Step by Step approach to build a model for SQuAD

Anjan Dwaraknath
Institute for Computational and Mathematical Engineering
Stanford University
Stanford, CA 94305
anjandn@stanford.edu

Abstract

We build multiple model architectures using basic building blocks such as GRUs, attention layers and fully connected layers. We study the impact of the different building blocks and observe which additions contributes significantly and which do not. We also explore adding another term in the objective function which mimics a language model to see if it helps convergence or improves the performance. Lastly we visualize the attention outputs to get a better insight into the internal workings of the model.

1 Introduction

Natural Language processing has made huge progress ever since it started using deep learning to build end to end models. As with all areas that use deep learning, the key is the availability of large high quality datasets. The Stanford Question Answering Dataset (SQuAD) by Rajpurkar et al. (2016) [1] has proven to be vital for progress in question answering. Research in the area has led to the identification of a few key primitives which have proven to be useful building blocks to build models. These primitives such as RNNs(LSTM, GRU), Attention, Self-Attention, Pointer Sentinel etc. if combined the right way can produce high performing models.

We explore each of these building blocks and determine when they are useful, by progressively building a model architecture and observing what works and what does not. Thus the final architecture is a product of a greedy search over adding these building blocks. We also explore a simple technique to incorporate language modeling as another term in the objective and whether that helps model convergence or performance.

2 Background

Many successful approaches have been developed to build an architecture for question answering. Most involve some type of RNN as the first layer. These can range from a simple vanilla RNN all the way to to bidirectional LSTM. The problem with RNNs is the vanishing gradient problem, as a solution to this and other problems, the next layer usually involves attention. For the case of question answering, we can have the question hidden states attending to the context hidden states and vice versa (BiDAF [2]). We can have more attention layers over this (as in coattention models [3]) or feed it into another RNN (as in R-Net [4]). Instead of computing attention between question and context, self-attention[4] can also be used.

3 Approach

Our approach was to abstract out the primitives from the methods discussed in the previous section and build a model step by step.

3.1 Baseline

We start with the baseline model, where we use GloVe embeddings for each word and feed it into a bidirectional GRU. We concatenate the hidden layer for both directions. Then each context hidden state attends to the question hidden state and this output is concatenated with the context hidden state to produce the blended representation. This is then passed through a fully connected layer with ReLU non-linearity. The FC layer and GRU both have dropout. The distributions for the span start and end are computed using a simple softmax layer over this. This can be represented using the following simple representation

$$GloVE \rightarrow biGRU \rightarrow C2QAttn \rightarrow FC \rightarrow SimpleSoftmax$$

Note that above and in the rest of this document, attention outputs are always concatenated with their inputs and this is what is fed to the next component In order to reduce the length of the representation we can contract the input and output layers as follows, but it is equivalent to the figure for the baseline model.

$$biGRU \rightarrow C2QAttn \rightarrow Span$$

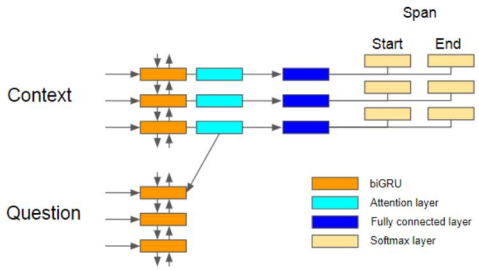

Figure 1: Baseline model

3.2 Step by Step additions

First we simply tried adding extra FC layers to see if improvement can be made that way.

$$biGRU \rightarrow FC \rightarrow C2QAttn \rightarrow FC \rightarrow Span$$

This did not show any improvements infact performance degraded as it overfit more. Next we add another attention layer,

$$biGRU \rightarrow biAttn \rightarrow FC \rightarrow C2QAttn \rightarrow Span$$

Here biAttn implies that we have both C2Q (context to question) attention as well as Q2C (question to context) attention. The FC layers do not share weights between context and question, this is so that it can learn different representations for both as they already share the same GRU. This step improved performance, so we added another layer of it.

$$biGRU \rightarrow biAttn \rightarrow FC \rightarrow biAttn \rightarrow FC \rightarrow C2QAttn \rightarrow Span$$

The extra layer had limited impact, so in the next step we decided to replace one of the FC layers with another biGRU.

$$biGRU \rightarrow biAttn \rightarrow FC \rightarrow biAttn \rightarrow biGRU \rightarrow C2QAttn \rightarrow Span$$

This along with an extra head and objective that mimics a language model, proved to be the best model.

