Question answering using weighted-loss Bi-Directional Attention Flow on SQuAD Dataset

Mengjie Cheng
Department of Management Science and Engineering
Stanford University
Stanford, CA 94305
Codalab username: Magie
mengjiec@stanford.edu

Abstract

Question Answering has been an exciting and challenging task in Nature Language Processing as it depends on how well systems can understand the text. In this paper, we tried different combinations of latest models and proposed the residual feature aggregation layer and the weighted loss function. We evaluated on SQuAd Dataset and achieved (F1: 75.42, EM: 65.345) on devset using a single model, (F1: 76.004, EM: 66.263) on devset and (F1: 77.036, EM: 67.922) on testset with 9 checkpoints ensemble.

(a potential better test set score is still under testing and will be updated by March 22nd 5:00 pm)

1 Introduction

Reading Comprehension has been a challenging task for machines, requiring both word level understanding of natural language and knowledge level understanding about the world.[4] Such problems have gained significant popularity over the years not only because of their vast applications in the real world, but also theoretical underpinnings for language and deep learning research. Just like the contribution of ImageNet for object recognition, a large and high-quality reading comprehension dataset Stanford Question Answering Dataset(SQuAD) released in 2016[4] has also led to many research works and significant breakthroughs in building effective reading comprehension systems. Especially, R-Net[1], BiDAF[5], DCN[2] and DCN+[3] investigated different kinds of attention models and their variants achieved breathtaking performance on SQuAD(82.482 EM for R-Net vs 82.304 EM for human performance)

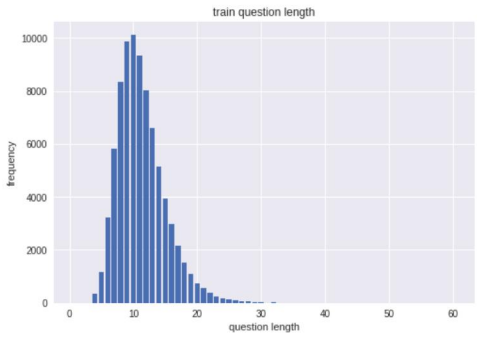
We tried to answer two questions in this paper: 1. How to get a good contextual representation for sentence. 2. How to directly make sure our optimization goal is consistent with the evaluation metrics. To do that, we started from the baseline model, evaluating different modules that can be inserted directly, and finally designed a model built on top of BiDAF with residual feature aggregation layers. We also propose F1-EM-weighted loss function particularly for this reading comprehension task. We find simply stacking multiple RNNs(or even residual version) leads to poor generalization performance. And attention layer is extremely important for the model to focus on certain parts of the question context. So we use BiDAF layer as our attention layer. We finally ensemble 9 models for testing and it achieves reasonably good results on leaderboard.

The rest of the paper is organized as following: Section 2 describes the dataset and the goal of the task Section 3 describes the model I've tried and the strategies I've explored to answer our two basic questions. Section 4 includes all the qualitative and quantitative analysis of each model, including error, loss and attention analysis. Section 5 is about model discussion and future work.

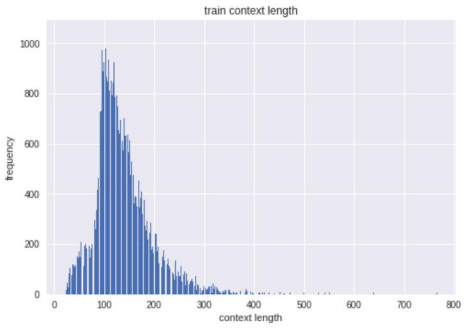
2 Data and Problem Description

The Stanford Question Answering Dataset (SQuAD) contains 107,785 question-answer pairs on 536 articles posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage[4]. The training dataset contains 86326 question-answer pairs (80% of the whole dataset). The dev contains another 10% and the hidden test set contains the last 10%. The distribution of the question, context, and answer length of the training set are listed below. From the Figure 2, we could see the length of most of the question is less than 30, most of the contexts are less than 400 and most of the answers are less than 20. We tried both 400, 500 and 600 for context length and we found the performance didn't change a lot. For speed purpose, we set the max length of context as 400. And from the Figure 2, we could see the position of answer inside the corresponding context, from the analysis of the answer's both starting and ending index, we could see the answer is more likely to appear at the beginning of the context.

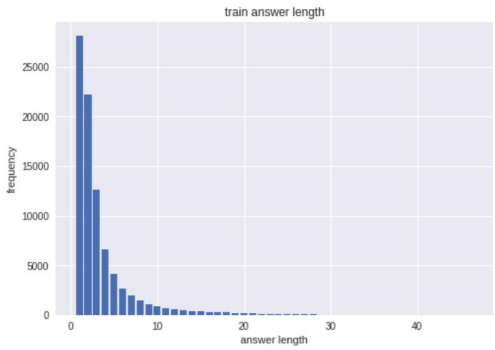
So that the problem needed to solve is given word sequence (embeddings from GloVe) of context c with length C and question q with length Q, we need to train a model to predict the start and end position respectively in the context c to answer question q.



