Question Answering System with Bidirectional Attention Flow

Hsu-kuang Chiu
Stanford University
hkchiu@stanford.edu

Ting-Wei Su
Stanford University
twsu@stanford.edu

Abstract

Question answering is a complicated task in natural language processing. It requires not only the comprehension of the question and the passage, but also finding out the location of the most critical words. In this project, we reimplemented the Bidirectional Attention Flow model (BiDAF) and analyzed the effectiveness of each layer. We focused on the analysis instead of developing a new model.

1 Introduction

Question answering, or reading comprehension, is an important topic in natural language processing. However, it remains to be a very challenging task. As neural networks become popular and challenge datasets like Stanford Question Answering Dataset (SQuAD) come out [2], more and more models have been developed. Most of these models focus on how to extract the most effective attention of a question and the corresponding context. R-net uses self-attention to predict the answer [3]. In Dynamic Coattention Network, they use coattention, which involves a second-level attention computation. Seo *et al.* apply bidirectional attention flow between question and context [1]. In addition to the attention, many other tips such as character-level embeddings and smarter span selection methods have been proposed.

Every model includes some special characteristics, but it is not so clear about the effectiveness of each of them. Therefore, we are interested in testing the effects of each layer inside the model to see if the most critical part of the model is the same as what the original paper claimed. In this project, we reimplemented BiDAF [1], which is one of the state-of-the-art model on SQuAD leader board. This model comprises character-level embeddings, word-level embeddings, phrase embedding layer, bidirectional attention flow, and modeling layers. Among these components, we especially want to test whether the most critical part of this model is the bidirectional attention flow. In our final discussion, we also analyzed the results and listed several common types of errors.

2 Method

The whole structure of our adjusted BiDAF is shown in Figure 1.

2.1 Embed Layers

At the beginning of the structure are three embed layers, character, word, and phrase embed layer. These layers will deal with the embeddings of different levels.

2.1.1 Character Embed Layer

For each character of a word, we give it an vector according to the character embedding lookup table. The lookup table is randomly initialized using Xavier initializer and will be trained throughout the

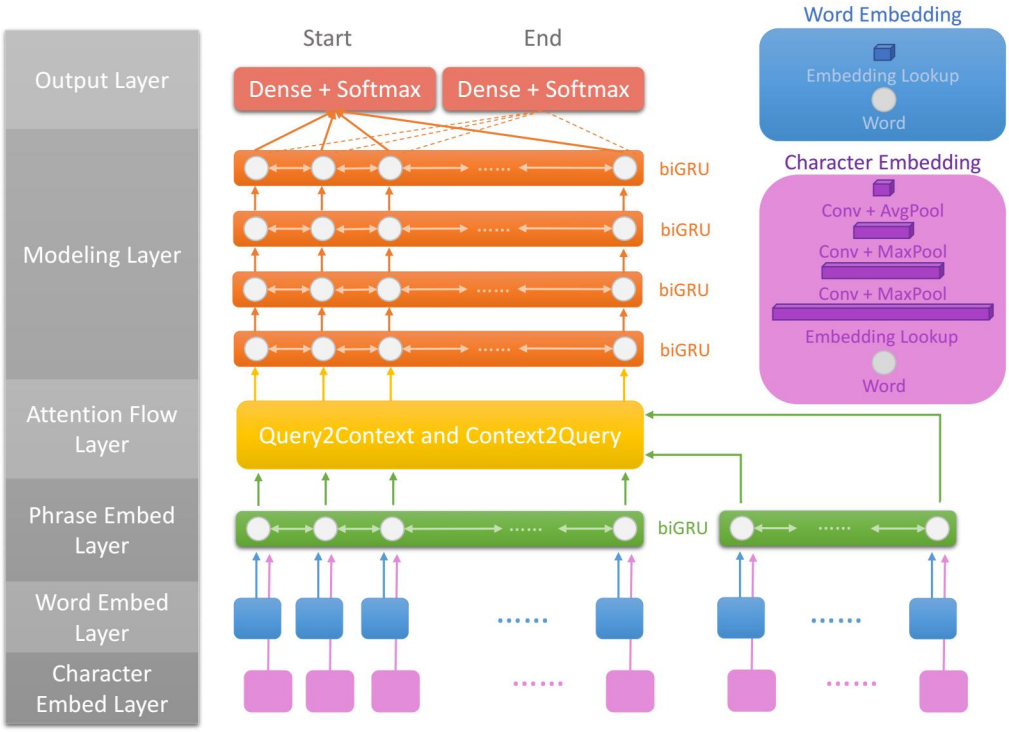


Figure 1: The flowchart of the proposed structure.

training process. The vectors will then go through several convolutional layers with each appended by a pooling layer. The output of the last pooling layer will be the character-level embeddings of this word. Our character dictionary is created from the existed characters in GloVe.

