Attention Mechanism in Machine Comprehension

Yingnan Xiao
Stanford University
Stanford, California, CA 94305
lxyxynt@stanford.edu

Abstract

Machine comprehension(MC) have seen increased interest and significant progress recently, especially with the latest releases of few Question-Answering(QA) dataset. Most of recent high-performance models have applied attention mechanism, which typically try to focus on a small portion of the context when generating hidden states of question(and vise versa). In this paper, instead of ensemble fancy models, we did detailed error analysis on a simple RNN models to better understand the weakness, and proposed our model based on existing high-performance attention-based models, with attention visualization to help understanding the model improvements. On the SQuAD test dataset, our best model achieves a single model performance of 58.815 F1 and 47.781 EM, improved baseline model performance by 38.7% on F1 and 45.1% on EM.

1 Introduction

Machine comprehension(MC) and Question Answering(QA) has been the most difficult tasks in Nature Language Processing research field, which aims to build a system responding to questions presented in natural language, based on some contexts that may contains the answer. Intuitively, researchers want to teach machines to read and understand the context paragraph(MC), and QA provided a nice measurement of machine comprehension quality. The industrial world has designed and deployed various successful production by rule-based QA system, like Siri[1], Google Assistant[2] and Cortana[3].

Question Answering systems have recently gained significant popularity and progress, one primary reason is the release of the SQuAD dataset in 2016[1]. The SQuAD dataset is built from 536 WikiPedia articles, contains 107,785 human-generated reading comprehension ¡questions, answer, context¿ triples, and the answer is guaranteed to be a span in the original context. Although this introduced some limitation on the dataset, it also enabled the possibility to quantify the result easily. Inspired by SQuAD, more and more researchers started work on machine comprehension dataset. For example, Tom et al. recently released NarrativeQA dataset[2], which is more open-ended and requires reader must answer questions by reading entire books or movie scripts. In this paper, we will mainly focus on SQuAD dataset.

Another key factor to the advancement has been the use of attention mechanism on end-to-end neural network model. Typically, the attention mechanism tries to focus on a small portion of the context when generating hidden states of question(and vise versa), which enabled the system to find the most relevant information from context to answer the question[3, 4]. Seo et al. introduced bi-directional attention flow[5] instead of traditional uni-direction attention, represented the context at different granularities and obtain query-aware context. Caiming et al. proposed Dynamic Coattention Net-

[1]https://www.apple.com/ios/siri
[2]https://assistant.google.com/
[3]https://www.microsoft.com/en-us/windows/cortana

work[6], which fuses co-dependent representations of the question and the document in order to let later components focus on relevant parts of both.

With the attention mechanism be published, the performance of most ensemble / single models get improved significantly[4]. In this paper, we will first create our attention-based neural network model for the QA task, then deeply analysis loss cases and work on different improvements to find some underlying myth.

2 Baseline Model

We first describe the overall structure of our baseline model, and subsequently the different attention mechanisms we want to explore. Our baseline machine comprehension model consists of four layers and works hierarchically.

2.1 Embedding Layer

For each SQuAD example (context, question, answer), we want to represent the context and question by a sequence of d-dimensional word embeddings. Specifically, we represent each context as $x_1, ..., x_N \in R^d$, and represent question as $y_1, ..., y_M \in R^d$ by pre-trained word embeddings.

2.2 RNN Encoder Layer

We build the first neural layer by one RNN encoder layer with bi-directional GRU. The bidirectional GRU produces a sequence of forward hidden states ($\overrightarrow{c_i} \in R^h$ for the context and $\overrightarrow{q_j} \in R^h$ for the question) and a sequence of backward hidden states ($\overleftarrow{c_i}$ and $\overleftarrow{q_j}$).

$$\{\overrightarrow{c_1}, \overleftarrow{c_1}, ..., \overrightarrow{c_N}, \overleftarrow{c_N}\} = biGRU(\{x_1, ..., x_N\})$$
$$\{\overrightarrow{q_1}, \overleftarrow{q_1}, ..., \overrightarrow{q_M}, \overleftarrow{q_M}\} = biGRU(\{y_1, ..., y_m\})$$

In the end, we concatenate the forward and backward hidden states to obtain the context hidden states c_i and the question hidden states q_j respectively:

$$c_i = [\overrightarrow{c_i}; \overleftarrow{c_i}] \in R^{2h} \; \forall i \in \{1..N\}$$
$$q_j = [\overrightarrow{q_j}; \overleftarrow{c_j}] \in R^{2h} \; \forall i \in \{1..M\}$$