(a) training set question length



(b) training set context length



(c) training set answer length

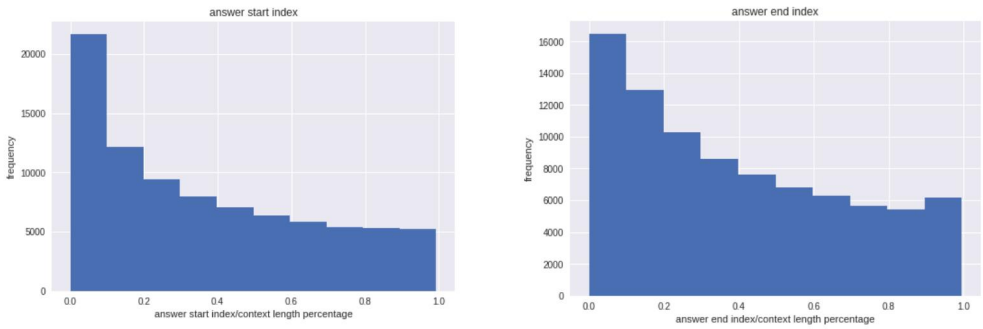
Figure 1: Training set question/context/answer length

3 Models

3.1 Residual RNN

As our first step, we add residual connection to the baseline model, hoping that can alleviate gradient vanishing problem and make the optimization process stable. From our experiments, there are some improvements but not as good as our expected. The way we construct residual RNN is:

$$\left\{\overrightarrow{h_1}, \overleftarrow{h_1}, ..., \overrightarrow{h_T}, \overleftarrow{h_T}\right\} = biGRU(\{x_1, x_2, ..., x_T\}) \tag{1}$$



(a) the start position of answer inside the context (b) the end position of answer inside the context

Figure 2: Position of answer appearing in the context

and

$$\overrightarrow{g_i} = \overrightarrow{h_i} + x_i, \tag{2}$$

$$\overleftarrow{g_i} = \overleftarrow{h_i} + x_i, \tag{3}$$

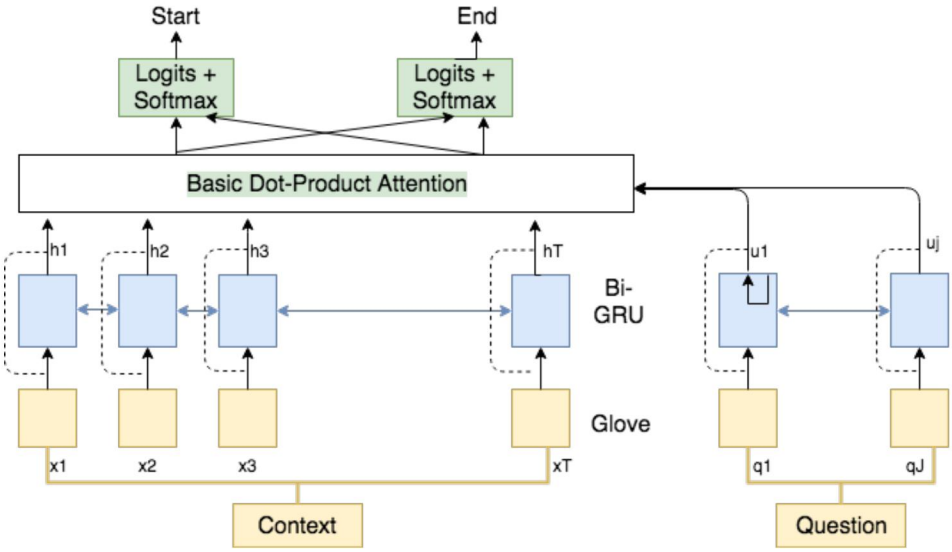


Figure 3: Residual RNN Model Architecture

3.2 Stack more RNN layers

In this section, we tried to answer the question: does the depth of RNN matter? So we stacked two bi-GRU layers but when we used more, the performance degenerated and there is overfitting problem.

