Bidirectional with Self Attention Question Answering on SQuAD

Leung Kin Lam
Department of Statistics and Actuarial science
The University of Hong Kong
u3527286@connect.hku.hk

Abstract

In this project, we create an end-to-end model for question-answering in SQuAD dataset. Our model combines several state-of-the-art methods such as complex attentions Bidirectional attention flow (Bidaf) and self attention with ELMo as input vectors instead of commonly used GLoVe vectors. Since this combination of these are extremely computationally expensive, our model simplifies some of the operations and achieved F1 score 75.9 and EM 62.0.

1 Introduction

Machine Comprehension (MC) has gained popularity in recent years. The Stanford Question Answering dataset (SQuAD) [1] has more than 100,000 questions posed by crowdworkers on a set of Wikipedia articles. Given a context, the task is to find the answer which is constrained to all possible spans within the context. SQuAD requires different forms of logical reasoning to find the answer. The model tries to extract the relevant sentence in the context as the answer.

The official baseline model based on logistic regression achieved F1 score 51 which is significantly lower than human performance 91.221. Many innovative attempts later can make this gap between machine and human close. Bidirectional Attention Flow (Bidaf) [2] tried to enhance the question-aware context representation. Self Attention [3] tried to further match the context against itself to find the most relevant answer in the context. Using ELMo word representation [4] which is pretrained to encode subword information and showed significant improvement over GLoVe word representation [5]. Using character n-gram to train word embeddings also shows improvements over GLoVe in several NLP tasks [7]. Dynamic Co-attention [6] attempts to iteratively find the potential answer spans so it can recover from initial local maxima corresponding to incorrect answers.

Our model has four layers. The embedding layers uses LSTM to encode both ELMo vectors and fastext vectors. The Bidirectional attention layer calculates two attentions – context to question (C2Q) and question to context (Q2C). These attentions avoid early summarization which casued information loss. The Self Attention layer strengthens the output from Bidirectional attention layer by considering context-to-context attention. It helps to find the most relevant answer if there are few in the context. Lastly, the output layer is the bi-LSTM to predict the start position and the end position of the context as answer.

The rest of this paper will firstly introduce our model architecture in section 2. Section 3 describes the dataset and implementation details of the model. Section 4 gives analysis of the performance of our model, both quantitative and qualitative. At the end, we conclude and give possible improvements for future model.

Passage: Tesla later approached Morgan to ask for more funds to build a more powerful transmitter. **When asked where all the money had gone, Tesla responded by saying that he was affected by the Panic of 1901**, which he (Morgan) had caused. Morgan was shocked by the reminder of his part in the stock market crash and by Tesla?s breach of contract by asking for more funds. Tesla wrote another plea to Morgan, but it was also fruitless. Morgan still owed Tesla money on the original agreement, and Tesla had been facing foreclosure even before construction of the tower began.

Question: On what did Tesla blame for the loss of the initial money?
Answer: Panic of 1901

Table 1: An example of SQuAD

2 Model

In this section, we introduce the architecture of the model. Figure 1 gives the overview of the model.

