NON-FACTOID ANSWER SELECTION

Manan Dilip Mehta
mmehta64@gatech.edu
903390470

Priyal Ajay Jhaveri
pjhaveri6@gatech.edu
903391526

Tharun Saranga
tsaranga3@gatech.edu
903457046

1 Abstract

We aim to have a thorough comparative study for various deep learning models for the Non Factoid Answer selection task. Comparing these models on various different evaluating metrics. We also aim to understand what benefits and shortcomings do each of the models pose for the corresponding task, i.e. identifying which types of question answer pair it classifies correctly and where a particular model fails. Thereby trying to understand how each of the models behaves when given various question answer pairs[5]. We started by performing a thorough literature review of each of the models for various tasks[8][2][3]. We implemented 4 different models and then performed ablation studies on each of the models which helps us understand the benefits of each model. The results show the BERT based model had a much better accuracy for the task as compared to the other 3 models. We explain which type of model performs well under which circumstances.

2 Introduction Related Work

The motivation behind our research study stem around non-factoid question-answer selection, and the various methodologies used around this. There is a large extensive study around

'Answer Passage Retrieval' that was found to be inadequate for this task as they annotated passage answers in order to retrieve from the passage

'Answer Retrieval Using Translation Models' that was found to have a lexical distance between the question-answer pair , used a query likelihood methodology for the answers and a translation approach for the questions.

'Answer Ranking' that used a methodology of extracting various translation, similarity and correlation features to rank answers but the database could be derived from many documents as compared to the candidate dataset which may not be as extensive.[7]

On a high level answer selection worked in the following manner, the question-answer pair representation is learned, a joint feature vector is constructed, and then this is converted into a classification problem. Our approach is of this last kind. The paper by [4] proposes to use BiLSTM along with CNN, BiLSTM along with attention models and a BiLSTM model to do the task of question-answer selection for long contextual sequences and BERT Model

Our project aims to run a comparative interpretability study on various deep learning models for this task on the following models:

- Bidirectional LSTM
- CNN - LSTM
- LSTM with Attention
- BERT

Here through our interpretability study we plan on analysing through various experiments the reason for accurate classification or not and justify the analysis through an ablation study that uses Accuracy, Precision and Mean Average Precision as the metric to exemplify this further. The models above have been implemented and tested in the papers mentioned above, there is no comparative study of the models in this task in terms of an extensive experimental analysis. The main task of our project here is to implement an interpretability and ablation study on the experimentation analysis we perform on basis of the results as run on the Insurance QA data set and the above mentioned models. There have been many deep learning based systems for Question Answering tasks but our task is slightly

different. We mainly aim at selecting the best answer from a group of candidate answers. These answers are usually long. Average length of the answers is 90 words.

3 Methods

3.1 Data

The main dataset for the experimentation is the InsuranceQA dataset[4]. It is a set of questions and answers gathered from insurance company claims processing and customer interactions. It has a training set, validation set and two test sets. The dataset contains Questions with Good Answers and Bad Answers. The number of questions and answers in the InsuranceQA is shown in the table 1.

	Train	Val	Test1	Test2
No.of Questions	12887	1000	1800	1800
No.of Answers	18540	1454	2616	2593

Table 1: InsuranceQA dataset details

The data is pre-processed such that each good answer and bad answer is associated separately with each question, increasing the number of data points. A single question may correspond to many answers. So, each question answer pair is separated and processed as such. Questions with no bad answers or no good answers are not considered.

The average length of the questions is 7 and the average length of answers is 94. The questions are much shorter than the answers making it difficult to predict the full context and content for the answer.

Examples of a question and answer from the data set after pre-processing are shown in Figure 1.

3.2 Baseline Models

LSTM and BiLSTM are the baseline models selected to compare the performance of the target model. These are the most common models used in the Natural Language tasks.

3.2.1 BiLSTM-CNN

Previously we used a simple Bidirectional LSTM model, but there was a drawback based on its architecture that we encountered during training the model. We realized that BiLSTM's store long term dependencies well and also is able to model the data in a sequential way. But it fails to capture local correlations of spatial or temporal structure. It also is unable to capture features in a parallel way. This is where a CNN architecture comes into the picture. We combine the strengths of an LSTM model and a Convolutional Neural Network to extract the high level local dependencies along with long term dependencies which are extracted using a BiLSTM model. The hidden states at each time stamps are passed as an input to the CNN architecture. For the window size 'm' we convolute the m consecutive hidden states with the m filters to generate the out-put. After which we perform max pooling to select the best representation of the question and the answers and perform cosine similarity as the previous model. We create two variants using mean pooling and max pooling and train the model on various hyper parameters to test them on the validation set.

