Comprehensive Natural Language Question Answering System

Manasa Krishnan
Khoury College of Computer Sciences
Northeastern University
Boston, USA
krishnan.man@northeastern.edu

Sriram Hariharan Neelakantan
Khoury College of Computer Sciences
Northeastern University
Boston, USA
neelakantan.s@northeastern.edu

Nandavardhan Chirumamilla
Khoury College of Computer Sciences
Northeastern University
Boston, USA
chirumamilla.n@Northeastern

***Abstract*—The study of natural language processing has increased in recent years, leading to the development of question-answering systems using conversational AI bots. These systems are useful in providing responses to queries posed by human users. Closed domain and open domain QA models exist, but we focus on a closed domain QA model using BERT and BiLSTM in this project. BERT and BiLSTM are complex deep learning models that have proven useful in understanding language and dealing with long-term dependencies. The project uses the Stanford Question answering dataset to develop a system that accurately answers queries based on a specific domain of knowledge using a combination of deep learning and natural language processing techniques.**

***Index Terms*—Question Answering, BERT, BiLSTM**

I. Introduction

In recent years, the study of natural language processing (NLP) has grown significantly. The creation of question-answering systems is one of the most intriguing and useful uses of NLP. These programs resemble conversational AI bots, which provide replies to queries posed by human users. Since it could take some time to browse through multiple documents to get the best solution, a question-answering system is highly helpful in many situations. There are both closed domain and open domain QA models. Open domain models receive responses from any domain, which may contain a vast corpus, as opposed to closed domain models, which only accept responses from a specific domain on which the dataset is built upon.

Over the years, a number of different approaches have been proposed including statistical methods, rule-based methods and machine learning methods for this domain. In this project, we focus on a closed domain QA model using BERT and BiLSTM to respond to queries based on a specified corpus of text. Question answering is a very challenging task for a machine, as it needs to understand the language and also needs relevant information to answer the question. So, to deal with such a complex task, we make use of complex machine learning models such as BERT and BiLSTM. With the introduction of transformer models, it has boosted the performance in tasks such as question answering significantly.

Identify applicable funding agency here. If none, delete this.

Bidirectional Encoder Representations from Transformers (BERT) is one such transformer-based model that has been trained on a very large corpus of text in an unsupervised fashion which has attained state of the art performance on many NLP tasks. The pre-trained BERT model is an excellent option for creating question answering models because it can be fine-tuned to a particular task.

Bidirectional Long Short-Term Memory (BiLSTM) is a type of recurrent neural network that can learn long-term dependencies much better than the standard recurrent neural networks. BiLSTMs proved to be very useful in dealing with tasks such as language modelling, text classification and POS tagging. Its unique ability to deal with long-term dependencies makes it very useful in building a question answering system.

To deal with the problem at hand, along with complex models that can model the data accurately, we also need a large and high quality reading comprehension dataset. We use the Stanford Question answering Dataset [8] for this purpose in our project.

To summarize, the objective of this project is to develop a system that can accurately answer questions based on a specific domain of knowledge. The model will be trained on a dataset of questions and answers and will use a combination of deep learning techniques and natural language processing techniques to generate accurate responses to user queries.

This report outlines the methodology used to build the QA model and the results obtained from the evaluation. We begin by discussing other related works in this domain, followed by going over the dataset and evaluation metrics used in the project. The subsequent section describes the BERT and BiLSTM models along with the baseline configurations, their training processes and the ablations settings used. We then present the results of the evaluation, followed by a conclusion of the project. Overall, this project aims to build a framework capable of performing question answering when given a context and a question.

