ILFQA: A PLATFORM FOR EFFICIENT AND ACCURATE LONG-FORM QUESTION ANSWERING

by

RHYS SEAN BUTLER

A dissertation submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Master of Science

Computer Science and Information Systems Program

2021

This report for the Master of Science degree by

Rhys Sean Butler

has been approved for the

M.S. Degree

by

Dr. Farnoush Banaei-Kashani, Advisor

Dr. Haadi Jafarian

Dr. Salim Lakhani

April 26th, 2020

Butler, Rhys Sean (M.S., Computer Science and Information Systems)

ILFQA: A PLATFORM FOR EFFICIENT AND ACCURATE LONG-FORM QUESTION ANSWERING

MS Project directed by Assistant Professor Farnoush Banaei-Kashani

ABSTRACT

We present an efficient and accurate long-form question-answering platform, dubbed iLFQA (intelligent Long-Form Question Answering). The purpose of iLFQA is to function as a platform which accepts unscripted, human generated questions and efficiently produces a semantically meaningful, explanatory, and accurate response. iLFQA introduces and integrates a number of natural language processing as well as supervised learning modules to provide answers to questions from a pre-defined and sizable knowledge corpus including numerous long-documents (e.g., books). iLFQA leverages Transformer Architecture, Zero-Shot Classification, and Term Frequency–Inverse Document Frequency to rapidly select paragraphs from a series of texts that are most relevant to a given question. Once the paragraphs have been selected iLFQA synthesizes an appropriate textual response for the question and returns it to the user. iLFQA's performance and limitations will be demonstrated.

The form and content of this abstract are approved. I recommend its publication.

Approved: Dr. Farnoush Banaei-Kashani

TABLE OF CONTENTS

CHAPTER

LIST OF FIGURES

FIGURE

CHAPTER I

INTRODUCTION

In this paper we present a long-form question answering platform named iLFQA: intelligent Long-Form Question Answering. This report is divided into six chapters. The introduction describes the motivation for our work and some important background information. Related work describes different question answering systems. Background provides a high level overview of many of the systems and architectures used in iLFQA's implementation. Methodology describes how iLFQA produces responses. Experimentation describes the hardware and software used to implement iLFQA, examples of output, and detailed metrics on the system's performance. Discussion describes our closing thoughts, and insights more interesting findings. We also detail future work that might be implemented to advance similar systems.

Question answering is a task within machine learning, where a model or system receives a question as input and produces a correct response. As an example, if you submit the question, "What is the average lifespan of a spotted owl?", to Google, Google will use the information in your question, and attempt to retrieve your answer as a span extracted from a relevant document 1. The answer returned for this question did not rely on any previous knowledge of the question. The response returned was also single sentence without explanatory information. Question answering has remained a difficult problem in the domain of natural language processing. Most question-answering techniques focus on returning factoid type answers, no longer than a sentence. In general, most question-answering systems focus on small inputs and retrieve the answer from a small passage, maybe a single paragraph in length. In practice, finding the answer to an open-ended question may require looking through a large number of unstructured documents, containing tens of thousands of words. In addition to difficulties in processing and comprehending long passages, the generated responses are usually extracted directly from the source document. Extractive answers generate answers by pulling information directly from the text. Normally, question answering systems are implemented using rule-based [5] or retrieval-based methods [6]. A rule-based system will produce responses based on a pre-defined set of rules implemented by if-

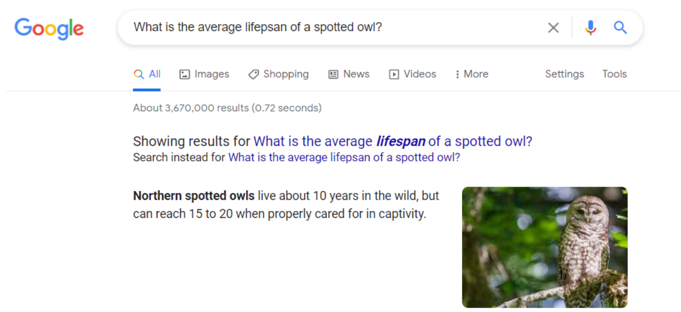

Figure 1: An example of question answering techniques in practice.

then conditions. When specific conditions are met, the system will select the course of action in line with its pre-defined set of rules. This would mean that a rule-based question answering system is limited in its capacity to respond to certain inputs. In contrast, retrieval-based systems operate by accessing a selection of documents, either on the web or stored locally [7]. When a user asks a question, a retrieval-based system will attempt retrieve the answer as a span of text from a short selection, usually a paragraph. Rule-based and retrieval-based question answering systems perform well in constrained circumstances, are easy and inexpensive to implement, and are somewhat fast [6]. Systems implemented in this way often fall short of interacting with humans in a natural and convincing way. Research in the field of generative question answering has become more common in recent years due to the advance of natural language processing techniques [8].