3.2.2 LSTM with Attention

Previously we described how we used Bi-directional LSTM and then added a Convolutional Layer to it. In this model after passing the question and answer through bi-lstm we add a self attention layer for the answer. Before we add a max / mean pool layer for the question / answers we apply softmax on the output of the bi-lstm using the question embeddings. In this way we focus on key words in the answer based on what the question demands and thus try to distinguish between good answers and bad ones.

$$\begin{aligned} \mathbf{m}_{a,q}(t) &= \tanh(\mathbf{W}_{am}\mathbf{h}_a(t) + \mathbf{W}_{qm}\mathbf{o}_q) \\ s_{a,q}(t) &\propto \exp(\mathbf{w}_{ms}^T \mathbf{m}_{a,q}(t)) \\ \widetilde{\mathbf{h}}_a(t) &= \mathbf{h}_a(t) s_{a,q}(t) \end{aligned}$$

[4] Here ha(t) = hidden state of answer at timestamp t. Oq = Output of bi-lstm for question after performing max / mean pooling One we get the new hidden state of the answer we follow the same procedure as in the above techniques and take the cosine similarity between the output question embedding and output answer embedding.

3.2.3 BERT

Bert stands for Bidirectional Encoder Representations From Transformer. It is one of the most widely used state of the art model for NLP tasks. We wanted to check how it works for answer selection task for non factoid answers. We have seen earlier how we get the Question and Answer embeddings and find similarity between them using cosine similarity. But for BERT it is slightly different. We used a pre-trained version of bert where we only perform fine tuning for our task. [3] The architecture is shown in Figure.2 [1] Since our task boils

```
Question: ['how', 'cancel', 'nationwide', 'renter', 'insurance']
Good answer: ['if', 'you', 'no', 'longer', 'have', 'the', 'need',
'for', 'your', 'Renters', 'Insurance', 'Policy', 'you', 'can',
'contact', 'your', 'Nationwide', 'Insurance', 'professional', ',',
'discuss', 'your', 'situation', 'and', 'your', 'result',
'decision', 'cancel', 'the', 'policy', 'and', 'they', 'will',
'be', 'able', 'help', 'you', 'do', 'that', 'quickly',
'efficiently', 'you', 'may', 'be', 'ask', 'provide', 'the',
'request', 'in', 'write', 'and', 'sign', ',', 'which', 'if',
'you', 'think', 'about', 'it', 'be', 'only', 'right', 'since',
'when', 'you', 'begin', 'the', 'coverage', ',', 'you', 'be',
'ask', 'sign', 'a', 'write', 'application']
Bad answer: ['Obamacare', 'be', 'available', 'to', 'all',
'American', 'during', 'open', 'enrollment', 'or', 'special',
'enrollment', 'period', 'many', 'applicant', 'will', 'qualify',
'for', 'advanced', 'premium', 'tax', 'credit', 'while', 'other',
'will', 'pay', 'full', 'price', 'for', 'their', 'select',
'insurance', 'regardless', 'if', 'you', 'want', 'a', 'plan',
'and', 'have', 'the', 'means', 'pay', 'for', 'it', 'you',
'simply', 'have', 'enroll']
```