II. Related Works

Recently, several academics have concentrated on Question Answering (QA) systems. Building effective QA models has been researched using a variety of methods, including statistical tests, machine learning, and rule-based systems. In this

```
{"data": [
{"title": "University_of_Notre_Dame", "paragraphs": [
{"context": "Architecturally, the school has a Catholic character. Atop the Main Building's gold dome is a golden statue of the Virgin Mary. Immediately in front of the Main Building and facing it, is a copper statue of Christ with arms upraised with the legend \"Venite Ad Me Omnes\". Next to the Main Building is the Basilica of the Sacred Heart. Immediately behind the basilica is the Grotto, a Marian place of prayer and reflection. It is a replica of the grotto at Lourdes, France where the Virgin Mary reputedly appeared to Saint Bernadette Soubirous in 1858. At the end of the main drive (and in a direct line that connects through 3 statues and the Gold Dome), is a simple, modern stone statue of Mary.",

"qas": [{"answers": [{"answer_start": 515, "text": "Saint Bernadette Soubirous"}], "question": "To whom did the Virgin Mary allegedly appear in 1858 in Lourdes France?", "id":
"5733be284776f41900661182"},

{"answers": [{"answer_start": 188, "text": "a copper statue of Christ"}], "question": "What is in front of the Notre Dame Main Building?", "id": "5733be284776f4190066117f"},

{"answers": [{"answer_start": 279, "text": "the Main Building"}], "question": "The Basilica of the Sacred heart at Notre Dame is beside to which structure?", "id":
"5733be284776f41900661180"},

{"answers": [{"answer_start": 381, "text": "a Marian place of prayer and reflection"}], "question": "What is the Grotto at Notre Dame?", "id": "5733be284776f41900661181"},

{"answers": [{"answer_start": 92, "text": "a golden statue of the Virgin Mary"}], "question": "What sits on top of the Main Building at Notre Dame?", "id": "5733be284776f4190066117e"}]}]}]}
```


Fig. 1. Sample SQuAD data in json format

section, we contrast the most recent articles on QA systems and talk about the variations in the researchers' methodologies.

Rule-based systems are among the earliest and most widely used techniques to QA systems. These systems extract data, analyze it, and produce results using pre-established criteria. However, because they rely so significantly on the accuracy and completeness of the rules, rule-based systems have limits. Numerous research studies have demonstrated that machine learning-based systems are more capable of producing accurate and efficient results than rule-based systems [4, 5].

QA systems that rely on machine learning have performed significantly better than other approaches, but this performance comes with a trade off on computationally intensive operations that need to be performed. Chuang Zheng et al [1] proposed a three-layered model that used BERT for model embeddings and BiLSTM for feature extraction. The model achieved an accuracy of approximately 78%. HongLiang Wang et al [2] proposed a combination of BERT, Neural, and Siamese Networks that outperformed a single BERT or Siamese network with an F1 score of 90%.

In QA systems, statistical tests like significance testing and hypothesis testing have also been applied. Different models can be compared, and their performance can be assessed using statistical tests. As an illustration, Han et al. [6] employed significance testing to evaluate the effectiveness of several QA models. According to the study, the BERT-based model performed better than the other models in terms of being efficient and accurate.

In conclusion, rule-based systems have drawbacks while machine learning-based approaches have demonstrated promising outcomes in developing effective QA systems. To assess and compare the effectiveness of various models, statistical tests can be performed. To create QA systems that are more efficient, further research can explore the integration of these methodologies.

III. Benchmark Dataset and Evaluation Metrics

A popular benchmark dataset for question answering language models is the Stanford Question Answering Dataset[8] (SQuAD)1.0 consisting of a large collection of Wikipedia articles. These articles are accompanied by 100,000 question-answer pairs.The data has been split into train,test and validation data for training purposes. Fig 1. shows a sample of the SQuAD dataset in json format.

The main evaluation metric chosen for our project is f1-score. The formula for the f1- score is as shown in Eq 1.

$$f1\text{-}score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{1}$$

$$\text{Avg. } F1\text{-}score = \frac{1}{N} \sum_{i=1}^{N} f1\text{-}score \tag{2}$$

F1-score is the harmonic mean of precision and recall which provides a good inference of the working of a model. The average F1-score across all question-answer pairs is the overall f1-score of the model. The formula for average f1-score is given in Eq 2.

IV. Methodology

A. Preprocessing

A combination of machine learning and deep learning techniques have been used in our approach to achieve optimal performance. Depending on the deep learning model being used, the level of preprocessing differs which has been discussed using ablation studies. After loading the data as question, context and answer triplets for each training example, we do a certain base level of preprocessing followed by additional preprocessing as per the ablation studies and then we create start and end tokens of the answer in the context, which are used during training. The base level of preprocessing involves converting text to lower case and tokenization. We convert questions, context and answers into lower case before any subsequent preprocessing.

