Implementation of R-NET Machine Comprehension Model for Question Answering

Sabarish Sankaranarayanan
Google Inc.
`sabarishs@google.com`

Abstract

We implemented the R-NET[1] machine comprehension model to solve the problem of question answering on the Stanford Question Answering Dataset (SQuAD). The best single model we were able to build achieved an F1 score of 72.6 and EM score of 62.2 on the test set.

1 Introduction

The problem of using learning algorithms to solve question answering on reading comprehension datasets has been gaining popularity in recent years. One particularly useful dataset for this challenge is the Stanford Question Answering Dataset (SQuAD) by Rajpurkar et. al (2016), which contains 100K context, query and answer triplets. The answers are represented as spans in the given context. As detailed in the paper, the dataset covers a wide variety of questions that requires different forms of logical reasoning to get a good performance.

Recently, several end-to-end algorithms have achieved performance competitive with humans on the SQuAD dataset. The best performing models typically employ some form of attention mechanism, introduced by Bahdanau et. al (2014), to process both the question and the context and build a representation from which the answer can be extracted. R-NET is an architecture by Wang et. al (2017), that achieves high single model performance and we implemented this model for the CS224N default course project.

2 Model Description

At a high-level, the important blocks in an R-NET model include an attention mechanism from the context to the query tokens to obtain a query-aware context representation. This is then refined through a self-attention mechanism wherein the context representation is improved by using attention mechanism on the context representation itself.

Shown in Figure 1 is a diagram of the model architecture. Below, we discuss the details of the different components.

2.1 Question and Passage Encoding

For word representation, we use pre-trained GloVe embeddings introduced by Pennington et. al (2014). In order to handle out-of-vocabulary words, trained character-level embeddings for the characters in each token are fed through an RNN layer. The final output of the character RNN layer

[1]Described in the work-in-progress technical report at https://www.microsoft.com/en-us/research/publication/mrc/

[2]Taken from technical report at https://www.microsoft.com/en-us/research/publication/mrc/

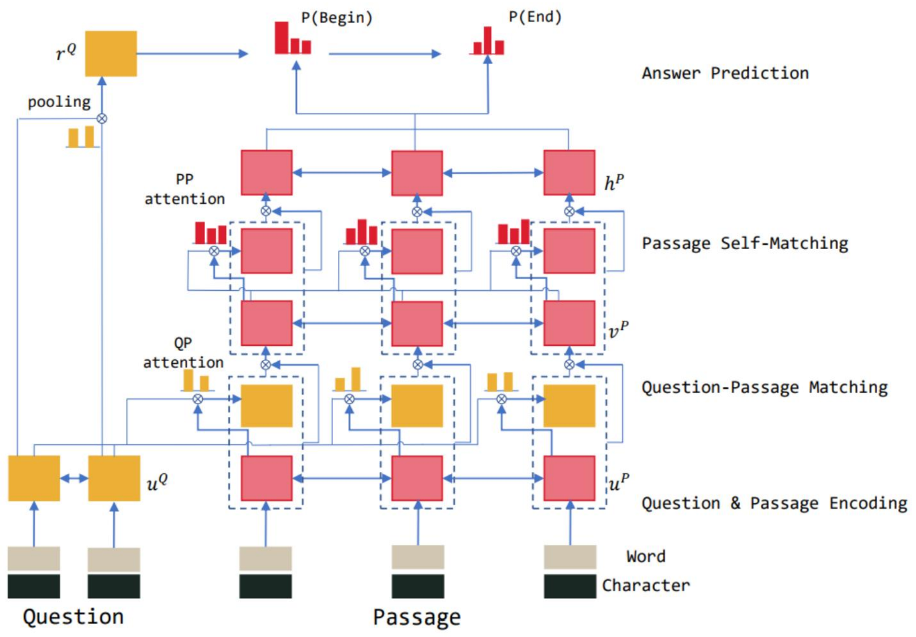

Figure 1: R-NET Model Architecture[2]

is concatenated with the word embedding and two layers of bidirectional RNNs are used to create the context and question representations.

