Finding Answers from Wikipedia for Open Domain Questions: A Survey

Viveksinh Solanki
vsolank3@stevens.edu
Stevens Institution of Technology
Hoboken, NJ

Kaviya Priya Kanakaraj
kkanakar@stevens.edu
Stevens Institution of Technology
Hoboken, NJ

1 ABSTRACT

We compared Bidirectional LSTM based models with and without attention mechanism for Question Answering(QA) task. We implemented BiLSTM without attention as baseline model, BiLSTM with Bahdanau attention and BiLSTM with self attention model. For our comparison, we used publicly available benchmark datasets for QA such as Google's NQ set and SQuAD 2.0 from stanford. Out of 3 models, BiLSTM with self attention mechanism gave the best F1 score of 60.3% and best Exact Match(EM) score of 49.5%. From this project we learned that attention mechanisms can significantly improve QA system performance. As for future improvements, transformer models, which have given state of the art performance on almost all of NLP tasks, can be tried for QA systems.

2 INTRODUCTION

Traditional research in the QA domain used a pipeline of conventional linguistically based NLP techniques which dealt with parsing, parts-of-speech tagging and coreference resolution with state of art models such as LSTM, GRU with attention mechanisms. As the research goes progressively in the QA domain, there are a lot of new advanced models which have been developed in the QA domain like Transformers models (i.e. BERT[4]), etc. We focus on the open domain question answering division which emulates how people look for certain information by reading the web. With the help of machine learning, the QA domain explores a lot of benefits and new models are being built continuously to improve the accuracy of the answers. Few challenges in the QA domain,

- All the existing models were trained on extracting answers from a short paragraph rather than reading an entire page of content for proper context.
- With the short paragraph trained model, Responses were complicated or lengthy whereas a good answer would be both succinct and relevant.

Understanding the problem, we wanted to find answers based on reading the entire page. Google's Natural Questions[5] Dataset provides one whole Wikipedia page for each question as context, this inspired us to work with this dataset. By researching the QA topic, we found few papers relevant to this problem which gave us idea to start the project with focusing on only the long answers as finding short answers from huge context would be a huge problem, comparing different RNN models to see how it performs with NQ dataset and at last to provide a comprehensive survey on LSTM based models using NQ dataset and SQuAD Dataset[8].

3 BACKGROUND

In January 2019, Google AI released an open dataset called **Natural Questions(NQ)[5]** to advance the research in open domain question answering. This dataset has been made publicly available under an open challenge. By taking part in this challenge[1], we wish to compare latest deep learning-based models for open domain question answering tasks on NQ dataset. We took motivation from this[2] document to utilize SQuAD[3] dataset[8] as well for our project. Also, we wish to present the best model out of all compared models at the end for both datasets.

4 RELATED WORK

Open-domain Question Answering was originally defined as finding answers in collections of unstructured documents, following the setting of the annual TREC competitions[4]. This paper[2] dealt with solving open domain question answering the challenge by using Wikipedia articles as context and the question and answer come from that specific article for each question. They developed a search component that combines bi-gram hashing and TF-IDF matching with a multi-layer RNN model to detect answers in Wikipedia paragraphs. They compared their model results with many other QA datasets to find the efficiency of the model. Natural Questions[5] is the research paper submitted by Google giving information about the Natural Questions dataset and providing baseline model to compare performance. They explain how the dataset was created and what information does the dataset hold. They explain robust metrics for the purpose of evaluating question answering systems and by demonstrating high human upper bounds on these metrics thereby providing baseline results by using competitive methods. They have also provided baseline BERT model performance on NQ dataset in the paper.
IntelligentQA[6] dealt with models based on Deep neural networks. They also proposed a multi-channel framework that handles operations such as word embedding, dot product, summation, splicing, and data layer transmission. The paper uses CNN and Bidirectional LSTM(BiLSTM) as the main model to learn details of words and capture long term dependency information of contexts in the SQuAD dataset.

