Building a QA System using R-net

Stanford CS224N Default Project

Tianyi Liu
Stanford University
`tliu26@stanford.edu`

Abstract

Question-answering (QA) task is an important problem for research in natural language processing, for which many deep learning models have been designed. In this report we implement R-Net and evaluate its performance on SQuAD 2.0. While the performance of R-Net itself is worse than BiDAF, it showed a strong capability of its attention mechanism. We have also experimented with an ensemble model using BiDAF and R-Net that achieved better performance than the baseline BiDAF. Our study suggests that a promising future direction is to combine BiDAF and R-Net for building better models.

1 Key Information to include

- Mentor: Lingjue Xie

2 Introduction

Reading comprehension is an important task for artificial intelligence both from a fundamental and a practical standpoint. It has many applications such as search engine and virtual assistant. Recently there have been a lot of progress on designing datasets and models for attacking the problem of question-answering (QA). A particularly popular dataset is the the Stanford Question Answering Dataset (SQuAD), which provides the data in a triple form: a passage,a question and an answer, which is an excerpt from the passage [1, 2]. Since the answer can be any span of the context passage and can involve reasoning across multiple sentences, the SQuAD dataset is more challenging than QA tasks such as multiple choice [3, 4] or Cloze style tasks [4, 5].

Since the release of the SQuAD dataset, researchers have proposed many deep learning models, such as bidirectional Attention Flow (BiDAF) [6], dynamic coattention networks [7], and R-Net [8, 9], as well as large transformer models such as BERT [10] and QANet [11]. In this report we focus mainly on the R-Net model, whose major difference from previouse works such as BiDAF is that R-Net consists of a self-matching attention. The motivation is that recurrent network in practice can only memorize information within a limited span of words, and that self-attention effectively incorporates information of the entire passage in encoding each timestep of the passage.

Here we report our attempt to implement the R-Net from scratch and characterize its performance on the SQuAD 2.0 dataset. Our implemented R-Net achieved 57.90 F1 and 54.46 EM score on the test set, which are less than the baseline BiDAF model (62.56 F1 and 59.04 EM), possibly due to the new no-answer examples in SQuAD 2.0 compared to SQuAD 1.0. We experimented with various methods to calculate attention-weights, such as simple dot product, linear kernel and L2 norm, and have found dot product to improve the performance. In addition, we have tried ensemble models by combining BiDAF and R-Net, which achieved 63.31 F1 and and 59.93 EM on the test set.

3 Related Work

There have been numerous proposed deep learning models for QA systems, and in particular for the SQuAD dataset. Equipped with features such as both context-to-query attention and query-to-context attention and memory-less attention, BiDAF is one of the early ones to achieve state-of-the-art result [6]. As attention proved to be essential for QA tasks, various form of attention mechanism were designed, such as coattention which involves attending to another attention output [7], as well as self-attention in R-Net, where the passage representation attends to itself [8, 9]. In addition, R-Net was partially inspired by match-LSTM [12], where the attention is performed squentially during the RNN computation. Another class of methods that demonstrated superior performance over previous models are the transformer models such as QANet [11], and pretrained models such as BERT [10].

In this work we implement the R-Net model and train and test its performance on the SQuAD 2.0 dataset. Previous work have shown R-Net to achieve state-of-the-art performance on SQuAD 1.0, so it is interesteing to test its performance on SQuAD 2.0 containing no-answer examples. The importance of the attention in QA tasks shown in previous work also motivates us to experiment with different types of attention for R-Net.

4 Approach

R-net consists of four parts: a passage and question encoder, an RNN with gated attention, an RNN with self-attention, and a pointer network for predicting answer. Following the original R-net paper, all the RNNs used are gated recurrent unit (GRU). In the following description, we elaborate on the procedure for character-level embeddings, which were not discussed in detail in the original paper, and briefly describe the other layers which were described in detail in the original paper. In addition, we use the BiDAF model with character-level embedding as our baseline [6].

4.1 Encoder

The encoder layer generates representations of passages and questions. First, it converts the text into word and character embeddings by looking up pretrained vectors. For example, for a question, the word vectors are $\{e_t^Q\}_{t=1}^m$, where m is the length of the question. For character-level embeddings, the character vectors of a word, $\{a_i\}_{i=1}^w$ where w is the length of a word, are fed into a bidirectional RNN (BiRNN),

$$h_{\text{fwd},i} = \text{BiRNN}(h_{i-1}, a_i), \tag{1}$$

$$h_{\text{rev},i} = \text{BiRNN}(h_{i+1}, a_i), \tag{2}$$

and the final hidden state is used as the character-level embedding for the word,

$$c = [h_{\text{fwd},w}, h_{\text{rev},w}]. \tag{3}$$

The concatenated word and character-level embeddings $\{[e_t^Q, c_t^Q]\}_{t=1}^m$ are then passed into another BiRNN to produce the representations for the text. The same procedure is also applied to the passage. The resulting vectors are u_t^Q and u_t^P.