Let e_t^Q and c_t^Q be the word and character embedding for the token at position t in the question, and e_t^P and c_t^P be the embeddings for the context token at position t.

$$u_t^Q = BiRNN_Q(u_{t-1}^Q, [e_t^Q, c_t^Q])$$
$$u_t^P = BiRNN_P(u_{t-1}^P, [e_t^P, c_t^P])$$

2.2 Query-Aware Context Representation

The model utilizes information from the query for a more suitable encoding of the context, v_t^P. This is done using an RNN layer which, in addition to u_t^P, takes as input an attention-blended query representation $c_t = \beta(u^Q, [u_t^P, v_{t-1}^P])$. The additive attention function β given a key k and a matrix of values V is defined as shown below

$$a_i = \theta^T tanh(W_v v_i + W_k k)$$
$$s_i = \frac{\exp(a_i)}{\sum_j \exp(a_j)}$$
$$\beta(V, k) = \sum_i s_i v_i$$

θ, W_v, W_k are trainable parameters. Finally, the query-aware context representation is derived as follows.

$$v_t^P = RNN(v_{t-1}^Q, g([u_t^P, c_t]))$$

g is a gating function that helps to filter out inputs that aren't relevant to answer the question and is defined as follows

$$g(x) = \sigma(W_g x) \odot x$$

W_g is a trainable parameter. Note that this gating is different from and in addition to the gates in the recurrent cell, which typically are applied only to the state and not on the inputs. Together, this layer is a gated-attention based recurrent neural network.

2.3 Self-Matching Attention

In dealing with passage, question pairs where there are multiple candidate answers, it is important to utilize this information and refine the passage representation. One way to achieve this is to have the question-aware passage attend to different portions of itself and use this attended representation of surrounding context at each position to refine the passage representation, leading to h_t^P.

$$\begin{aligned} c_t &= \beta(v^P, v_t^P) \\ h_t^P &= BiRNN(h_{t-1}^P, g([v_t^P, c_t])) \end{aligned}$$

g is the gating function defined previously.

2.4 Output Layer

Since the answer is a span in the original passage, it can be represented by the indices of the start and end tokens, which are obtained through the use of pointer networks.

2.4.1 Original

In the R-NET paper, the start pointer, p^s, is calculated using the following procedure. First, a trainable parameter V_r^Q attends to the question representation u^Q to produce r^Q, which is then used to attend to the question-and-context-aware passage representation h^P. The resulting attention weights, a^s, are taken to be the probability of each location to be the beginning of the answer and an argmax operation gives the final prediction.

$$\begin{aligned} r^Q &= \beta(u^Q, V_r^Q) \\ \mathrm{a}^s &= \beta_w(h^P, r^Q) \\ p^s &= argmax(a_1^s, a_2^s, \ldots, a_n^s) \end{aligned}$$

Here β_w uses the same additive attention mechanism but returns the attention weights instead of the sum of attention-weighted values.

The end pointer, p^e, is calculated similarly. First, we calculate c_t, the sum of passage vecotrs in h^P weighted by the start probabilities a^s. This becomes an input to an RNN cell whose input state is set as r^Q. The output of the RNN, $\tilde{r}^Q$, is used as the key for the attention mechanism over h^P to produce the end probabilities.

The equations to calculate the end pointer are as follows.

$$\begin{aligned} c_t &= \sum_i a_i^s h_i^P \\ \tilde{r}^Q &= RNN(r^Q, c_t) \\ \mathrm{a}^e &= \beta_w(h^P, \tilde{r}^Q) \\ p^e &= argmax(a_1^e, a_2^e, \ldots, a_n^e) \end{aligned}$$

2.4.2 Modified Implementation

While our start pointer implementation is identical to what is described in the original paper, the end pointer implementation is slightly different. Instead of deriving a separate key $\tilde{r}^Q$ for the end pointer, we reuse r^Q as the key but instead use a bidirectional RNN to process h^P and create a separate passage representation for the purpose of end pointer calculation.

$$\begin{aligned} \tilde{h}_t^P &= BiRNN(\tilde{h}_{t-1}^P, h_t^P) \\ \mathrm{a}^e &= \beta_w(\tilde{h}^P, r^Q) \\ p^e &= argmax(a_1^e, a_2^e, \ldots, a_n^e) \end{aligned}$$

Table 1: Performance comparison of our implementation with some top-performing single models

Model	Test Set EM	Test Set F1
Dynamic Coattention Networks (Xiong et. al, 2016)	66.2	75.9
BiDAF (Seo et. al, 2016)	68.0	77.3
R-NET (Wang et. al, 2017)	72.3	80.7
Our implementation	**62.2**	**72.6**

This choice was a result of not being able to get the original version to work, which could just been due to a bug in our program. The modified version seemed to work well, so we did not attempt to investigate further.

3 Implementation Details

We used 300-dimensional pre-trained GloVe embeddings that were fixed during training. For the character embeddings, a 1-D CNN is used on the characters of each token followed by max pooling across the sequence of characters for each filter. This method was described in the BiDAF paper [4]. The concatenated word and characater embeddings were processed using two layers of bidirectional Gated Recurrent Unit (GRU) based networks to produce the question and context representations. Context length was limited to 400 for training, question length to 30. The character length was limited to 10, embedding dimension set to 50 and 50 filters of width 5 were used for the character CNN.