5 DATASET

- **Google's Natural Questions(NQ):**

The Natural Questions(NQ) dataset has questions consisting

[1] https://ai.google.com/research/NaturalQuestions?authuser=0
[2] https://web.stanford.edu/class/cs224n/project/default-final-project-handout.pdf
[3] https://rajpurkar.github.io/SQuAD-explorer/
[4] http://trec.nist.gov/data/qamain.html

of real anonymized, aggregated queries issued by the Google search engine. To collect this dataset, An annotator is presented with a question along with a Wikipedia page from the top 5 search results, and annotates a long answer (typically a paragraph) and a short answer (one or more entities) if present on the page, or marks null if no long/short answer is present. The NQ corpus consists of Context, Questions, Long answers, Short answers with their start & end tokens, the Wikipedia website URL from where the data has been extracted from and question-id. The Dataset contains about 307K training examples, 8K development examples, and an extra 8K examples for testing.

- **SQuAD 2.0 (Stanford Question Answering Dataset):**

  SQuAD is a reading comprehension dataset, consists of questions posed by crowdworkers on a set of Wikipedia articles. Unless like NQ, the SQuAD dataset has short paragraphs and a mix of long and short answers. Same as NQ, the SQuAD corpus consists of Context, Questions, Answers with their start and end tokens, Question-id. The Dataset contains 88621 training examples, 11873 development examples, and test evaluation scripts.

6 APPROACH

We have designed our approach based on the learning from the course. We apply 3 different variants of Bidirectional LSTM based models to publicly available benchmark datasets SQuAD and Google's Natural Questions. By doing so, we aim to provide a comprehensive survey of Bidirectional LSTM encoders and their performance comparison for mentioned datasets.
In our case, we are dealing with text data and hence we are utilizing GloVe[7] word embeddings as feature vectors. For question answering task, the most widely used metrics in machine learning are F1-score and Exact Match(EM). We are going to use these metrics for evaluating our models. These metrics will help in understanding the overall performance of our algorithms.

7 EXPERIMENTAL DESIGN

7.1 Pre-Processing

NQ dataset is a huge dataset with size in GBs. Because of the lack of resources, we needed to reduce the size of dataset. In order to reduce size, we only considered rows that contain long answers and removing rest of the records. Another problem, we faced for our datasets, was the dimensionalty of each record. We tried reducing the dimensionalty by considering only records with context/paragraph length <= 10000 for Natural Questions dataset and length <= 300 for SQuAD to account for available compute resources and to make the training much more faster.
All the Tags like <p> <tr> <h1> and special characters were removed to improve modeling performance. To convert text records to vectors, we utilized vectorize function from this[5] source. We did data splitting by using validation split size as 0.2. Hence, split original train set into 80% - 20% splits (training & validation). And used dev set from respective authors as test set for our project.

[5]https://github.com/wentaozhu/recurrent-attention-for-QA-SQUAD-based-on-keras

Dataset	Train	Dev	Test
SQuAD	40,000	10,000	5000
NQ	1120	280	255

Table 1: final Dataset stats after preprocessing

Inputs and Outputs for our models are as follow,

- Input: Context/Paragraph and Question pair embeddings
- Output: Start & End Token embeddings

7.2 Data Modeling

Refer table 1 to check number of remaining samples for each train, dev(which is split from original train set) and test set(original dev set) after preprocessing step. We have utilized 100 Dimenstional GloVe[6] embeddings to create feature vectors.

7.2.1 ***BiLSTM without attention - Model 1.*** Architecture for model 1 is given in figure 1. First layer is embedding layer in which we pass the GloVe[7] word embeddings for both question and paragraph separately. Then we create two separate BiLSTM layers for question and paragraph, which gives two hidden states QueH and ParaH. Now, we concatenate these two hidden states to get combined layer (QueH_ParaH). We pass this combined layer as input to new BiLSTM layer. Next, to capture the similarity between question and paragraph we concatenate paragraph hidden state ParaH with output of previous BiLSTM layer. This combined layer is sent to two separate BiLSTM layers called start_bilstm and end_bilstm. start_bilstm is passed to lstm with softmax activation to get final prediction of start token. Similarly, end_bilstm is passed to another lstm with softmax activation to get final prediction of end token.

- **Advantages**: As compared to normal LSTMs, Bidirection LSTM(BiLSTM) can preserve information from past as well as from future as data runs forward as well as backward in BiLSTM. This gives better context capturing as compared to normal LSTMs.
- **Disadvantages**: Because of two way flow of data in BiLSTM, they are slower than LSTM.

