CS 224N Default Final Project: Question Answering

Raghunath Krishnamurthy(kichyrag)
SCPD - Stanford University
kichyrag@stanford.edu

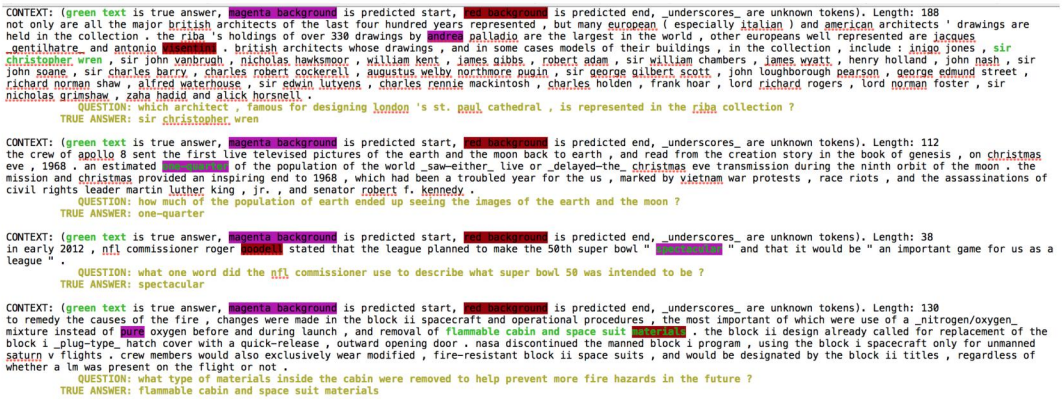
Abstract

The default project is to build a system for the Stanford Question Answering Dataset **(SQUAD)**. The goal is to understand the context from a provided paragraph, followed by query(question) and system should output the correct answer. Basic structure of the system implemented will be improvements to the baseline model, which has **RNN encoder layer**, which encodes both the context and the question into hidden states, an **Attention Layer**, that combines the context and an output layer, which applies a fully connected layer and then two separate softmax layers, one of them to get the start location, and the other one to determine the end location of the answer. The improvements attempted, were to implement **Bidirectional attention flow (BiDAF)**[1], which was quite unsuccessful because of decode layer was kept same softmax, and this impacted the dev F1 and EM score to be very low (2.6%) and failed to improve even after 10k iterations. Finally, several adjustments to the baseline model was done such as dropout, learning rate decay and LSTM instead of GRU. Finally the model with LSTM and dropout of 0.20 achieved the best results. Future work is to improve on the BiDAF and implement LSTM instead of softmax for decoding to achieve higher accuracy.

1 Introduction

The intution behind attention mechanism is that the model is able to recognize the difference between context encodings, where the question pays more attention.

Some examples of the Questions and the true answers:



2 Approach

Approach 1

Basic structure of the system implemented will be improvements to the baseline model, which has **RNN encoder layer**, which encodes both the context and the question into hidden states, an **Attention Layer**, that combines the context and an output layer, which applies a fully connected layer and then two separate softmax layers, one of them to get the start location, and the other one to determine the end location of the answer.

The context and question are fed into an encoder, which had two independent BiLSTM with dropout. The context is represented as sequence of d-dimensional word embedding, and the question by a sequence of d-dimensional word embedding. The GloVe embedding are fed into a 1-layer bidirectional LSTM, which produces and sequence of hidden states, both forward and backward.

Attention layer is a dot-product attention, with context hidden states attend to question hidden states, which produces attention distribution. The attention distribution is used to produce weighted sum of question hidden states.

The attention output is then concatenated to context hidden states to obtain blended representation.

Decoder The fully connected layer is then fed into two separate softmax layers, one of them to get the start location, and the other one to determine the end location of the answer.

Approach 2

The next model attempted to improve on the basic attention, instead of using dot-product is using BiDAF [1], which uses similarity matrix to compute context-to-question attention using weighted sum of question hidden states with softmax on similarity matrix. The next step is perform question-to-context attention, which takes max of the corresponding row of the similarity matrix. The attention distribution here would be softmax of max of similarity matrix, which is used to take a weighted sum of context hidden states.

Finally the attention output is determined by combining context hidden states, context-to-question attention output and question to context attention output. This is then fed to the decoder from approach 1.

3 Experiments

Based on the readings and recommendation, BiDAF would have been the best model. The model suffered from decoder as it was fed into the softmax and needed something like Bidirectional LSTM to hav e had good accuracy.

I decided to change for tuning better hyperparameters to fit the basic attention model to perform better. Couple of the experiments that were undertaken were 1. Change the Batch size (Increase the batch size)
2. Increase the size of hidden states 3. increase the embedding size of the pre-trained word vectors

Moat of the above changes worked well on training, but did not work well on official evaluation and suffered OOM on GPU.

Finally the best combination found by changing the dropout to 0.20, which performed well on dev and test set. I was able to achieve 44% F1 accuracy with that setting.

4 Conclusion and Next steps

The basic dot-product attention structure works well in achieving good results, and the key takeaway was just increasing the embedding size, batch size or hidden states is not good. The important future works are defenitely adding regularization to prevent over-fiiting. I am planning on continuting to improve and work on completing the BiDAF decoder and other models like co-attention to improve the accuracy.

References

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