2.1.2 Word Embed Layer

We use GloVe for word embeddings. After both embeddings are extracted, we concatenate them to be the input of Phrase Embed Layer.

2.1.3 Phrase Embed Layer

This layer contains one bidirectional RNN layer. For both Context and Question, they share the same parameters in this layer. The output of this layer will be

$$\mathbf{c}_i = [\overrightarrow{\mathbf{c}_i}, \overleftarrow{\mathbf{c}_i}] \in \mathbb{R}^{2h}, \forall i \in \{1, ..., N\}, \text{(for context)}$$
$$\mathbf{q}_j = [\overrightarrow{\mathbf{q}_j}, \overleftarrow{\mathbf{q}_j}] \in \mathbb{R}^{2h}, \forall i \in \{1, ..., N\}, \text{(for question)}$$

where N is the maximal length of context, M is the maximal length of question, and h denotes the number of hidden units in Phrase Embed Layer.

2.2 Attention Flow Layer

Given the bidirectional output $\mathbf{c}$ of context and $\mathbf{q}$ of question, we first compute the similarity matrix $S \in \mathbb{R}^{N \times M}$, and

$$\mathbf{S}_{ij} = \mathbf{w}_{sim}^T [\mathbf{c}_i; \mathbf{q}_j; \mathbf{c}_i \circ \mathbf{q}_j] \in \mathbb{R},$$

where $\mathbf{w}_{sim}^T \in \mathbb{R}^{6h}$.

Then, we get the Question-to-Context (Q2C) attention by

$$\alpha^i = softmax(S_{i,:}) \in \mathbb{R}^M, \forall i \in \{1, ..., N\}$$
$$\mathbf{a}_i = \sum_{j=1}^{M} \alpha^i \mathbf{q}_j \in \mathbb{R}^{2h}, \forall i \in \{1, ..., N\}$$

Next, the Context-to-Question (C2Q) attention is

$$\beta_i = softmax(\max_j \mathbf{S}_i j) \in \mathbb{R}^N, \forall i \in \{1, ..., N\}$$
$$\mathbf{c}' = \sum_{i=1}^{N} \beta_i \mathbf{c}_i \in \mathbb{R}^{2h}$$

Finally, the output of attention flow layer is

$$\mathbf{b}_i = [\mathbf{c}_i; \mathbf{a}_i; \mathbf{c}_i \circ \mathbf{a}_i; \mathbf{c}_i \circ \mathbf{c}'] \in \mathbb{R}^{8h}, \forall i \in \{1, ..., N\}$$

2.3 Modeling Layer

This part is constructed with several bidirectional RNN layers. The input vector, 2nd layer output, and the last layer output are eventually concatenated to become the final output of it.

2.4 Output Layer

The output layer has two independent fully-connected layers that predict the start position and end position respectively.

3 Experiments

3.1 Dataset

The dataset we use for this project is the Stanford Question Answering Dataset (SQuAD)[2]. SQuAD contains more than 100k context-question-answer tuples, which are collected by crowd-workers using Wikipedia articles as the source of the context paragraphs. Among all the 100k plus tuples, 86,326 of them are distributed into the training set, 10,391 of them are in the development set, and others are in the hidden holdout test set.

3.2 Hyper-parameters

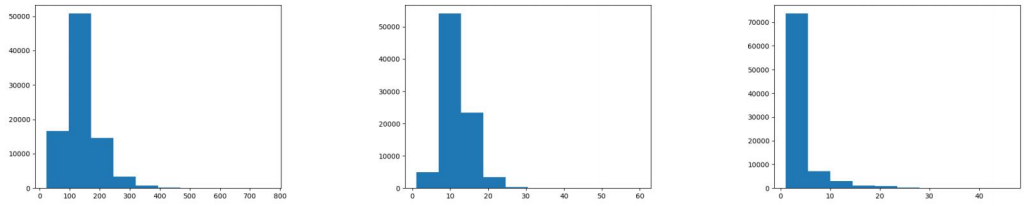
First we create the histogram on the length of context, question, and answer in the training set, as shown in Figure2. Based on the histogram, we preprocess the context data to have at most 300 words, the question data to have at most 30 words, which reduce the training time significantly. We also limit our system to only predict the answer with length less than or equal to 15, which improves the prediction accuracy, described in the later section in more details.