7.2.2 ***BiLSTM with Bahdanau Attention - Model 2.*** For our second model, we are borrowing attention mechanism developed by Bahdanau[1]. Refer figure[7] 2 to understand this attention mechanism. In regular encoder-decoder models, a fixed length context vector is fed into decoder unit. In this attention mechanism, Instead of fixed length context vector, we are feeding dynamic context vector, which is created by addition of all hidden states and simple feed forward network. By doing this, context vector will be able to consume information about alignment between question and paragraph vectors, which can significantly help in finding right answer for given question and paragraph pair.

Architecture for model 2 is given in figure 3. Similar to Model 1, we are creating two different embeddings for question and paragraph and passing them to two separate BiLSTM layers to get two hidden states QueH and ParaH. We pass the concatenated layer of

[6]http://nlp.stanford.edu/data/glove.6B.zip
[7]https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html

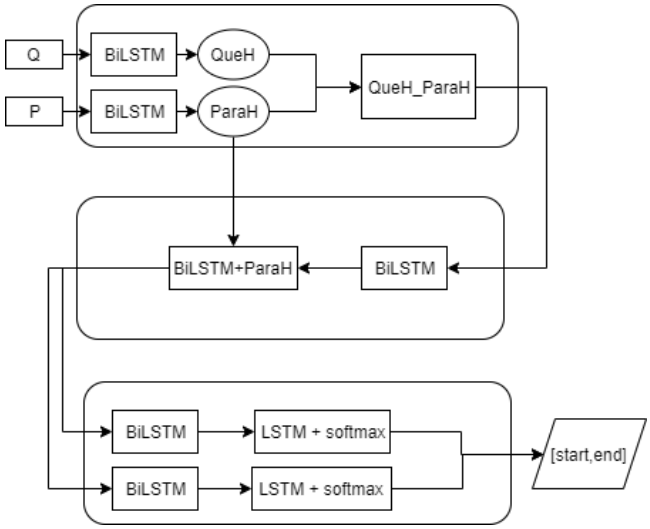

Figure 1: Bidirection LSTM without Attention mechanism

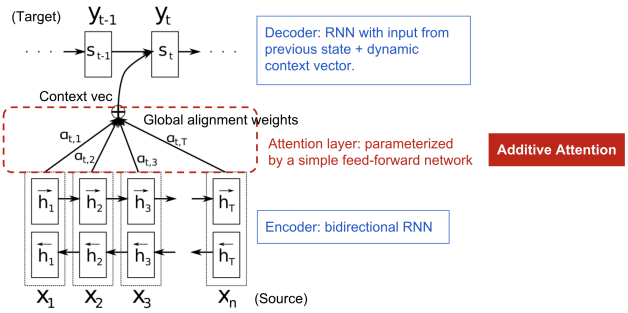

Figure 2: The encoder-decoder model with additive attention mechanism in Bahdanau et al. 2015[1].

these two hidden states to attention layer. As Bahdanau attention takes concatenation of forward and backward source hidden state, it will give good results when used along side BiLSTM layers. Next, to reduce overfitting we introduce first dropout layer with value 0.25. We pass this output to new BiLSTM layer to get intermediate hidden states. The output is passed to concatenated layer of Paragraph hidden state ParaH and previous BiLSTM. This concatenation is performed to get good similarity between question and paragraph pairs. Next, we again introduce dropout with rate of 0.5 and pass its output to two separate BiLSTMs, one for start token and one for end token. Final layers are same as model 1, which includes two separate LSTM layers with softmax activation to get start token and ned token output.

- **Advantages**: As we know, in normal seq2seq models,fixed length context vector will not remember long sequences. Most of the time, It will forget the first part at the end of processing. To resolve this problem, Bahdanau[1] introduced attention mechanism, which will try to remember long sequences better than fixed context vector seq2seq models.
- **Disadvantages**: Even though attention will help in remembering long sequences, LSTMs still won't be able to remember sequences longer than 1000s.

