Question Answering System on SQuAD dataset

Liuming Zhao
Stanford University
lxz299@stanford.edu

Taiming Zhang
Stanford University
tzhang55@stanford.edu

Abstract

In this project, we focused on building a question answering system for reading comprehension. We aim to improve the performance of existing BiDAF baseline model which produces more accurate answers. We built upon the BiDAF baseline model with self-attention layers, character-level embedding, and no-answer layers. Additionally, we adapted DrQA to our baseline BiDAF model. We finally ensembled different models and achieved an F1 score of **63.745** and EM score of **60.152** on test non-PCE leaderboard.

1 Introduction

Reading comprehension has always been a challenging task in machine learning and natural language processing. In reading comprehension task, we are given a context and a query, and our goal is to predict a segment of the context that represents the answer to the query. With the development of deep learning, researchers have achieved impressive performance in reading comprehension. Such achievements mostly relies on a powerful dataset that the model can be trained on. Stanford Question Answering Dataset (SQuAD) [1] has gained increasing popularity and become one of the most fundamental datasets for researchers in reading comprehension task.

In this paper, we built upon the baseline BiDAF model with the addition of character-level embedding and self-attention. We also adapted some state-of-the-art model architecture such as DrQA [2] to our baseline model. We will discuss the detailed implementation in the following sections. We then show our experiment details including the dataset and several important hyperparameters, and finally, we summarize the performance of each modification we made and provide analysis to these modifications.

2 Related work

In recent years, there have been extensive researches on model architectures and implementation on SQuAD challenge. Seo et al proposed a Bi-Directional Attention Flow network [3], a multi-stage hierarchical process to represent the context at different level. They introduced the bidirectional attention flow technique to generate the query-aware context representation. Clark et al [4] proposed the DocQA that samples multiple paragraphs from the documents and used a shared normalization training objective that encourages the model to produce globally correct output. Moreover, Yu et al [5] avoided using recurrent neural networks which are often slow for both training and inference due to the sequential nature of RNN. Instead, they proposed the QANet, which combines local interaction using convolution models and global interaction using self-attentions models.

Besides improvements on model architectures, language embedding has also been extensively studied. Botha et al [6] presented a scalable method for integrating compositional morphological representations into a vector-based probabilistic language model. Furthermore, instead of word embedding, character-level convolutional neural networks has gained increasing popularity for a wide range of natural language processing tasks. Kim et al [7] developed a simple neural language

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

model that relies only on character-level inputs which achieves state-of-the-art performance while reducing 60% total number of parameters. In SQuAD 2.0 dataset, lots of unanswerable questions makes the inference even harder when models tried to infer a questions without an answer. Hu et al [8] presented a read-then-verify system which not only allows the neural reader to extract candidate answers and produces no-answer probabilities, but also leverages an answer verifier to decide whether the predicted answer is entailed by the input snippets.

3 Approach

3.1 Baseline

3.1.1 Word embedding layer

We used GloVe pre-trained word vectors to retrieve vector representation of each word in context and query [4]. Highway network will be used to output vector with a fixed size of 300 for each word. We can finally get vector representations of context and query with the size of (D, C_{len}) and (D, Q_{len}) respectively, where D is the dimension of each word vector (300 in our case), C_{len} is the sequence length of the context sentence and Q_l is the sequence length of the query sentence.

3.1.2 contextual embedding layer

The contextual embedding layer will further extract embeddings of the words based on the surrounding words in the context [3]. The baseline model used bi-directional LSTM network to refine 300 dimensional pre-trained GloVe word vector into context based word vectors. The output of forward and backward LSTM will be concatenated together to produce context based word vectors with the size of (2D, C_{len}) and (2D, Q_{len}) for the words in context sentences and query sentences respectively. In our baseline model, D is equal to 300 to produce 600 dimensional feature vectors for each word.