Before defining the hyper-parameters of character-level embed layer, we have to look into the distribution of word length(number of character in a word), shown in Figure 3. Since most of the words are shorter than 20 characters, the maximal length of character level input is set to 20. We then go through 3 CNN layers all with 200 filters, filter size 3, and stride size 1. All these layers are appended with a max-pooling layer with stride size and pooling size equal to 2.

We've tested LSTM and GRU on our BiDAF model, and they turned out to have similar performance. For cheaper computation cost, all the RNN layers in our BiDAF model use GRU cell. Also, all RNN layers have 100 hidden units.

3.3 Settings

In SQuAD[2], each context paragraph contains the exact associate answer, so our question answering systems are designed to predict the distribution of the starting and ending positions in the context



(a) Histogram of context length. (b) Histogram of question length. (c) Histogram of answer length.

Figure 2: Histogram of the training set.

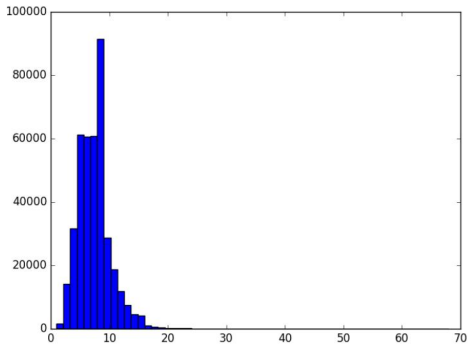


Figure 3: Distribution of Word Length in GloVe

as the system outputs, and choose the final answer span from the predicted distributions. Therefore, based on the predicted distribution, we use the cross-entropy loss as the training optimization target.

We train our models mainly using the Adam optimizer with initial learning rate 0.001. We keep training until we see the convergence of dev set F1 and EM scores in the tensorboard, which usually happens before 30k iterations. We use batch size 64 or 40 depending on different architectures of our models, due to the GPU memory limitation of the Azure NV6 machines.

We built our models mainly based on the architecture of BiDAF[1]. More specifically, we built multiple models by incrementally including each of the BiDAF components, in order to have a deeper understanding of how each component contributes to the final prediction performance. We also experimented on different types of optimizers and different RNN cells.

3.4 Evaluation

To evaluate the performance of question answering systems, F1 scores and the exact match(EM) scores are used to compare the final answer predictions with the ground truth answers.

3.5 Results

Table1 shows the summary of our models' performance evaluation using the F1 and the EM scores. We also include the information about how much improvement each component contributes to the prediction scores in Table2. In the following subsections, we will analyze the impacts of the following components: bidirectional attention layer, RNN cells, the modeling layer, the output layer, the embedding layer, and the optimizers.

3.5.1 Bidirectional Attention Flow Layer

The main innovative idea of BiDAF[1] is introducing the bidirectional attention flow, so we started to include this component on top of the baseline code, which only has the basic single directional

Table 1: Summary of F1 and EM scores of our models.

Models	F1 (dev)	EM (dev)
Baseline	43.751	34.920
Bi-Attn-Flow only	43.704	37.037
Bi-Attn-Flow + Modeling Layers(2)	72.514	62.337
Bi-Attn-Flow + Modeling Layers(3)	73.117	62.696
Bi-Attn-Flow + Modeling Layers(4)	73.392	62.895
Bi-Attn-Flow + Modeling Layers(4) + Smarter Span	74.192	63.652
Bi-Attn-Flow + Modeling Layers(4) + Smarter Span + Char CNN	74.832	64.342

Table 2: Summary of F1 and EM scores improvement from adding each layer.