4.2 Gated attention-based recurrent networks

This layer is an RNN applied to the passage representation, where part of the inputs are attention-pooling vectors of the entire question, using the soft-alignment attention proposed by Rocktaschel et al [13]. The inputs are the concatenation of the passage representation and the attention-weighted representation, multiplied by a gate with sigmoid activation on the input that controls information flow.

4.3 Self-attention

The self-attention layer is similar to the previous layer, except that the attention is from the word in the passage to all of the words in the same passage.

4.4 Pointer network

Pointer network, an RNN, is used to predict the start and end point of the answer [14, 15]. This layer uses the attention from the hidden states to the passage representation and selects the passage word corresponding to the highest attention score as the answer. The input to the RNN is itself an attention-pooling vector over the entire passage.

4.5 Other attention mechanisms

In addition to the standard R-Net implementation, we have also tried implementing three other kinds of attention by modifying the interaction between the key and query vectors. These include:

- Dot-product, where we simply take the dot product of the context and query vectors as the attention score
$$s_j^t = u_j^Q \cdot u_t^P \tag{4}$$
- Linear kernel, where we use a learn-able matrix W to model the interaction
$$s_j^t = u_j^Q W u_t^P \tag{5}$$
- L2 norm, where we use the distance between two vectors as the attention score
$$s_j^t = |u_j^Q - u_t^P|^2 \tag{6}$$

The attention weights are then calculated using a softmax function as usual.

4.6 Ensemble models

For the ensemble model, we combine the baseline BiDAF model and R-Net, by calculating a weighted average of the predicted probabilities

$$p = rp_{\mathrm{R-Net}} + (1-r)p_{\mathrm{BiDAF}} \tag{7}$$

where $0 < r < 1$, $p_{\mathrm{R-Net}}$ and p_{BiDAF} are probabilities of the start and end positions of the answer predicted by R-Net and BiDAF. We optimize the weight r on the dev set.

4.7 Code implementation

For the standard R-Net and the ensemble methods we used our own implementation using PyTorch. For trying out different attention mechanisms, we used an R-Net code we found from one GitHub repository[1], because the training time for this implementation is faster and we are limited by time. We believe the faster implementation ignores masking the paddings for passages and simplified a few linear layers by simply multiplying vectors with a random matrix, which make the training faster. In fact, as we show later, this implementation does not perform as well as our own implementation. However, we believe that it is still meaningful to compare how different attention mechanisms perform using this compromised implementation. For the baseline, we use the provided starter code and implemented a layer for character-level embedding following the method in the BiDAF paper [6].

5 Experiments

5.1 Data

We use the Stanford Question Answering Dataset (SQuAD) 2.0, which consists of about 130,000 examples for the training set and about 6000 examples for dev and test sets. In this dataset, given a passage of text (or context) and a question regarding the text, the goal is to find the answer to the question. In the SQuAD dataset, the answer is restricted to a span of the passage. The question answering system therefore needs to predict the start and endpoint of the answer within the passage [1, 2].

[1]`https://github.com/heliumsea/RNet-pytorch`

Models	F1	EM
BiDAF without character-level embeddings	60.48	57.18
BiDAF with character-level embeddings	63.36	60.36
R-Net	58.39	55.03
Ensemble, $r = 0.4$	63.55	60.56

Table 1: F1 scores and EM for the BiDAF model, R-net and ensemble model on the SQuAD **dev** set for IID SQuAD track

Models	F1	EM
BiDAF with character-level embeddings	62.56	59.04
R-Net	57.90	54.46
Ensemble, $r = 0.4$	63.31	59.93

Table 2: F1 scores and EM for the BiDAF model, R-net and ensemble model on the SQuAD **test** set for IID SQuAD track

5.2 Evaluation method

To evaluate the model we use exact match (EM), which is the percentage of predicted answers that exactly matches the ground truth, and F1 score, which is a harmonic mean of the precision and recall.