Two layers of bidirectional GRU network were used to produce the final question-and-context-aware passage representation h^P. The hidden state dimension was set to 75 for all RNN layers and the hidden attention dimension for the additive attention was also set to 75. Dropout (Srivastava et al., 2014) regularization was applied between all layers with a dropout rate of 0.2. Training was done using the AdaDelta (Zeiler, 2012) optimizer with a learning rate of 1.

4 Model Performance

Shown in Table 1 is the performance for our implementation along with that of a few other high-performing models on the SQuAD dataset. As can be seen, there is a big performance difference between the original implementation and our replication. In a later section, we offer some hypothesis in this regard.

In Figure 2, we look at how the performance changes with respect to some properties of the data. We can see that there is noticeable degradation in the performance for context lengths beyond 400 and for longer answers. There is almost no dependence on the question length. Performance across different types of questions are also shown. Notably, EM scores for ‘why’ questions are much lower than for other types. This must be due to lower proportion of this type of question in the dataset as well as the fact that such questions have longer answers with ambiguous boundaries.

4.1 Error Types

In this section, we provide some error modes of the model along with examples below

4.1.1 Syntactic complications and ambiguities

Context: from here, the situation becomes more complicated, as the dutch name rijn no longer coincides with the main flow of water. two thirds of the water flow volume of the rhine flows farther west, through the waal and then, via the merwede and nieuwe merwede (de biesbosch), merging with the meuse, through the hollands diep and haringvliet estuaries, into the north sea.

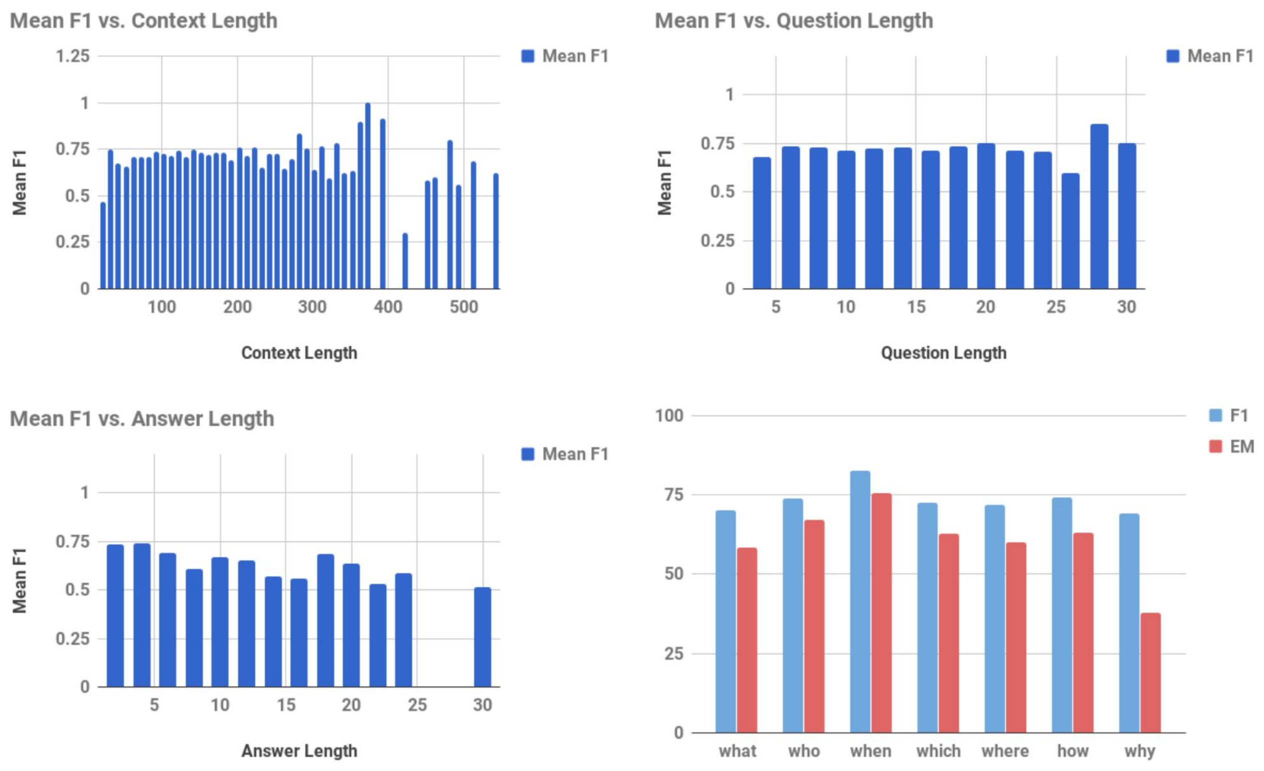

Figure 2: Performance comparison with respect to context length, question length and answer length and question type.