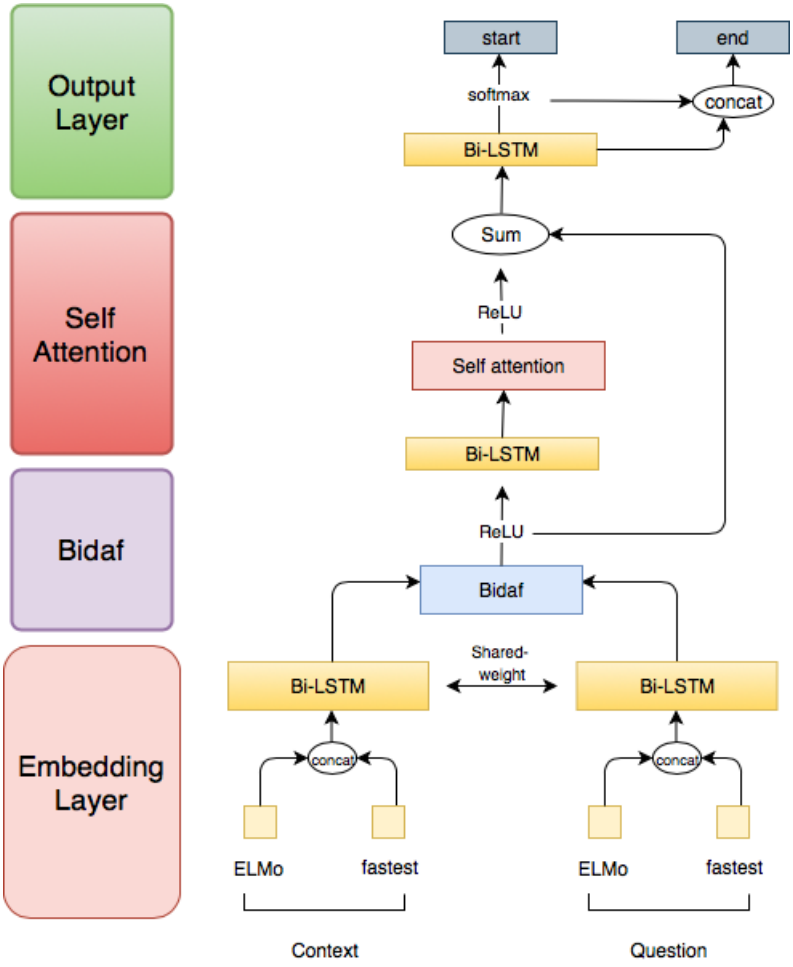

Figure 1: Overview of the complex attention architecture

2.1 Embedding Layer

For each word in both question $Q = \{w_t^Q\}_{t=1}^n$ and context $C = \{w_t^P\}_{t=1}^m$, we map it using Common Crawl 300-D fastext vectors and ELMo vectors $E = \{e_t^Q\}_{t=1}^n$ (and corresponding context $E = \{e_t^P\}_{t=1}^m$) which are the weighted sum of outputs of 2 pre-trained Bi-LSTM layers in language model. Then, for each word we get a vector as the representation. Bi-LSTM is good at giving representation of a whole sentence and here we use shared weight Bi-LSTM for the calculation. Shared weight can enrich both question and context representations. Later the similarity matrix can benefit this enrichment.

$$h_t^P = BiLSTM_P\left(h_{t-1}, [w_t^P, e_t^P]\right)$$

$$u_t^Q = BiLSTM_Q\left(u_{t-1}, [w_t^Q, e_t^Q]\right)$$

2.2 Bidirectional Attention Layer

After getting the encoded representations from previous layer, this layer firstly calculates similarity matrix where entry $\mathbf{S}_{tj}$ is the similarity between t-th context word and j-th question word. Therefore, for each row it can be interpreted as how the context word is relevant to each query word. This similarity

$$\mathbf{S}_{tj} = w_s^T \cdot \left[h_t^P; u_j^Q; h_t^P \circ u_j^Q\right]$$

is calculated by the dot product where the circle means elementwise multiplication, [;] represents concatenation.

Context to query (C2Q) attention : The goal of this attention is to find which query words are more relevant to each of the context word. For t-th context word, take softmax for that row in similarity matrix **S** to get a score for each of the query w.r.t. that context word.

$$a_t = softmax\left(S_{t:}\right)$$

Then, we can compute the attended question vector.

$$\tilde{u}_t = \textstyle\sum_j a_{tj} u_j^Q$$

Query to Context (Q2C) attention : The goal of Q2C is to compute the new representation for each context word by considering the attention calculated from question. Q2C calculation is different from C2Q where here we consider the attention weights by the maximum value in each col (question word).

$$b_t = softmax\left(\max_{col}(S)\right)$$

$$\tilde{h} = \textstyle\sum_t b_t h_t^P$$

Finally, we concatenate the above attentions and pass it through a fully connected layer with $\mathbf{ReLU}$ to get question-aware context representation $\mathbf{G} = [g_1, ..., g_m]$ where

$$g_t = \mathbf{ReLU}\left(W_g \cdot [\tilde{h}_t^P; \tilde{u}_t; h_t^P \circ \tilde{h}; h_t^P \circ \tilde{u}_t]\right)$$