Figure 1: Example of a single question with good and bad answer

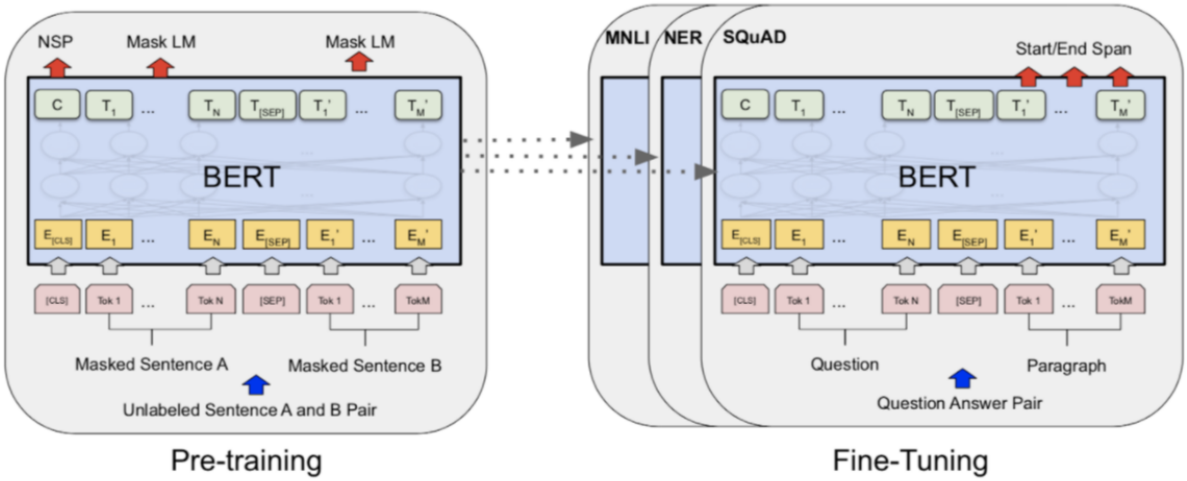

Figure 2: BERT Architecture.

down to a binary classification problem where we try to identify for a given question which answer is the most apt. We insert the question as the first sentence and the answer as the second sentence. We use the BERT base model which consists of 12 Layers, 768 Hidden Dimension and 12 Self Attention Heads. We then get the joint embedding from the CLS token of the last layer of the BERT model. This 768 dimensional representation is then passed on to a linear layer whose output is either 0 or 1 based on the fact that if the answer is the correct answer for this question or not. Input needs to be in the form as shown 3.[1] For token embeddings we use bert tokenizer. The first word is a special token CLS and the question and answer are separated using another special token called SEP. We then merge it with segment embeddings which indicate if the word is a part of the question or the answer. Finally we add positional information by giving the bert model positional information about the words using positional embeddings. We perform 2 types of training

- When we only learn weights for the last layer and do not propagate the gradient through the bert layer.
- When we train everything including the BERT layers.

4 Results

4.1 Experimental Setup

The data already is divided into train, validation and test sets. The same data format has been used for all the tested models. The BiLSTM-(CNN and Attention) models are first trained on the train set with batch size of 64 and 0.1 validation split from the train set. The models were further tuned on

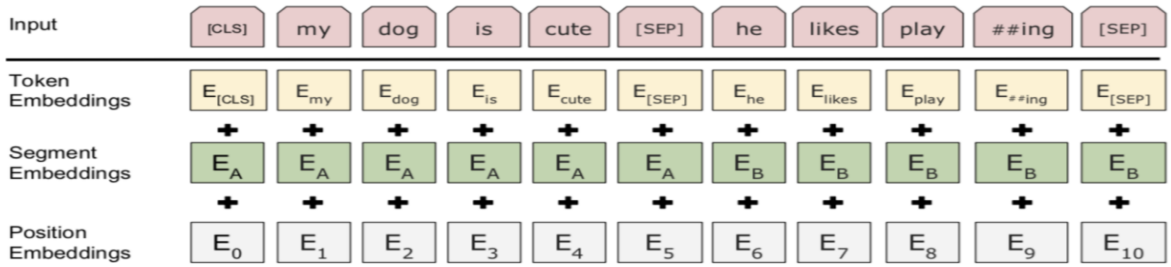

Figure 3: Input Form.

the Validation set by tuning the hyper parameters. The input to the models is a question plus a good answer and a set of bad answers. The model selects an answer from the given list. If the answer is the good answer then the model would have predicted correctly or else it would be wrong. Various experiments are conducted using a similar setup. The model performance is measured using Precision metric. Precision is calculated as the total correctly classified questions divided by total number of questions presented to the model.

4.2 Result Comparison

4.2.1 Metrics Used

$Precision = \frac{TruePositives}{TruePositives+FalsePositives}$

$MeanReciprocalRank = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i}$

The rank of the data is calculated from scipy.stats rankdata.