2.3 Attention Layer

Here we use the the most straightforward attention flow is basic dot-product attention. For each context hidden state c_i, we first attends all question hidden states to calculate attention distribution $\alpha i \in R^M$:

$$e^i = [c_i^T q_1, ..., c_i^T q_M] \in R^M$$
$$\alpha_i = softmax(e^i) \in R^M$$

Then we can use distribution to take weighted sum of the question hidden states q_j, producing the attention output a_i:

$$a_i = \Sigma_{j=1}^M \alpha_j^i q_j \in R^{2h}$$

The final output of attention layer is simply concatenate context hidden states with attention output:

$$b_i = [c_i; a_i] \in R^{4h} \; \forall i \in 1..N$$

2.4 Output Layer

The basic output layer contains a fully-connected layer, and followed by a linear layer with softmax for start position and end position individually:

[4]https://rajpurkar.github.io/SQuAD-explorer/

$$b_i^{'} = ReLU(W_{FC}b_i + v_{FC})$$
$$p^{start} = softmax(w_{start}^T b_i^{'} + u_{start})$$
$$p^{end} = softmax(w_{end}^T b_i^{'} + u_{end})$$

3 Error Analysis on Baseline Model

We randomly selected 25 loss questions (based on EM) and try to analysis the result by visualizing the attention distribution. In summary, 44% of errors are due to the incorrect answer boundaries(which means the F1 value is not fully wrong), 40% captured wrong attention position, 8% require external knowledge, and 8% need to cross different sentences to get final answer. We will focus on the first two cases for further improving. Here're few typical examples:

3.1 Incorrect answer boundary

Context: ...the most important of which were use of a nitrogen/oxygen mixture instead of pure oxygen before and during launch , and removal of flammable cabin and space suit materials.
Question: what type of materials inside the cabin were removed to help prevent more fire hazards in the future ?
Answer: flammable cabin and space suit materials
Predicted: flammable cabin and space suit

This is a typical case that our model works well, but we failed to identify the correct answer boundary. One possible improvement can be add another layer between starting position and end position predict.

3.2 Incorrect attention position

Context: ...the primes 41 , 43 , 47 and 53 appear in the third etude, "neumes rythmiques" . according to messiaen this way of composing was " inspired by the movements of nature , movements of free and unequal durations.
Question: in which etude of neumes rythmiques do the primes 41 , 43 , 47 and 53 appear in ?
Answer: the third etude
Predicted: etude

This case looks like our baseline model made wrong boundaries again, but actually not. By looking into the attention distribution in figure 1, we find the baseline model actually failed to focus on the critical answer(the third), but **only focused on some shallow patterns** and found the keyword 'etude'. Some stronger attention model may help on this case.

Context: ...where the different institutions can not agree at any stage, a "conciliation committee" is convened, representing meps, ministers and the commission to try and get agreement..
Question: what entity is created if the three different institutions can not come to a consensus at any stage ?
Answer: conciliation committee
Predicted: parliament , a minority in the council

This is another typical case that our attention model failed to focus on correct part. The context length of this case is really long(318 words), which made the attention task more difficult.

3.3 Require External Knowledge

Context: southern california is home to many major business districts . central business districts (cbd) include downtown los angeles, downtown san diego, downtown san bernardino , downtown bakersfield, south coast metro and downtown riverside.
Question: what is the only district in the cbd to not have " downtown " in it's name ?
Answer: south coast metro
Predicted: central business districts

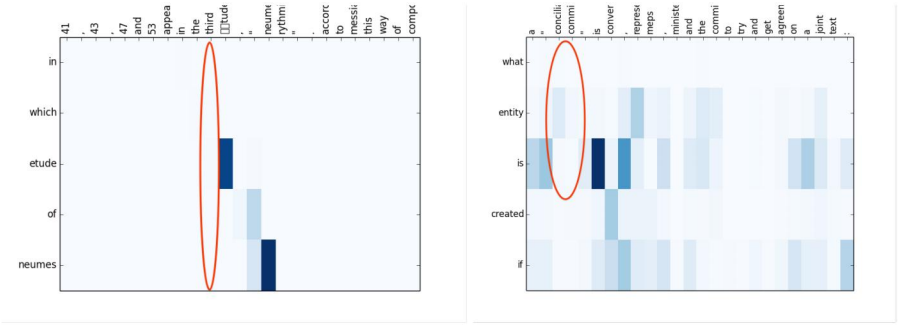

Figure 1: Baseline Attention on Loss Cases(**Full visualization can be found in appendix**)

