Machine Comprehension Task with BiDirectional Attention Flow on SQuAD Dataset

Shim-Young Lee
Department of Statistics
Stanford University
sylee1@stanfordedu

Abstract

In this default project (assignment 4) for the course CS224n Natural Language Processing with Deep Learning, a challenging NLP task called machine comprehension was explored in the form of question answering. The SQuAD dataset provides the contexts and corresponding questions, and the task is to correctly highlight, or mark the beginning and the end of, a word sequence in the context paragraph that answers the question. I augmented the baseline model with the attention mechanism of the BiDirectional Attention Flow and Bidirectional LSTMs in the contextual embed layer and modeling layer of the neural network. A single model achieved THIS F1 and THIS EM.

1 Introduction

The goal of this project is to explore a natural language processing (NLP) task of machine comprehension in the form of question answering, where the level of comprehensive capability of a machine is assessed by evaluating the machine generated answers given a context paragraph and a question about the context. Machine comprehension is a challenging task, as the machine must be able to extract information from a relatively long sequence of words and represent, or summarize that information in a form that will allow the machine to use when answering a question. It shares many challenges of machine translation as it must understand the relationship between words and sentences; however, instead of mapping a word sequence in one language to another, it also must extract useful information in both the context and the question and model the complex interaction between them in order to come up with a correct answer. The "answer" in this specific task with SQuAD dataset[4] is given by marking the start and the end position of a word sequence in the context paragraph.

In this project, a baseline model with GRU[1] cells in the encoding layer and a basic attention mechanism was provided. The task was to replace or augment this provided baseline model in a single or multiple modules of the network architecture to achieve better performance. I chose to implement Bidirectional Attention Flow (BiDAF) [5] whose architecture has the similar modularization/hierarchy as the baseline model. BiDAF is one of the highest performing model for this task one SQuAD dataset.

2 Related Work

As previously stated, the related work that was most influential to this project is the Bidirectional Attention Flow model, whose attention mechanism was implemented here. The general organization of components or layers other than the attention layer is also very similar for the BiDAF and the version I implemented for this project. The full BiDAF model has other extensions such as character-level CNN embedding layer.

Other important model that performed well on SQuAD dataset is Dynamic Coattention Network by Xiong *et al*[7]. Similarly to the BiDAF model, one of the major contributions of the Coattention Network is its attention mechanism in the Coattention layer, which uses a two-way attention between the context and the question, as well as the second-level attention computation that attends over attention outputs.

3 Dataset and Features

The dataset used for the question answering task here is the Stanford Question Answering Dataset (SQuAD). SQuAD is a reading comprehension dataset; it consists of about 100,000 context paragraph-question pairs extracted from Wikipedia articles as well as the answers to the questions, built through crowdsourcing. From the perspective of a model, each input is a pair of context paragraph and a question about that paragraph, and the goal is to answer the question correctly. The training set has roughly 85,000 context-question pairs, and the dev set has about 10,000 pairs. The test set is similar to the dev set in terms of size, and I did not have direct access to the test set, and it was used only for the final evaluation/submission.

To make the assessment of the correctness of the machine generated answers easy, the answers are always taken directly from the context paragraph. Specifically, the answer is specified by marking the start and the end positions of the word sequence in the context paragraph, so the question answering system needs not generate a new word sequence. The accuracy of the answers are evaluated simply by comparing these machine generated start and end positions with the answers provided by the dataset.