4.2.2 Comparison with Base Line

All the models implemented perform better than the baseline model (Bi-lstm). The CNN - LSTM model is able to use the advantages of LSTM based model as well as is able to capture the local dependencies with the help of Convolutional layer. Hence this architecture performs better than the baseline LSTM model. The local dependencies of the answers which are usually long are well captured by the convolutional layer. The LSTM with Attention Model performs better than the baseline model. The reason for this is that the LSTM with Attention model is able to give relevant words in the answer more weight based on the question that needs to be answered. This helps the model differentiate between a good answer and a bad answer. In fact the LSTM with Attention model performs slightly better than CNN LSTM model. This could be explained by the fact that CNN LSTM finds local dependencies based on the filter size and may not be able to capture long term dependencies between important words. Whereas the attention model helps us weigh words based on the question embeddings and hence helps us find more relevant answers. Bert model on the other hand out performed the baseline model as well as the other 2 models. By using the pretrained bert model we see that by passing the question and answer we get an embedding that is able to closely distinguish between the good answers and the bad ones. The biggest reason for its success is the multi head attention modules which work parallel across the length of the input. This successfully incorporates all the salient features which are not captured by the baseline model.

Model	Val	Test1	Test2
BiLSTM	52	51.4	51
Conv LSTM	63.2	60.4	61.1
Attention-LSTM (max-pool)	68	66.2	61.1
Attention-LSTM (avg-pool)	68.5	68.1	62
BERT (Freezing all BERT layers)	78	80.2	75.1
BERT (Fine-tuning all layers)	90	88.6	82.1

Table 2: The table represents the Mean Average Precision of all 3 Models compared to the baseline

4.2.3 Hyper parameters of the Models

BiLSTM CNN The Hyper parameters used for this model are learning rates (1e-1, 2e-3, 2e-5). Different activations are also tested ('sigmoid', 'tanh'). tanh performs better than the other in this particular model.

BiLSTM with Attention The hyper parameters used for this model are learning rate (2e-5 , 2e-3, 5e -5), hidden dimensions and the maximum epochs.

The advantage of this model is that it tries to give attention to relevant words in the answer based on the embedding of the question. This helps separate the good answers from the bad ones.

BERT The hyper parameters used for bert are max-len of question and answer, batch size, learning rate, max-epochs and whether we want to freeze the bert layers for back propagation or not. We tested the model on various learning rates (2e-5 , 2e-3, 5e -5), and max len(30, 60, 90). One of the disadvantages of the Bert model is that it requires fixed size input and since the inputs are varying we need to either cut short the answer length or pad it. Hence it was important to note which value of max-len is most apt. We also analyzed results based on either freezing all BERT layers for back prop or not. This helped us understand what are the pros and cons of each method as there is a tradeoff between time taken vs accuracy of the model.

4.2.4 Ablation Study

BiLSTM CNN Adding dropouts at different points in the model doesn't improve the performance much but adding dropouts before the last layer improved the performance of the model. Another study of changing the number of CNN layers is also conducted. We found that two CNN layers are optimum for this particular model.

BiLSTM with Attention Studies showed that adding dropouts helped the model to generalize better. Thus we tried adding dropout in the model and experienced a 2% increase in the Mean Average Precision score. Which corroborates our fact that the model generalizes better.

BERT For BERT we tested things in different ways. Although we just fine tune the BERT model we perform 2 experiments. First we just tested the model using distillation ie. we froze all the BERT layers and back propagated the loss only on the last linear layer. We ran it the second time by back propagating the loss through the entire BERT architecture. We had this experiment since one of the biggest disadvantages of using BERT is that it is very computational heavy and thus causes a long time to train. Thus, the aim of this experiment was to understand that if the first run gives us comparable accuracies as compared to the second run then we can say that we can use a transfer learning technique which gives us great results but also takes much less time to train. For the first run we received accuracy of 80% on the test dataset and it took under 60 mins to train the model. Whereas using all the Layers of the BERT model for back propagation we notice an increase in the performance by 15% although it took almost double the time. The reason why we see such an improvement in accuracy and precision is that we realize the BERT model was pretrained on a specific task which was different from ours. We also realize that since our questions are highly domain specific, tuning all the weights would help us achieve better results as compared to just tuning the weights of the last layer.

	Good Answers	Bad Answers
Good Answers (Class 1)	TP = 2749	FN = 282
Bad Answers (Class 0)	FP = 540	TN = 3693

Table 3: Confusion Matrix for BERT: Best Performing

4.2.5 Visualization, Extended Analysis, Error Analysis

BiLSTM-CNN Layer output for Correct and Wrong classifications: Observing the layer outputs from the model it is prioritizing the word bunches from the starting of the question. The penultimate layer output can be seen in the 4

The model classifies correctly when the words present in the question are also present in the answer.

