such QA and QG system in our project.

In order to operate the QA system, the context related to the question must be used as input. However, not all context sentences are used to find the answer, only a part of them are. The context retriever is a method to find the sentence related to the question (Karpukhin et al., 2020). The retriever process is as follows. First, the context is created as a vector using the retriever model, and the question is also encoded in the same way. After that, similar sentences are found through a similarity measurement between context sentences and question vectors. We used approximate search as the vector similarity search method. This makes it possible to find similar vectors in less time than an exhaustive search, and we used HNSW (Malkov and Yashunin, 2018) among the searching methodologies. In our project, we tried to reduce the processing time of the QA model by using a retriever, and we were able to get an accurate answer in less time. The detailed implementation method is described in 2.2

After the context retrieval process, the selected paragraphs and query sentence is given as input to the QA model. We used DistilBERT (Sanh et al., 2019), which is a small, fast, cheap, and light Transformer model based on BERT. As a student model, the number of layers is reduced and token-type embeddings and poolers are removed. We used DistillBERT to generate the answers with low time consumption.

For the question generation system, we use ProphetNet (Qi et al., 2020) which shows great performance on QG on SQuAD 1.0 (Rajpurkar et al., 2016). ProphetNet, with the highest BLEU-4 score, scored 23.910, only 1.5 percent shorter than the highest score of ERINE-GENLARGE. We use pre-trained ProphetNet for our project. Instead of optimizing one-step-ahead prediction, ProphetNet is optimized by N-step ahead prediction that predicts the next N tokens simultaneously.

2 System Architecture

2.1 Preprocessing

For question answering and generation systems, the language model such as BERT(Devlin et al., 2018) or ProphetNet retrieves embeddings from the given context input to generate the right answer or questions. However, the original Wikipedia context includes lots of accented words, non-English words such as Chinese or French, and unnecessary short titles. In a pre-trained language model, those kinds of words might generate strange information far from the original intentions of each sentence. Therefore, to prevent this, the context should be preprocessed into paragraphs where language models can easily compute and obtain appropriate embeddings.

In the traditional NLP model, text preprocessing includes removing stop words, tokenization, stemming, lemmatization, and so on. However, in our case, tokenization or lemmatization is not necessary since we use the API which takes input as a whole sentence. Hence, what we did is removing URLs, words inside parentheses, and accented characters. In the case of parenthesis words, we decided to delete them because there are many additional descriptions of keywords in different languages depending on the document. For example, the name of the Pokémon character is described in Japanese in parentheses. In addition, we removed the extra newlines or white spaces. We also removed the short titles for the question generation system. After the context is preprocessed, the text file has one paragraph for each row that can be used as input of question answering and generation models.

2.2 Question Answering

The question answering process was processed through two processes. First, sentences related to the question were extracted from the context with the retriever model, and then answers were extracted with the language model. As described above, in order to find the retriever sentence, we first create an encoded vector. We used multi-qa-mpnet-base-cos-v1[1] of sentence-transformer as a model for this. This is a model learned with various question-and-answer pairs that create 768 dimensional dense vectors. SQuAD2.0 (Rajpurkar et al., 2018), which is similar to the data of the project, is also used. Vector created using context can be used for multiple questions. Therefore, saving them in advance and reusing them further speeds up the QA system. Therefore, in this experiment, the context encoded vector was stored and used with the Pinecone database system. We deleted duplicated vectors before uploading them to the database. When the QA system starts with a specific context, we check whether vectors regarding this context are uploaded or not. If it is already uploaded, we used that. The question also created an encoded vector in the same way as the context sentence. Based on this, a similarity search was conducted with the vectors in the Pinecone database, and a new context was constructed with the top k sentences with the highest similarity. The approximate search was performed as a similarity search method, and it is provided in the Pinecone database API. Finally, with this completion, we put the new context and question into the question answering model to extract the answer. For the question answering model, we used DistilBERT (Sanh et al., 2019), which is a light and fast BERT-based Transformer model. The whole pipeline of the QA system is shown in Figure 1

2.3 Question Generation

Coreference We wished to create a new sort of context from the original text because we wanted the question generation to make questions from a variety of sentences from the original text, not just the ones that are next to each other. In order to do this, we would randomly select 3 paragraphs from the original text, then randomly select 20 percent out of all the sentences that make up those 3 paragraphs.

