Question Answering with Bi-directional Attention and Character Embedding

Xiangcao Liu
shawn610@stanford.edu

Yuting Sun
ytsun@stanford.edu

Abstract

In this project, we extended the baseline model of reading comprehension system for SQuAD[1] following the BiDAF paper [2] and applied smarter span [3] during prediction time. Besides architecture improvements, we have trained models with different strategies to combine hidden states, switch RNN types, change encoder layers, and tune hyper-parameters strategically. In the end, our single model is able to achieve 74.695% F1 score and 64.37% EM score.

1 Introduction

Reading comprehension(RC) and question answering(QA) have achieved remarkable results during the past few years, and since the release of Stanford Question Answering Dataset (SQuAD) [1], the community has made great progress and the best models could achieve high F1 and EM scores which are comparable to human performance. In our project, we did literal study on a couple of papers, and did an implementation of Bi-Directional attention flow model (BiDAF) [2] and select the predicted span smarter following the DrQA paper [3].

2 Model

As shown in figure 1, our reading comprehension model consists of multiple layers. For input layer, we implemented character-level Convolutional Neural Network CNN [4][2] and then applied LSTM [5] encoding over the hybrid input representation which consists of both character-level encoding and word-level encoding. On attention layer, we implemented Bi-Directional Attention Flow [2]. We added modeling layer with 2 layers of bidirectional LSTM. For post processing, we added smarter span [3] selection at test time.

2.1 Architecture

2.1.1 Input layer

Character embedding layer To better handle out-of-vocabulary words, we added character-level encoding, which start with a trainable character embeddings, apply Convolutional Neural Network on character embeddings and apply max pooling to obtain character-level embeddings.

Word embedding layer We use the pretrained GloVe vectors [6] which were trained on 6 billion words of Wikipedia and Gigaword to get word level embeddings.

RNN encoder layer The concatenated word embedding and character-level embedding are fed into a 2-layer bidirectional LSTM, and the bidirectional LSTM produces a sequence of forward hidden states and backward hidden states. We concatencate/avergae/max pooling the forward states and backward states to obtain the context hidden states and questions hidden states.

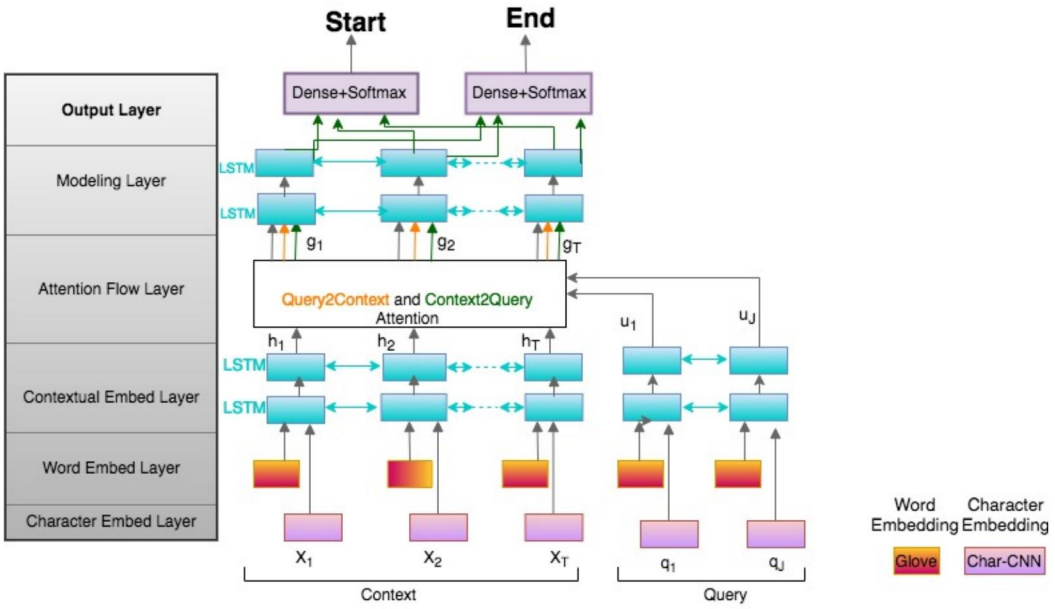

Figure 1: High-level illustration of our systems

2.1.2 Attention layer

This layer calculates attentions in both directions and produces a query-aware representation of each context word. The input to this layer are context vectors $H \in R^{2d \times T}$ and query vectors $U \in R^{2d \times T}$ from input layer.