CHAPTER II

RELATED WORK

There has been previous work that focuses on generative question answering [8][6][5], knowledge grounded chat bots [7], and ambiguous question answering [1]. Generative question answering refers to a process where a response is dynamically generated. This is different than retrieving a span of text for a given question. Knowledge grounded chat bots can be defined as any chat bot that incorporates knowledge from a domain or set of domains when interacting with a user. Ambiguous question answering can be defined as answering questions that may contain multiple plausible answers. Long-form question answering can be defined as the task of producing answers by retrieving them from appropriate documents and generating a multi-sentence response.There are many question answering systems, but few long-form question answering systems. Some question answering systems use recurrent neural networks. Recurrent neural networks (RNNs) are effective for generating semantically accurate and meaningful responses. Some RNNs are auto-regressive. That is to say that each output token in a sequence of text makes up the input for the next token. So, given a long sequence of text to process as input, or to generate as output, each word must be processed one at a time. This makes interpretation and generation of long passages difficult. A new recurrent neural network architecture, named Transformers has been introduced that makes processing longer passages more feasible. One project that showcases the capabilities of new technologies for use in the question answering domain is called AmbigGQA [1]. In their paper, "AmbigQA: Answering Ambiguous Open-domain Questions", the authors explain that one of the difficulties in generative question answering is that answering dynamically generated questions requires knowledge not necessarily gained from previous user inputs. For instance, if I ask how many movies are in a particular franchise, the system must understand the context of the question, then seek the correct result from a series of documents. The authors of AmbigQA create a database that can be used to train models to disambiguate similar questions within the same passage 2.

To perform the necessary tasks on the AmbigQA dataset, a model first selects spans of text from

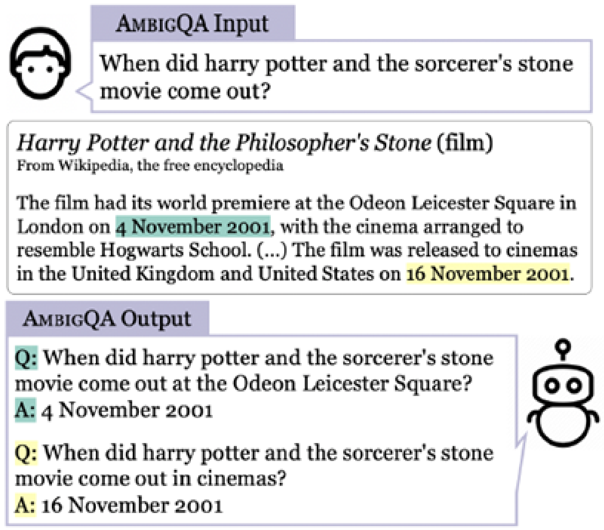

Figure 2: An example of AmbigQA output. [1]

its document base (Wikipedia passages). It then ranks those passages and concatenates the most promising ones together up until it has reached a total of 1024 tokens. At that point a sequence to sequence model is used to generated the answer from the concatenated spans. The model then generates a series of questions that are paraphrased from the original input. It tries to match the best spans to the whole series of generated questions and the original to find the most likely answer. Another system, StockQA [8] attempts to answer questions in the domain of stock trading. StockQA accepts open ended questions about stocks and attempts to return answers based on those questions. StockQA is of note because some questions may require calculations to be performed based on information contained within the database. Both StockQA and AMBIGQA offer interesting examples of question answering systems that seek to perform useful tasks. However, neither of them address the issue we seek to solve with our work. They do not tackle the problem of long-form question answering. Long-form question answering systems are rare. Part of this is due to a lack of resources necessary to train appropriate models [9]. ELI5 is the only database that focuses on long-form responses to questions at the time of this report [10]. The lack of appropriate databases remains a critical bottleneck for more advanced research into the field of Long-Form Question Answering.

CHAPTER III

BACKGROUND

3.1 TRANSFORMERS

We use AmbigQA as an example because its method of question answering similar to what was used for iLFQA. When seeking the passages that most likely contain the best answer, AmbigQA leverages a technology named Transformers [11]. AmbigQA uses Transformers to find the passages that most closely correlate to the input question. The Transformer is a neural network architecture that has become popular for use in translation and language comprehension tasks [2]. Before Transformers were proposed, Long short Term Memory (LSTM) systems were heavily used for on language translation. When using LSTMs, each input of a sequence must be passed into the network one at a time. This means that each token from an input sequence must be processed before the next input can be accepted. Transformers improve upon this method by allowing multiple tokens from an input sequence to be processed in parallel [2]. This leverages the capabilities of modern graphics processing units, which are designed to run computations in parallel. By taking advantage of the parallel capabilities of graphics processing units, Transformers can perform tasks much more quickly than their LSTM counterparts. Transformers create a mapping for language by embedding each word in a vocabulary to a point in an embedding space. Each token in a language's vocabulary corresponds to a point in this multi-dimensional space. The embedding space can be thought of as the Transformer's understanding of the language [2]. When the Transformer receives tokens for input it does its best to match that token to a point in the embedding space. Transformers implement a concept called self-attention to correctly determine which tokens correlate to which points in the language embeddings [2]. Self-attention can be briefly described as an understanding of each individual token in an input sequence on its own. For instance, if an input sequence read "This dog is better than that dog", self-attention would differentiate the first dog token from the second one. Self-attention allows more accurate predictions to be made by ensuring that each token is considered on its own. LSTM models of the past would have trouble differentiating between the two different dogs in the sentence above. An example of the Transformer architecture can be viewed in 3.