For example:

Question: [‘Why’, ‘do’, ‘not’, ‘medicare’, ‘shingle’, ‘vaccine’]

Good Answer: [‘exempt’,’vaccine’]

BERT The BERT Layer when fine tuned over the entire architecture gives the following scores.

As we can see the BERT model performs much better as compared to the other 2 layers. Mainly due to the fact that Bert involves multi head attention and it also takes into consideration bidirectional text.

We used an open source visualization tool called BERTviz [6] which helps us visualize 2 types of views of the BERT Model. Instead of using the default BERT model we loaded the model that we fine tuned on our task. Below we will go over each of the views in depth and understand it with a given example.

We used an open source visualization tool called BERTviz which helps us visualize 2 types of views

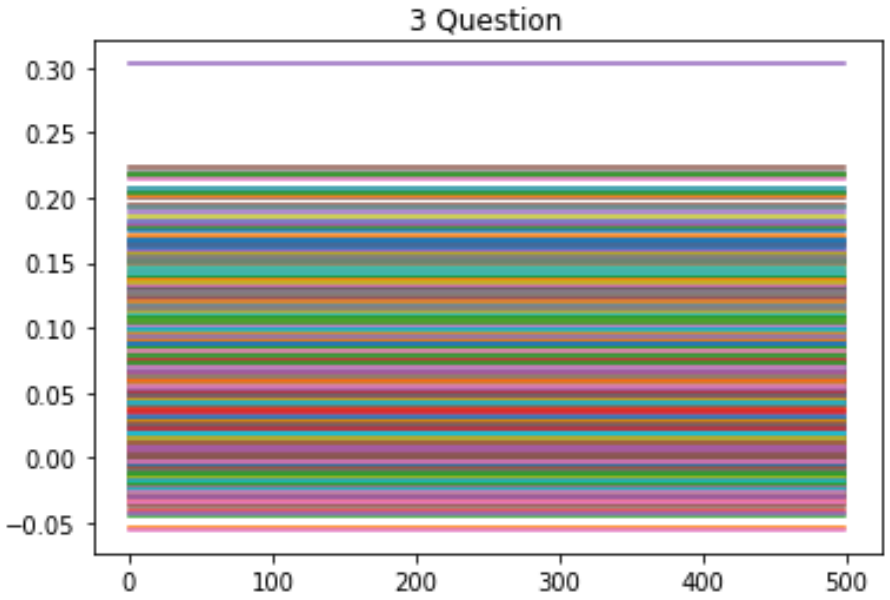

Figure 4: Penultimate layer output in BiLSTM-CNN model

	Good Answers (Class 1)	Bad Answers (Class 0)
Precision	90.4	88.7
Recall	83.13	90.7
F1 Score	86.6	89.5

Table 4: Precision,Recall,F1 Score BERT

of the BERT Model. Instead of using the default BERT model we loaded the model that we fine tuned on our task. Below we will go over each of the views in depth and understand it with a given example.

Example Question : How To Get Health Insurance When Pregnant?
Example Answer : You may be able to get coverage through Medicaid. There are no open enrollment period restrictions for this program. However there are needs based criteria which are often extended for pregnant women. Each state has very different rules for who qualifies. Contact the Medicaid office in your county to see if you qualify.

Head View Figure 5 is a view which helps us understand the attention patterns from the attention heads of one out of the 12 layers of the BERT model. The left side indicates the words that are attending and the right indicates the one being attended. The different colours represent the 12 different attention heads in each Layer. We could select the layer we want to visualize and can also select if we want to see between Sentence A -¿ Sentence B or vice versa. We can also visualize each sentence's attention with itself. The intensity of the lines indicate the attention score. The attention heads tend to capture important information like Named Entity Recognition, subject - verb pairs etc. As we can see for the example in Fig 1 above even though "Medicaid" is a proper noun we see that the word "health" is attending to "med" of the "Medicaid". This clearly depicts the power of the BERT model justifying its advantages of the other 2 models. In fig 2 we can see how "coverage" is attending words like "health" , "insurance", "pregnant". Which again depicts how the model trains and learns to attend to specific words. Since each attention head has independent parameters, they all learn unique attention mechanisms. [6]