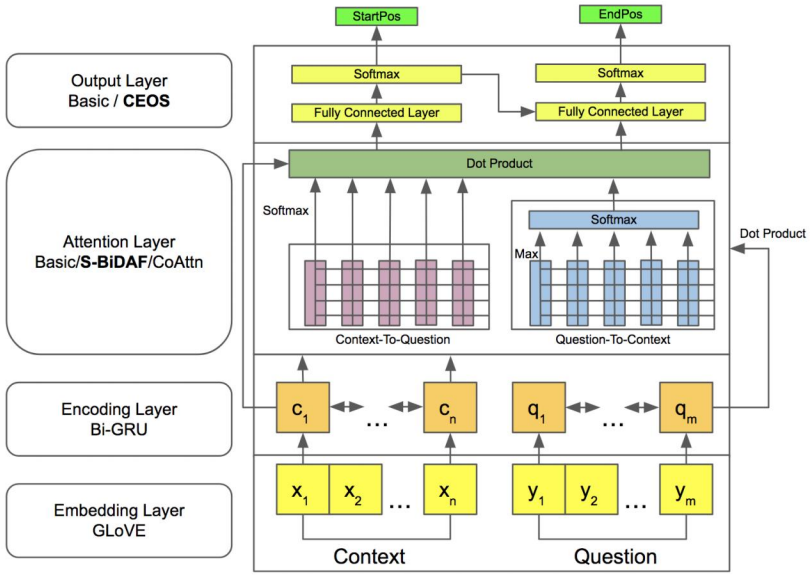

Figure 2: Model Overview

This case falls to the minority class, where require the model learn some external knowledge to understand 'not have $keyword in name'. Such case also happened like numerical knowledge(which is the 3rd one).

4 Model Improvements

In this section, we proposed some improvements on baseline model based on error analysis. Similar to baseline model hierarchy, we kept the four-layer model, but proposed some alternative layers to replace the original layer, as Figure 2 showed.

4.1 Condition End Position on Start Position(CEOS)

As we saw in previous section, we found most of EM loss are due to incorrect answer boundaries. We proposed CEOS, to add another layer neural network between start position and end position. More specifically, we first apply fully-connected layer followed by linear layer to decide start position, as we did in basic output layer:

$$b_i^{'} = ReLU(W_{FC_S} b_i + v_{FC_S})$$
$$p^{start} = softmax(w_{start}^T b_i^{'} + u_{start})$$

Then we take the weighted sum on blended input by p^{start}, followed by another fully-connected layer to encode information into a hidden state h, so we can let end position condition on start position by apply the hidden state as a linear layer before softmax:

$$h = \Sigma_{i=1}^{N} b_i p_i^{start}$$
$$h^{'} = ReLU(W_{FC_H} h + v_{FC_H})$$
$$p^{end} = softmax(bh^{'} + v_{FC_E})$$

4.2 Attention Layers

Another direction is to introduce more powerful attention models. Here we want to explore some attention models, including S-BiDAF, BiDAF and CoAttention.

4.2.1 Simple Bi-Directional Attention Flow(S-BiDAF)

BiDAF[5] is one of the high-performance attention mechanism recently. The basic idea is pretty straightforward, but improved our baseline model a lot. It introduced bi-directional attention flow instead of traditional uni-direction attention, and represents the context at different granularities so we can obtain query-aware context. From the paper, the most value that it added is the bi-directional flow and modeling layer, and the core part is still the dot product.

We proposed a simple version of BiDAF, that removed similarity matrix and use dot product directly. We first take the row-wise softmax of S to obtain attention distributions α_i, which we use to take weighted sums of the question hidden states q_j, yielding context-to-question attention outputs:

$$\alpha_i = softmax(S_{i,:}) \in R^M$$
$$a_i = \Sigma_{j=1}^{M} \alpha_j^i q_j \in R^{2h}$$

Secondly, we take question-to-context attention outputs. For each context location, we take the max element from corresponding row in similarity matrix $m_i = max_j S_{ij} \in R$, and take the softmax over the resulting vector to get attention distribution $\beta = softmax(m) \in R^N$, for weighted sum of the context hidden states $c^{'}$:

$$c^{'} = \Sigma_{i=1}^{N} \beta_i c_i \in R^{2h}$$

Lastly, we concatenate the results as the final attention output:

$$b_i = [c_i; a_i; c_i \circ a_i; c_i \circ c^{'}] \in R^{8h}$$

4.2.2 Co-Attention

Co-Attention[6] is proposed by Caiming et al., which fuses co-dependent representations of the questions and the document in order to let later components focus on relevant parts of both. It introduced similar two-way attention between question and context, which is a second-level attention layer on previous attention output. This model encoded the word before dot product, which is not only primarily depends on word similarity. Co-Attention also introduced another Bi-LSTM layer at last, which actually also worked as a modeling layer.