2.3 Self Attention Layer

Through the bidirectional attention flow, the question-aware context representation $\{g_t\}_{t=1}^m$ can pinpoint relevant parts of the context given the question. However, there may be several parts have relation to the question. Since the answer only lies one of these parts, we need to find the most relevant part that is important to the question. Therefore, from the inspiration from Rnet [3], we match

the question-aware context representation against itself. Instead of directly follow the attention mechanism in Rnet which is computationally expensive, we use similar mechanism in bidirectional attention layer which calculates the similarity matrix. In this layer, it is defined as

$$\tilde{S}_{tj} = W_{self} \cdot [g_t; g_j; g_t \circ g_j] - \infty \cdot \mathbb{1}\{t = j\}$$

In order to avoid word always matching to itself, the above similarity ignores the attention when g_t matches to g_j for $t = j$.

$$\tilde{a}_t = softmax(\tilde{S}_{t:})$$

$$m_t = \sum_j \tilde{a}_{tj} g_j$$

where m_t is the t-th context word presentation.

Since both self attention and bidaf are powerful techniques in question-answering, to avoid serious overfitting and inspired by Resnet in computer vision, here residual connection is used to allow gradient flow directly to the previous representation g_t from bidaf layer. One interpretation is It increases flexibility of the model that the bidaf representation can directly go to output layer for prediction if self-attention does not help the prediction or pass through the self attention layer for further improvement. The output of this layer $V = [v_1, \cdots, v_m]$ is

$$v_t = g_t + \mathbf{ReLU}(W_{res} \cdot [m_t; g_t; m_t \circ g_t] + b_{res})$$

where $t \in [1, ..., m]$

2.4 Output Layer

The final layer summaries the output from self attention layer using Bi-LSTM and get $\tilde{\mathbf{V}} = [\tilde{v}_1, \cdots, \tilde{v}_m]$. To predict the start and end positions, we calculate the probability over all the context words using softmax as activation function of a fully connected layer without bias.

$$\mathbf{p_{start}} = softmax(w_{start} \cdot \tilde{V}_{start}) \in \mathbb{R}^m$$

Since the end position should depend on the start position, in order to enhance the connection, we concatenate $[p_start, \tilde{V}]$ and pass it to RNN to get $\tilde{\mathbf{V}}_{\mathbf{end}}$ then use softmax to predict the end position.

$$\mathbf{p_{end}} = softmax(w_{end} \cdot \tilde{V}_{end}) \in \mathbb{R}^m$$

2.5 Dynamic Programming test time prediction

In some cases, the prediction of our model will give the end position index larger than the start position index. Therefore, in test time, we use linear time dynamic programming to find the span $i, j, i \geq j$ such that $p_{start,i} \cdot p_{end,j}$ has largest possible value.

3 Experiments

3.1 Dataset

We use SQuAD dataset version 1.1 for training and evaluation. It consists of more than 100,000 questions on Wikipedia articles with 80% of the examples are used for training, 10% are used for validation and the last 10% are test set which is hidden to the public. 99.3% of the questions are less than or equal to 30 words and 99.9% context has lower or equal to 350 words. We set the maximum length of question and context to be 30 and 350 respectively. The evaluation metric is F1 score and Exact Match (EM).

3.2 Implementation details

We used case sensitive 300-D fastext word embeddings which is pre-trained on Common Crawl containing around 600 billions tokens and has 2 millions word vectors. Fastext embeddings are concatenated with ELMo embeddings which is 1024-D. The embedding is fixed during training. The hidden size of all layers are set to 75. Dropout rate 0.2 is used across the outputs of every layer except embedding. In Bi-LSTM, variational dropout [8] is used with 0.2 dropout rate. Variational dropout applies dropout between the cells in LSTM rather than its output and could more successfully reduce overfitting. Since the ELMo embedding consumes lot of memory, we use rather small batch size 32.