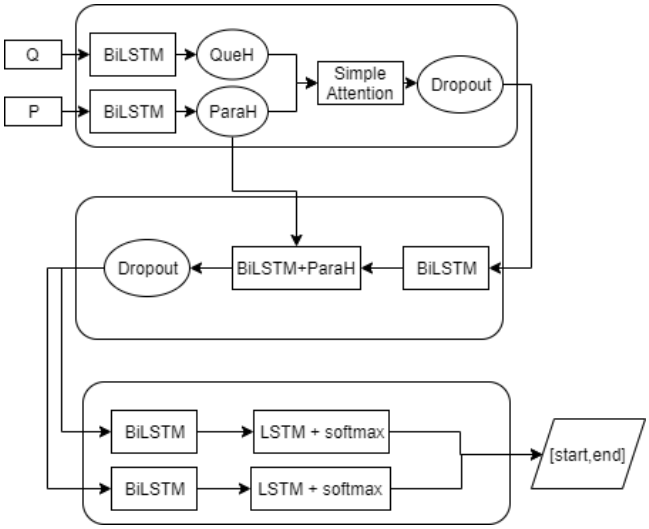

Figure 3: Bidirection LSTM with Bahdanau Attention mechanism

7.2.3 ***BiLSTM with self attention - Model 3.*** : Architecture for model 3 is given in figure 4. As you can notice, architecture for Model 2 and Model 3 are same, we are just replacing Bahdanau attention with self-attention[3] layer. As we know, self attention is an attention mechanism which will try to relate different parts of single sequence. Our motivation for utilizing self-attention for the QA task was self-attention is proved to be very useful for machine-reading and abstractive summarization tasks.

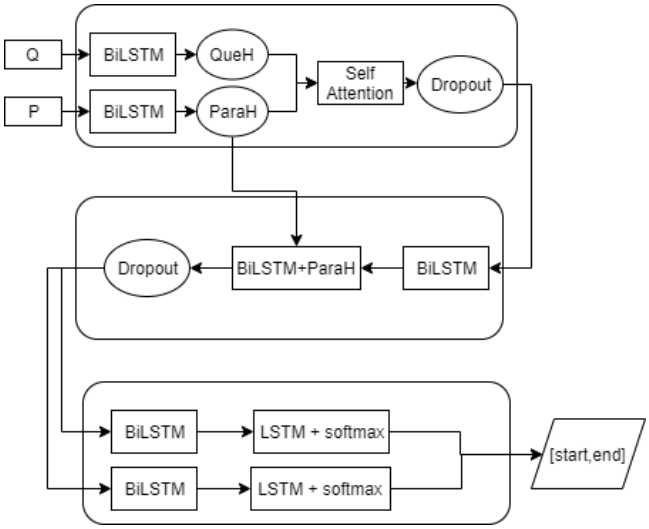

Figure 4: Bidirection LSTM with Self Attention mechanism

	F1(%)		EM(%)	
Model/Dataset	**SQuAD**	**NQ**	**SQuAD**	**NQ**
BiLSTM without Attention	35.73	-	25.2	-
BiLSTM with Bahdanau Attention	57.80	-	45.8	-
BiLSTM with Self Attention	60.3	-	49.5	-

Table 2: Performance comparison on dev set

8 EXPERIMENTAL RESULTS AND ANALYSIS

8.1 Hyperparameters setting

- Feature vectors = 100D GloVe[7] embeddings
- Number of units(neurons) in each layer = 100
- Dropout-1 value = 0.25, Dropout-2 value = 0.5
- Final Activation layer = Softmax
- Optimizer = Adam with learning rate of 0.003
- Loss = 'categorical crossentropy'
- Early stopping with patience = 7
- Validation split size = 0.2
- Batch size = 128
- Epochs = 10

8.2 Results and Analysis

All results are mentioned in table 2. Hyperparameters settings can be referred to in the previous section.

Google NQ dataset:
Initially, we started with whole dataset(300k rows) and tried to train our models. But, it resulted in memory errors due to very high dimensionality of records. So, we started experimenting with different 'max allowed paragraph length' for each record. After several trial and errors, we decided to go with max allowed paragraph length <= 5000. So, after doing mentioned pre processing we only kept records which have paragraph length <= 5000.

We ran our first model, which is Bidirectional LSTM without any attention. After training, when we tried to predict answer tokens, it was predicting all same tokens for all true start tokens and all same tokens for all true end tokens. So, BiLSTM without attention model didn't work for NQ set. Next we tried BiLSTM with Bahdanau[1] attention and BiLSTM with self-attention[3] one by one. But, still results were same. Same tokens for all true start tokens and same tokens for all true end tokens.