Tokenization is the process of splitting a given sentence or corpus into individual words also called tokens. This is a very important step taken to break down sentences into words so

Table 1. Ablation Settings with BERT base model

No.	Pretrained Weights	Pre-processing	#Transformer layers	Dropout	Optimizer	Validation loss
1	BERT Base	SW, LM	6	0.1	Adam	1.23
2	BERT Base	SW	6	0.1	AdamW	1.21
3	BERT Base	SW, ST	12	0.2	AdamW	1.18
4	BERT Base	LM	12	0.1	AdamW	1.33
5	BERT Base	-	6	0.2	AdamW	1.19
6	BERT Base	-	6	0.1	AdamW	1.20
7	BERT Base	-	12	0.1	AdamW	1.10
8	BERT Base	-	12	0.2	Adam	1.25
9	BERT Large	-	24	0.1	AdamW	0.68

that we can build a vocabulary. For BERT and DistilBERT, we use the WordPiece tokenizer which has the advantage of being able to handle words that are not commonly used and record word morphological changes, which can enhance the performance of the model on specific tasks. We do not remove any text from the corpus like numbers, special characters or any sequence pattern because removal of any text from the data can be treated as loss of information when dealing with tasks such as question answering systems where the required answer can be anything from the given context. Other preprocessing techniques included in our approach are removal of stop words, tokenization, stemming and lemmatization.

Removal of stop words involves removal of words that do not carry any significant meaning such as “is”, ”and”,” the”, etc. Removal of stop words is very useful especially in situations where the frequency or counts of words is used in generating embedding vectors or for modeling purposes. But in our case, removal of stop words may result in the loss of context in many scenarios, hence we decide on the inclusion of stop words based on the ablation results. Stemming and lemmatization are methods of converting a word into its root form, this process aids in grouping words with similar meanings together. For instances, words such as “acted”, “acting”,” acts” are all reduced to the root word “act”. Stemming and lemmatization differ from each other in terms of the process used to attain the root words. In stemming, the suffix of the word is removed to attain the root word whereas in lemmatization, it makes use of vocabulary and morphological analysis to reduce words to the root form.

B. Embedding

Embedding vectors are generated to represent words as dense vectors that capture the words semantic and syntactic characteristics. These techniques enable the model to extract relevant features from the text data, making it more effective in understanding the context of the questions and providing accurate answers. In the BiLSTM approach, different embedding techniques like tf-idf, Glove, count vectorizer and DistilBERT were used to find the optimal embedding technique that can maximize performance. GloVe vectors are built using a big corpus of text’s co-occurrence matrix of words. The underlying premise is that words that commonly occur together in the same situations are probably related to one another or have meanings that are like one another. Using this data, GloVe creates word vector representations that accurately reflect their semantic links with other words. DistilBERT is a much lighter version of BERT that has been used to generate embedding vectors in our case.

C. Approach

Baseline

The baseline model of our project is the BERT with these configurations: Pre-processing involves stop words removal, pretrained BERT base weights with 6 transformer layers, dropout rate of 0.1 and AdamW optimizer. The weight decay regularization is already present in AdamW’s optimizer unlike Adam in which weight decay is added after each update. This can enhance the model’s overall performance by enabling the optimizer to more effectively regulate the weight decay term’s magnitude.

This model produced a loss of 1.21. The metrics used are the same as that of the benchmark and we obtained a precision of 0.80, recall of 0.646 and f1score of 0.715. We tried ablation techniques, changed configurations and improved this score.

BERT

We have utilized two main algorithms for our project: BERT (Bidirectional Encoder Representations from Transformers) and BiLSTM(Bidirectional Long Short-Term Memory). Both these algorithms perform exceptionally in understanding the context of words, which is a necessity for our problem. The input features are the contexts and questions, and the output features are the start and end positions of the answers which would then be converted into texts.

By taking into account the words that occur before and after each word in the input sentence, BERT’s transformer-based design enables it to encode the context of each word. As a result, the model can comprehend the context in which the words are being used and capture intricate links between words and phrases.