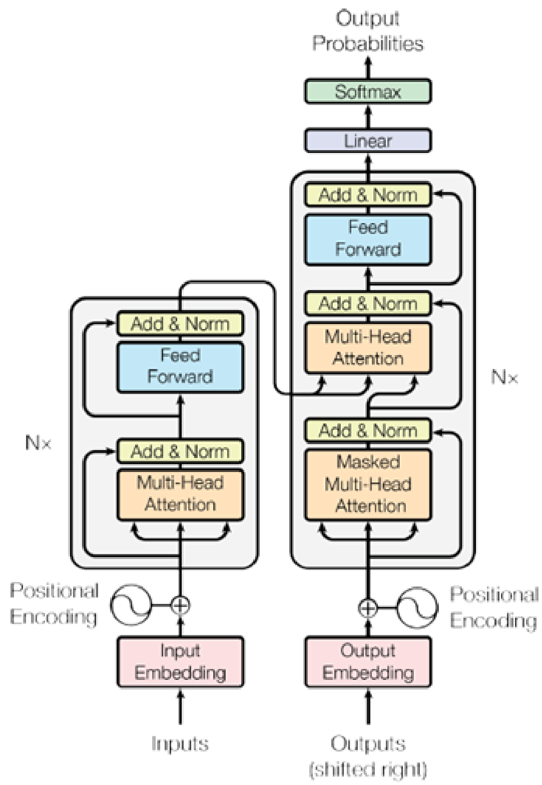

Figure 3: A diagram of the Transformer architecture. [2]

3.2 BERT

BERT stands for Bidirectional Encoder Representation from Transformers. BERT consists of a series of stacked Transformer encoders [11]. When a token is passed to the encoder of a Transformer, the output is a vector that contains the positional embedding information from the token. Recall that the embedding space represents the model's understanding of the vocabulary and language. So, each token that is passed through the encoder is transformed into a single vector that contains information on the token that helps place it somewhere in the embedding space. BERT is used to pre-train models [11]. Once a BERT model has learned a language or vocabulary, it is then fine-tuned to perform some specific task.

3.3 BART

BART stands for Bidirectional and Auto-Regressive Transformers. BART combines a stack of Transformer encoders and series of stacked Transformer decoder blocks as in GPT-2 [8][9]. Transformers are composed of an encoder and decoder and GPT-2 utilizes the decoder portion of the architecture, usually to perform text generation tasks [9]. The creators of GPT-2 wanted a system that could perform multiple different natural language processing tasks [10]. BART combines techniques from BERT and GPT-2 to perform text generation, summarization, and question answering. See 4 for an illustration of the architecture.

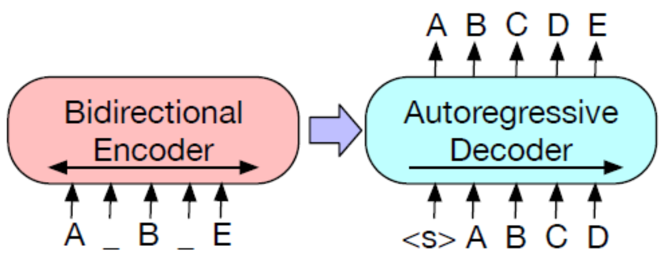

Figure 4: A diagram of the BART architecture. [3]

BART does have a drawback. BART generates text in an auto-regressive fashion, similar to an LSTM. Unlike LSTMs, BART can utilize the embedding of an input to create more accurate outputs. BART is also better suited to generating long sequences of text. [3].

3.4 ZERO-SHOT CLASSIFICATION

Zero-Shot-Learning consists of learning new categories and classes without having been exposed to instances of them in the past [4]. In the context of this project, Zero-Shot-Classification refers to a classification model that can classify a question input by topic based on a series of provided labels, even if those labels were not present in the training set. An example of the implementation of zero-shot-classification used in our work can be seen in 5.There is evidence from GPT-2 and the large BERT models that training a model on a huge amount of data with many parameters can help the model perform tasks it has not been fine-tuned for [12].

Figure 5: An example code snippet and output of Zero-Shot classification. [4]

CHAPTER IV

METHODOLOGY

This chapter will include a high level overview of iLFQA, a detailed explanation of each module, and a description of the datasets used to train the various models. In addition, this chapter provides explanation of the metrics we used for quantitative evaluation.

4.1 SYSTEM OVERVIEW

iLFQA leverages three distinct methods of machine learning to create a generative question answering platform. When a question is asked, it is sent to a classifier module. The classifier associates the question with one or more topics that are present in iLFQA's knowledge corpus. Each topic is associated with a corresponding set of documents. Once a question is assigned a topic label, only the documents associated with that label are retrieved. iLFQA then calculates the TF-IDF of each paragraph in the given documents to determine the passages most likely to contain relevant answers. The 5 most likely paragraphs are then scored, and the likely answers from each paragraph are extracted. At this point, the possible answers and the highest scoring paragraph are sent to a text generation module. The text generation module takes the input text and synthesizes a long form answer for the input question. A diagram of the entire system can be seen in 6.