5 Experiments

In this section, we will first try to evaluate some shared hyper-parameters like length and hidden states sizes on baseline model, and deeply understand how our improvements works.

5.1 Training Details

We trained and evaluated our model under tensorflow v1.4.1 with SQuAD dataset, which used F1 and Exact Match (EM) metrics. Exact Match is a binary measure (i.e. true/false) of whether the

Table 1: Hyper Parameters used in our model

Hyper Parameter	Value
fixed context length	400
fixed question length	30
embedding size	100
max gradient norm	5
learning rate	0.006
dropout	0.15
batch size	100
hidden state size	200

Table 2: Results on dev set

Model	F1	EM	#params
Baseline	42.678	33.784	521,802
Baseline-lr0.05	6.192	4.109	521,802
Baseline-200d	42.201	32.923	641,802
S-BiDAF	56.99	46.036	681,802
S-BiDAF-lr0.006	**57.488**	**46.187**	681,802
S-BiDAF-nodropout	49.003	41.358	681,802
S-BiDAF+CEOS	56.568	44.995	722,401
BiDAF	49.542	38.213	683,002
CEOS	42.146	32.498	562,401
CoAttn	53.679	42.062	8,607,802
CoAttn-lr0.006	57.414	46.178	8,607,802

system output matches the ground truth answer exactly, while F1 is the harmonic mean of precision and recall, which basically measures how well the predicted result overlaps with the ground truth.

For the embedding layer, we applied different dimensions of pre-trained GLoVE word vectors[x]. In addition, we enforced the fixed length of context to 400 and fixed length of question to 30 based on our statistics. By choosing these two threshold, we ensured 99.9% contexts and questions will not be affected, while we can filter some outliner and make the training faster.

We also tried to choose different hyper parameters based on baseline model. We first tried different word vector dimensions on baseline model, which showed increasing from 100 to 200 didn't really improve the performance, but slowed down the training process a bit. we also tried some small dropout value and it turns out model get over-fitted quickly. Although the training performance is increasing, the performance on dev dataset starts drop and loss has significantly increased.

We listed all hyper parameters we used in Table 1. **More training details and stats can be found in appendix**.

5.2 Experiment Result

The overall results on different models are shown in table 2. Besides primary models, we also tried to tune different parameters per-model basis.

6 Improvements Analysis

From experiment results, we can see powerful attention models significantly improved our baseline model, while CEOS actually didn't help a lot. We will do some deep analysis on how each components helped.

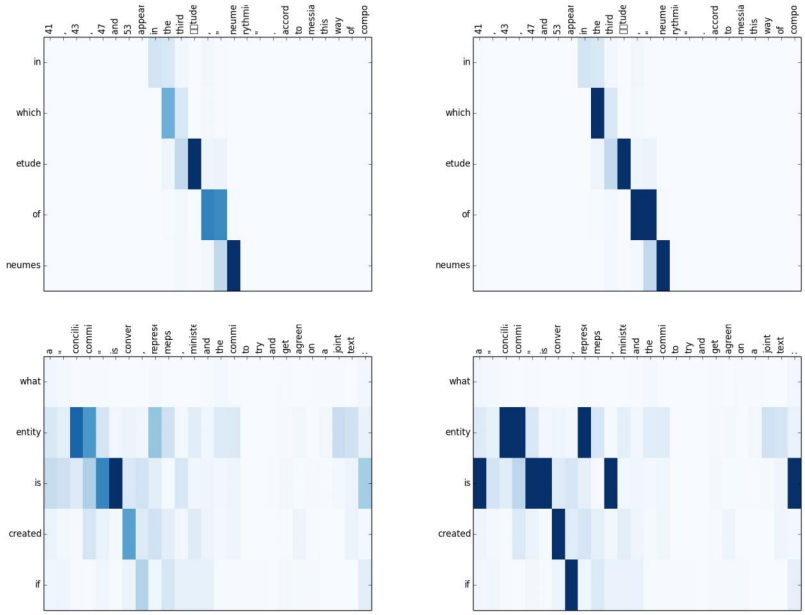

Figure 3: BiDAF Attention distributions(**Full visualization can be found in appendix**)

6.1 CEOS

From experiment results the CEOS model didn't get the model improved, but intuitively it should. We tried to dig more loss cases, and find previous cases actually get better, but the model also introduced new losses. For the first case we listed in error analysis section, **CEOS is the only model that solved it with Exact matching answer**, while all other models can't. However, we found that CEOS also made some other cases answer boundary wrong. That means CEOS did changed the output layer behavior, but it may not strong enough to capture all the information it needs.