Similarity
We chose the fusion function

$$\alpha(h, u) = w_S^T[h; u; h \circ u]$$

to create similarity matrix S=alpha(h,u), and

$$S_{tj} = \alpha(H_{:t}, U_{:j})$$

Our experiment showed the performance of this complex fusion function only improved performance trivially comparing to basic dot product function.

Context-to-query attention
Context-to-query attention is used to produce an attended query vector for each context word. For each context word t , an attended query vector $\in R^{2d}$ will be the weighted sum of all query word vectors and the weights are acquired by applying soft-max to the t-th row of similarity matrix: S[t:]. All attended query vectors composes a matrix

$$\tilde{U} \in R^{2d \times T}$$

Query-to-context attention
Query-to-context attention is used to produce an attended context vector and it will use the same attended context vector for each context word. The attended context vector $\tilde{h} \in R^{2d}$ is a weighted sum of all context word vectors and the weights are acquired by applying soft-max to the vector whose t-th item is max(S[t:]). Repeating $\tilde{h}$ for each context word generates another matrix

$$\tilde{H} \in R^{2d \times T}$$

Output
The output of this layer combines attended query vectors $\tilde{U}$, attended context vectors $\tilde{H}$, and also context word vectors H directly from input layer. We used a simple formula to generate the output of bidirectional attentional layer: $G = [H, \tilde{U}, H\tilde{U}, H\tilde{H}] \in R^{8d \times T}$. Alternatively, this could be a trainable neural network and could be further explored.

Model	F1	EM
Baseline	43.81%	34.693%
+ Bi-Directional Attention layer	57.85%	46.982%
+ 2 layers of RNN encoder layer	58.108%	46.906%
+ Character embedding + Smarter span	60.153%	48.265%
+ Modeling layer	74.695%	64.376%

Table 1: F1 score and EM score for our models

2.1.3 Modeling layer

This layer aims to encode interaction between query and context words by encoding the input G which is output from bidirectional attention layer. Similar to what we did in input layer, we encode the input using a two layer bidirectional LSTM with output size of d for each direction. Hence we get the output M $\in R^{2d \times T}$.

2.1.4 Output layer

The blended representations generated by modeling layer are fed to softmax layer after a downprojecting linear layer, to get a probablity distrubution p^{start} and p^{end}.

Smarter span To predict the span for each question, at test time, we take argmax of i, j that maximizes $p^{start}(i)p^{end}(j)$ and $i <= j <= i + 15$

2.2 Model Analysis

We have taken an iterative approach to develop our model to measure the performance of each feature we added. As shown in Table 1, adding Bi-Directional Attention layer and LSTM Modeling layer improve the model significantly, adding smarter span could improve accuracy at prediction time, while character embedding doesn't help to improve the model. The hyper-parameter to achieve our best single model is: learning rate = 0.001, dropout rate = 0.25, batch size = 75, hidden unit size = 100, word embedding dimension = 100, character embedding dimension = 20, maximum length of span selection = 20, RNN encoder layer = 2.

2.3 Model visualization

For the Bi-Directional attention layer, to better understand the performance of our model, we highlighted the Query-to-Context(Q2C) attention and Context-to-Query (C2Q) attention. As shown in Figure 2 the highlighted word in Q2C attention indicates the context words that have the closest similarity to one of the query words, and as shown in Figure 3 the highlighted word C2Q indicates the query words that are most relevant to each context word.

2.4 Beside architecture improvements

Due to limited resource, we didn't tune hyper-parameter exhaustively for each model. The following observation are based on limited hyper-parameter tuning.

Hyperparamter tuning We have tried different combination of hyperparameter, 2 is an example shows different F1 score we get with changing dropout rate only. As show in 4, larger dropout rate help to reduce over-fit problem effectively.

Sharing weights At modeling layer, sharing weights between 2 LSTM layers reduce the model size significantly, which help to speed up the training speed.

