Question Answering for Nanyang Technological University's GEM Explorer Program

Hussain Khozema Kheriwala
School of Computer Science and Engineering

Dr Sunil Sivadas
Singtel Cognitive and Artificial Intelligence Lab for Enterprises

Dr Yan Shi Xing
School of Computer Science and Engineering

***Abstract* –** Question Answering systems are part of the Natural Language Processing (NLP) domain and are becoming more and more common in today's world. This paper presents a system which provides answers pertaining to Nanyang Technological University's GEM Explorer student exchange program.

It first splits large documents containing information regarding the program into several appropriately sized documents. The ElasticSearch information retriever then gets top k documents and ranks them based on relevancy to the query. It then uses a reader to extract answers from these documents to directly answer the question to the query. The paper further investigates the accuracy of various pretrained transformer models. This system will be demonstrated via a graphical user interface, where the user will be able to ask a question and receive an answer together with a score.

Keywords – Question Answering, Natural Language Processing, GEM Explorer

1 INTRODUCTION

Question Answering (QA) systems [1] are a form of information retrieval whose task is to automatically provide an answer for a given question. They have been adopted and used more and more over the years. There is an abundance of data in today's world more than ever before, and that makes finding of important information quickly an important task. Question Answering is a growing research field and better and more accurate methods are being developed to deliver short, precise question-specific answers.

GEM Explorer is an outbound student exchange program under Nanyang Technological University (NTU). It allows students to spend a semester abroad either to pursue courses or to conduct research. It is a highly popular program in NTU which allows students to experience diverse cultures across the world while earning credits for their studies.

The motivation for this paper is to explore and build a closed-domain question answering system which can answer questions about the GEM Explorer program in NTU.

This report will provide the details of the project including the question answering pipeline, collection of documents, pre-trained models, retrievers to retrieve relevant documents, reader to get the correct answer from the documents, and the evaluations done on the retriever and reader as well as the different pre-trained models.

2 CORE COMPONENTS

A typical question answering system consist of three core components [2]: question/query processing, document processing and answer extraction.

2.1 QUERY PROCESSSING

Query processing uses linguistic techniques such as parts of speech tagging, named entity recognition, parsing, removal of stop words to gain better insights on the type of question asked and the kind of answer that it expects. A lot of the times it might be hard to find the terms used in the queries in the documents, so query expansion and paraphrasing techniques might also be used to tackle this issue. Question classification is also often used to identify the class the question belongs to (eg., where, who, what, why) which can tell the system what type of answer to expect.

2.1 DOCUMENT PROCESSING

Document processing is also known as information retrieval whose job is to retrieve a set of documents which is most relevant to the query. Sparse and dense retrievers are the widely used retrievers which we will discuss more in section 3.5. There are also two main types of data where information can be retrieved from, structured and unstructured [3]. Unstructured data are free text which does not have a predefined format and can be easily accessible, such as Wikipedia. Structured data on the other hand are predefined knowledge base often stored in databases and are often manually constructed. After the documents are retrieved, paragraph filtering might be used to

extract the relevant paragraphs from the documents. This can reduce the number of documents and the text which contains the answer to the query. These paragraphs are then ordered and ranked based on the probability of them containing the correct answer. A popular ranker is InferSent Ranker which uses sentence representations to rank passages based on semantic similarity with the question.

2.1 ANSWER EXTRACTION

Answer extraction is the identification and retrieval of the answers from the paragraphs provided. The accuracy of the answer extracted is dependant on the complexity of the question, the expected answer type, the documents retrieved from the document processing stage and the method used to extract the answer.

In most of the modern question answering systems, a pre-trained reader is used to extract the answer from the document. There are two forms of readers: extractive reader which predicts the answer from the retrieved documents, and generative reader which generates answers in natural language using a sequence-to-sequence model. Extractive reader assumes that the answer certainly exists in the documents and usually predicts the start and end position of the answer from the retrieved document. Bidirectional Encoder Representations from Transformers (BERT) reader is very commonly used for question answering systems. We will discuss more about BERT in section 3.6.

3 ARCHITECTURE

The overall implementation of the question answering pipeline is shown in the Figure 1 below.

