SQuAD Reading Comprehension

Xinyi Jiang
Department of Computer Science
Stanford University
xinyij@stanford.edu

Abstract

One important task in Natural Language Understanding is Reading Comprehension. Given a piece of text, we want to be able to answer any relevant questions. Using Stanford Question Answering Dataset(SQuAD), which is a new reading comprehension dataset consisting of 100,000+ questions posed by crowdworkers on a set of Wikipedia articles, we built a reading comprehension model that attains 75.2% F1 score and 65.0% Exact Match (EM) on the test set.

1 Introduction

One important task in Natural Language Understanding is Reading Comprehension. Given a piece of text, we want to be able to answer any relevant questions. An ideal model for Reading Comprehension produces answers that are indistinguishable from human produced answers. In practice, we use Stanford Question Answering Dataset(SQuAD), which is a new reading comprehension dataset consisting of 100,000+ questions posed by crowdworkers on a set of Wikipedia articles. Answers to every question is a segment of text, or *span*. [3]

2 Background

Like many other machine learning tasks, progress in Reading Comprehension relies heavily on quantity and quality of datasets. Datasets previous to SQuAD are either too small [4][5] or are semi-synthetic [6][7] and do not share the same characteristics as explicit reading comprehension question.[3] SQuAD was made available in 2016. Using logistic regression, the authors of SQuAD got 40.4% Examct Match(EM) accuracy on the test set. They also provided human performance results, according to which humans got 77.0% EM score. (From the SQuAD website, however, human got 82.3% EM.)

Since SQuAD was made available, there have been many progress in the task. One important category of methods is the attention method. Coattention Technique improved the performance to 66.2% [8] and Bi-Directional Attention Flow(BiDAF) Technique improved the performance to 68.0% with single model, and 73.3% with ensemble model [2] Coattention and BiDAF both computes Context-to-Question attention and Question-to-Context attention using a similarity matrix computed directely from context and question. The difference is Coattention has second level attention computation. Self-attention is another popular way to compute attention for reading comprehension models. It is usually used together with BiDAF attention in the model.[10][9] As its name suggests, self-attention computes the attention of a context representation to itself.

A second category of methods aims at enriching the inputs. A naive way is to apply pre-trained word vectors directly as input. BiRNN or BiLSTM are often applied to generate question and context hidden states from the raw input.[R-net, MNemotic Reader] One way to improve the input is to use character level CNN which addresses the problem of unseen words.[2] Adding features relevant to

the Reading Comprehension task explicitly also helps improve the accuracy. The feature choices include parts-of-speech tags, name-entity tags, or explicit question category.[9]

A third category of methods focuses on enhancing the output. The final layer of the model often uses mixed context and question final representations as inputs and outputs distributions of predicted answer span start position and end position. Fully connected layers and answer pointers [10][9] are common practices for the final task. After the output probability distributions for start and end positions are predicted, the positions with the maximum probability are chosen as the start and the end.

3 Model Architecture

We show an overall architecture first and describe individual modules used in this model in more details in the next subsections.

The model consists of encoder layer, attention layers, and output layer. The encoding layer encodes a context-question pair into context representations and question representations. The attention layers includes two BiDAF layers and one Self Attention. Immediately after each attention layer is a Bidirectional GRU layer. The attention layers learns about the interactions between context and question. Lastly, the output layer uses mixed final representations to compute the distributions for the start position of the spand and the end position of the span.