3.1.3 BiDAF attention flow layer

The BiDAF attention flow layer is mainly responsible for drawing connections between the context word vector and query word vector obtained from contextual embedding layer [3]. In our baseline model, two similarity matrices will be constructed to compute the similarities between the context and query. One matrix will compute similarities between one context sentence and all query sentences in order to find the most relevant query for each context sentence. Another matrix will compute similarities between one query sentence and all context sentences in order to find the most relevant context for each query sentence. Row-wize softmax will be used to compute the highest similarity between context and query. Finally, the attention features from both of two matrices will be concatenate to produce context word vectors with the dimension of (8D, C_{len}).

3.1.4 modeling layer

The modeling layer is mainly responsible to capture the dependencies among query-based context word vectors [3]. In our baseline model, two-layer LSTM model is used to output feature vectors with dimension of (2D, C_{len}).

3.2 Our improvements

3.2.1 Character-level embedding

In baseline model, we are provided GloVe pre-trained word embedding that converts word into word vectors. It is more preferable to consider character-level embedding and combine both word and character level embeddings as our input to encoder. Therefore, we adapted the character embedding from assignment 5 that allows out baseline model to both generate embedding vectors for both word-level and character-level. In order to feed both vectors into our encoder. We concatenate the word and character vectors and feed them to a highway layer. Highway layer is inspired by the gates of an LSTM. It adaptively carry some dimensions of the input directly to the output. Because we concatenated the word and character vectors into one vector, the hidden size of the resulting output doubled; we therefore added a linear projection layer to halve the hidden size.

3.2.2 Self-attention layer

In recent literature, self-attention has outperformed RNNs in various tasks. Self-attention can connect distant words through shorter network paths. Attention defines what portion of the input that the model focuses on, and it is largely used on perception in its early days. Self-attention refers to the attention that words pay to each other within a sentence. At every time step of an RNN, a weighted average of all the previous states are used as inputs to compute the next state. In this project, we implement self attention layer according to [4]. Unlike [4] using Bi-GRU, we used Bi-LSTM to convert word vectors with large embedding size to small one and then compute similarity matrix of the passage with itself. The implementation of similarity matrix is similar to the one for the BiDAF attention layer in order to be more memory efficient. Three trainable weight vectors has been used to perform linear transformation of the feature vectors of the paragraph according to [4]:

$$a_{ij} = w_1 \cdot h_i + w_2 \cdot h_j + w_3 \cdot (h_i \odot h_j)$$

Finally, we used additional fully connected layer to convert paragraph embedding with large hidden size to the smaller one in order to concatenate with the output from BiDAF attention layer.

3.2.3 POS embedding and NE embedding

In addition to self-attention layer, another improvement is to add POS(part of speech) and NE(name entity) tag as additional feature to the word embedding according to [2]. We used nltk to generate POS and NE tags for each word in sentence and use zero index as word padding. We also treat OOV as padding index for both POS and NE tags. This addition improves about 1.5 in both F1 and EM scores.