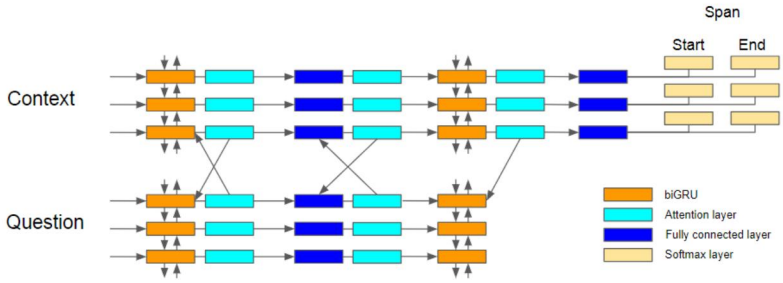

Figure 2: 3 layer model with 2 biGRU layers, the best model architecture we found

3.3 Other unsuccessful additions

A natural next step was to try to replace another FC layer with a biGRU

$$biGRU \rightarrow biAttn \rightarrow biGRU \rightarrow biAttn \rightarrow biGRU \rightarrow C2QAttn \rightarrow Span$$

However this proved to be too memory intensive to be feasible. We also tried adding another attention layer

$$biGRU \rightarrow biAttn \rightarrow FC \rightarrow biAttn \rightarrow biGRU \rightarrow biAttn \rightarrow FC \rightarrow C2QAttn \rightarrow Span$$

We also changed one of the biAttn layers to self-attention.

$$biGRU \rightarrow selfAttn \rightarrow FC \rightarrow biAttn \rightarrow biGRU \rightarrow biAttn \rightarrow FC \rightarrow C2QAttn \rightarrow Span$$

We noted no significant improvements

3.4 Extra Language modeling objective function

The most successful architecture was the one which had three layers and 2 GRUs

$$\begin{aligned} biGRU \rightarrow biAttn \rightarrow FC \rightarrow biAttn \rightarrow biGRU \rightarrow C2QAttn \rightarrow Span \\ \searrow FC \rightarrow L2Norm \end{aligned}$$

We explored an idea where we add an extra FC layer after the first biAttn layer to predict the word vector for the next word. The L2 norm between the predicted vector and actual vector is added as another term in the objective. The idea is to train representations in the biGRU that would be present in a language model. This can also be thought of as an auto encoder for the word vectors. After training the joint objective till plateau, the extra term in the objective is removed and the model is trained again to obtain a refined model.

3.5 Improvements in Span calculation

Instead of just computing the argmax of the start and end distributions, a more sophisticated approach is to find the pair i,j (i¡j) such that $P_{start}(i)P_{end}(j)$ is maximized. This can be computed easily by computing the matrix of products for all i and j and find the maximum in the upper triangle region of the matrix. This produced a significant improvement in score of about 2 points for both EM and F1.