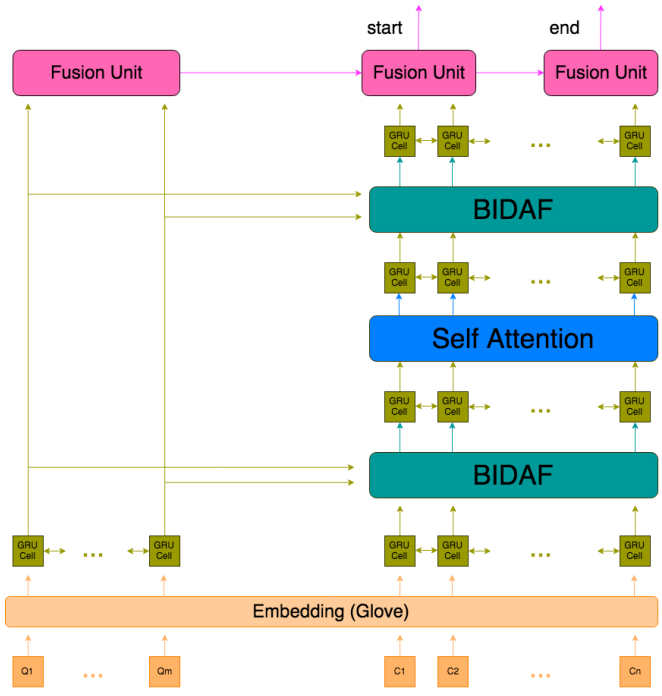


Figure 1: Model architecture

3.1 Encoder Layer

For each context-question pair, we denote context as $x = [x_1, x_2, \cdots, x_n]$ and question as $y = [y_1, y_2, \cdots, y_m]$. Each word $x_i \in \mathbb{R}^d$ or $y_j \in \mathbb{R}^d$ is represented by pre-trained Glove vector.[11] The encoder layer uses 1-layer bi-directional GRU.

$$c = [c_1, c_2, \cdots, c_n] = BiGRU(x)$$
$$q = [q_1, q_2, \cdots, q_m] = BiGRU(y)$$

,where $c_i \in \mathbb{R}^{2h}$ for all i and $q_j \in \mathbb{R}^{2h}$ for all j.[**?**]

3.2 Attention Layers

The model uses combinations of Bidirectional Attention Flow(BiDAF) and Self-Matching Attention. For both BiDAF and Self-Matching Attention, we apply one-layer BiGRU immediately after the attention output.

3.2.1 Bidirectional Attention Flow(BiDAF)

Given context hidden states c and question hidden states q, we compute $S \in \mathbb{R}^{n \times m}$ be the similarity matrix between context and question.

$$S_{ij} = w^T[c_i; q_j; c_i \circ q_j] \in \mathbb{R}$$

, where $\circ$ represents element-wise multiplication.

Context-to-question attention $a \in \mathbb{R}^{n \times 2h}$ is computed as follows:

$$a_i = (softmax_{row}(S))_{ij} q_j$$

Question-to-context attention $c' \in \mathbb{R}^{2h}$ is computed as follows:

$$m_i = \max_j S_{ij}$$

$$c' = softmax(m)_i c_i$$

BiDAF output is $b \in \mathbb{R}^{n \times 8h}$

$$b_i = [c_i; a_i; c_i \circ a_i; c_i \circ c']$$

[2]

3.2.2 Self Attention

Given input representation $b \in \mathbb{R}^{n \times d_b}$, self attention layer computes the attention of the input with itself $h \in \mathbb{R}^{4h}$.

$$b' = (b^T W b)/\sqrt{d_b}$$

$$a'_i = (softmax_{row}(b'))_{ij} b_j$$

$$h_i = [b_i; a'_i]$$

This is almost the same with the R-Net Self Attention except for the first equation. In the R-Net paper, $tanh$ attention is applied, but because it's hard to parallelize the $tanh$ equation, we chose to use this one. In additional, the factor $\frac{1}{\sqrt{d_b}}$ is applied because it effectively improves training for large dimensional inputs. [10][1]

3.3 Output Layer

Recapping that in this SQuad text understanding challenge, the output is constrained to be parts of the contexts. Therefore, we could simply point out the exact position of the start and end word, then the sentence between whom will be the output. Here we describe two alternatives for the output layer.

3.3.1 Fully Connected

The key point is to find some good enough representative for the start and end position. One simplest way is to feed hidden representations h into a fully connected layer to reduce the vector size to avoid overfitting. Here, either the same fully connected layer or different ones could be used for start and end positions, but the output of which should be the input of two different softmax classifiers. Finally, the jth output of the start-word softmax classifier p^j_{start} represents the probability of that the jth word is the start word, and the same for the jth output of end-word softmax classifier.

We use the sum of start position cross entropy lossand end position cross entropy loss as the loss of our model.

3.3.2 Fusion Output

One drawback of simple fully connected layer is that we do not utilize the information of the start position when we predict the end position, which is especially harmful when the answer length is relatively long. Thus, inspired by the work of [13], here we propose a simpler alternative to the complex pointer network, which is named as fusion output. The underneath idea is that when we predict the start position, the information of questions could be reused, and when we predict the end position, the information of start position could be used. The fusion output layer follows the equations below:

$$
\begin{aligned}
q_f &= tanh(W^{(q)}[q_1, q_2, \cdots, q_M] + b^{(q)}) \\
h_{\text{start}} &= tanh(W^{(s)}h + W^{(sq)}q_f) \\
h_{\text{end}} &= tanh(W^{(e)}h + W^{(es)}h_{\text{start}})
\end{aligned}
$$

and

$$
\begin{aligned}
p_{\text{start}} &= \text{Softmax}(h_{\text{start}}) \\
p_{\text{end}} &= \text{Softmax}(h_{\text{end}}).
\end{aligned}
$$

We use the sum of start position cross entropy lossand end position cross entropy loss as the loss of our model.