The model uses simplified cross entropy where the distribution is provided as one-hot encoding. We have further performed many fine-tuning techniques which will be reported later. One method is we introduced a better method to correctly assess the start and end positions. As BERT model is sometimes robust to space and no characters in the answers, especially if they occur as end tokens, we made the model ignore such special characters during prediction. After preprocessing the dataset, we move on with the steps involved in BERT which are

Table 2. Ablation settings with BiLSTM model

No.	Model	Embedding Types	Pre-processing	#layers	Loss Function	Optimizer	Validation Loss
0	BiLSTM	Count Vectorizer	SW, LM	2	Cross Entropy	ADAM	5.79
1	BiLSTM	Glove	SW, LM	2	Cross Entropy	ADAM	3.84
2	BiLSTM	Tf-IDF	SW, LM	2	Cross Entropy	ADAM	4.87
3	BiLSTM	DistilBERT	SW, LM	2	Cross Entropy	ADAM	3.72
4	BiLSTM	DistilBERT	LM	2	Cross Entropy	ADAM	4.22
5	BiLSTM	DistilBERT	ST	2	Cross Entropy	ADAM	4.37
6	BiLSTM	DistilBERT	–	2	Cross Entropy	ADAM	4.69
7	BiLSTM	DistilBERT	SW, LM	4	Cross Entropy	ADAM	3.50
8	BiLSTM	DistilBERT	SW, LM	8	Cross Entropy	ADAM	3.43

Tokenization: The contexts(paragraphs) and questions are tokenized into unit words and special tokens using the WordPiece Tokenizer. [CLS] indicates the beginning of the sequence, [SEP] indicates the end of contexts and questions separately and [MASK] can mask some input tokens so that the model can learn to predict unseen tokens based on the context. The [MASK] token is optional.

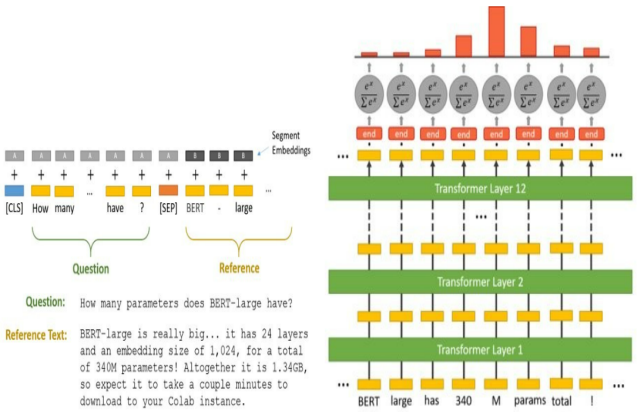

Fig. 2. Architecture of BERT base

Encoding: Two variations of BERT models(encodings) are used for our project: BERT base and BERT large. BERT base has 12 transformer blocks and is trained on 110 million parameters, whereas BERT large has 24 transformer blocks and is trained on 340 billion parameters. Our project is mostly based on BERT base as it was compatible with our computing power.

Prediction: The final classifier layer of BERT will predict the start and end positions of the answer. The span(start,end) with the highest probability is chosen as the final output. The model provides the textual answer with the help of the span of the positions in the contexts.

We have performed several variations of fine-tuning to select the best model for our project.

Ablation Studies with BERT

We experimented with several variations of the BERT model. Some of the pre-processing techniques included are removal of stop words, stemming and lemmatization. With the computing power, we were able to train only one BERT large model. The optimizers used are AdamW, Adam. The variations of transformer layers are 6, 12 and 24 and the attention heads are 12 and 16. We varied the dropout rates between two values, 0.1 and 0.2, but this rate did not have a major change in the result. If we vary the number of layers and attention heads, we will know the effects of overfitting or underfitting and try to understand the contextual representation made by BERT with respect to our dataset and choose the optimal number. Even though dropout did not have any effect, we tried to infer if there was a chance of overfitting especially when we dealt with 16 layers. Table 1 shows the results of various ablations studies discussed here, which are further discussed in detail in the Evaluation section.

BiLSTM

We have implemented a BiLSTM model to compare it with the BERT model. BiLSTM is a common choice for sequence labeling problems because it can capture dependencies between past and future contexts in the input sequence.

The model is trained to estimate where the answer will begin and end inside the supplied section. A sequence of tokens is fed into the model, which is then sent via an embedding layer to generate dense vector representations of the tokens. These representations are then passed into the BiLSTM layer, which captures the contextual information of each token in both the forward and backward directions based on its nearby tokens. The BiLSTM layer output is then sent through a linear layer to generate the start and end logits, which denote the likelihood of each token being the beginning or ending of the response.