Model View In 7 we can see each of the attention patterns for each layer and each of its attention heads within the layer. That means it would be a 12 X 12 grid. Once you click on a cell you can get a detailed view of that specific attention head. This view indicates what each attention head within each layer has learned. This view enables us to identify which attention heads are relevant. It may be further used to mask some of the attention heads in order to reduce inference time. Figure 3 depicts the 9th Layer of the Bert Model whereas

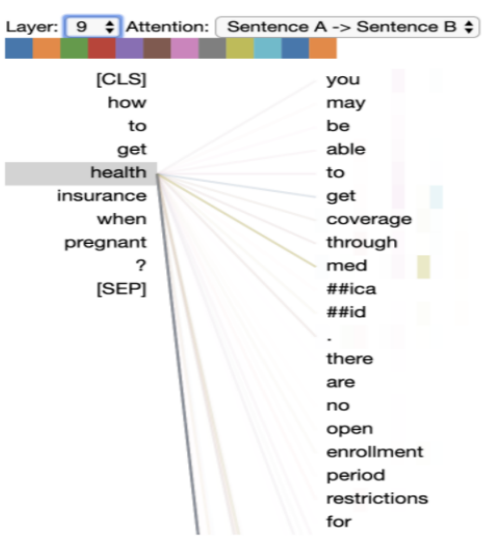

Figure 1

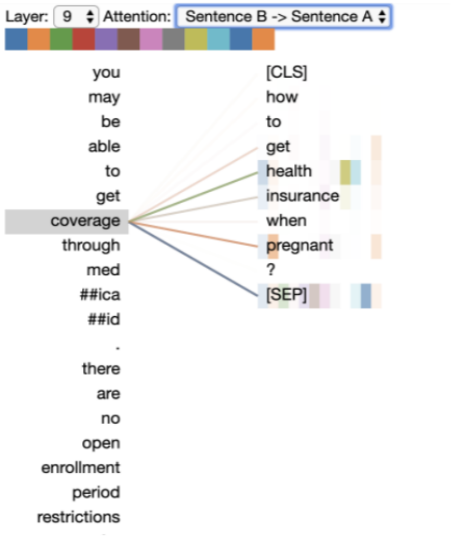

Figure 2

Figure 5

Figure 6: Head View

Fig 4 indicates the 5th attention head of this layer. Once we hover over a specific word (Fig 5) we can see what "insurance" attends to. [6]

LSTM with Attention Some Negative Examples
For the same example LSTM with Attention suggests that the following is a good answer: "Health insurance is important because it is the way we finance health care . Health care is expensive with office visits , lab tests , x-rays , and surgery whether inpatient or outpatient . Most people do not have money saved up for medical expenses and therefore purchase health insurance to help pay for medical care . 80 % of Americans spend less than $ 900 annually on medical care and if you are young and healthy you may not think you need health insurance . Health insurance also protects us against large and unexpected medical expenses. The large health care expenses young adults have are usually caused by outdoor activities and sports where injuries can create a high medical bill . Those are the main reasons why young adults should purchase health insurance to protect against larger unexpected medical bills."

This would not be a good answer as it barely answers the question. Due to the fact that we are attending important words in answers we see that since this answer is very long and it has used the words "health insurance" multiple times, the model starts to consider this answer as a good answer. This is a general trend we see for the LSTM with Attention model that longer the length of the answer more is its tendency to give a wrong answer. This behaviour can be justified. Even though LSTM models are better than Vanilla RNN in diminishing the vanishing gradient problem. Some form of this issue yet persists when the length becomes almost 90-100 which is the average size of our answers. Thus Even with simple attention we are not being able to retain much of the information. This is one of the disadvantages of any recurrent model.