Question: where does two thirds of the rhine flow outside of germany?
True Answer: waal
Predicted Answer: farther west

4.1.2 Imprecise answer boundaries

Context: from italy, the disease spread northwest across europe, striking france, spain, portugal and england by june 1348, then turned and spread east through germany and scandinavia from 1348 to 1350. it was introduced in norway in 1349 when a ship landed at asky, then spread to bjrgvin (modern bergen) and iceland. finally it spread to northwestern russia in 1351. the plague was somewhat less common in parts of europe that had smaller trade relations with their neighbours, including the kingdom of poland, the majority of the basque country, isolated parts of belgium and the netherlands, and isolated alpine villages throughout the continent.
Question: where was the disease spreading between 1348 and 1350?
True Answer: germany and scandinavia
Predicted Answer: east through germany and scandinavia

4.1.3 External Knowledge

Context: cbs provided digital streams of the game via cbssports.com, and the cbs sports apps on tablets, windows 10 , xbox one and other digital media players (such as chromecast and roku) . due to verizon communications exclusivity, streaming on smartphones was only provided to verizon wireless customers via the nfl mobile service. the espn deportes spanish broadcast was made available through watchespn.
Question: which game console could viewers stream the game on?
True Answer: xbox one

Predicted Answer: cbssports.com

4.1.4 Paraphrasing Errors

Context: jochi died in 1226, during his father's lifetime. some scholars, notably ratchnevsky, have commented on the possibility that jochi was secretly poisoned by an order from genghis khan. rashid al-din reports that the great khan sent for his sons in the spring of 1223, and while his brothers heeded the order, jochi remained in khorasan. juzjani suggests that the disagreement arose from a quarrel between jochi and his brothers in the siege of urgench. jochi had attempted to protect urgench from destruction, as it belonged to territory allocated to him as a fief. he concludes his story with the clearly apocryphal statement by jochi: "genghis khan is mad to have massacred so many people and laid waste so many lands. i would be doing a service if i killed my father when he is hunting, made an alliance with sultan muhammad, brought this land to life and gave assistance and support to the muslims. "juzjani claims that it was in response to hearing of these plans that genghis khan ordered his son secretly poisoned; however, as sultan muhammad was already dead in 1223, the accuracy of this story is questionable.
Question: why is jochi 's reported alliance with the muslims historically suspect?
True Answer: sultan muhammad was already dead in 1223
Predicted Answer: genghis khan is mad to have massacred ... genghis khan ordered his son secretly poisoned

5 Self-Matching Attention Visualization

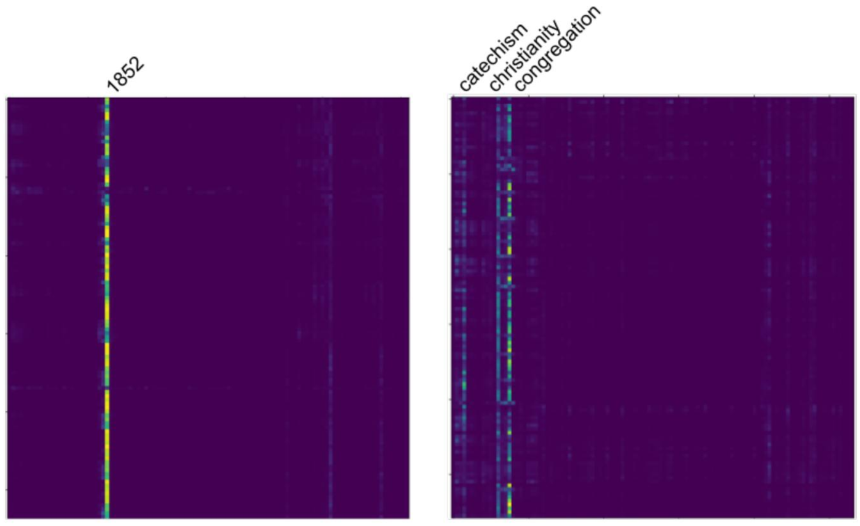

Figure 3: Self-Matching Attention Weights for two examples a and b. Each row in the matrix represents the attention weights for a particular word in the context

Figure 3a corresponds to the following example.