Tokenization and Embedding: The main model uses a BERT tokenizer and embedding model. The Bidirectional Encoder Representations from Transformers (BERT) model has been pre-trained on a vast text corpus, allowing it to generate high-quality embeddings for any given text input. The BERT tokenizer is used to preprocess incoming text into a sequence of tokens that can be fed into the BERT model. It employs WordPiece tokenization, which divides the input text into subwords. This enables the model to handle terms that are not in the model's lexicon and to capture the semantics of phrases that may have numerous meanings. In the context of the project , the BERT tokenizer and embedding model are used to preprocess the input text and generate embeddings for the text and the question. The embeddings are then supplied into a BiLSTM model, which performs the actual work of predicting the start and end positions of the answer in the input text.

The BiLSTM is made up of two LSTM layers: one that processes the input sequence forward and one that processes

Table 3. Performance Comparisons of different BERT Model on Test Data

Model	Pre-processing	#Layers	LR	Optimizer	F1 Score
BERT Base	SW, LM	6	0.1	Adam	0.723
BERT Base	SW	6	0.1	AdamW	0.715
BERT Base	SW, ST	12	0.2	AdamW	0.740
BERT Base	LM	12	0.1	AdamW	0.742
BERT Base	-	6	0.2	AdamW	0.802
BERT Base	-	6	0.1	AdamW	0.824
BERT Base	-	12	0.1	AdamW	0.846
BERT Base	-	12	0.2	Adam	0.808
BERT Large	-	24	0.1	AdamW	0.885

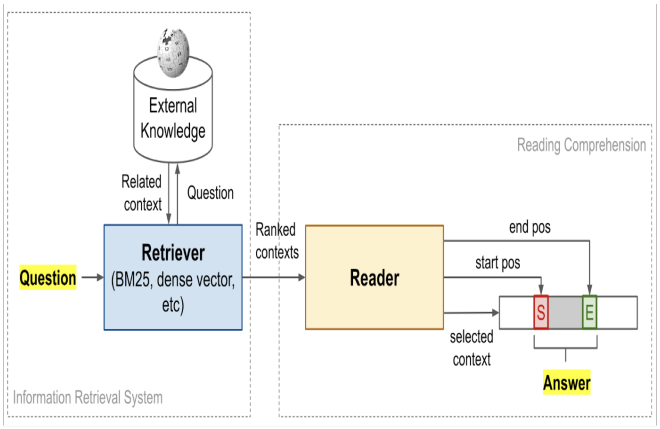

Fig. 3. Architecture of BiLSTM

the input sequence backward. When making predictions, this architecture enables the model to capture both past and future context information. The model takes as input a series of tokens created by the BERT tokenizer. The tokens are subsequently processed by the BERT embedding model, which converts each token into a high-dimensional vector representation. These embeddings are then sent into the BiLSTM model as input. A bidirectional LSTM layer, a fully linked layer, and a final output layer comprise the BiLSTM model. The BERT embeddings are sent into the bidirectional LSTM layer, which outputs a sequence of hidden states for each token in the input sequence. The fully connected layer receives these hidden states as input and does non-linear transformations before applying a ReLU activation function. A linear layer outputs the start and end logits for each token in the input sequence as the final output layer.

Training: The model is optimized during training by comparing the predicted logits to the true start and end indices of the answer span. Adam, a gradient descent optimization technique that adjusts the model's weights during training, was utilized as the optimizer.

Ablation Studies with BiLSTM

BiLSTM models were trained using a variety of embedding types, including GloVe, Tf-IDF, and DistilBERT, as well as pre-processing techniques including stop-word removal (SW) and lemmatization (LM). The models were trained with two, four, and eight layers, with cross-entropy loss function and ADAM optimizer.

The validation loss using the GloVe embedding model was 3.84, which was lower than the validation losses for the models trained using Tf-IDF (4.87) and DistilBERT with stop-word removal and lemmatization (3.72).

Embedding using the DistilBERT model worked well in all circumstances, with validation losses ranging from 3.43 to 4.69. DistilBERT with stop-word removal and lemmatization and eight layers produced the best validation loss, suggesting that a deeper network with pre-processing techniques produced more accurate predictions. Furthermore, using simply stop-word removal resulted in a somewhat worse validation loss than using both stop-word removal and lemmatization, implying that lemmatization assisted the model to better capture the semantic meaning of the input text.

V. Evaluation

In this project, we experimented with a variety of preprocessing methods and several model configurations of BERT and BiLSTM on GPU P100 available on kaggle to examine their effects on the functionality of the closed domain QA system. Due to a lack of computational resources, we were only able to train one BERT big model.