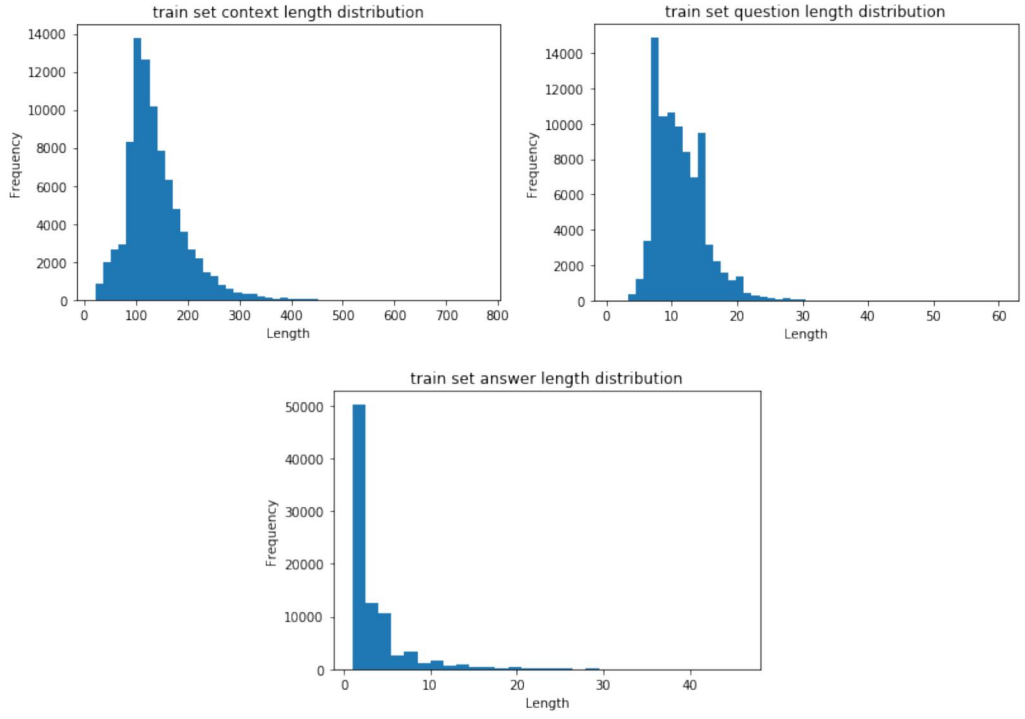

Figure 1: Histogram of sequence lengths for context paragraphs, questions and answers in training set.

Table 1: 95 and 99 percentile of the sequence lengths for context paragraphs, questions and answers in training set.

Word sequence type	95 percentile length	99 percentile length
Context	245	325
Question	18	23
Answer	10	18

The training and inference with the model involve tensor multiplications, so the maximum sequence lengths of a context paragraph and a question are hyperparameters to be set. To determine the cut-off length for each effectively, histograms of sequence lengths were used. The figure 1 and table 1 summarize the distributions of lengths for context paragraphs, questions and answers. Based on these statistics, the maximum length for context paragraphs was set to 300 and for questions 25.

4 Model Architecture

First, I provide the higher level description of the modularization and hierarchy of the model architectures that all models tested follow.

Both the baseline model and the BiDAF models consist of three major components, or layers. The first layer (besides the very first word embedding layer) is the RNN encoder layer, which takes in the word level embedding vectors of the sentence and uses some RNN encoder (GRU or LSTM) to encode the word sequence into hidden states. In this layer, context paragraph and question are encoded with the same RNN cell, so the weights are shared.

The next layer is the attention layer. The interaction/similarity between context and question hidden states are computed, with dot-product in the simplest case, and the score is turned into a distribution by applying softmax to the scores. The distribution is then used as a weight in the weighted sum of either context or question hidden states, which is blended together to be fed into following layers.

The layer following the attention layer is the output layer, where the blended representation of reweighted context and question vectors are fed into the usual fully connected layers. Specifically, affine transformation followed by nonlinear activation (ReLU was used for all models discussed) is applied to the blended representation, and the output once again goes through another affine transformation and then the softmax is applied to turn these logits into a probability distribution. Note that the probability distribution here is the probability of each word in the context paragraph being the start/end position of the highlight span, so two output layers are needed, each for computing the distribution for the start position, and the end position. The final prediction is made simply by taking the argmax of the probability distribution.

Next, the implementation details for each component of respective models are dicussed.