Layer	Improvement of F1 (dev)	Improvement of EM (dev)
Bi-Attn-Flow Layer	-0.047	2.117
Modeling Layers(4)	29.688	25.858
Smarter Span Output Layer	0.800	0.793
Character-level CNN Embedding Layer	0.640	0.690

attention flow. The first two rows of Table1 show the F1 and the EM scores of our models with the bidirectional attention flow layer, compared with the baseline model. Surprisingly, the final performance does not improve much. And we think it is probably due to the fact that baseline code does not have any modeling layer as described in BiDAF[1], so the model does not have enough expressive power and suffer from under-fitting. And then we started to include the modeling layer into our model.

3.5.2 Modeling Layer

In this set of experiment, we added different numbers of layers of bidirectional RNN as the modeling layer, and we can see that simply adding 2 layers significantly improves the performance, as shown in Table1 and Table2. And we further experimented on 3 layers and 4 layers of of bidirectional RNN. We can see that the performance keep increasing, but the margins seem to be saturated. Besides, the training time increases a lot, so we decided to stop with 4 layers bidirectional RNN as the modeling layer and moved on to create other components. In summary, adding the modeling layer significantly improves the F1 scores and the EM scores by roughly 30%. But 4 layer RNN as the modeling layer only outperforms 2 layer RNN by less than 1%.

3.5.3 Output Layer

In the final prediction stage, we also implemented the smarter span to confine the answer length up to 15, according to the histogram statistics of the SQuAD[2] training set as shown in Figure2. This enhancement improves the overall performance by roughly 0.8%, as shown in Table1 and Table2.

3.5.4 Character Embed Layer

Finally, we implemented the Character-level CNN layer to create character-level embeddings for each word. And we concatenate them with the GloVe word embeddings as the new input embeddings. The results are shown in Table1. We can see that Character-level CNN layer improves the performance by roughly 0.6%.

3.5.5 Other Experiments

In above subsections we only show the experiments that have improvements from each of the components. Additionally, we also experimented on different training optimizers(Adam and Adadelta), and different RNN cells(GRU and LSTM). We also implemented the Self Attention Layer. However, we did not find improvements in the F1 and EM scores in those experiments.

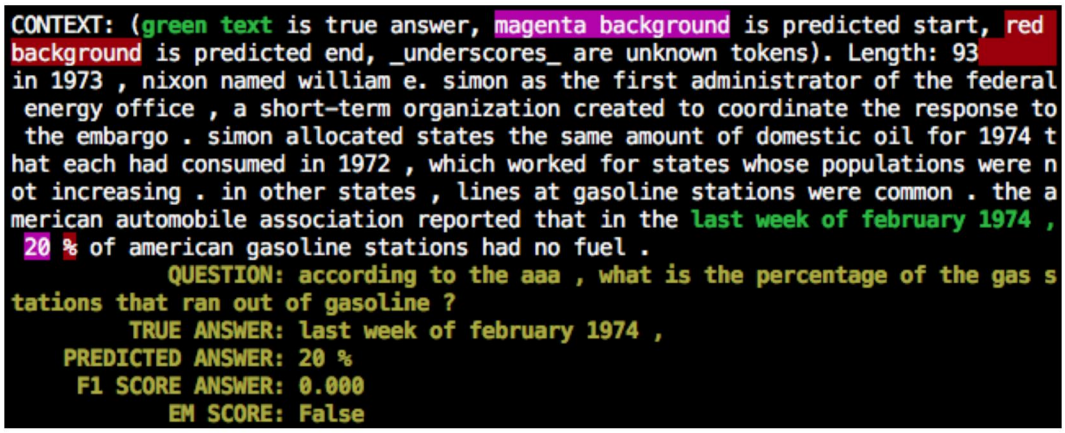


Figure 4: Incorrect labeling example.

3.5.6 Summary

We implemented the BiDAF[1] model, and our final model's best result in the dev leaderboard achieves F1 score 74.832, and EM score 64.342, as of the time when we submit this write-up, although we still have several models being evaluated in the dev leaderboard.

3.6 Error Analysis

We inspect the cases that our final model predicts incorrect answers. Different type of errors are discussed in the following subsections with the example.

3.6.1 Incorrect Labeling

Some errors are due to incorrect labeling of the answers. In such case, although our system is able to provide the correct answer, the evaluation system still give 0 F1 and EM scores. Here is one of the examples:

The question asks on "what is the percentage ...". However, the ground truth labeling seems to answer to a "when" question. There is no way to recover this type of errors other than asking the SQuAD owner to fix. And such data with wrong labeling may have slightly negative impacts for the models to learn.