However, if we put together sentences from random places in the original source, there was a possibility that the pronouns in the sentences would make the question generator misunderstand the text. For instance, when the sentences “Sarah ate an apple as a snack”, and “She loves cake” are put together, it seems like the pronoun “she” in the second sentence indicates Sarah. However, it would be a wrong interpretation if the second sentence originally came after “Jane went to the bakery.”

To avoid this sort of confusion, we decided to apply coreference resolution to the original text before creating a new context from it. We used a model named Crosslingual Coreference[2] and chose to use spanBERT (Joshi et al., 2020), an extension of BERT, as a pretraining option. By applying this model to the original text, we were able to change pronouns, nouns that refer to other nouns, and phrases into nouns that are cluster heads.

Paraphrasing After all the pronouns were changed to nouns, data was preprocessed using

[1] https://huggingface.co/sentence-transformers/multi-qa-mpnet-base-cos-v1

[2] https://github.com/Pandora-Intelligence/crosslingual-coreference

paraphrasing, which expresses these sentences in different ways. Pre-training with Extracted Gap-sentences for Abstract Summarization Sequence-to-sequence models, or PEG (Zhang et al., 2020), were used to insert a process to express one sentence in a different way. For example, the sentence "Eevee is best known for being the Pokémon with the most potential evolutions, with eight possible evolutionary forms." can be paraphrased by "The Pokemon with the most potential evolutions is Eevee." Through paraphrasing, the sentence was changed more clearly and expected better performance than simply adding context. After paraphrasing, question generation was performed using prophetnet-large-uncased-squad-qg fine-tuned weights.

3 Experiments

After the preprocessing applied in the QA system was applied. When paraphrasing, the length of beam was determined to be 10, and the max length of one batch was set to 60.

3.1 Question Answering

Our question answering system uses the context path and the path of related questions as arguments. Therefore, for this project, we picked a specific context in the given CSV file and made text files containing questions related to it. Also, the answers to the questions were saved as a separate text file. As a parameter, the number of sentences to be retrieved as a query of the retriever database was used. This is the number of sentences with the highest value based on the similarity between the question and the stored sentence vector.

To evaluate whether the question answering system is working well or not, the SQuAD dataset used two different metrics which are an exact match (EM) and F1 score (Rajpurkar et al., 2016). The EM measures the percentage of predictions that match any one of the ground truth answers exactly. In contrast, the F1 score measures the percentage of overlaps between predictions and the ground truth answers. The overlap is computed based on the tokenized words of the answers. We used these two metrics in experiments on the given dataset which is made of students. The tested dataset is 159 questions and answers generated from Wikipedia's Eevee document.

We compared our question-answering model with the baseline distilBERT without any preprocessing and paragraph retrieval process. In the case of the baseline, since the model needs to extract the hidden representations of the entire context, it takes quite a long time to process a long document. However, when we used the sentence retriever, the computation time is shortened according to the number of paragraphs we are considering. We compared the accuracy results according to the Top K numbers of relevant paragraphs retrieved from the original context. The K is set as 1, 3, 5, and 7. As shown in Table 1, both EM and F1 score has the highest value when we used the Top 3 sentences as a context input. Using only one sentence as an input has significantly reduced computational time, but for hard problems, there might not be enough information. However, as the number of input paragraphs increases, the model accuracy decreases because the model is confused by a lot of information.

For the generated answers, we noticed that our proposed model cannot generate yes or no answers. This is the reason that EM and F1 score is significantly lower than the result from the original papers. Hence, we checked only for the questions without a confirmation response. The result is shown in Table 2.

3.2 Question Generation

In the QG experiment, we took 10 questions that were generated using Wikipedia's Eevee document as context.

3.2.1 Answerable Questions

To see if the questions were answerable, which here means that the questions have answers that can be found in the context, we had to use human judgment. One of our team members examined the input context to see if the 10 questions our QG model had generated had answers based on the context.

Out of the 10 questions, 7 questions were judged to be answerable(Questions 1, 3, 4, 5, 6, 8, 9 in Generated Question part of Table 3), 1 was considered to be ambiguous(Question 7 in Generated Question part of Table 3), and 1 was unanswerable(Question 10 in Generated Question part of Table 3).