3.4 Dynamic Programming Span Prediction

Although we can simply pick the index of p_{start} and p_{end} having the largest value as the predicted position, it might violate the fact that the end word should not appear before the star word. Thus, instead of maximize p_{start} and p_{end} separately, we want to maximize the value of $p_{\text{start}}^{(i)} p_{\text{end}}^{(j)}$, where there is $i \leq j$. The optimal (i, j) pair will separtelu be the start and end position. We could apply dynamic programming to achieve a linear running time.

4 Experiments

4.1 Dataset

Glove Vectors pretrained on wikipedia texts are applied as word embeddings, and we trained, developed and tested our model on SQuAD.

4.2 Model Configurations

We used 100d Glove word vector embeddings. We did a simple stastic of the SQuAD training and dev set, and concluded that context length 400, question length 30 should cover almost all question-answer pairs. The model hidden size h is used consistently throughout different layers, and is set as 100 We applied a dropout rate of 0.2 yet still observe big overfittings. We adjust the learning rate throughout the training manually. The initial learning rate is 0.001. We applied Adam Optimizer in the model. The batch size is set to 64 based on our computational resource limit. Max gradient norm for gradient clipping is 5.

4.3 Evaluation Metrics

The goal of the model is to produce answers as close to human produced answers as possible. Following the evaluation metric on the SQuAD leaderboard, we used both F1 score and Exact Match(EM) as evaluation metrics. Both scores are computed using the three SQuAD crowdworker produced answers as ground-truths. For both scores, first the answer score with respect to each one of the three ground truth answers is computed, and then the maximum of the three is taken as the final score. F1 or EM for the model is simply the average of F1 or EM on individual context-answer pairs.

F1 score with respect to a ground truth answer span is the harmonic average of precision and recall, case-insensitive.

Exact Match score with respect to a ground truth answer span is 1 if the predicted span matches the ground truth span exactly, case-insensitive.

4.4 Results

We first categorize context-questions pairs by average ground truth answer lengths and look at the model performance on different question lengthx.

	F1 Score	EM	Number of Questions
1-2	75.6	70.6	4147 (39.2%)
2-4	76.9	66.1	4362 (41.3%)
4-8	69.2	47.8	1651 (15.6%)
8-16	59.6	31.8	398 (3.8%)
16-	47.5	30.8	12 (0.1%)
Overall	**74.2**	**63.5**	**10570**

Figure 2: A table that shows prediction accuracy for answers of different lengths

We notice the majority ($> 80\%$) of context-question pairs have very short answers (< 4 words), and the model performance on these context-question pairs is good. There are two possible explanations:

1. There are not enough context-question pairs with long answers for training.

2. Long answers usually involve more complex reasonings, and the model is not complex enough to deal with that.

Then we categorize each context-question pair by the question type: Who, Where, When, Why, Which, How, Others, and examine the prediction scores. Question type is determined by the first word in questions. Others categories may include questions that start with words such as "in which" or "for whom", but we don't further categorize these questions.

	F1 Score	EM	Number of Questions
Who	**79.1**	**72.3**	**1061 (10.0%)**
Where	70.3	57.1	433 (4.1%)
When	**85.8**	**81.8**	**696 (6.6%)**
Why	**58.5**	**28.3**	**151 (1.4%)**
What	71.9	59.4	4753 (45.0%)
Which	75.4	64.8	454 (4.3%)
How	75.5	65.2	1090 (10.3%)
Others	75.7	65.9	1932 (18.3%)
Overall	**74.2**	**63.5**	**10570**

Figure 3: A table that shows prediction accuracy for different question types

There's again a huge performance difference on differenct categories. We notice WHEN and WHO questions are answered particularly well, but WHY questions are answered strikingly bad.

We suspect thats because: The model learns to answer ground-truth questions (WHEN and WHO) from statistics, but the cant deal with complex logic (WHY).