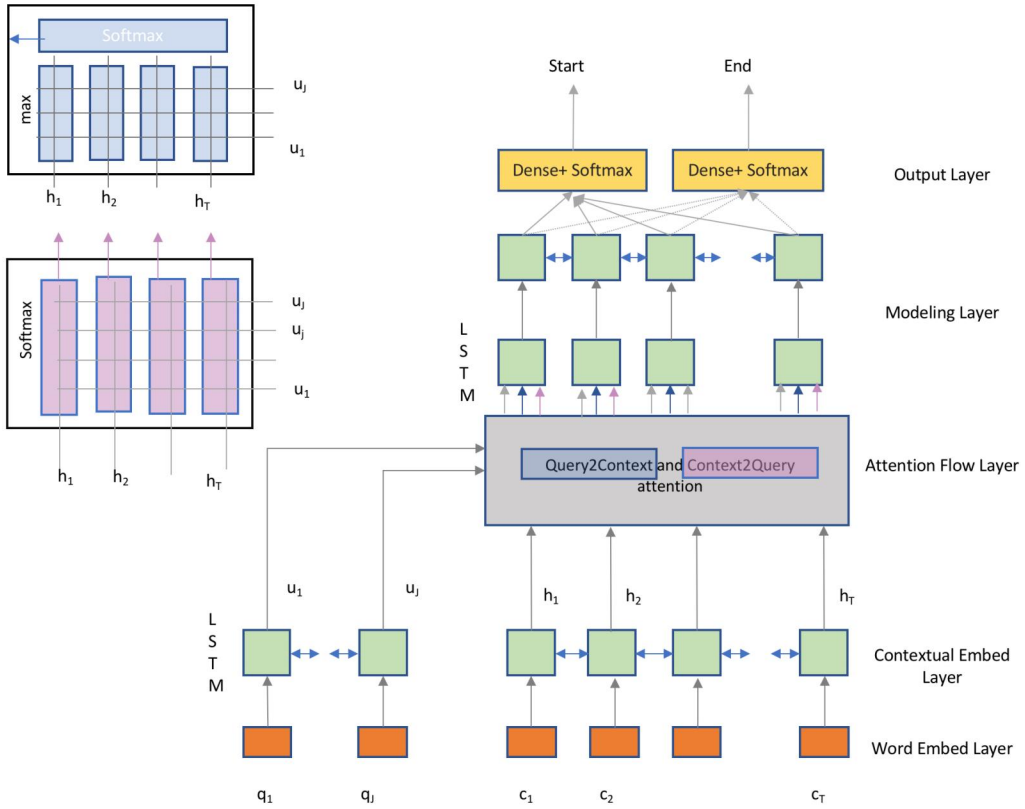

Figure 2: Model Architecture diagram for BiDirectional Attention Flow model

4.1 Baseline model

Embedding Layer: The GloVe [3] word embedding vectors of size 100 was used.
RNN Encoder Layer: A bidirectional GRU was used for both context and the question. An embedded sequence (GloVe) of context and query vectors $\boldsymbol{c} = [c_1, c_2, \dots, c_T]$ and $\boldsymbol{q} = [q_1, q_2, \dots, q_J]$ are fed into the RNN cell and the representation matrix $\boldsymbol{H} \in \mathbb{R}^{T \times 2h}$ and $\boldsymbol{U} \in \mathbb{R}^{J \times 2h}$ each for context and question are output ($2h$ since bidirectional, where h is the size of a single hidden state).
Attention Layer: A simple dot-product attention was used: similarity matrix $\mathbf{S} = \mathbf{H}_{t:}\mathbf{U}_{j:}^{T}$, attention weight vector $\boldsymbol{a}_t = softmax(\mathbf{S}_{t:}) \in \mathbb{R}^{J}$ was computed and used to reweight the question vectors, yielding $\widetilde{\mathbf{U}}_{t:} = \sum_j \boldsymbol{a}_{tj}\mathbf{U}_{j:}^{T} \in \mathbb{R}^{T \times 2h}$.
Output Layer: A simple output layer described above was used.

4.2 Bidirectional Attention Flow

Embedding Layer: The GloVe word embedding vectors of size 100 was used. GloVe vectors of higher dimension were not used nor tested due to the training speed and the GPU memory limitation.