5.3 Experimental details

For R-net, we follow the original implementation, using 1 layer of RNN for character-level embedding and 3-layers for encoding passages and questions. All hidden sizes are 75. Dropout is applied between layers, with a dropout rate of 0.2. For the BiDAF model, we add character-level embeddings based on CNN to the starter code. The kernel size of the CNN is $5 \times h$ where h is the size of the character vector, and the number of output channels is 100, which means the final character-level embedding of the word is a $100 - d$ vector. We use AdaDelta optimizer, with $\rho = 0.9$ and $\epsilon = 10^{-6}$. The learning rate is 0.5 for BiDAF and 1.0 for R-Net, and we train the model until the EM and F1 scores stop increasing, fluctuating around some value. This usually takes about 15 epochs.

5.4 Results

The F1 and EM scores for the baseline BiDAF models and R-Net are shown in table 1 and 2 for dev and test sets, respectively. R-Net performs worse than the baseline model. This is perhaps not unexpected, as the official R-Net paper reported results on SQuAD 1.0, while here we train and evaluate the model on SQuAD 2.0. This result indicates that R-Net in the current form does not perform well on questions that might have no answer. For the ensemble model, we found that choosing an $r = 0.4$ (see Eq. 7) yields the best result on the dev set. It achieves the best result on the test set: 63.31 F1 and 59.93 EM.

6 Analysis

6.1 Model performance

Considering that R-Net performed worse than BiDAF on the dev set, we have done several analyses to make sure that our model was correctly implemented. In Fig. 1, we show the F1 score and negative log likelihood on the dev set for the models we trained. The evolution of both quantities for R-Net is similar to the baseline models, suggesting that our implementation of R-Net is probably reasonable. Note that R-Net suffers from overfitting more severely than BiDAF models, as shown in Fig. 1 (b), which might be one reason for the worse performance. Possible ways of reducing overfitting include adding regularization on the weight gradients during training, or increasing drop-out probabilities or drop-out layers.

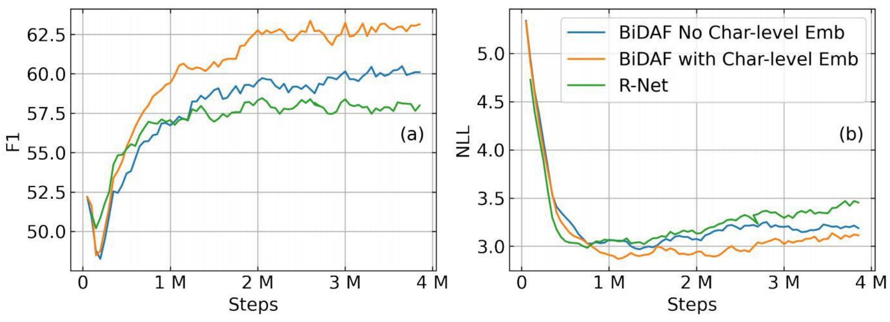

Figure 1: F1 scores (a) and negative log likelihood (b) on the SQuAD dev set, as a function of training steps for the BiDAF model with and without character-level embeddings, and R-Net.

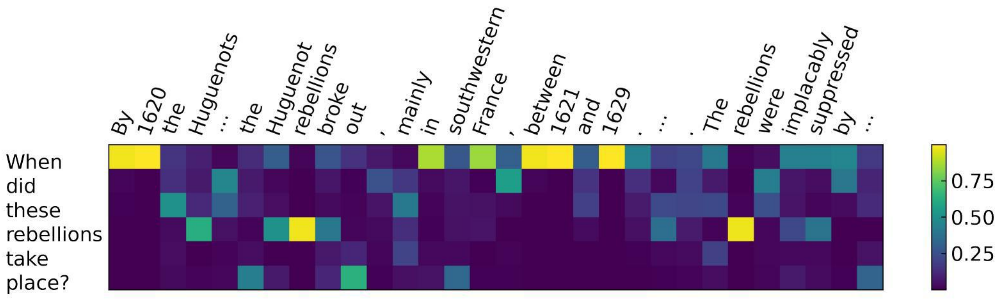

Figure 2: R-Net context-to-question attention weights, each column showing the attention of a context word to the a specific question word.

6.2 Attention analysis

As attention obviously plays an important role in QA tasks, it is quite interesting to analyze the attention weights produced by the two different models. In Fig. 2 we plot the R-Net context-to-question attention weights, each column showing the attention of a context word to the a specific question word. Here “By 1620" and “between 1621 and 1629" attend to “When", indicating that the model can correctly identify information related to time as requested by the query sentences. Another pair that show strong attention weights are “rebellions" in the context and query, which is expected as they are the same words. We also note that “out" seems to attend to “place" very strongly, while ideally we would like “broke out" to attend to “take place" strongly. In this case the model might not have actually understood the synonymy between the two phrases and the connection it drew might be in the sense that “out" carries a meaning of physical direction, which is related “place". The fact that “southwestern" in the context attends strongly to “place" also supports this interpretation.