Context: private schooling in the united states has been debated by educators, lawmakers and parents, since the beginnings of compulsory education in massachusetts in 1852. the supreme court precedent appears to favor educational choice, so long as states may set standards for educational accomplishment. some of the most relevant supreme court case law on this is as follows: runyon v. mccrary, 427 u.s. 160 (1976); wisconsin v. yoder, 406 u.s. 205 (1972); pierce v. society of sisters, 268 u.s. 510 (1925); meyer v. nebraska, 262 u.s. 390 (1923).
Question: in what year did massachusetts first require children to be educated in schools?
True Answer: 1852
Predicted Answer: 1852

We can observe from the attention matrix that almost all passage words are paying attention to only "1852" and not taking into account information from the surrounding context.

Figure 3b corresponds to the following example.

Context: luther devised the catechism as a method of imparting the basics of christianity to the congregations. in 1529, he wrote the large catechism, a manual for pastors and teachers, as well as a synopsis, the small catechism, to be memorised by the people themselves. the catechisms provided easy-to-understand instructional and devotional material on the ten commandments, the apostles' creed, the lord's prayer, baptism, and the lord's supper. luther incorporated questions and answers in the catechism so that the basics of christian faith would not just be learned by rote, "the way monkeys do it", but understood.
Question: what did luther devise to teach christianity to the congregation?
True Answer: catechism
Predicted Answer: catchism

Similar to example a, we see that most passage words are paying attention to the answer candidate "catechism". In addition, the attention is also strongly focused on words from the question such as "christian" and "congregation". There are other similar examples that lead us to believe that instead of using the self-matching attention layer to extract information from the surrounding context, the model has trained in a way that this layer acts like a combination of answer pointer and query to context attention layer. As a contrast, Figure 4 shows an example of self-attention weights from Wang et. al (2017). The behavior is quite different, where words that are important for the answer pay attention to other relevant context words while irrelevant words in the context end up with diffuse attention weights. The difference could be attributed to bugs in our implementation, differences in architecture, initialization methods, optimization settings, etc. This could also partly explain the difference in performance that we achieve compared to what is reported by Wang et. al (2017).

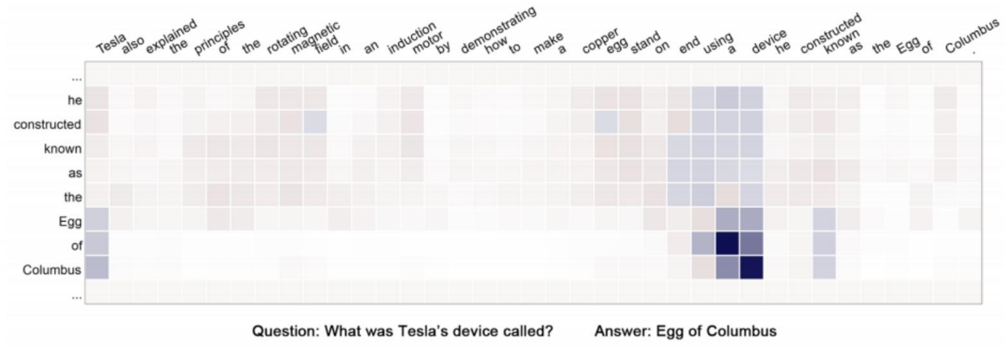

Figure 4: Attention Visualization[3]

6 Other Improvements Attempted

A few improvements we tried did not result in any meaningful performance benefits.These are listed below.

- Adding regularization on all attention weights. We used the L2 norm of each attention vector and applied a max-margin loss to keep it within limits, so that attention is neither too focused nor too diffuse. This seemed to affect neither the training dynamics nor the overall performance.
- Replacing GRU cells with LSTM cells.
- Using two attention heads for the self-matching attention layer, but with half the hidden dimension size (37) so that the model would fit in GPU memory.

[3]Image source: Wang et. al (2017)

7 Conclusion And Future Work

We have implemented the R-NET architecture for solving question answering on the SQuAD dataset. Our best-performing single model achieves a score of 72.6 F1 and 62.2 EM on the test set. We identified potential issues with the self-matching attention layer implementation, which when fixed could further boost the performance and bridge the gap with the original R-NET implementation. It would also be valuable to perform an ablation study to understand the importance of the following pieces in the architecture - gated-attention, self-matching attention, character embeddings and the answer pointer.

Acknowledgments

We would like to thank the course instructor, Dr. Richard Socher and the course TAs for help with the project, Microsoft for providing GPU access on Azure and fellow classmates for asking and answering insightful questions in the class forum.

References

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

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

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

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

[5] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 2014.

[6] Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. Gated self-matching networks for reading comprehension and question answering. *Association for Computational Linguistics (ACL)*, 2017.

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

[8] Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.