3.2.2 Grammar

We took 10 questions that were created by our QG system and ran them through the T5 Grammar Cor-

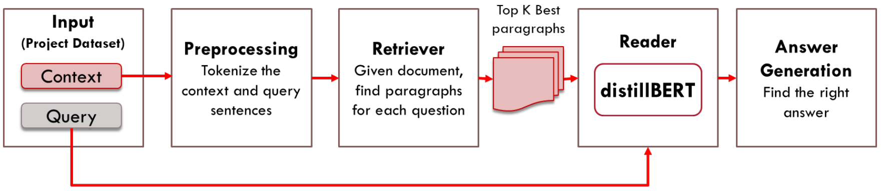

Figure 1: The pipeline for question answering system

Table 1: The model accuracy for test QA dataset

Metrics	baseline	Ours with Top 1	Ours with Top 3	Ours with Top 5	Ours with Top 7
EM	33.33	30.19	**34.59**	27.67	28.931
F1	38.45	36.31	**42.25**	34.62	37.64
Times	800.77	**60.96**	117.27	187.06	255.01

Table 2: The model accuracy for test QA dataset without confirmation questions

Metrics	baseline	Ours with Top 3
EM	45.69	**46.55**
F1	52.36	**56.32**
Times	348.78	**72.02**

rection[3], which used HappyTransformers[4] and the dataset JFLEG (Napoles et al., 2017), model to test if our questions were grammatically correct.

However, the raw results could not be used to discern whether the input questions were grammatically correct because the questions had the word "Eevee" in them. This can be seen in Table 3. The model marked that word as incorrect, so the results had to go through human judgment. After examination from one of our teammates on the results from the T5 Grammar Correction model, we found that 1 question was deemed completely correct by the model, 8 were found to be correct from human judgment (which means that we overlook "Eevee" as something to be corrected), and 1 question had part of its structure modified completely, so we deemed it as undecided.

3.2.3 Other Metrics

We wanted to use a more objective metric other than human judgment, so originally we planned to use the Answerability-Metric (Nema and Khapra, 2018), which claimed to be a metric that overcame problems concerning answerability that the more traditional metrics like BLEU (Papineni et al., 2002), which was originally used to assess machine translation, had. However, it required reference questions that matched the generated questions for it to output evaluation scores. Since we did not have these reference questions, we were unable to use this metric.

4 Discussion

For question answering and generation, we constructed a model system that works with the unstructured Wikipedia dataset. However, there are several limitations to our system. First of all, the response model cannot generate a confirmation response, such as Yes or No. Instead of these answers, we find that the model generates the correct answers in consideration of the question and context. For example, when the ground truth answer to the question "Was the Eevee pokemon designed by Donald Trump?" is "No", our model gives the answer "Created by Motofumi Fujiwara" which is the right explanation for the question.

On the other hand, in the question generation system, we found that several questions are too ambiguous to have a single answer or cannot find an answer from the given context. However, it was not easy to find the evaluation metrics for the generated questions since it is impossible to find the answers for each question manually. In addition, the same questions are generated because sentences are randomly selected from the entire context and put as the system's input.

[3] https://huggingface.co/vennify/t5-base-grammar-correction

[4] https://github.com/EricFillion/happy-transformer

Table 3: Question Generation Grammar

Generated Question	Grammatically Corrected Question
in 2015, eevee was the most traded pokemon?	In 2015, eevee was the most traded pokemon?
what was the name of may's eevee?	What was the name of May's **Eve**?
is there a reason why an eevee can't evolve?	Is there a reason why an **evolution** can't evolve?
is eevee a photographable pokemon?	Is **there** a photographable pokemon?
what is eevee best known for?	What is **evee** best known for?
when did the player receive an eevee from professor oak?	When did the player receive an **email** from professor oak?
when did the player receive an eevee?	When did the player receive an **email**?
is eevee photographable outside of the main series?	Is **it possible to photograph** outside of the main series?
what was the original english name for eevee?	What was the original english name for **evee**?
what happened to eevee when flareon forced her to evolve?	What happened to **Eve** when flareon forced her to evolve?

5 Conclusions

We constructed the question answering and generation system using the pre-trained model of distilBERT and ProphetNet. The SQuAD dataset which is similar to our given dataset is used for the training. Also, several preprocessing and context selection methods are used to improve the model's accuracy. For future work, it would be better to include the question type classification process to generate the confirmation responses. In addition, evaluation metrics for grammar and fluency of generated question sentences are needed.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836.

Courtney Napoles, Keisuke Sakaguchi, and Joel R. Tetreault. 2017. JFLEG: A fluency corpus and benchmark for grammatical error correction. *CoRR*, abs/1702.04066.

Preksha Nema and Mitesh M. Khapra. 2018. Towards a better metric for evaluating question generation systems.

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

Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training. *arXiv preprint arXiv:2001.04063*.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.