Types of RNNs At RNN encode layer, we have tried bothe GRU cell and LSTM cell, we didn't find there is obviously difference between the 2 RNNs.

Combining forward and backward hidden states We have tried concatenate, sum and average the forward hidden states and backward hidden states from the bidirectional RNN. Concatenate the state has slightly better performance.

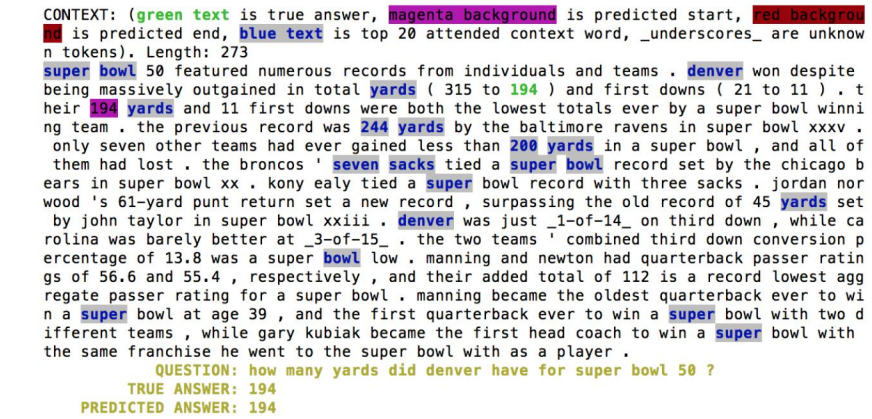

Figure 2: Query-to-Context attention indicates the context words that have the closest similarity to one of the query words

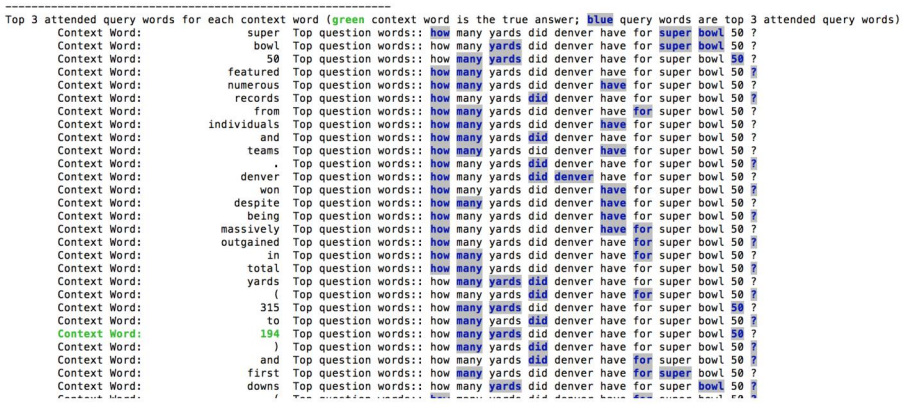

Figure 3: Context-to-Query indicates the query words that are most relevant to each context word

dropout	F1	EM
0.15	74.695%	64.376%
0.2	73.056%	62.526%
0.25	72.867%	62.194%
0.35	69.641%	58.316%

Table 2: F1 and EM core by different dropout rate

Figure 4: train and dev score with different dropout

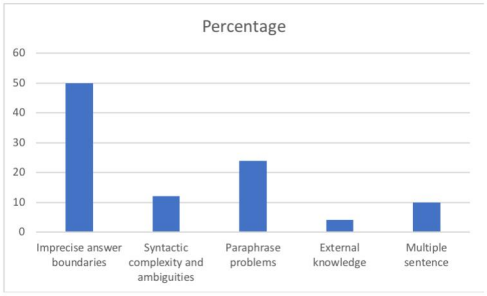

Figure 5: Percentage of error types

Size of hidden states In our best model, we use 100 hidden units in RNN encoder layer and Modeling layer. We found this hidden unit size give us relatively good result, reducing over-fit problem at some level and doesn't slow down the training speed significantly.

3 Result Analysis

We looked at more than 70 error cases. We categorize the errors into 5 types [2] and the ratio of each error type in our tests are shown in Figure 5.