3.6.2 Inaccurate Attention

Some of the errors are due to inaccurate attention. Figure 5 shows the example question, and Figure 6 contains the distribution of our system's question-to-context attention, start prediction, and end prediction. We can see that our system put more attention to the word "signed" in the context, because the question also contains the key word "signed." However, another word "maastrich" is much more important in this question. Although our system also pays attention to the word "maastrich" in the context, but with a lower weighting. And that causes our model to choose "1992" as the answer, which is closer to the word "signed." We think increasing the attention weights on named-entities may solve this problem.

3.6.3 Unable to Understand Negator Words

Figure 7 shows an example in which the error is caused by the fact that our system is unable to handle negate words correctly. In this context, we can see that "cbs" is the closest network entity word to any instance of the key word "spanish." However, there exists the word "not" between them. Additionally, another negator word "unlike" separates the phrase "nbc and fox " from the word "cbs." Our system seems to ignore both and select the incorrect answer including "cbs".

CONTEXT: (green text is true answer, magenta background is predicted start, red background is predicted end, _underscores_ are unknown tokens). Length: 243
the principal treaties that form the european union began with common rules for coal and steel , and then atomic energy , but more complete and formal institutions were established through the treaty of rome 1957 and the maastricht treaty 1992 (now : tfeu) . minor amendments were made during the 1960s and 1970s . major amending treaties were signed to complete the development of a single , internal market in the single european act 1986 , to further the development of a more social europe in the treaty of amsterdam 1997 , and to make minor amendments to the relative power of member states in the eu institutions in the treaty of nice 2001 and the treaty of lisbon 2007 . since its establishment , more member states have joined through a series of accession treaties , from the uk , ireland , denmark and norway in 1972 (though norway did not end up joining) , greece in 1979 , spain and portugal 1985 , austria , finland , norway and sweden in 1994 (though again norway failed to join , because of lack of support in the referendum) , the czech republic , cyprus , estonia , hungary , latvia , lithuania , malta , poland , slovakia and slovenia in 2004 , romania and bulgaria in 2007 and croatia in 2013 . greenland signed a treaty in 1985 giving it a special status .

QUESTION: when year was the maastrich treaty signed ?
TRUE ANSWER: 1992
PREDICTED ANSWER: 1985
F1 SCORE ANSWER: 0.000
EM SCORE: False

Figure 5: Inaccurate attention example.

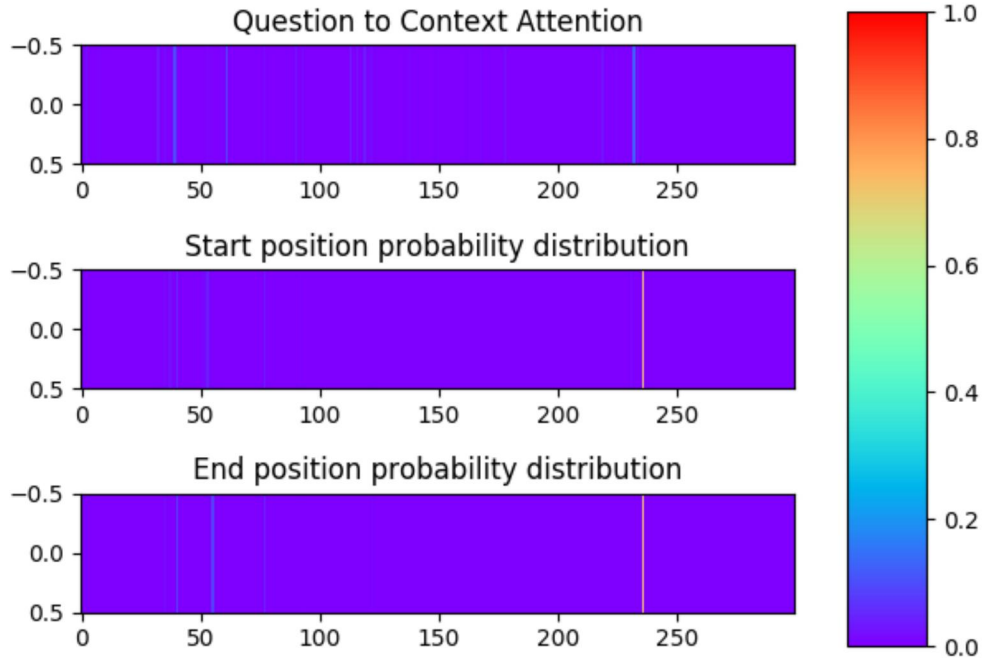


Figure 6: Visualization of inaccurate attention example.