To get a better sense with these conjectures, we examined some sample model predictions.

example 39
true start position: 22
true end position: 30
CONTEXT: (green text is true answer, magenta background is predicted start, red background is predicted end, _underscores_ are unknown tokens). Length: 95
the historian francis aidan gasquet wrote about the _'great_ pestilence ' in 1893 and suggested that " it would appear to be some form of the ordinary eastern or bubonic plague " . he was able to adopt the epidemiology of the bubonic plague for the black death for the second edition in 1908 , implicating rats and fleas in the process , and his interpretation was widely accepted for other ancient and medieval epidemics , such as the justinian plague that was prevalent in the eastern roman empire from 541 to 700 ce .
QUESTION: what did gasquet think the plague was ?
TRUE ANSWER: some form of the ordinary eastern or bubonic plague
PREDICTED ANSWER: able to adopt the epidemiology of the bubonic plague
F1 SCORE ANSWER: 0.400
EM SCORE: False

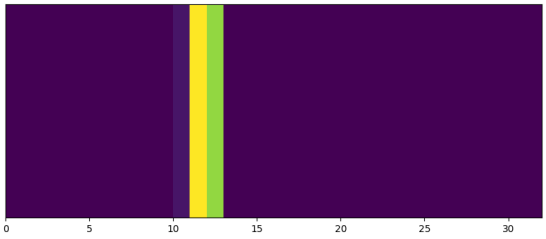
Figure 4: Example of a question with long answer

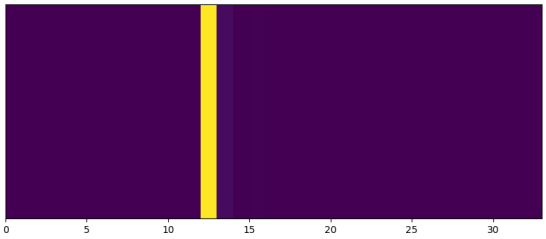
example 36
true start position: 5
true end position: 10
CONTEXT: (green text is true answer, magenta background is predicted start, red background is predicted end, _underscores_ are unknown tokens). Length: 76
in 1874 , tesla evaded being drafted into the austro-hungarian army in _smiljan_ by running away to _tomingaj_ , near _gračac_ . there , he explored the mountains in hunter 's garb . tesla said that this contact with nature made him stronger , both physically and mentally . he read many books while in _tomingaj_ , and later said that mark twain 's works had helped him to miraculously recover from his earlier illness .
QUESTION: why did tesla avoid by fleeing smiljan ?
TRUE ANSWER: being drafted into the austro-hungarian army
PREDICTED ANSWER: this contact with nature made him stronger , both physically and mentally
F1 SCORE ANSWER: 0.000
EM SCORE: False

Figure 5: Example of a why question context-question pair.

We see the predicted answers are in fact drastically wrong–they are not even in the same sentence as the ground truth answers.

Another analysis we did was to visualize the model output probabilities for start and end position. For many context-question pairs, the model has surprisingly high probability for one span. Many of the examples we've seen exhibit this pattern, among which most have correct predictions.





example 13
true start position: 11
true end position: 12
CONTEXT: (green text is true answer, magenta background is predicted start, red background is predicted end, _underscores_ are unknown tokens). Length: 98
one of the earliest surviving examples of european quilting , the late 14th-century sicilian tristan quilt , is also held by the collection . the collection has numerous examples of various types of textiles designed by william morris , including , embroidery , woven fabrics , tapestries (including _'the_ forest ' tapestry of 1887) , rugs and carpets , as well as pattern books and paper designs . the art deco period is covered by rugs and fabrics designed by marion dorn . from the same period there is a rug designed by serge chermayeff .
QUESTION: when is the sicilian tristan quilt dated ?
TRUE ANSWER: late 14th-century
PREDICTED ANSWER: late 14th-century
F1 SCORE ANSWER: 1.000
EM SCORE: True

Figure 6: Start position predicted distribution; end position predicted distribution; correspondent context-question pair

For those context-question pairs that our model is uncertain about the answers, we found the output result is in fact incorrect for the majority of times.