6.2 Attention Mechanisms

Why does different attention worked so well? We chose two very different type of attentions and want to look into the reasons. The BiDAF/S-BiDAF are based on dot product, and Co-Attention works more like an encoder.

6.2.1 S-BiDAF

S-BiDAF removed the first encoding layer, but still got the similar performance on original BiDAF model. It introduced two-way attention models, which is simple and easy to explain. From attention distribution visualization, we can see context-to-question and question-to-context distribution together captured the correct position successfully, as Figure 3 shows.

However, by S-BiDAF took advantage of the core dot product, the attention actually directly depends on word similarity only. That means, the model may only learnt shallow patterns and can be easily hacked if the context contains very similar sentences with question. We tried a simple hack on a case that attention model predicted correctly, and found this as the weakness of the model:

Context: ...one important rock found during the apollo program is dubbed the genesis rock , retrieved by astronauts david scott and james irwin during the apollo 15 mission. **the name of the rock found during the apollo 14 mission is called WRONG** .
Question: what was the name of the rock found during the apollo 15 mission that kreep was discovered in ?
Answer: genesis rock
Predicted: WRONG

6.2.2 CoAttention

Co-Attention is more difficult to explain, as it encoded question context first, introduced two-layer attention output, and feed them into another Bi-RNN to generate the final output. From attention visualization(**see appendix**), we the distribution is more averaging on lots of words. Intuitively, we assume this model depends on this attention layer captured much more information than baseline model, instead of smart attention mechanism. Afterall, Co-Attention model introduced more than 8,000,000 parameters, but only get mostly same result on S-BiDAF.

As it's not depending on simple word similarity directly, it may not only depends on shallow patterns. We also tried to hack it by the case above, and the model worked well. From this point of view, the CoAttention itself may not be a smart attention model, but it have good performance and potential with other network structure, like Dynamic CoAttention Network[6].

7 Conclusion

In this paper, we explored two state-of-art attention mechanism, S-BiDAF and CoAttention, proposed S-BiDAF and CEOS to optimize traditional output layer. The experimental evaluations on Stanford Question Answering Dataset (SQuAD) showed that our model achieves significant improvements. We also conducted analysis and/or detailed visualization, in order to reason each components we explored. However, we also found that our model somehow only learnt some shallow patterns in some cases. Therefore, it can be easily beat by some sentences similar to question itself, or similar to the answer sentence.

In future, we can explored following directions to further improve our model:

- To Explore even more powerful attention layer instead of word similarity, which may help the model to capture more information.
- To Explore more network structures. The original BiDAF paper claimed the modeling layer have also improved baseline model a lot.
- To Investigate to introduce external knowledge base or text resource to help model learn some generic knowledge, to better understand the context semantics.

Acknowledgments

The authors would like to thank all the CS224N teaching staff for their helpful insights, and Microsoft for their generosity of sponsoring our experiments on Azure.

References

[1] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, Squad: 100,000+ questions for machine comprehension of text, arXiv preprint arXiv:1606.05250.

[2] Koisk, T., Schwarz, J., Blunsom, P., Dyer, C., Hermann, K. M., Melis, G., & Grefenstette, E. (2017). The NarrativeQA Reading Comprehension Challenge. arXiv preprint arXiv:1712.07040.

[3] Xiong, C., Merity, S., & Socher, R. (2016, June). Dynamic memory networks for visual and textual question answering. In International Conference on Machine Learning (pp. 2397-2406).

[4] Yang, Z., He, X., Gao, J., Deng, L., & Smola, A. (2016). Stacked attention networks for image question answering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 21-29).

[5] Seo, M., Kembhavi, A., Farhadi, A., & Hajishirzi, H. (2016). Bidirectional attention flow for machine comprehension. arXiv preprint arXiv:1611.01603.

[6] Xiong, C., Zhong, V., & Socher, R. (2016). Dynamic coattention networks for question answering. arXiv preprint arXiv:1611.01604.