CONTEXT: (green text is true answer, magenta background is predicted start, red background is predicted end, _underscores_ are unknown tokens). Length: 139
on december 28 , 2015 , espn deportes announced that they had reached an agreeme
nt with cbs and the nfl to be the exclusive spanish-language broadcaster of the
game , marking the third dedicated spanish-language broadcast of the super bowl
. unlike nbc and fox , cbs does not have a spanish-language outlet of its own th
at could broadcast the game (though per league policy , a separate spanish play
-by-play call was carried on cbs 's second audio program channel for over-the-ai
r viewers) . the game was called by espn deportes ' monday night football comme
ntary crew of alvaro martin and raul allegre , and sideline reporter john sutcli
ffe . espn deportes broadcast pre-game and post-game coverage , while martin , a
llegre , and sutcliffe contributed english-language reports for espn 's sportsce
nter and mike & mike .
QUESTION: which network broadcast the game in spanish ?
TRUE ANSWER: espn deportes
PREDICTED ANSWER: nbc and fox , cbs
F1 SCORE ANSWER: 0.000
EM SCORE: False

Figure 7: Negator word example.

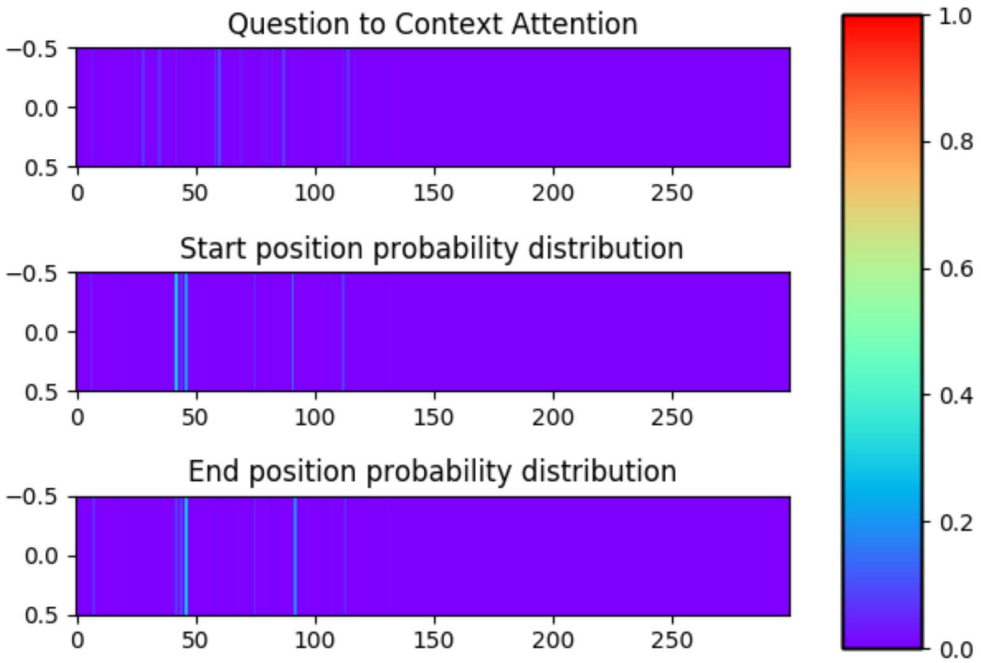


Figure 8: Visualization of negator word example.

CONTEXT: (green text is true answer, magenta background is predicted start, red background is predicted end, _underscores_ are unknown tokens). Length: 106
st. george 's united methodist church , located at the corner of 4th and new str
eets , in the old city neighborhood of philadelphia , is the oldest methodist ch
urch in continuous use in the united states , beginning in 1769 . the congregati
on was founded in 1767 , meeting initially in a sail loft on dock street , and i
n 1769 it purchased the shell of a building which had been erected in 1763 by a
german reformed congregation . at this time , methodists had not yet broken away
 from the anglican church and the methodist episcopal church was not founded unt
il 1784 .
QUESTION: when was the methodist episcopal church founded ?
TRUE ANSWER: 1784
PREDICTED ANSWER: 1767
F1 SCORE ANSWER: 0.000
EM SCORE: False

Figure 9: Another negator word example.