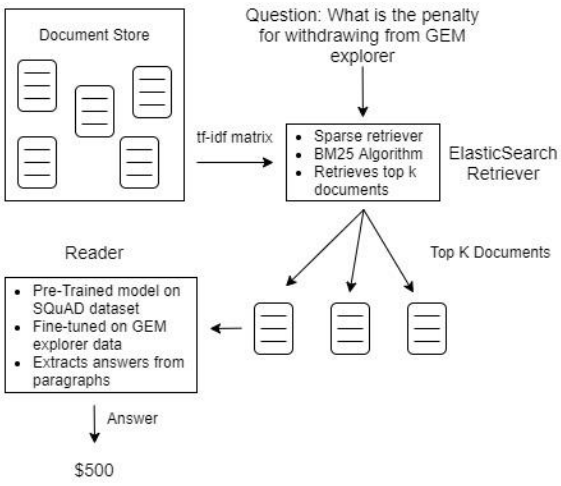

Figure 1 Question Answering Architecture

As seen from the architecture diagram [4], there are several components involved in building a question answering system. When a query is entered, the retriever will get the documents from the document store, including the one consisting of the answer. A similarity algorithm will then be used to rank these documents according to how similar and relevant they are to the query. These top-ranking documents are then passed to a reader. The reader has been pre-trained on the SQuAD dataset and then further fine-tuned using the GEM Explorer dataset. The reader then extracts the top answers from these documents.

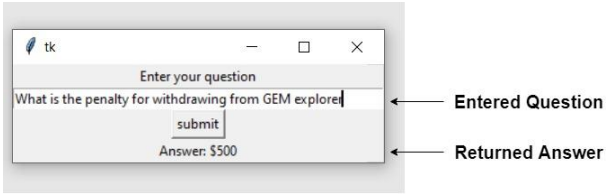

Figure 2 Graphical User Interface for the Question Answering System

The figure above shows the user interface created which allows the user to enter a question, and the system will return the answer accordingly. Python's tkinter library is used for this implementation.

3.1 DATA COLLECTION

This is a closed-domain question answering system which deals with questions only regarding Nanyang Technological University (NTU) GEM Explorer program. Therefore, it is required to collect documents relevant to GEM Explorer.

There are 25 documents extracted from the NTU's website which contains all the information about GEM Explorer. These documents are then split in into 59 smaller documents and stored in a documents store. Document store is further discussed below.

3.2 DOCUMENT STORE

Document store is essentially a database which contains all the documents. All the answers for the queries are looked up in the documents in the document store.

It is paramount to limit the size of the documents stored. A very large document with a lot of words can greatly impact the performance of the retriever since the relevant context in the document can be overwhelmed by the rest of the text in the document. Limiting the size of the document can optimise the retrieval process since only the most relevant sections of text are retrieved. The reader will also take up more time to find the answer in a large document. Therefore, limiting the size of the document will also speed up the reader.

In this implementation, the documents collected are first pre-processed by removing consecutive empty lines, cleaning any whitespaces at the beginning and end of each line in the text. These documents are then split into 59 smaller documents of 100 words each. The sentence

boundaries are respected so the sentences do not get split up halfway through the document. The 59 documents are then stored into the Elasticsearch document store.

3.3 TF-IDF

Term frequency-inverse document is a statistical measure [5] which evaluates how relevant and important a word is to the document. It is calculated simply by multiplying the term frequency of a word in a document and the inverse document frequency across the set of documents.

$$\mathrm{tf}(i,j) = \frac{Number\ of\ times\ word\ i\ occurs\ in\ document\ j}{Total\ number\ of\ words\ in\ document\ j} \quad (1)$$

$$idf(i) = \log_2\left(\frac{Total\ number\ of\ documents}{Number\ of\ documents\ with\ word\ i}\right) \quad (2)$$

$$tf\text{-}idf(i,j) = \mathrm{tf}(i,j) \times idf(i) \quad (3)$$

3.4 BM25

BM25 is a sparse retrieval method which ranks the documents. It is a variant of tf-idf and is known to perform much better than tf-idf. The BM25 similarity scoring is shown as below.

$$score(i,j) = \sum_{n=1}^{|i|} idf(i_n,j) \cdot \frac{tf(i_n,j)\cdot(k+1)}{tf(i_n,j) + k\cdot\left(1 - b + b\cdot\frac{len(j)}{avg\ document\ length}\right)} \quad (4)$$

The number of occurrences of a relevant word directly affects the term frequency score. A word occurring twice as much in a document compared to another document does not necessarily mean it is twice as relevant to the document. Once a document gets saturated with occurrence of a term, more occurrences should not have that much impact on the score. BM25 tackles this limitation by introducing a variable k which controls the contribution term frequency to the overall tf-idf score. It can limit how much a single query term can affect the tf-idf score of a given document. This also helps a query with a lot of terms since it will match documents with more of the terms in the query rather than match documents with just a lot of occurrences for one of the terms.

BM25 also introduces the ratio of the length of the document over the average document length [6]. This allows us to reward short documents in which the term occurs and penalise larger documents in which the same term occurs. This ratio is multiplied by the variable k in the denominator. For a document of average length, there is no impact. However, if the document length is shorter than average, it will have a greater impact on the similarity score.