3.1 Examples

3.1.1 Imprecise answer boundaries

Context: there are over 10,000 objects made from silver or gold in the collection, the display (about 15% of the collection) is divided into secular and sacred covering both christian (roman catholic , anglican and greek orthodox) and jewish liturgical vessels and items.
Question: the silver and gold collection of the v a is divided into which categories ?
True answer: secular and sacred
Predicted answer: secular and sacred covering both christian (roman catholic , anglican and greek orthodox) and jewish liturgical vessels and items

3.1.2 Syntactic complications and ambiguities

Context: following the merger , thomas s. murphy left abc with robert igertaking his place as president and ceo.

Question: who took thomas murphy 's place after the disney acquisition of abc
True answer: robert iger
Predicted answer: president and ceo

3.1.3 Paraphrase problems

Context: colonialism is the builder and preserver of the colonial possessions in an area by a population coming from a foreign region . colonialism can completely change the existing social structure , physical structure and economics of an area ; it is not unusual that the characteristics of the conquering peoples are inherited by the conquered indigenous populations .
Question:what do conquering people pass down to native populations
True answer:characteristics
Predicted answer: conquering peoples are inherited by the conquered indigenous populations

3.1.4 External knowledge

Context: super bowl 50 featured numerous records from individuals and teams . denver won despite being massively outgained in total yards (315 to 194) and first downs (21 to 11) . their 194 yards and 11 first downs were both the lowest totals ever by a super bowl winning team . the previous record was 244 yards by the baltimore ravens in super bowl xxxv . only seven other teams had ever gained less than 200 yards in a super bowl , and all of them had lost . the broncos ' seven sacks tied a super bowl record set by the chicago bears in super bowl xx .
Question: what team had the lowest downs and yards ever in the super bowl as of super bowl 50 ?
True answer: the broncos
Predicted answer: baltimore ravens

3.1.5 Multiple sentence

Context: a variety of alternatives to the y. pestis have been put forward . twigg suggested that the cause was a form of anthrax, and norman cantor (2001) thought it may have been a combination of anthrax and other pandemics . scott and duncan have argued that the pandemic was a form of infectious disease that characterise as hemorrhagic plague similar to ebola . archaeologist barney sloane has argued that there is insufficient evidence of the extinction of a large number of rats in the archaeological record of the medieval waterfront in london and that the plague spread too quickly to support the thesis that the y. pestis was spread from fleas on rats ; he argues that transmission must have been person to person . however , no single alternative solution has achieved widespread acceptance . many scholars arguing for the y.
Question: what does graham twigg propose about the spread of disease?
True answer: a form of anthrax
Predicted answer: the y. pestis was spread from fleas on rats

3.2 Analysis

Imprecise answer boundaries are the most common type of errors in our model. One explanation is our model did not take features from neighboring words when creating word embedding vector. To alleviate this problem, we can try adding concatenate the word vector from neighboring words in a window size W for each word.

Syntactic complexity and ambiguity and paraphrase problems are also very frequent. To overcome this issue, we can add more token features into the representation of words, such as the Part-of-Speech tag, the Named Entity type and the Normalized Term Frequency, etc. Lack of these information from input or model's inability to deduce these information from input could be why we see these type of error.

4 Conclusion and Discussions

During developing our model, we have taken an iterative approach and trained our models continuously each time we implemented a new improvement. This helped us seeing the effectiveness of some improvements and the limit of others. We found bidirectional attention flow is very effective in capturing the context words more relevant to the query, which explains why it improved the dev F1 score by around 10; Changing from basic dot product to more complex formula while calculating similarity matrix for attention barely helped; Adding modeling layer with LSTM also boosted the dev F1 score significantly, due to its ability to capture the memory of longer sequences. During post processing, smarter span helped improving predicted span accuracy and yet does not increase the model complexity at all.

It is interesting to see the character embedding vector provided little boost to our model. We suspect it is because we implemented it as a trainable vector as part of this model, instead of pre-trainning it. The intuition is we want to get extra information by adding the character embedding vector. However, training the character level embedding vector together with the entire model may hurt that purpose as the complexity of the model grows equally as the input. Pre-training the character embedding vectors could be worth exploring.

References

[1] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv: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] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*, 2017.

[4] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[5] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[6] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.