4.2 SYSTEM MODULES

4.2.1 Classification Module

When a question is given to iLFQA, the question is passed to a classification module. Our classification module uses a BERT model to embed the question and a list of topics into the same space. Then the topic that is most similar to the question is selected. Each topic has a list of associated documents. The classification module's main purpose is to limit the amount of data that needs to be processed for any given question. Without classifying the question, iLFQA will be forced to calculate TF-IDF for every paragraph in the knowledge corpus. Classifying questions helps increase the accuracy of the responses and improves the overall response time. Once a question has been classified, the appropriate documents are retrieved, and they are sent to the passage selection module along with the user's question.

4.2.2 Passage Selection Module

The passage selection module accepts a series of tokenized documents and the user's question as input. TF-IDF is calculated for every paragraph in the input document(s). We select the five most relevant paragraphs by the TF-IDF calculation. Once we have selected the best candidates, those paragraphs are sent through a BERT [13] model that calculates a confidence score for each paragraph. The confidence score is used to determine which of the five most relevant paragraphs contains the correct answer. In addition to a confidence score, we also determine the span of text within each paragraph that may contain the answer. iLFQA is designed to answer questions in a long-form format. However since the spans are only a single sentence, they are only one part of the input of our text generation module. We concatenate the best spans and the paragraph with the highest confidence score together and pass that as an input for our text generation module.

4.2.3 Text Generation Module

The text generation module takes the user's question, a series of possible answer spans, and a single paragraph as input. We use a BART model pre trained on the ELI5 [10] dataset to produce our text. The BART model is sequence to sequence text generation model. It takes the text we have generated as input, and the question, and attempts to abstract an answer. The ELI5 dataset is comprised of a series of long form answers to questions. Answers are encouraged to be explanatory and easy to understand. The text to text model abstracts the information from the best answer spans and the selected paragraph into an answer that addresses the user question.

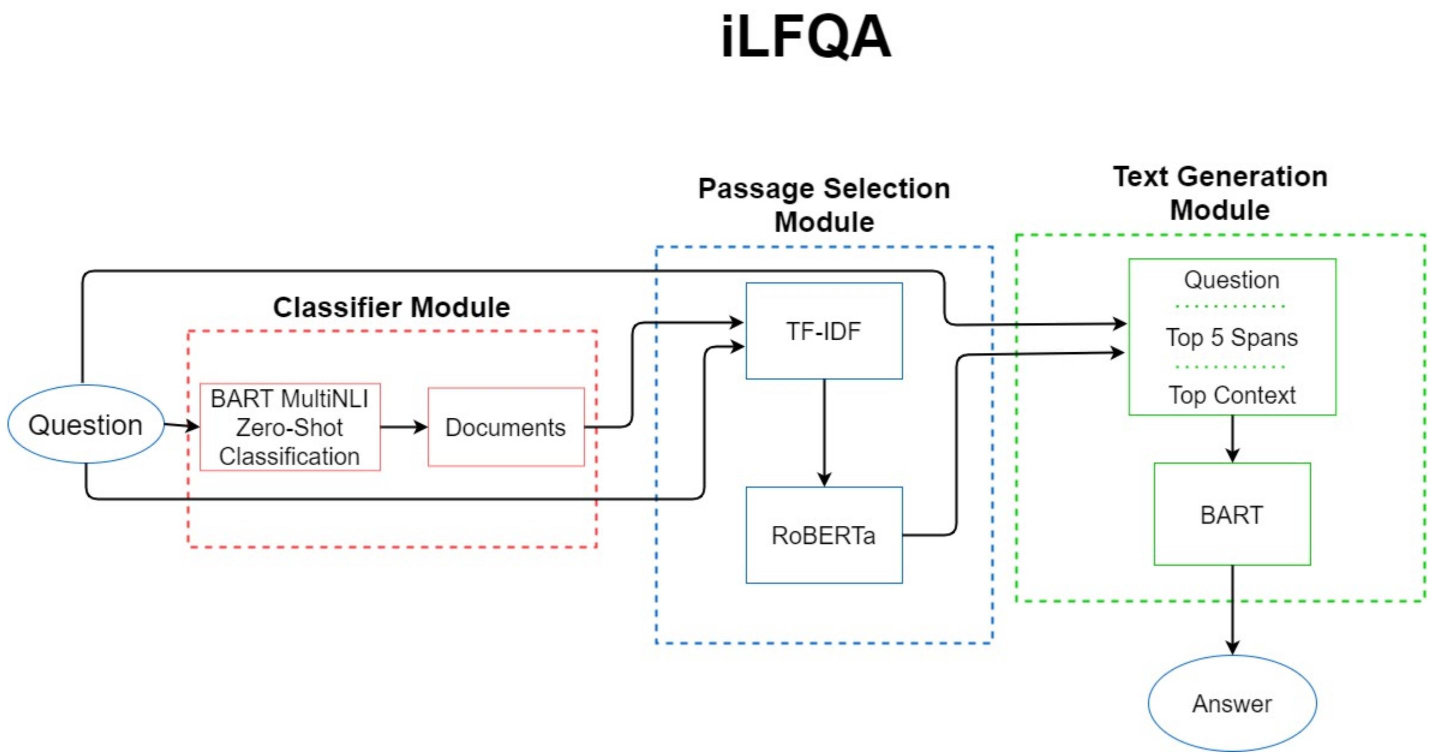

Figure 6: A diagram of iLFQA.