We are also able to control the impact of this ratio of the length of the document over the average document length by introducing another variable b which takes the value between 0 to 1. The bigger the value of b, the bigger the effect of the length of the document compared to the average length.

Overall, we can see that BM25 overcomes the limitations posed by tf-idf and is a better algorithm [7]. We will be using BM25 for our information retrieval process.

3.5 RETRIEVER

The retriever is meant to quickly go through the document store and get a set of documents which is relevant to the query. It filters out the documents which are obvious negative cases. This speeds up the querying process since it removes the documents which does not contain the answer, reducing the work needed to be done by the reader.

Dense and sparse are two types of retrieval methods. Dense retrieval indexes and stores the documents by passing it thorough a neural network and storing the resulting vectors generated. It is also sensitive to the order and meaning of the words. Sparse retrievers on the other hand treats text as a bag-of-words and leverages frequency of terms in the query appearing in the document. Tf-idf and BM25 are most common sparse retrieval methods that are used. We will be using ElasticSearch retriever which uses BM25 to retrieve the relevant documents.

3.6 READER

Reader is a main core component of a question answering system. BERT is a method of using transformers, which is an attention mechanism that learns the contextual relations between the words in a text, to generate a language model with bidirectional encoder representations [8]. The transformer reads the entire sequence of words at once rather than in a sequential order. This allows the model to learn the context of the words based on all its surrounding words. During the training process, 15% of the words in each sequence are masked and the model tries to predict the original masked words. This is the loss function for a BERT model.

In this implementation we will be using DistilBERT [9], which is a transformer model pretrained on the same corpus as BERT. However, it was optimised to be smaller and faster than the original BERT model. This model is then trained on SQuAD [10] dataset, which is a reading comprehension dataset consisting of 100,000+ question and answer pairs on 500+ articles.

Once we get the model trained on SQuAD dataset, the model is then further fine-tuned using the

question answering dataset relating to GEM Explorer. This allows the model to better answer domain specific questions. We can then use this fine-tuned model to answer questions about GEM Explorer.

4 EVALUATION AND RESULTS

4.1 RETRIEVER ACCURACY

To evaluate the accuracy of the retriever, we take note of whether the set of retrieved documents contains the correct document.

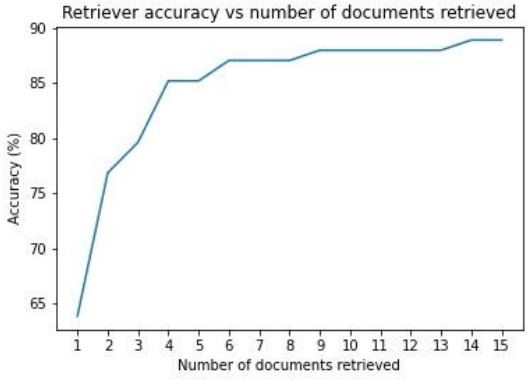

Figure 3 Line graph of retriever accuracy vs the number of documents

The graph in figure 3 shows how the accuracy of the retriever varies when the number of documents in increased from 1 to 15. We can see that the accuracy increases at a decreasing rate. This is because as more documents are retrieved, the probability of the correct document to be one of those documents also increases. For the implementation, we will be retrieving 10 documents before passing it to the reader as we can see that the accuracy does not increase significantly after 10 documents. For 95 questions out of 108, it was able to get the correct document out of the 10 documents retrieved with an accuracy of 87.96%.

4.2 READER ACCURACY

To get the accuracy of the reader, we will see whether the reader is able to predict the correct answer given a set of documents including the correct one.

For this experiment, a set of documents is retrieved using the retriever. If the documents do not contain the relevant document which contains the answer. One of the incorrect retrieved documents is replaced with the correct one. These documents are then passed to the reader, which extracts the correct answer from these documents.

The reader used is the distilled version of BERT trained on SQuAD dataset and fined tuned on GEM Explorer dataset.

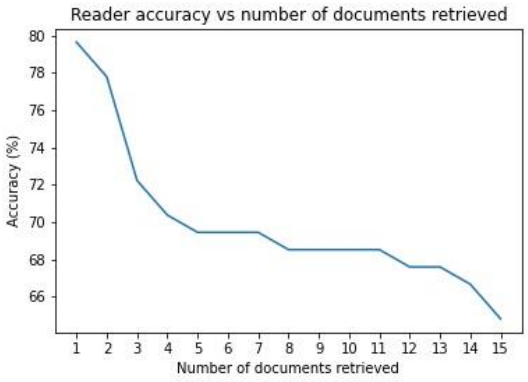

Figure 4 Line graph of reader accuracy vs the number of documents

This graph above shows the accuracy of the reader as compared to the number of documents retrieved including the correct document. We can see that as more documents are retrieved, the lesser the accuracy of the reader gets. This is because more documents result in the reader looking through more documents which can cause the reader to extract the wrong answer. When only one correct document is passed to the reader, we can see that the reader is able to correctly extract the answers for almost 80% of the questions.