4.2.6 Work Division

Every Member Contributed to all the analysis and results but these were the assigned task leaders:
Tharun Saranga: Results and Experimental analysis for BiLSTM-CNN
Priyal Jhaveri: Results and Experimental analysis for BiLSTM with Attention and BERT Head View Visualization
Manan Mehta: Results and Experimental analysis for BERT and BERT Model View Visualization

5 Conclusion

Comparing all the models, we see that BERT performs better than the other layers widely because it uses multi-head attention as compared to BiLSTM-attention which tends to use a simple attention model. This is because the independent power

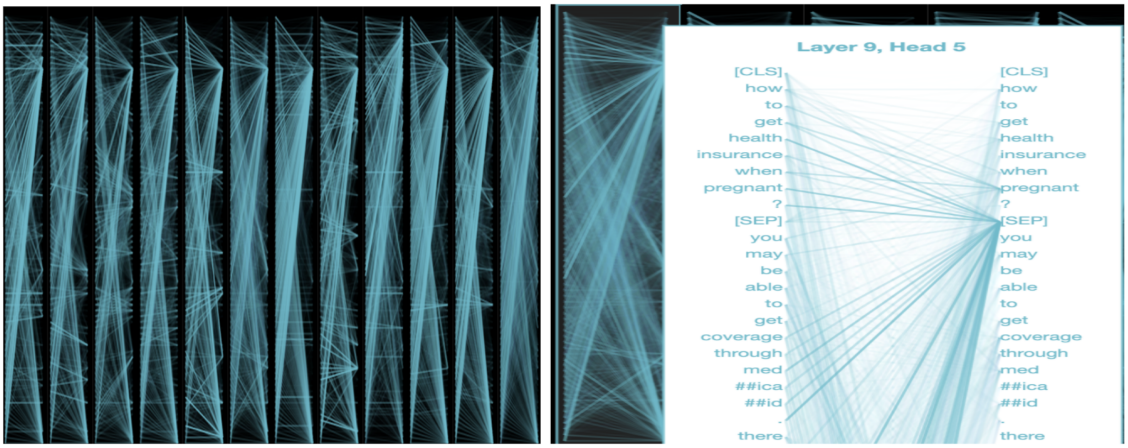

Figure 7

Figure 8: Model View

to learn to train and learn weights for each attention head, causes a fine tuning at each layer which helps to encapsulate different contextual relations between the question-answer pair. BERT also tends to perform better than BiLSTM-Attention and BiLSTM-CNN since it takes into consideration bidirectional text, altogether whereas in a BiLSTM architecture of any kinds, it would take into consideration bidirectional text per timestamp causing the vanishing gradients problem for long-contextual texts. Overall, We see this for multiple such examples, as it handles paragraph type answers way better than the other two models.

The results of our models, comply with the results we see in the papers. However our experimentation sheds further light on how hyper-parameter tuning, such as hidden dimension size happened to increase the accuracy for BiLSTM-Attention as it happened to retain more information especially for long-contextual answers. Freezing the layers and Unfreezing the layers for BERT during backpropogation improved the speed drastically and increased the accuracy respectively highlighting the trade-off for both these experiments widely. BiLSTM-CNN works good for questions of more length and questions with similar features as the answer. It fails in all the other cases. The number of CNN layers affect the performance of the model. We found that two CNN layers are optimum for the feature extraction from the embeddings. For future work compressed versions of BERT can be considered which may be computationally less intensive. A more balanced dataset could be opted for as well.

6 Code Repository

https://github.gatech.edu/pjhaveri6/question$_a nswer_s election$

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[2] Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, and Bowen Zhou. Applying deep learning to answer selection: A study and an open task. *CoRR*, abs/1508.01585, 2015.

[3] Siddhant Garg, Thuy Vu, and Alessandro Moschitti. Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection, 2019.

[4] Ming Tan, Bing Xiang, and Bowen Zhou. Lstm-based deep learning models for non-factoid answer selection. *CoRR*, abs/1511.04108, 2015.

[5] Nam Khanh Tran and Claudia Niederée. A neural network-based framework for non-factoid question answering. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, page 1979–1983, Republic and Canton of Geneva, CHE, 2018. International World Wide Web Conferences Steering Committee.

[6] Jesse Vig. A multiscale visualization of attention in the transformer model. *arXiv preprint arXiv:1906.05714*, 2019.

[7] Liu Yang, Qingyao Ai, Damiano Spina, Ruey-Cheng Chen, Liang Pang, W Bruce Croft, Jiafeng Guo, and Falk Scholer. Beyond factoid qa: Effective methods for non-factoid answer sentence retrieval. In *European Conference on Information Retrieval*, pages 115–128. Springer, 2016.

[8] Lei Yu, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman. Deep learning for answer sentence selection. *CoRR*, abs/1412.1632, 2014.