Table 3 shows the performance of BERT on various ablation settings. Results have been truncated to show the most relevant results that can help us draw meaningful conclusions about the effect of each aspect on the model's performance. All the ablation settings for the BERT approach have been covered in more detail in the BERT subsection of methodology . We can see from the table that BERT large model with 24 transformer layers and without any level of preprocessing trained using the AdamW optimizer seemed to perform the best, but the training time was extremely high (>8 hours). Hence, it was not suitable to train the BERT large model for multiple ablations. As a result, we focused on finding a lighter architecture of BERT that can give significantly good performance. We ran iterations to check the effect of preprocessing techniques on BERT and found that surprisingly additional preprocessing is only inhibiting the performance of BERT. Under further analysis, it was concluded that BERT works better without any preprocessing due to its ability to capture complex linguistic patterns and dependencies in addition to its pre-training process that makes it robust to stop words and words not in base form. So, in this case, removal of stop words and suffixes and prefixes has led to a loss of information, for example punctuation and stop words play a very important role when sarcasm is involved and removal

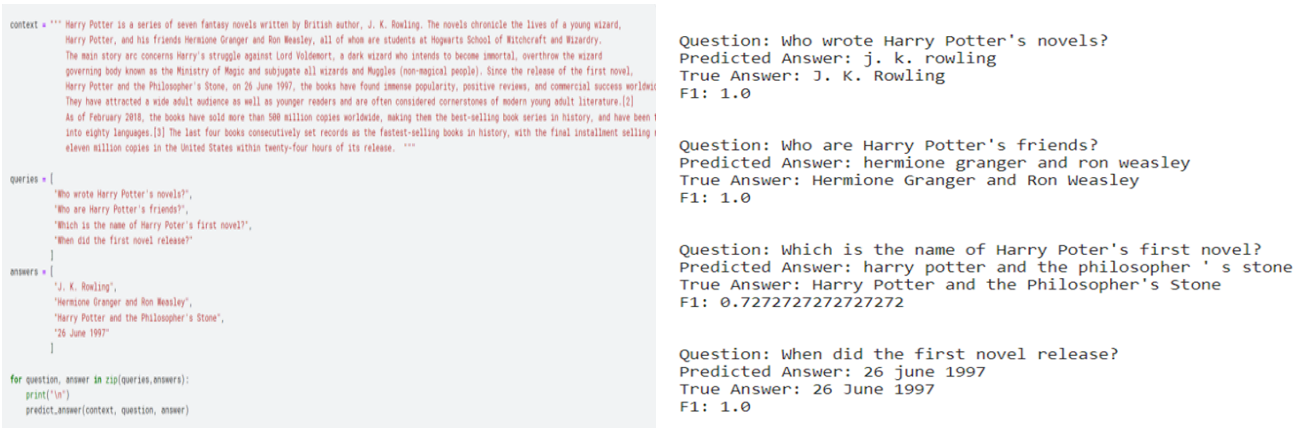

Fig. 4. Sample Input and Output on Test Sample

of stop words changes the meaning entirely. However, it is still important to note that the use of these pre-processing techniques may still be beneficial in some cases, depending on the specific dataset and task at hand. With the increasing number of transformer layers, the performance also increased as expected as with the increasing number of parameters it is able to model more and more complex relationships. Varying the drop out did not affect the performance in any significant way. To conclude, BERT model with 12 transformer layers and 0.1 dropout trained using the AdamW optimizer led to the optimal performance with an f1 score of 0.846. Some cases where the BERT model failed were cases in which numbers were involved in the numeric format. We couldn't figure out the exact root cause for this behaviour but intend to take it up as part of the future work on this project.

Table 4. Performance Comparisons of different BiLSTM Model on Test Data

Model	Embedding Types	Pre-processing	#layers	F1-Scores
BiLSTM	Count Vectorizer	SW, LM	2	0.680
BiLSTM	Glove	SW, LM	2	0.784
BiLSTM	Tf-IDF	SW, LM	2	0.752
BiLSTM	DistilBERT	SW, LM	2	0.789
BiLSTM	DistilBERT	LM	2	0.760
BiLSTM	DistilBERT	ST	2	0.751
BiLSTM	DistilBERT	–	2	0.705
BiLSTM	DistilBERT	SW, LM	4	0.803
BiLSTM	DistilBERT	SW, LM	8	0.811

From the Table 4, we can see that the models utilizing the DistilBERT embedding outperform those using the Count Vectorizer, Glove, and Tf-IDF embeddings. This is most likely due to the fact that DistilBERT is a more advanced and strong language model that captures more nuances and dependencies in text data. Furthermore, we can observe that adding stop word and lemmatization pre-processing techniques can improve the model's performance, as evidenced by higher F1-scores in the SW and LM columns.