4 Experiments

4.1 Training

All the models were trained with the same learning rate (0.001) and only the promising ones had it reduced by a factor of 5 at later stages. Not much hyper-parameter search was attempted as the goal was to do an architecture search with basic building blocks. A good change in architecture produced

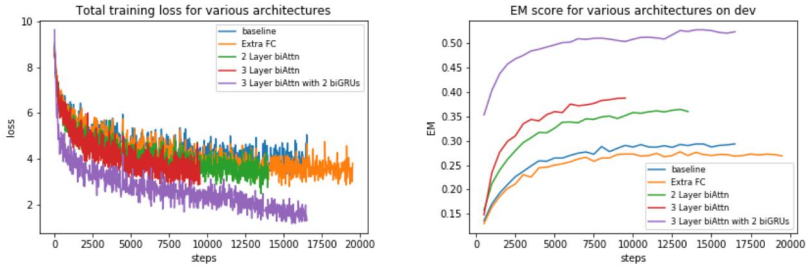

Figure 3: Training loss and dev EM score for each of the models.

a significant boost in performance, while hyper-parameter tuning can then be applied to the best architecture at the end. Main observations during the architecture search were the following. When we added fully connected layers to the baseline model, the performance on the dev set actually dropped, possibly because the increase in parameters caused the model to overfit.

When another biAttn layer and FC layer was added, the performance significantly improved, indicating that the attention mechanism is more important than just the extra FC layer. When a third biAttn and FC layer pair was added, the performance did increase but not by as much. The FC layer in the third biAttn layer was then replaced with a biGRU. This again increased the performance significantly, and proved to be the best feasible model. A model with another biAttn layer was also tested, although performance was slightly better, the training was too slow.

The training process for the language model experiment was very simple, we trained with the language model objective, till plateau, then we remove that term in the objective and further train the model.

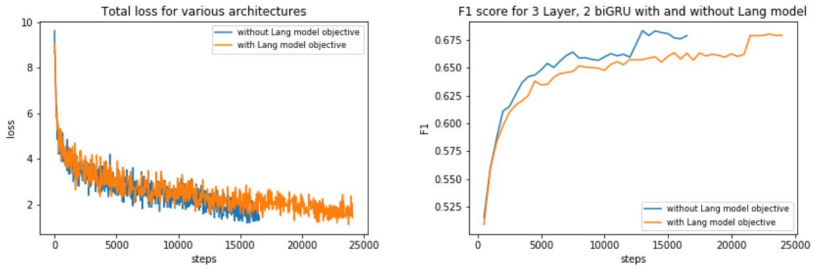

Figure 4: Training loss and dev EM score for each of the models.

There was no significant improvement from this procedure, just a marginal increase in performance on dev set. It did produce the best model amongst all the others, thus it was submitted to the leader board.

Below is the performance on the test set.

Table 1: Results on hidden test dataset

Score	Value
F1	74.382
EM	63.799

4.2 Basic analysis of data

We performed some basic analysis of the data, such as how many of each type of question is present in the dev set. We also looked at the length of answer for the different types of questions. We also

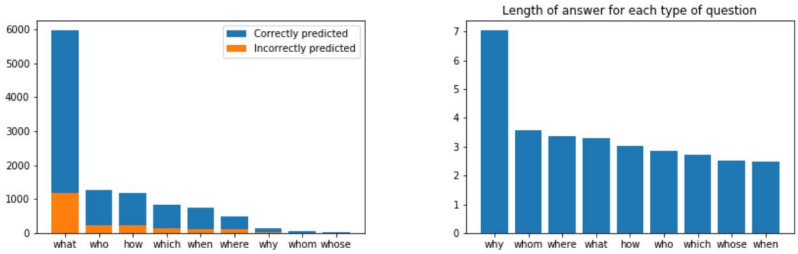

Figure 5: Analysis based on type of question

show in orange the portion of the questions that our best model gets very bad i.e. has an F1 score less than 0.01. As we observe, the most common type of question involves the word ”what” and also the most incorrect answers. Questions with the word ”why” tend to have significantly longer answers as they usually need to be descriptive, but are not very common.