RNN Encoder Layer: Two layers of bidirectional LSTM was used. This is the contextual embed layer in the figure 2 (as the authors of BiDAF called it).

Attention Layer: The similarity matrix $\mathbf{S}_{tj} = \mathbf{w}_{sim}^{T}[\mathbf{H}_{t:}, \mathbf{U}_{j:}, \mathbf{H}_{t:} \circ \mathbf{U}_{j:}] \in \mathbb{R}$ was computed, then the Context2Question attention, $\alpha^t = softmax(\boldsymbol{S}_{t:}) \in \mathbb{R}^J$, $\widetilde{\mathbf{U}}_{t:} = \sum_{j=1}^{J} \alpha_j^t \mathbf{U}_{j:}$ and the Question2Context attention $\boldsymbol{m}_t = \max_j \boldsymbol{S}_{tj} \in \mathbb{R}$, $\beta = softmax(\boldsymbol{m}) \in \mathbb{R}^N$, $\mathbf{h} = \sum_{t=1}^{T} \beta_t \boldsymbol{H}_{t:}$, and $\widetilde{\boldsymbol{H}}$ is constructed by tiling $\mathbf{h}$ to match the dimension of the final blended representation, $\mathbf{B} = [\mathbf{H}; \mathbf{U}; \mathbf{H} \circ \widetilde{\mathbf{U}}, \mathbf{H} \circ \widetilde{\boldsymbol{H}}]$

Modeling Layer: This is a layer not described in the higher lever overview in the beginning of the section. Instead of feeding the blended representation right into the output layer, two layers of bidirectional LSTM were used to encode the representation.

Output Layer: A simple output layer described above (same as the baseline model) was used.

The figure 2 summarizes the overall architecture and flow of computation.

4 Result and Discussion

In assessment of the performance of question answering task, F1 score and Exact Match (EM) score are used. F1 score is the harmonic mean of precision and recall, and Exact Match is a simple accuracy score that computes the proportion of predictions that exactly match the answer span (given by the start and the end positions).

The baseline model achieves F1 score of 43.027 and the EM score of 34.134 in the dev set with about 16k iterations of training step with batch size 100.

The first augmented model trained was the BiDAF model without the modeling layer (BiDAF1), that is, the baseline model with replaced attention mechanism (one layer of GRU RNN Encoder layer, BiDAF attention layer, and output layer). This model achieved the F1 score of 48.77 and EM score of 35.47 in the dev set with over 20k iterations of training steps with batch size 100. The improvement in performance was fairly minor given the increase in the complexity of the attention layer.

Fnally, the BiDAF model with one layer of bidirectional LSTM and two layers of bidirectional LSTM in the contextual embed layer and the modeling layer respectively, and the BiDAF attention mechanism (BiDAF2), achieved the F1 score of 73.115 and EM score of 63.084 in the dev set, and F1 score of 73.553 and EM score of 64.219 in the test set with only about 12k iterations of batch size 64.

Table 2: The F1 and EM scores of each model.

Model	F1	EM
Baseline(dev)	43.027	34.134
BiDAF1(dev)	48.77	35.47
BiDAF2(test)	73.553	64.219

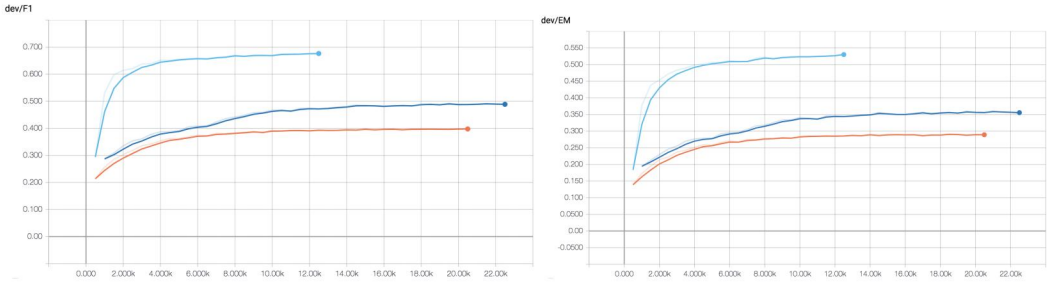

Figure 3: Dev set F1 score (left) and Dev set EM score (right) as a function of iterations.
Orange: baseline, navy: GRU BiDAF model with no modeling layer,
sky blue: LSTM BiDAF with a modeling layer