Experiments with more layers (4 and 8) utilizing DistilBERT embedding and SW, LM pre-processing yielded the best F1-scores of 0.803 and 0.811, respectively. This implies that increasing the number of layers can assist the model in capturing more complicated relationships in the text input, resulting in improved performance.

Another observation from the table suggests that all models had pretty close F1-scores ranging from 0.680 to 0.811. This suggests that the performance of the BiLSTM model is more dependent on the overall architecture of the model than on the precise combination of embedding types and pre-processing procedures used. As a result, the embedding and pre-processing techniques used should be determined by the features of the text data and the study issue.

VI. Conclusion

In conclusion, the comparison between BERT and BiLSTM models revealed that BERT performed better than BiLSTM. Although BiLSTM produced better results with pre-processing steps, the overall performance of BERT was superior without any pre-processing steps. Additionally, the training time for BERT was less compared to BiLSTM since pre-trained weights were used for BERT and BiLSTM model had to be trained from scratch.This highlights the advantage of pre-trained models like BERT in NLP tasks. One of the reasons for the better performance of BERT is its use of transformers which can effectively handle the contextual representation of the input sequence. Therefore, BERT can be considered as a more efficient and effective approach for question-answering tasks.

VII. Future Scope

Using other models: While the research has made use of two powerful algorithms, BERT and BiLSTM, there are numerous additional models that may be investigated for this goal. One may, for example, experiment with other transformer-based models such as RoBERTa, ALBERT, or T5. Other architectures, such as Convolutional Neural Networks (CNNs) or Recursive Neural Networks (RNNs), could also be investigated.

Methods for pre-processing, embedding, or vectorizing: Pre-processing steps like as text cleaning, stemming, or lemmatization could be adjusted to see how they affect model performance. In addition, instead of BERT embeddings, other

word embeddings such as Word2Vec or FastText could be used. These modifications to the preprocessing, embedding, or vectorizing phases may enhance performance and provide insight into which methods are best suited for this task.

Ensemble models: Ensemble models can be used to increase the performance of the question-answering system by integrating multiple models such as BERT and BiLSTM. Training numerous models on the same data and then aggregating their predictions yields the ensemble model. The forecasts from the BERT and BiLSTM models, for example, can be averaged or weighted to achieve the final prediction. Alternatively, the ensemble can be built by training several hyperparameter versions of the same model. The ensemble model might potentially increase the accuracy and robustness of the question-answering system by combining the strengths of several models or variations of the same model.

In addition to these, we also intend explore the failure cases with existing approaches and figure out reasons for certain extreme behaviour observed, for example in cases of BERT model where it fails when the answers are in numeric format.

REFERENCES

[1] Chuang Zheng, Wei Wu, Yuedong Yang, and Xiaodong Liu. 2019. Synchronous bidirectional and unidirectional representation learning for QA matching. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 5984–5994.

[2] HongLiang Wang, Jun Zhou, and Jiaqing Liang. 2020. A hybrid model based on BERT, neural and siamese network for faq matching in financial domain. In Proceedings of the 2020 IEEE International Conference on Big Data (Big Data), pages 3246–3253.

[3] Shima Foolad, Amirhossein Roshanzamir, and Mohammad Ali Jabraeil Jamali. 2021. A BERT-based model for question answering with attentive features and logistic regression. Journal of Information Science, 47(1):76–95.

[4] Bhavani R, M Deepak. 2014. Comparative Analysis of Question Answering Systems: A Survey. International Journal of Computer Applications, 101(14):1–8.

[5] Shitong Wang, Hongyuan Zha, Xiaoqiang Liu. 2020. A study of rule-based and machine learning approaches to question answering. Knowledge-Based Systems, 196:105794.

[6] Xu Han, Xiaofei Liu, and Tian Shi. 2020. NERQA: A Named Entity Recognition-based Question Answering System for COVID-19. Journal of Biomedical Informatics, 110:10354

[7] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[8] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.