CHAPTER V

EXPERIMENTATION

5.1 Setup

All results were obtained on a Google Cloud Platform Virtual Machine instance. Our VM instance uses an Intel Broadwell CPU, one NVIDIA TESLA K80 GPU with 12GB of memory, and a 64GB persistent SSD. The operating system is Ubuntu 18.04 (Bionic Beaver). The source code for iLFQA, as well as all the results presented in this paper are available at the following location: https://github.com/rhysb05/iLFQA. Our method of calculating TF-IDF originates from a heuristic proposed by Clark and Gardner in their paper "Simple and Effective Multi-Paragraph Reading Comprehension" [14]. We process the term frequency for each paragraph without considering the entire corpus. By doing so we can avoid giving too much weight to words that might be very common in the entire corpus. For instance, if we wanted to ask a question about training a dog to do a trick, the word dog probably appears often in the document that we inspect for the answer. By calculating the term frequency of each word in only one paragraph, the word dog becomes less important, and the other words such as trick, will be weighted more heavily. This helps select relevant paragraphs. We follow their research specifically because they sought to improve neural-net-based model multi-paragraph reading comprehension. The questions we used to test our system were selected heuristically. The knowledge corpus that iLFQA has available to it are textbooks. Questions were written to have existing answers in some paragraph of the entire set of texts. In the raw results, the question, and the paragraph containing the answer can be seen side by side. The raw data for the results presented in this report are available at the github repository. It is common to train and evaluate natural language processing models on different datasets and compare your own results to previous benchmarks. We will briefly describe the datasets that were used to train the models in iLFQA.

MultiNLI

The Multi-Genre Natural Language Inference corpus, MultiNLI for short [15], is used in the Zero-Shot classification module. MultiNLI consists of a large number of entries that consists of a sentences and genre labels. MultiNLI is appropriate for our zero-shot classification model because we are expecting single sentence inputs from the user. The zero-shot classification model uses a BART model pre trained on MultiNLI.

SQuAD 2.0

Our implementation includes a language comprehension model trained on SQuAD 2.0. SQuAD stands for the Stanford Question Answering Dataset. The original SQuAD dataset contained questions paired with a context (a paragraph) that contains the answer to the question. SQuAD 2.0 [16] includes many unanswerable questions. In SQuAD 2.0 the unanswerable questions were designed to have relevance with the paired context. So, a model trained on SQuAD 2.0 would need to be able to understand that at times, a question is unanswerable even when a plausible answer exists in the given context [16]. The BERT model used in iLFQA is RoBERTa [13]. RoBERTa was trained and evaluated on SQuAD 2.0.

ELI5

ELI5 [10] was created to address the issue of long form question answering. The authors of the ELI5 paper state that they sought to create a dataset that could train models to answer open ended questions requiring in depth explanations. The ELI5 paper notes that most question answering datasets are extractive. Meaning, the answer exists within a span of text in each paragraph, and there is only one correct answer. ELI5 is made up of questions and lengthy answers from the Reddit subreddit "Explain Like I'm Five". Answers to questions on this subreddit are encouraged to be easily understandable. The authors of the ELI5 paper do note that models trained on their dataset are not perfect and that human raters prefer the gold standard answers to answers generated by models trained on ELI5 86% of the time [10].

5.2 Results

We have evaluated iLFQA by two metrics named BERTScore [17] and BLEU. BLEU stands for Bilingual Evaluation Understudy. These scores were obtained by submitting a total of 115 questions to iLFQA. BLEU evaluates a machine generated text by matching n-grams between the generated text and the reference text. An n-gram is a sequence of text that appears contiguously. BLEU can be calculated by setting n to different values. Typically, the higher the value of n, the more rigorous the evaluation.

BERTScore and BLEU Evaluation			
BERTScore P	BERTScore R	BERTScore F1	BLEU
0.836836127	0.8337135	0.834965846	0.476107211

Table V..1: iLFQA evaluation metrics

iLFQA Response Times			
Classification Module	Passage Selection Module	Text Generation Module	Total
0.153520549	1.831399663	8.217388176	10.20230839

Table V..2: iLFQA times by module

BLEU is an excellent method for evaluating direct translation, however BLEU fails to capture semantic meaning. There are few methods for evaluating the semantic correctness of generative question answering platforms. We have chosen BERTScore because it was designed to go beyond n-gram matching. BERTScore uses a pre-trained BERT model to evaluate the cosine similarity between the generated text and the reference text. The embeddings of the generated text and reference text are compared to determine the similarity in meaning between the two texts. This method of evaluation is more rigorous and appropriate for this application than BLEU. The similarities of the two texts are compared token by token. Precision, recall, and F1 are calculated based on the matches by similarity of tokens in the languages embedding space. BERTScore was designed to evaluate machine translation, but it is appropriate for iLFQA. BERTScore is used to compare the similarity of the input and output of the text generation module. The average BLEU and BERTScore for iLFQA can be seen in figure V..1. BLEU and BERTScore were calculated for individual, question response. The scores in V..1 are the average of the evaluation of all questions from iLFQA. We present the response time for each module of iLFQA in V..2, the distribution of response times can be seen in 7.