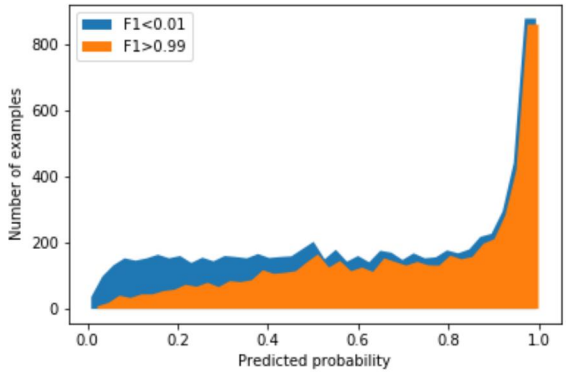

Figure 6: Histogram of either fully correct or fully wrong answers vs predicted probability of being correct

We also study how the model’s own confidence in its answer relates to whether it is actually correct. We partition all the examples in the dev data set into whether the model got it right exactly F1¿0.99 (orange) and got it completely wrong F1¡0.01(blue). We then plot the combined histogram of the predicted probability of the interval, which is the product of the start and end probabilities of the span.

As we see in the graph, evidenced by the peak at 1, the model gives an answer with very high probability in a significant number of cases, and it is correct in most of these cases. As its confidence drops, so does the number of cases where it gets it correct.

This is interesting because if we allow the model to say that it is not confident of the answer and thus will refuse to answer, it can significantly lower the chance of giving a wrong answer. This might be of more interest in a more practical setting, where such a system will be useful.

4.3 Experimenting with simple inputs and looking at attention outputs

In order to get a better understanding of how the model works, we can inspect the attention outputs of the various attention layers. For every biAttn layer each word in both the question and context

attends to every other word in the other paragraph, to make it easier to analyze, we look at simple examples.

In the example in the figure, we have the simple context ”the cat is on the table . a dog is below the table” and we ask the question ”what is above the dog ?”. The answer we expect is underlined, and the model prediction is in bold. The probability of it being the start and end of the answer span for each word is shaded with different colours. In this example the model answered with ”the table”, which is not completely wrong, but not usually what we mean when we ask the question.

Figure 7: The probability and attention outputs for a simple example

If we now look at the attention outputs, we can make certain observations. In most cases if a word in the question is present in the context, then the most relevant instance of that word in the context is attended to, the most. Of course it need not be the exact same word, it could be the most relevant word semantically, which could be a synonym or antonym etc. In the example above, the word ”above” attends to the word ”below”. This shows that the model does make these semantic connections. Similar observations for the C2Q attentions, the word ”below” attends to ”above”.

We can look at another example, with more a complex question and context. In the second example, we test the model’s ability to bring together different parts of the context to find the correct answer. We also test its knowledge of complimentary relations such as ”brother” and ”sister”. The question never mentions ”julia”, it only uses ”adam ’s sister”, but we see that the model correctly resolves the reference if we look at the attention output, the word ”sister” strongly attends to ”julia”. We also added a distracting statement, with another entity ”crossing the street” for a different reason, but it was not affected by this.

In these examples the higher attention layers do not have a significant pattern, as the context and question are simple, but for more complex questions with a lot of composite statements, the higher

Figure 8: The probability and attention outputs for another example

attention outputs are used in similar fashion. More such examples can be found in the supplementary materials.

5 Conclusion

By constructing the model step by step we can learn the impact each of these building blocks have on performance. Attention seems to be very important, more than just fully connected layers on single words. Adding more attention layers does help, but marginal improvement is lower. Extra biGRU layer had the most significant impact. The extra term in the objective that mimics a language model had very limited impact, but this idea could possible be explored further, by building a more sophisticated language model on the same dataset.

The attention outputs provided insights as to how reference resolution is represented in the model. All attention layers do not show salient activity for simple examples. The more complex the question, more the higher attention layers are used. This can be investigated further is future work by looking at what types of questions the model fails on as we successively add more attention layers.

References

[1] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. CoRR, abs/1606.05250, 2016.

[2] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. arXiv preprint arXiv:1611.01603, 2016.

[3] Caiming Xiong, Victor Zhong, and Richard Socher. Dynamic coattention networks for question answering. arXiv preprint arXiv:1611.01604, 2016.

[4] Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov. Gated-attention readers for text comprehension. arXiv preprint arXiv:1606.01549, 2016.