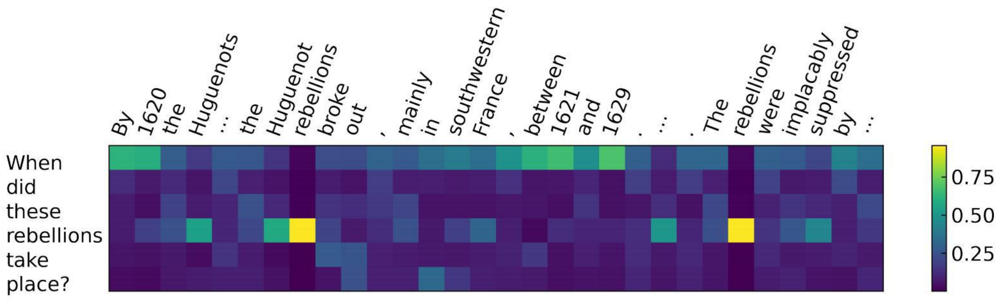

Figure 3: BiDAF context-to-question attention weights, each column showing the attention of a context word to the a specific question word.

The context-to-question attention weights for BiDAF is plotted in Fig. 3. The attention weights distribution is similar to that of R-Net, but the attention to the question word “When" is less prominent than R-Net. This shows that R-Net has a more effective attention mechanism, possibly due to a more complex architecture. We also note that in R-Net the attention is performed sequentially, taking into account previous hidden states. This might also help the attention to be more effective.

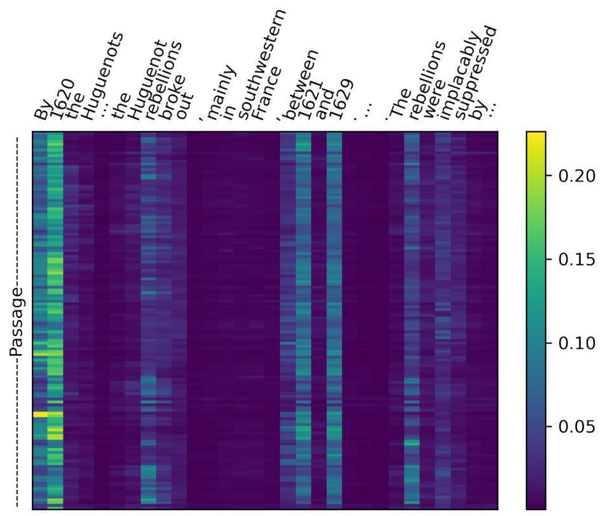

Figure 4: R-Net self attention weights within the context passage, each row showing the attention of a context word to the context itself.

Another key component for R-Net is self-attention, illustrated in Fig. 4, where each row shows the attention weights of a context word to the context themselves. Here, it seems that for each context word w, the context words that w attends to the most are the ones that are related to the question, such as “By 1620 ...", “the Huguenot rebellions ..." and so on. This is expected as self-attention is performed after the passage-question attention, and so the information of the question is already encoded into the passage representation. The self-attention layer acts as a kind of reinforcement for the question awareness of the passage.

6.3 Error analysis

Here we inspect a few examples where R-Net does not produce the correct answer.

Semantic relations

- **Question:** Who was Kaidu’s grandfather?
- **Context:**: Instability troubled the early years of Kublai Khan’s reign. Ogedei’s grandson Kaidu refused to submit to Kublai and threatened the western frontier of Kublai’s domain. The hostile but weakened Song dynasty remained an obstacle in the south. Kublai secured the northeast border in 1259 by installing the hostage prince Wonjong as the ruler of Korea, making it a Mongol tributary state. Kublai was also threatened by domestic unrest. Li Tan, the son-in-law of a powerful official, instigated a revolt against Mongol rule in 1262. After successfully suppressing the revolt, Kublai curbed the influence of the Han Chinese advisers in his court. He feared that his dependence on Chinese officials left him vulnerable to future revolts and defections to the Song.
- **Answer:** Ogedei vs. **Prediction:** Grandson
- **Analysis:** The model did not learn the connection between the word “grandson" and “grandfather". Since the context itself does not contain information on this connection, the model will have to learn this semantic relation from training data, which should contain examples that explicitly indicate such relation either within the context itself or through the question-answer pair, as in this example.