Initially an LSTM model was used to calculate the confidence scores during paragraph selection. Using an LSTM model as part of our passage selection module resulted in total response times close to 50 seconds on average. The classification module was added because it increased the accuracy and response time of the entire system. The results of iLFQA with and without the classification module are shown in V..3. Overall accuracy and response times were increased. It should be noted that the response time was decreased even though there is extra computation involved in the classification module. This overall improvement is due to the fact that classifying questions allows us to process fewer total tokens when selecting a passage.

Classification Comparison					
Version	BLEU	BERTScore P	BERTScore R	BERTScore F1	Response Time
iLFQA	0.476107211	0.836836127	0.8337135	0.834965846	10.20230839
without classification	0.481451919	0.837526725	0.831856726	0.83424287	10.78334703

Table V..3: iLFQA results with different heuristics.

Many heuristic approaches to improving the accuracy and performance were attempted during the implementation of iLFQA. Several different combinations of best span and best paragraph concatenation were tried. These methods included using only best spans, using all the paragraphs retrieved at the TF-IDF stage, using two of the best paragraphs, and several other methods. The results of those methods can be seen in V..4.Each version indicates a different method for generating the input for the text generation module. Five best answers sends the four best answer spans, and the highest scored paragraph to the text generation module. Four best context sends the four highest scoring paragraphs to the text generation module. Two answer two para sends two spans and two paragraphs to the text generation module. Two answer one para sends two spans and the highest scored paragraph to the text generation module. The configuration we have settled on uses the method described for five best answers. In addition to

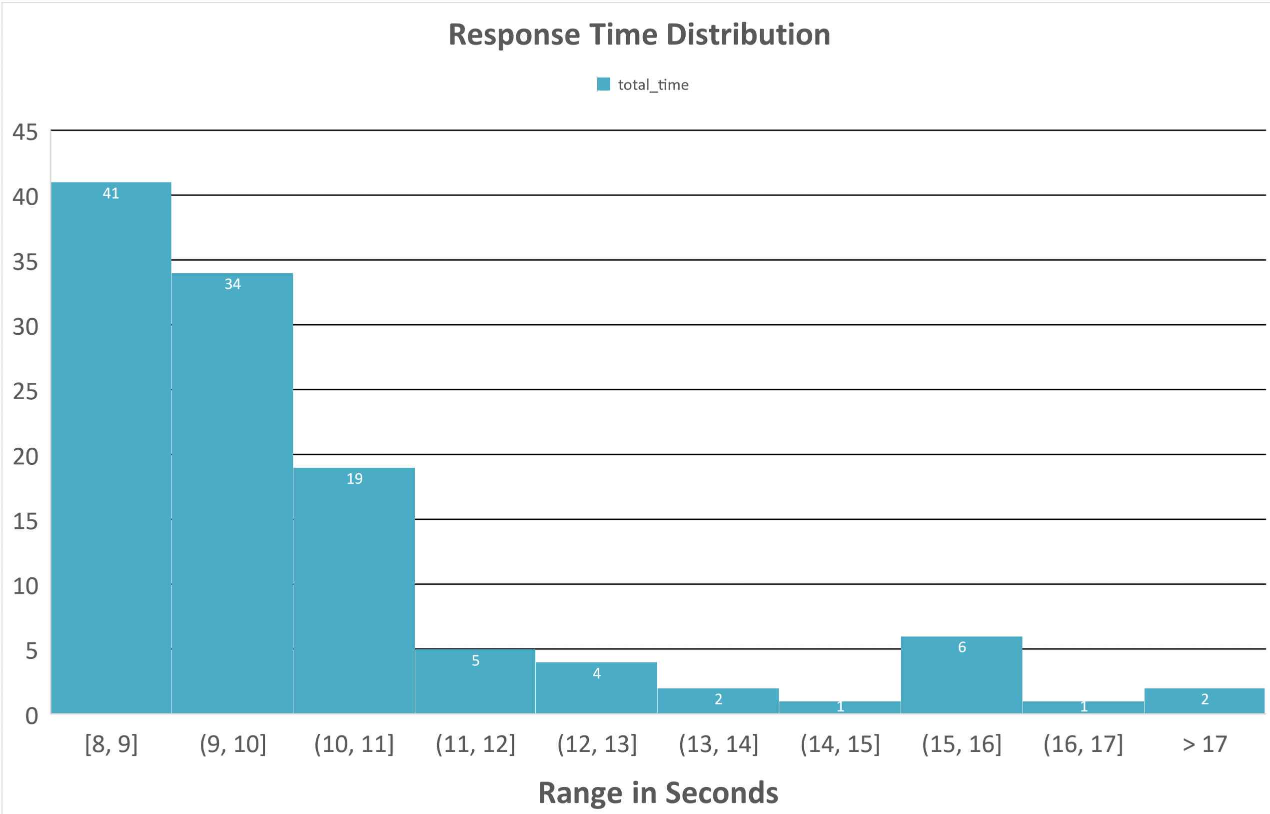

Figure 7: Histogram representing distribution of response times for 115 questions.