One probable reason for this could be the number of records we considered as part of our training set. But, we believe that the main reason should be the model itself. As we know, LSTMs are good with remembering long term dependencies as compared to other simple RNNs. But, It is also true that LSTMs cannot remember very long sequences of 1000s or more. This was one of the findings for our experiment with NQ set.

SQuAD dataset:
Because of the negative results on NQ dataset, we decided to test out hypothesis models on one more similar dataset, which is Stanford Question Answering Dataset[8]. After understanding dimensionality problem in NQ set, we decided to keep 'max allowed paragraph length' <= 300 to have faster training loops and to be able to train on as many as possible number of rows.

Our baseline model of BiLSTM without attention ran successfully. It gave F1-score of 35.73% and Exact Match(EM) score of 25.2%. We can expect EM score to be always less than F1-score. Because EM is stricter than F1-score in the sense that, It will only consider a token to be true if it 100% matches with ground truth token.

Next, we ran preprocessed SQuAD set against BiLSTM with Bahdanau attention. As expected, attention mechanism significantly improved F1 score as well as EM score as mentioned in table 2. We got 57.8% F1 score and 45.8% EM score. This experiment proves that attention mechanism is really helpful in improving performance of QA systems. To support this hypothesis even further, we ran SQuAD set against BiLSTM with self-attention, which is new variant to original attention mechanism. This model improved scores even further. This time we got F1-score of 60.3% and EM score of 49.5%. If we compare with baseline model, then our model-3 gave really good improvement for both evaluation metrics.

Hence, these experiments prove that attention mechanisms should be part of any QA model.

9 CONCLUSION

In this project, we compared various BiLSTM based models along with two attention mechanisms on Squad and Google's NQ datasets. We implemented BiLSTM without attention as baseline model, BiLSTM with Bahdanau attention and BiLSTM with self attention. We can see clear improvement in performance with attention based models. BiLSTM with self attention gave the best F1-score of 60.3% and EM score of 49.5%. Also, we learned that to train LSTM based models on dataset with very long sequences, you require a lot of compute power and time. One more learning is that LSTM and its variant can remember sequences of 100s, but it performs poorly when sequences are longer than 1000s or more.

10 FUTURE WORK

Bahdanau and self attention mechanisms can be utilized with Bidirectional flow attention[9] model to see if it improves performance or not. Also, in recent years, we have seen so many state of the art performances on various NLP tasks using transformers[10]. So, we believe that applying transformers will definitely improve performance for given datasets.

11 CONTRIBUTION

Please refer table 3 for individual member contribution for this project.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR* abs/1409.0473 (2014).

Task	Kaviya	Viveksinh
Data Preprocessing	✓	
Model 1 implementation	✓	
Model 2 implementation		✓
Model 3 implementation		✓
Apply models on NQ set	✓	
Apply models on SQuAD		✓

Table 3: Individual Contribution

[2] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to Answer Open-Domain Questions. *CoRR* abs/1704.00051 (2017). arXiv:1704.00051 http://arxiv.org/abs/1704.00051

[3] Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long Short-Term Memory-Networks for Machine Reading. *CoRR* abs/1601.06733 (2016). arXiv:1601.06733 http://arxiv.org/abs/1601.06733

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805

[5] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: a Benchmark for Question Answering Research. *Transactions of the Association of Computational Linguistics* (2019).

[6] Chengfei Li, Li Liu, and Fan Jiang. 2018. Intelligent Question Answering Model Based on CN-BiLSTM. In *Proceedings of the 2018 2Nd International Conference on Computer Science and Artificial Intelligence (CSAI '18)*. ACM, New York, NY, USA, 447–450. https://doi.org/10.1145/3297156.3297261

[7] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543. http://www.aclweb.org/anthology/D14-1162

[8] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100, 000+ Questions for Machine Comprehension of Text. *CoRR* abs/1606.05250 (2016). arXiv:1606.05250 http://arxiv.org/abs/1606.05250

[9] Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional Attention Flow for Machine Comprehension. *CoRR* abs/1611.01603 (2016). arXiv:1611.01603 http://arxiv.org/abs/1611.01603

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *CoRR* abs/1706.03762 (2017). arXiv:1706.03762 http://arxiv.org/abs/1706.03762