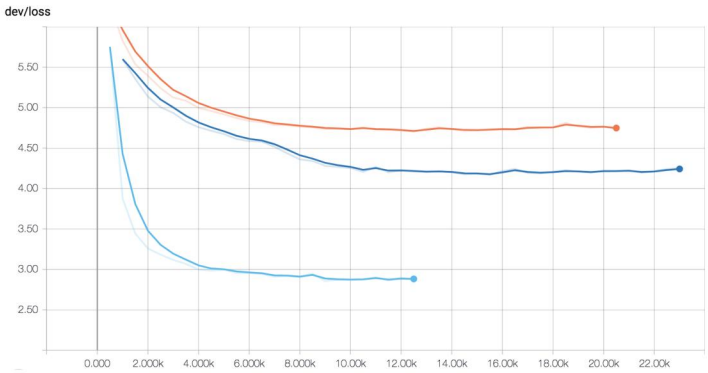

Figure 4: Dev set loss as a function of iterations.
Orange: baseline, navy: GRU BiDAF model with no modeling layer,
sky blue: LSTM BiDAF with a modeling layer

The figures above show the dev set evaluation of F1 and EM scores as well as the loss of three models as a function of iterations. (I did not save the loss and F1/EM scores separately, so I took the figures directly from the TensorBoard visualization tool.)

All three plots show that BiDAF2 achieves loss and performance in only a few thousand iterations the other two models couldn't achieve in more than twice the training steps. Also keep in mind that the batch size was smaller for BiDAF2, due to the GPU memory constraint.

The overwhelming difference in the performance of BiDAF1 and BiDAF2 suggests that BiDAF's attention mechanism alone cannot improve the performance of the baseline model without the modeling layer. Even with the increase in model complexity in BiDAF2 due to the change in the RNN encoder from GRU to LSTM and two layers of modeling layer, overfitting was much more severe in BiDAF1, where the training F1 and EM scores went up as high as over 85. On the other hand, for BiDAF1, the F1 and EM scores for training set and dev set remained roughly similar throughout the training, and the model performed better on the test set than on the dev set. This might be due to the decrease in the size of hidden states for all RNN encoders (was forced to do so due to the memory constraint).

5 Conclusion and Future Work

For this default project, I implemented some of the components of the BiDirectional Attention Flow to augment the baseline model, and was able to achieve improvement of about 25 in F1 score and 30(%) in EM score. The more complex attention mechanism, however, improves the model significantly only when the contextual embed layer and the modeling layer are sufficiently complex.

One possible, obvious extension to the current model is the character-level CNN, which uses the character embedding to represent the characters and provides more subtle and refined interaction of word sequences in character level in addition to the word level embedding.

References

[1] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," ArXiv eprints, Jun. 2014.

[2] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp.1735-1780, Nov. 1997.[Online]. Available: http://dx.doi.org/10/1162/neco.1997.9.8.1735

[3] Pennington, J., Socher, R., & Manning, C. D. "GloVe: Global Vectors for Word Representation," 1532–1543. Retrieved from http://www.aclweb.org/anthology/D14-1162, 2015

[4] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ Questions for Machine Comprehension of Text," ArXiv e-prints, Jun. 2016.

[5]M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," arXiv preprint arXiv:1611.01603, 2016.

[6] Sutskever, I., Vinyals, O., and Le, Q.. "Sequence to sequence learning with neural networks," In *Advances in Neural Information Processing Systems*. 2014

[7] C. Xiong, V. Zhong, and R. Socher, "Dynamic Coattention Networks For Question Answering," ArXiv e-prints, Nov. 2016.