Context Generation Comparison			
Version	BLEU	Perplexity	Response Time
five best answers	0.476107211	72.9835029	10.20230839
four best context	0.478781827	73.62670021	10.21829904
two answer two para	0.47847055	72.8883316	10.1626531
two answer one para	0.480523763	73.39233174	10.08110274

Table V..4: iLFQA implemented with different context selection methods

different heuristic approaches to constructing the input for the text generation module, different methods of text generation were experimented with. T5 [18], GPT-2, and an alternate BART model were all experimented with. For each different experiment, we removed the current iLFQA text generation model and replaced it. The metrics for the results of those experiments can be seen in V..5. One abstractive summarization model based on the T-5 architecture [18] sometimes produced results that were acceptable and semantically correct very quickly. This behavior was not consistent. Only the model used in the final iLFQA implementation has been trained on the ELI5 dataset. Training the models we did not select on a proper long form dataset would yield much better results. Finally, we present some examples of iLFQA output. See 8 and 9 for an example of an iLFQA response. Note that for 8 and 9 the answers are contained within iLFQA's knowledge corpus.

Different text Generation model architectures					
Version	BLEU	BERTScore P	BERTScore R	BERTScore F1	Response Time
iLFQA	0.476107211	0.836836127	0.8337135	0.834965846	10.20230839
T-5	0.697096606	0.843155287	0.805740035	0.82375022	5.252131491
GPT-2	0.558380775	0.810880085	0.810336336	0.81031971	5.399227395
DistilBART	0.539802913	0.820264539	0.817357293	0.818603523	6.840499799

Table V..5: Experiments with different text generation architectures

Question:

How does Data Science use statistical techniques?

Selected context:

make robust conclusions based on incomplete information Applications to real data sets choosing between hypotheses based on data in random samples careful reasoning attempt to choose between two views of the worldFor whatever aspect of the world we wish to study – whether it ' s the Earth ' s weather , the world ' s markets , political polls , or the human mind – data we collect typically offer an incomplete description of the subject at hand . A central challenge of data science is to make reliable conclusions using this partial information . In this endeavor , we will combine two essential tools : computation and randomization . For example , we may want to understand climate change trends using temperature observations . Computers will allow us to use all available information to draw conclusions . Rather than focusing only on

Figure 8: Question

iLFQA does not perform well when a key phrase from the question is missing in the knowledge corpus, see 10 and 11.. In these cases, iLFQA will return a response that is grammatically correct, but semantically incorrect. That is to say, the question may read as if it was human generated, but it will lack meaning with regard to the question. If the specific information that is asked for does not exist in the iLFQA reference documents, the text generation model will attempt to synthesize a response based on the question. Ultimately iLFQA returns an answer that

Generated answer:

Data Science doesn't use statistical techniques, it uses data analysis techniques. Data analysis is the study of data and how it relates to other data. For example, if you want to know the average temperature of a region, you can look at the average temperatures of a bunch of people in that region. If you look at that data you can make a pretty good guess as to what the average is. You can also look at how the data relates to the other data you're looking at, and use that to make a more nuanced analysis.

Figure 9: Response

Question:

How old was Tim Duncan when he won his first championship?

Selected context:

3.2 SDs above average 100 years chance 18 100 " years Sometimes , adjustments have to be made in the first or last bin , to ensure that the smallest and largest values of the variable are included . You saw an example of such an adjustment in the Census data studied earlier , where an age of " 100 " years actually meant " 100 years old or older . " We can see that there are 10 bins (some bars are so low that they are hard to see) , and that they all have the same width . We can also see that none of the movies grossed fewer than 300 million dollars ; that is because we are considering only the top grossing movies of all time . It is a little harder to see exactly where the ends of the bins are situated . For example , it is not easy to pinpoint exactly where the value 500 lies on the horizontal axis . So it is hard to judge exactly where one bar ends and

Figure 10: Hard Question

Generated answer:

Depends on how old he was when he won his first championship. If he was 16, then he would have been 18 years old at the time of the championship. But if he was 23, then it would be a little over a year older than the NBA's maximum age limit, which is 25 years old. So Tim Duncan would have turned 23 on the day he won the championship, which would have made him 23 years old when the NBA changed the age limit to 25 years.

Figure 11: Hard Response

briefly mentions the key topic of the question, but does not answer the question. At times when the key phrase or topic presented by the question is not contained in the reference documents, the text generation model will pull some information from the data that it was pre trained on. In this case it is obvious that the ELI5 dataset contains some questions and explanations about the NBA. Even though it supplies this information in the response, the answer does not provide an adequate explanation for the question. We have managed to deploy a prototype version of our system to Google Assistant. Currently google assistant requires that it receive a response from any application associated back-end within 10 seconds. It can be seen from our results that we are not able to consistently hit that mark. We could significantly improve our response times by upgrading the GPU used in our implementation. However, this is a proof of concept, and should show that with reasonable resources, systems like ours can be deployed. This opens up opportunities for more rigorous testing and increased human evaluation.