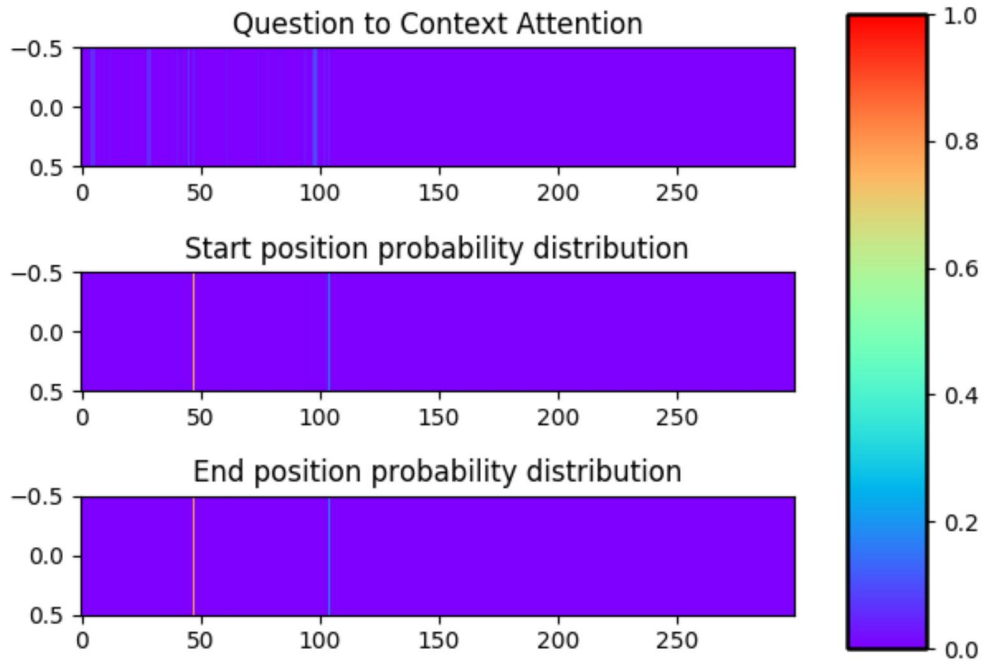


Figure 10: Visualization of another negator word example.

Another example is shown in Figure 9. Our system is unable to understand that ”was not founded until” is equivalent to ”was founded in.” So our system picks the incorrect answer ”1767”, which is closer to the phrase ”was founded in”, although our model actually pay more attention on the correct entity ”the methodist episcopal church”, as shown in Figure 10

For the negator words issue, we think including Tree Recursive Neural Networks may help the system understand the associate effects and solve this problem.

4 Conclusion

We implemented the BiDAF question answering system, and evaluated the system performance on the SQuAD dataset. We also analyzed each component's contribution to the final prediction accuracy. One interesting finding is that the bidirectional attention flow layer itself is unable to achieve good performance without the cooperation with the modeling layer. Add the modeling layer boosts the performance by roughly 30%. Besides, we also implemented other components, such as Character-level CNN, the smarter answer span, based on the data histogram statistic. And those layers also helps improving final prediction F1 and EM scores.

Additionally, we presented the visualization of question-to-context attention and the probability distribution of answer start/end predictions. The visualization shows that our system does a good job on paying attention to the important key words in the questions and the contexts in general. The visualization is also helpful to our error analysis. From the error analysis, we discover some mistakes of the SQuAD[2] dataset labeling, and some weakness of our implementation of the BiDAF model: unable to understand negator words, slightly inaccurate attention. We also proposed several ideas for future works to solve above issues for the potential future work: using Tree Recursive Neural Networks to learn the negator words, and increasing the attention weights on name-entities to better focus on the most important key words in the context.

References

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

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

[3] Wang, Wenhui and Yang, Nan and Wei, Furu and Chang, Baobao and Zhou, Ming. *Gated self-matching networks for reading comprehension and question answering* Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, 2017.

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