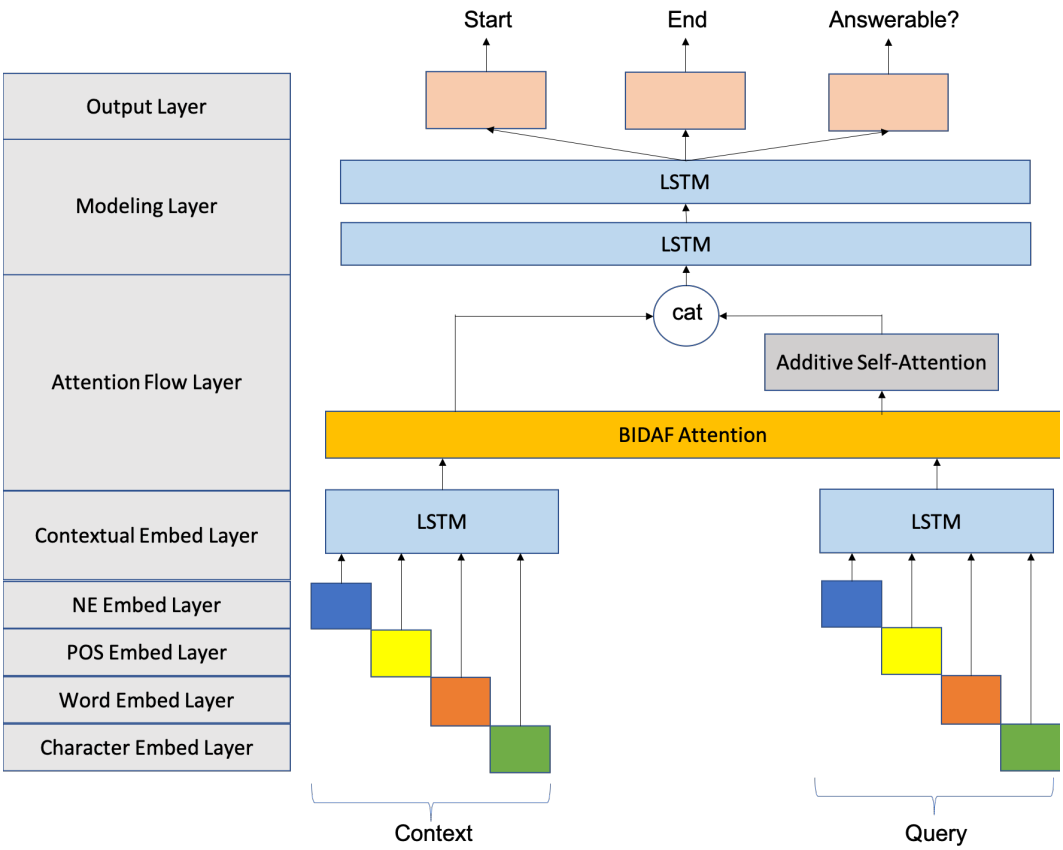

Figure 1: Model architecture

3.2.4 No-answer layer

Our baseline model uses negative log likelihood as our objective function shown in Equation [1]. It predicts "no-answer" when both start index and end index equal to zero that yields the highest probability. In our approach, we add another output layer which outputs score z that gives us the probability whether the question is answerable [4]. Therefore, our revised objective function becomes Equation [2]. However, in our experiment, this modification does not produce much difference compared with baseline model.

$$NLL = -log(\frac{e^{s_a g_b}}{\sum_{i=1}^{n}\sum_{j=1}^{n} e^{s_i g_j}}) \quad [1]$$

$$Loss = -log(\frac{(1-\delta)e^{z} + \delta e^{s_a g_b}}{e^{z} + \sum_{i=1}^{n}\sum_{j=1}^{n} e^{s_i g_j}}) \quad [2]$$

where s_i and g_j are the scores for the start and the end bounds produced by the model for token j, a and b are the correct start and end tokens, z is the possibility of "no-answer" option, and δ is 1 if an answer exists and 0 otherwise.

4 Experiments

4.1 Data

In this project, we used Stanford Question Answering Dataset (SQuAD 2.0) to train and test our model, where the dataset comes from the questions posed by crowdworkers on Wikipedia articles and answers by a span, quoting a segment of the text. The dataset consists of 100,000 questions from SQuAD 1.1 with over 50,000 new, unanswerable questions in order for the model to be adaptive to problems with no answers. For this project, we are only given training set and dev set for SQuAD 2.0 because the test set is kept secret and we split the dev set into dev set and test set. For each question in the dataset, we are given three gold answers which we will make comparison with our predicted result.

4.2 Evaluation methods

We evaluate our model with two evaluation metrics, Exact match (EM) and F1 score. In Exact Match metric, we will consider the result accurate if it matches any one of the three golden answers whose equation as shown in Equation [1]. In F1 score metric, we take the result and each golden answer as bags of words and evaluate precision, recall, and finally the harmonic mean F1 shown in Equation [3]-[6].

$$EM = \frac{E_{ExactMatch}}{N_{Total}} \quad [3]$$

Equation [1]: Exact Match metric

$$precision = \frac{TP}{TP + FP} \quad [4]$$

$$recall = \frac{TP}{TP + FN} \quad [5]$$

$$F1 = \frac{2 * precision * recall}{precision + recall} \quad [6]$$

Equation [3]-[6]: F1 score metric, where TP is "True Positive", FP is "False Positive", and FN is "False Negative"