CHAPTER VI

CONCLUSIONS

The field of question answering has been advancing rapidly in recent years. Some of the most promising advances, such as Transformer architecture, have improved our ability to perform generative question answering. We have implemented a question answer platform, named iLFQA: intelligent Long-Form Question Answering, that seeks to take advantage of these and other advances in natural language processing.. iLFQA seeks to provide accurate, meaningful, and explanatory answers to user generated questions in a reasonable amount of time. iLFQA works by accepting a question as input, classifying the question by topic, and deducing the correct answer from an associated document. The information for the correct answer is synthesized into a response by and returned to the user. The architectures and systems that iLFQA leverages have been presented along with a detailed overview of each module. iLFQA has been evaluated with a meaningful metric that most accurately captures its semantic performance. There are very few long form generative question answering systems that currently exist. It is our hope that this work will contribute to progress in that field.

CHAPTER VII

FUTURE WORK

It seems that a promising area of future work on this project or related subjects could be to fine tune an abstractive summarization model on a long form documents. Many summarization models produce single sentence outputs and may repeat themselves when longer outputs are required. However very few examples of research on long-form summarization exist. Future work on iLFQA should consider training a summarization module on long form documents and using that as a portion of the text-generation module. This might require actually creating a suitable database. Some summarization databases used for pre training and fine tuning provide single sentence explanations of whole articles. This is not ideal for our work. Also, as mentioned earlier. BERTScore [17] is best used as an evaluation of machine translation. We can use it in our case because the reference context and generated text can be compared for evaluation in a manner consistent with how our system was designed. However, this is not an ideal metric to capture the correctness of the generated answer as it relates to the original question. A recently proposed metric named KPQA [19] may capture the relationship between then question and response more accurately. KPQA determines key phrases in the question and attempts to determine if machine generated answers capture the meaning of the key phrases from the question. Future implementations of iLFQA should incorporate this metric or any similar future metrics that are designed to evaluate the semantic accuracy of a generated response.

REFERENCES

[1] S. Min, J. Michael, H. Hajishirzi, and L. Zettlemoyer, "AmbigQA: Answering ambiguous open-domain questions," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Online), pp. 5783–5797, Association for Computational Linguistics, Nov. 2020.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[3] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, (Online), pp. 7871–7880, Association for Computational Linguistics, July 2020.

[4] T. Romera-Paredes B., "An embarrassingly simple approach to zero-shot learning," 2017.

[5] Y. W. Chandra and S. Suyanto, "Indonesian chatbot of university admission using a question answering system based on sequence-to-sequence model," *Procedia Computer Science*, vol. 157, pp. 367–374, 2019. The 4th International Conference on Computer Science and Computational Intelligence (ICCSCI 2019) : Enabling Collaboration to Escalate Impact of Research Results for Society.

[6] A. Chowanda and A. D. Chowanda, "Generative indonesian conversation model using recurrent neural network with attention mechanism," *Procedia Computer Science*, vol. 135, pp. 433–440, 2018. The 3rd International Conference on Computer Science and Computational Intelligence (ICCSCI 2018) : Empowering Smart Technology in Digital Era for a Better Life.

[7] S. Kim, O.-W. Kwon, and H. Kim, "Knowledge-grounded chatbot based on dual wasserstein generative adversarial networks with effective attention mechanisms," *Applied Sciences*, vol. 10, no. 9, 2020.

[8] Z. Tu, Y. Jiang, X. Liu, L. Shu, and S. Shi, "Generative stock question answering," 2018.

[9] K. Krishna, A. Roy, and M. Iyyer, "Hurdles to progress in long-form question answering," 2021.

[10] A. Fan, Y. Jernite, E. Perez, D. Grangier, J. Weston, and M. Auli, "ELI5: Long form question answering," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 3558–3567, Association for Computational Linguistics, July 2019.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference*

of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), (Minneapolis, Minnesota), pp. 4171–4186, Association for Computational Linguistics, June 2019.

[12] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models are Unsupervised Multitask Learners," 2019.

[13] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized BERT pretraining approach," 2019.

[14] C. Clark and M. Gardner, "Simple and effective multi-paragraph reading comprehension," *CoRR*, vol. abs/1710.10723, 2017.

[15] A. Williams, N. Nangia, and S. Bowman, "A broad-coverage challenge corpus for sentence understanding through inference," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, (New Orleans, Louisiana), pp. 1112–1122, Association for Computational Linguistics, June 2018.

[16] P. Rajpurkar, R. Jia, and P. Liang, "Know what you don't know: Unanswerable questions for SQuAD," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, (Melbourne, Australia), pp. 784–789, Association for Computational Linguistics, July 2018.

[17] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with BERT," *CoRR*, vol. abs/1904.09675, 2019.

[18] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *CoRR*, vol. abs/1910.10683, 2019.

[19] H. Lee, S. Yoon, F. Dernoncourt, D. S. Kim, T. Bui, J. Shin, and K. Jung, "Kpqa: A metric for generative question answering using keyphrase weights," 2020.