Cross entropy loss is used for start and end probabilities over all context word and also the true span of the answer. To prevent exploding gradients, we set max gradient clipping to 5. During training, Adam optimiser is used with learning rate 0.001 which gives stable convergence. With all the configurations, we trained the model for around 20 hours using single Tesla-p100 GPU.

4 Results and Analysis

4.1 Model Comparison

As shown in table 2, our single model achieved 75.9 F1 and 62.0 EM and performs competitively with state-of-the-art models.

	Dev Set	**Test Set**
Single Model	**F1 / EM**	**F1 / EM**
Logistic Regression Baseline[1]	51.0 / 40.0	51.0 / 40.4
Dynamic Chunk Reader[9]	71.2 / 62.5	71.0 / 62.5
Dynamic Coattention Networks [6]	75.6 / 65.4	75.9 / 66.2
BiDAF [2]	77.3 / 67.7	77.3 / 72.3
R-net [3]	80.6 / 72.3	80.7 / 68.0
Our Model	**75.9 / 62.0**	- / -

Table 2: Performance comparison of state of the art models

4.2 Qualitative Analysis

4.2.1 Imprecise span

As shown in the figure, our model can find the relevant word for the answer so the attention mechanism is successful. However, sometimes it outputs imprecise answer but the predicted answer, though not exactly the same as the model answer, can be viewed as correct if judged by human.

Passage: The objective is typically a course of study , lesson plan , or a practical skill . A teacher may follow standardized curricula as determined by the relevant authority . The teacher may interact with students of different ages , from infants to adults , students with different abilities and students with learning disabilities .

Question: What type of disability would a teacher help a student with ?
Answer: learning
Predicted Answer: learning disabilities
F1 Score: 0.667
EM Score: False

Table 3: Imprecise Span

5 Conclusion

In this project, we proposed and implemented an end-to-end deep learning architecture that combines state-of-the-art techniques used in question-answering task. These include Bidirectional attentional flow, Self-attention, improved word embedding ELMo. The mixed attention mechanism successfully captures relevant words as answer in the context and achieved F1 score 75.9, EM 62.0. However, from the error analysis, it cannot perform logical reasoning and sometimes span imprecisely. While the latter is fine for most cases because the differences between predicted answer and model answer share the same semantic meaning, the former one does require further improvement of the model such as iterative reasoning. Due to limited computational resources, we cannot ensemble different models having different hyper-parameters which can boost the performance and not able to compare changes in our model to evaluate the effect of part of the architecture such as how adding residual connect affect (increase/decrease) F1 score, and how much it does.

References

[1] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, Percy Liang. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *CoRR* arXiv:1606.05250, 2016.

[2] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, Hannaneh Hajishirzi. Bidirectional Attention Flow for Machine Comprehension. *CoRR* arXiv:1611.01603v6, 2016

[3] Natural Language Computing Group (2017). R-NET: Machine Reading Comprehension with Self-matching Networks. Retrieved at https://www.microsoft.com/en-us/research/wp-content/uploads/2017/05/r-net.pdf

[4] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer. Deep contextualized word representations. *CoRR* arXiv:1802.05365v2, 2018

[5] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global Vectors for Word Representation, 2014

[6] Caiming Xiong, Victor Zhong, Richard Socher. Dynamic Coattention Networks For Question Answering. *CoRR* arXiv:1611.01604v4, 2016

[7] P. Bojanowski, E. Grave, A. Joulin, T. Mikolov. Enriching Word Vectors with Subword Information. *CoRR* arXiv:1607.04606, 2016

[8] Yarin Gal, Zoubin Ghahramani. A Theoretically Grounded Application of Dropout in Recurrent Neural Networks. *CoRR* arXiv:1512.05287v5, 2016

[9] Yang Yu, Wei Zhang, Kazi Hasan, Mo Yu, Bing Xiang, Bowen Zhou/ End-to-End Answer Chunk Extraction and Ranking for Reading Comprehension. *CoRR* arXiv:1610.09996v2, 2016