4.3 COMPARING DIFFERENT PRE-TRAINED MODELS

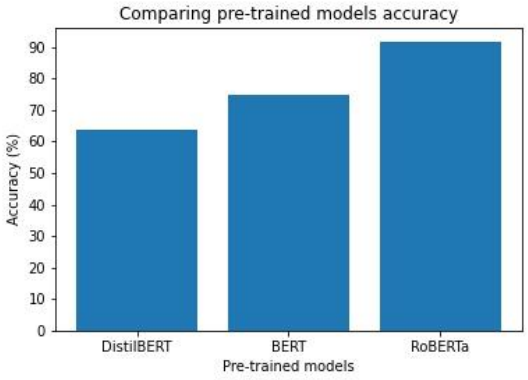

Figure 5 Bar graph comparing accuracies of various pre-trained models.

The graph above shows the accuracy of the reader using various pre-trained models. The three models, DistilBERT, BERT as well as RoBERTa, trained on SQuAD dataset were fine tuned on GEM Explorer dataset.

To evaluate the accuracy, the reader was provided with the question and 5 documents including the document containing the correct answer. We then take note of the number of questions out of 108 questions the reader was able to correctly extract the answer for.

We can see that the RoBERTa model is able to achieve the highest accuracy of about 90%, significantly higher as compared to the accuracy of

DistilBERT and BERT. The RoBERTa model was able to correctly extract the answer for 99 questions out of 108 questions when provided with 5 documents including the correct document.

5 CONCLUSION

This paper explores the question answering systems using machine comprehension in today's worlds. It goes into details about the various components needed for a question answering system and how they work together to return a short, precise, and accurate answers to a given query. It also talks about the various methods and approaches commonly used in today's world. Evaluations on the reader, retriever as well as the various pre-trained models are also performed to see the accuracy of our closed domain question answering system with regards to Nanyang Technological University's GEM Explorer program.

For future work, the various pre-trained models can be explored in depth to explain the significant difference in the accuracy of the models.

ACKNOWLEDGMENT

I would like to thank Dr Yan Shi Xing for his generous support during this project. I would also like to show my greatest appreciation to Dr Sunil Sivadas and his team members, Dr Josey Mathew, Yi Ren Leng and Jing Hao from the Singtel Cognitive and Artificial Intelligence Lab (SCALE@NTU) for their thorough guidance during this project. They supported me with their expertise and knowledge, going through important concepts and terminologies, and clarifying any doubts presented to them.

I would like to acknowledge the funding support from Nanyang Technological University – URECA Undergraduate Research Programme for this research project.

REFERENCES

[1] Marco AntonioCalijorne Soares, Fernando SilvaParreiras, "A literature review on question answering techniques, paradigms and systems", 6 July 2020; https://www.sciencedirect.com/science/article/pii/S131915781830082X

[2] Ali Allam, Mohamed H Haggag, "The Question Answering Systems: A Survey", September 2012.

[3] Fengbin Zhu, Wenqiang Lei, Chao Wang, Jianming Zheng, Soujanya Poria, Tat-Seng Chua, "Retrieving and Reading: A Comprehensive Survey on Open-domain Question Answering", 8 May 2021.

[4] André Macedo Farias, "How to create your own Question-Answering system easily with python", 8 July 2019; https://towardsdatascience.com/how-to-create-your-own-question-answering-system-easily-with-python-2ef8abc8eb5

[5] Bruno Stecanella, "What Is TF-IDF?", 11 May 2019; https://monkeylearn.com/blog/what-is-tf-idf/

[6] Shane Connelly, "Practical BM25 - Part 2: The BM25 Algorithm and its Variables", 19 April 2018; https://www.elastic.co/blog/practical-bm25-part-2-the-bm25-algorithm-and-its-variables

[7] Rudi Seitz, "UNDERSTANDING TF-IDF AND BM-25", 20 March 2020; https://kmwllc.com/index.php/2020/03/20/understanding-tf-idf-and-bm-25/

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", 24 May 2019.

[9] Victor Sanh, Lysandre Debut, Julien Chaumond, Thomas Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter", 1 March 2020.

[10] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, Percy Liang, "SQuAD: 100,000+ Questions for Machine Comprehension of Text", 11 October 2016.