4.3 Experimental details

We have built upon our baseline model with modifications mentioned in Approach section and the model architecture is illustrated in Figure 1. Specifically, we have added character-level embedding, self-attention layer, POS embedding and NE embedding, and no-answer layer to our baseline model. With all the modifications added, the total training time is around 1 hour and 20 minutes each epoch. We tried to vary learning rate between 0.2 and 0.8, hidden size between 50 and 150, and dropout probability between 0.2 and 0.4. We then ensemble our model with different types of self-attention layers to get our best result, and the results are shown in next section.

4.4 Results

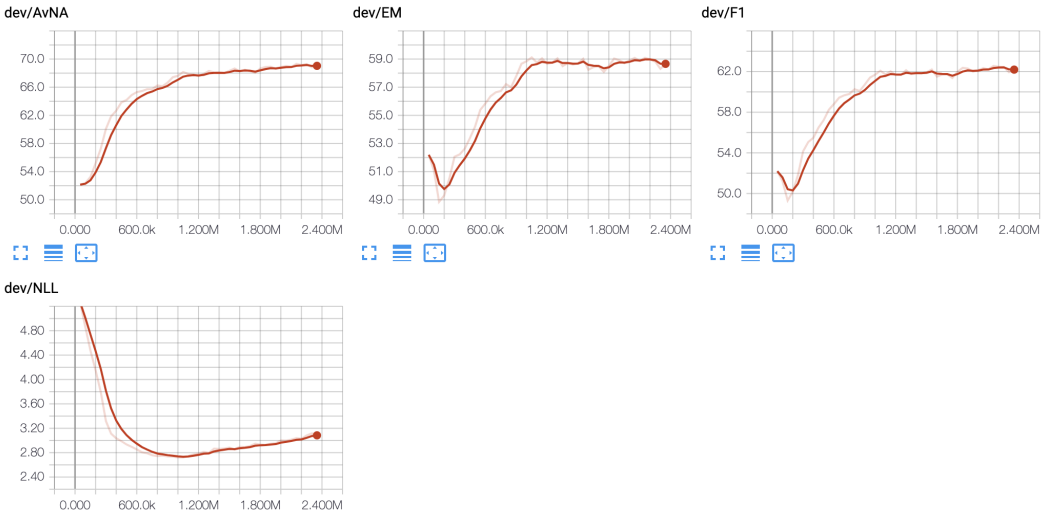

Figure 2: Training curve on Tensorboard

Model Variations	EM score	F1 score
Baseline model	56.298	59.920
Self-attention	57.16	60.25
DrQA	59.02	62.38
No-answer layer	57.03	60.08
Self-attention+DrQA	59.08	62.55
Ensemble	60.21	63.40

Table 1: Results for different model modifications in DEV set

Our results are summarized in Table 1 (in DEV set). Our final EM and F1 scores on test non-PCE leaderboard are **60.152** and **63.745**, respectively. In table 1, we can see that the largest increase is due to the DrQA, which indicates that it contributes a lot to our final result. We also noticed that there is little improvements on no-answer layer, and we believe that there are more effective way of predicting the no-answer option.

5 Analysis

- When looking through the predicted outputs and ground truth outputs in dev set, we can clearly see a lot of questions that output model falsely predict an answer while the question indeed is unanswerable. This is the common issue when dealing with SQuAD 2.0 dataset. We can conclude that our model still is not well capable of predicting no-answer scenarios. We have read some papers discussing how to predict whether the question is answerable. In [8], Hu et al proposed a model that is specifically used to determine whether the question is answerable by using read-and-verify system, which we believe is a great example to try in our future work.

- We have also discovered some wrong answers when the model falsely aligned the attention between the context and the question.