Attention problem

- **Question:** Who proved that these exist practical relevant problems that are NP-complete in 1961?
- **Context:** In 1967, Manuel Blum developed an axiomatic complexity theory based on his axioms and proved an important result, the so-called, speed-up theorem. The field really began to flourish in 1971 when the US researcher Stephen Cook and, working independently, Leonid Levin in the USSR, proved that there exist practically relevant problems that are NP-complete. In 1972, Richard Karp took this idea a leap forward with his landmark paper, "Reducibility Among Combinatorial Problems", in which he showed that 21 diverse combinatorial and graph theoretical problems, each infamous for its computational intractability, are NP-complete.
- **Answer:** N/A vs. **Prediction:** Leonid
- **Analysis:** The question is very similar to a subsequence of the context, except the question specified a year 1961 which is different from the context 1971. R-Net attention mechanism might have been too strong that it thinks the difference between “1961" and “1971" is too small. This might be a general problem. In an end-to-end model liek R-Net, every word or character are treated equally (embeddings), while in reality some words are more significant than others. In this case the model did not learn the importance of the specified year.

Incomplete answers

- **Question:** What is the term for the lack of obsevable free quarks?
- **Context:** The strong force only acts directly upon elementary particles. However, a residual of the force is observed between hadrons (the best known example being the force that acts between nucleons in atomic nuclei) as the nuclear force. Here the strong force acts indirectly, transmitted as gluons, which form part of the virtual pi and rho mesons, which classically transmit the nuclear force (see this topic for more). The failure of many searches for free quarks has shown that the elementary particles affected are not directly observable. This phenomenon is called color confinement.
- **Answer:** color confinement vs. **Preduction:** called color
- **Analysis:** Here the model does understand the question and was able to locate the vicinity of the correct answer in the passage, but nevertheless produces an incomplete answer. This might be an indication that the RNN is affected by local features. As gating controls how much to “forget" some features, We could try improving the gating applied to the passage representations.

6.4 Different attention mechanisms

We have also trained R-Net with modified interaction for the attention mechanism, using either dot product, a linear kernel or L2 norm. For this study we used existing code from an open repository as discussed in Sec. 4.7. Table 3 shows that in this implementation of the R-Net, using dot product is better than the original attention mechanism, which involves passing key and query vectors through linear layers and applying a tanh activation. Linear kernel performed similarly as the standard R-Net. For L2 norm kernel, the model suffers from severe overfitting where the performance on the dev set drops considerably in later stage of training. This indicates that simple distance measure of the vectors is not helpful as a similarity score.

Attention	F1	EM
R-Net standard	35.08	34.33
Dot product	43.88	43.56
Linear kernel	33.69	32.83

Table 3: F1 scores and EM for the BiDAF model and R-net on the SQuAD test set for IID SQuAD track

7 Conclusion

We have implemented the R-Net model for QA tasks in the SQuAD 2.0 dataset. We found that R-Net has a worse performance than our baseline model, BiDAF. By inspecting the attention weight distribution of R-Net and BiDAF we found that the attention mechanism in R-Net is more effective in context-to-question attention, and that the self-attention mechanism helps reinforce the question-awareness of the context representation. We have also found that a simple ensemble model combining BiDAF and R-Net achieves marginally but consistently better performance than BiDAF.

The good performance of the ensemble model indicates that BiDAF and R-Net may be combined in more optimal ways to improve the model. Therefore a good future direction would be to identify the strengths and weakness of the two models and design a hybrid model.

References

[1] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics.

[2] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Association for Computational Linguistics (ACL)*, 2018.

[3] Matthew Richardson, Christopher J.C. Burges, and Erin Renshaw. MCTest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 193–203, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[4] Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. The goldilocks principle: Reading children's books with explicit memory representations. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[5] Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 1693–1701, Cambridge, MA, USA, 2015. MIT Press.

[6] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension, 2018.

[7] Caiming Xiong, Victor Zhong, and Richard Socher. Dynamic coattention networks for question answering, 2018.

[8] Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 189–198, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[9] Microsoft Research Asia. R-net: Machine reading comprehension with self-matching networks, 2017.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[11] Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. Qanet: Combining local convolution with global self-attention for reading comprehension, 2018.

[12] Shuohang Wang and Jing Jiang. Machine comprehension using match-lstm and answer pointer, 2016.

[13] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiský, and Phil Blunsom. Reasoning about entailment with neural attention, 2016.

[14] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *NIPS*, pages 2692–2700, 2015.

[15] Shuohang Wang and Jing Jiang. Learning natural language inference with LSTM. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1442–1451, San Diego, California, June 2016. Association for Computational Linguistics.