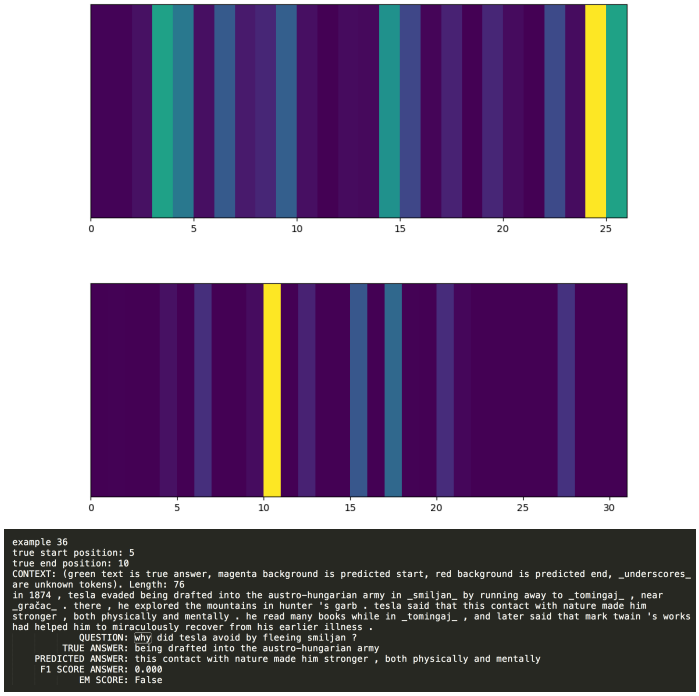


Figure 7: Start position predicted distribution; end position predicted distribution; correspondent context-question pair

Finally, the model training progress is visualized via tensorboard.

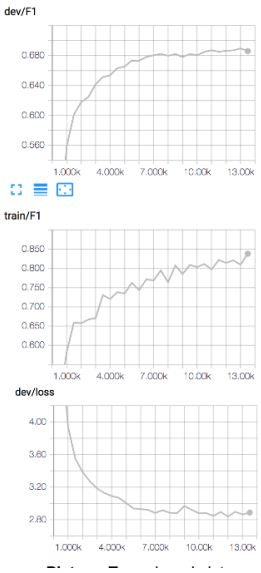


Figure 8: Tensorboard plots

The dev loss drops quickly to around 4 and the dev F1 rises quickly to 55.8Around the 5th epoch, 5k iteration, the dev loss drops to around 3, and the dev F1 rises consistently to 66.5After that the

model converges on the dev set, but still increases accuracy on the training set in a linear fashion. The model starts to overfit.

5 Conclusions

In conclusion, our model performs well on short answer length tasks, and on ground-truth questions such as WHEN and WHO questions, but performs badly on long answer length tasks, and on more logic intensive questions. Asides, it still suffers from over-fitting,

Based on the results, we think future works should be devoted to 1) use longer and better context and question embeddings and tune the model to reduce overfit. 2) Improve performance on complex questions aka questions that cant be answered in a few words and questions that have vague answer spans such as WHY questions. This potentially requires architecting a more complex model.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, Attention is all you need, arXiv preprint arXiv:1706.03762, 2017.

[2] Seo M, Kembhavi A, Farhadi A, et al. Bidirectional attention flow for machine comprehension[J]. arXiv preprint arXiv:1611.01603, 2016.

[3] Rajpurkar, Pranav, et al. Squad: 100,000+ questions for machine comprehension of text. arXiv preprint arXiv:1606.05250, 2016.

[4] Richardson, Matthew, Christopher JC Burges, and Erin Renshaw. "Mctest: A challenge dataset for the open-domain machine comprehension of text." Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. 2013.

[5] Berant, Jonathan, et al. "Modeling biological processes for reading comprehension." Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014.

[6] Hermann, Karl Moritz, et al. "Teaching machines to read and comprehend." Advances in Neural Information Processing Systems. 2015.

[7] Hill, Felix, et al. "The goldilocks principle: Reading children's books with explicit memory representations." arXiv preprint arXiv:1511.02301 (2015).

[8] Xiong, Caiming, Victor Zhong, and Richard Socher. "Dynamic coattention networks for question answering." arXiv preprint arXiv:1611.01604 (2016).

[9] Hu, Minghao, Yuxing Peng, and Xipeng Qiu. "Mnemonic reader for machine comprehension." arXiv preprint arXiv:1705.02798 (2017).

[10] Natural Language Computing Group, Microsoft Research Asia. "R-Net: Machine Reading Comprehension with Self-Matching Networks" Microsoft research (2017).

[11] Pennington, Jeffrey, Richard Socher, and Christopher Manning. "Glove: Global vectors for word representation." Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014.

[12] Stanford 2017-2018 Winter CS224N Teaching Team. "CS 224N Default Final Project: Question Answering." (2018)

[13] Shuohang Wang and Jing Jiang. Machine comprehension using match-lstm and answer pointer. arXiv preprint arXiv:1608.07905 , 2016.