  Context: "Advances in polynomial algebra were made by mathematicians during the Yuan era. The mathematician Zhu Shijie (1249–1314) solved simultaneous equations with up to four unknowns using a rectangular array of coefficients, equivalent to modern matrices. Zhu used a method of elimination to reduce the simultaneous equations to a single equation with only one unknown. His method is described in the Jade Mirror of the Four Unknowns, written in 1303. The opening pages contain a diagram of Pascal's triangle. The summation of a finite arithmetic series is also covered in the book."

  Question: "What modern math concept did Zhu Shijie do work similar to?"

  Ground truth: "matrices"

  Predicted: "Jade Mirror of the Four Unknowns"

  The model fails to align "work similar to" in the question with "equivalent to" in the context. Consequently, the model predicts the wrong answer. One possible improvements to overcome this problem is to try to use convolution-based attention layer to replace our attention layer in baseline model.

- We have also seen a few question that our model roughly predicts the correct results but with an incorrect span. This kind of mistakes is hard to avoid for machine and even for human ourselves. The below example illustrates our point.

  Context: "Both B cells and T cells carry receptor molecules that recognize specific targets. T cells recognize a "non-self" target, such as a pathogen, only after antigens (small fragments of the pathogen) have been processed and presented in combination with a "self" receptor called a major histocompatibility complex (MHC) molecule. There are two major subtypes of T cells: the killer T cell and the helper T cell. In addition there are regulatory T cells which have a role in modulating immune response. Killer T cells only recognize antigens coupled to Class I MHC molecules, while helper T cells and regulatory T cells only recognize antigens coupled to Class II MHC molecules. These two mechanisms of antigen presentation reflect the different roles of the two types of T cell. A third, minor subtype are the T cells that recognize intact antigens that are not bound to MHC receptors." "

  Question: "What kind of T cells have the purpose of modulating the immune response?"

  Ground truth: "regulatory T cells"

  Predicted: "regulatory"

6 Conclusion

In this paper, we have made several modifications on the baseline model. Specifically, we have added character-level embedding, self-attention layer, POS embedding and NE embedding, and no-answer layer to our baseline model. The results showed improvements on each modification that we made, with DrQA being the best improvement. The final EM and F1 scores we have on test non-PCE leaderboard are 60.152 and 63.745, respectively. Throughout this project, we have learned the ability to improve the model performance by researching on state-of-the-art methods and analyzing the advantages and disadvantages of different modifications.

In the future, we would like to implement QANet which is reported to have higher EM and F1 scores on SQuad 2.0 dataset. We can also perform data augmentation on our dataset [5]. Specifically, we can translate English to French by machine translation and then translate it back to English. Finally, we can implement the read-and-verify system [8] which first predicts whether the question is answerable and then try to predict the answer if the question is answerable.

References

[1] Rajpurkar, Pranav, Robin Jia, and Percy Liang. "Know What You Don't Know: Unanswerable Questions for SQuAD." arXiv preprint arXiv:1806.03822 (2018).

[2] Chen, Danqi, Adam Fisch, Jason Weston, and Antoine Bordes. "Reading wikipedia to answer open-domain questions." arXiv preprint arXiv:1704.00051 (2017).

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

[4] Clark, Christopher, and Matt Gardner. "Simple and effective multi-paragraph reading comprehension." arXiv preprint arXiv:1710.10723 (2017).

[5] Yu, Adams Wei, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. "Qanet: Combining local convolution with global self-attention for reading comprehension." arXiv preprint arXiv:1804.09541 (2018).

[6] Botha, Jan, and Phil Blunsom. "Compositional morphology for word representations and language modelling." In International Conference on Machine Learning, pp. 1899-1907. 2014.

[7] Kim, Yoon, Yacine Jernite, David Sontag, and Alexander M. Rush. "Character-aware neural language models." In Thirtieth AAAI Conference on Artificial Intelligence. 2016.

[8] Hu, Minghao, Yuxing Peng, Zhen Huang, Nan Yang, and Ming Zhou. "Read+ verify: Machine reading comprehension with unanswerable questions." arXiv preprint arXiv:1808.05759 (2018).