3.3 Bi-directional Attention Flow (BiDAF)

Word embedding layer: Word embedding layer maps each word in context and question to a high-dimensional vector space. We use pre-trained word vectors GloVe[8] provided by the existing code to obtain the fixed word embedding of each word. So that for each SQuAD example, the context is represented by a sequence of d-dimensional word embeddings $x_1, x_2, ..., x_T \subset R^d$, the

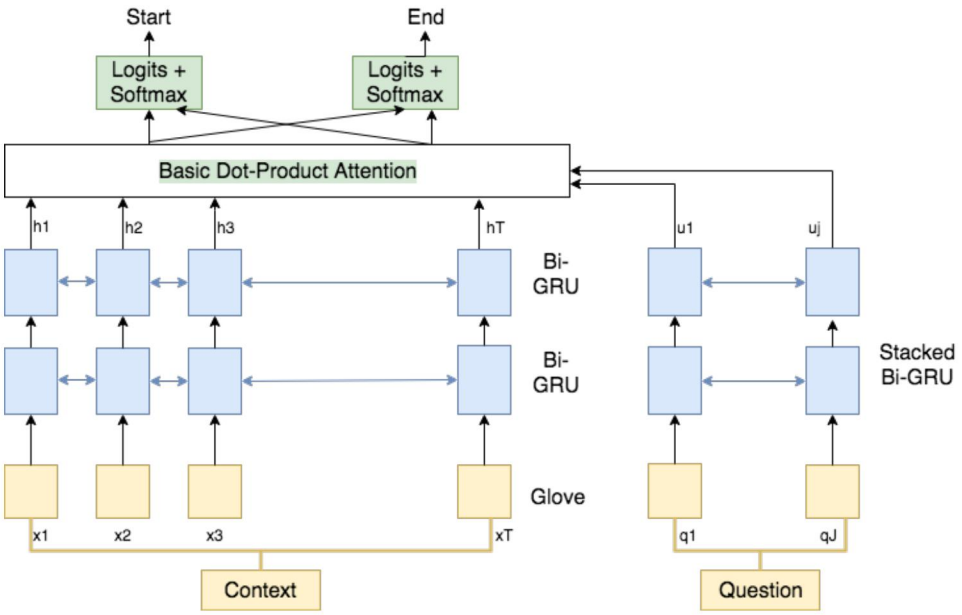


Figure 4: Stack RNN Model Architecture

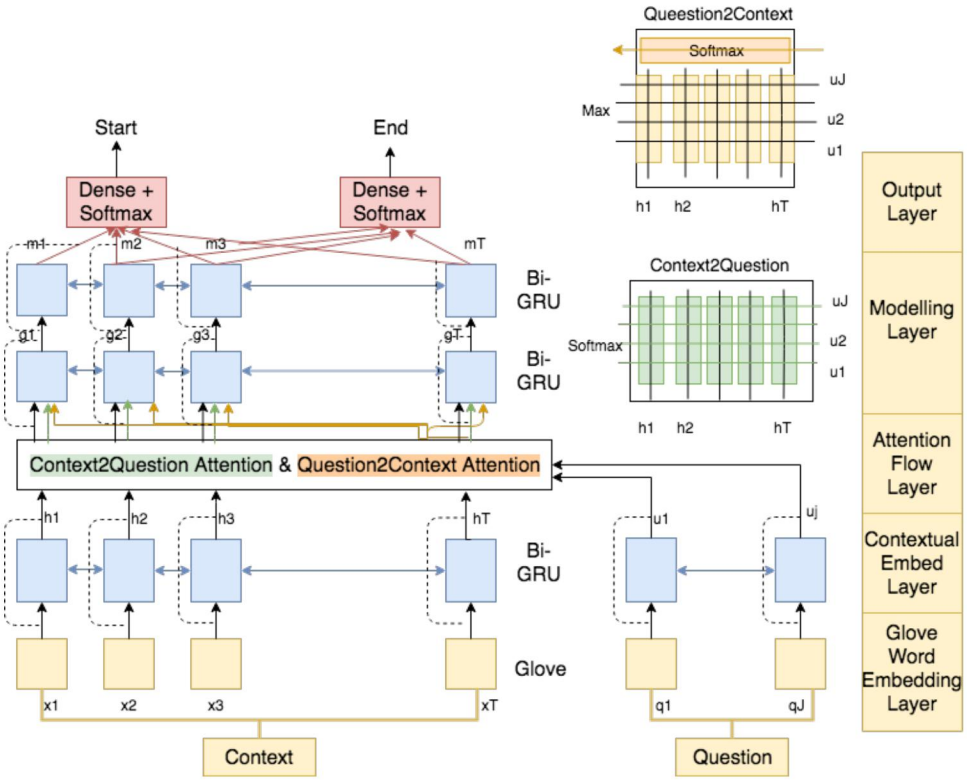


Figure 5: BiDAF Model Architecture

question is also represented by a sequence of d-dimensional word embeddings $q_1, q_2, ..., q_J \subset R^d$. And we fix our embedding size d as 100 because I didn't see the model performance improvement if I increase embedding size to 200.

Contextual Embedding Layer: This layer uses bidirectional GRU to utilize contextual cues from surrounding words to refine the embedding of the words. We also try a bi-directional LSTM (same as the paper [5]), but it doesn't work good compared with bidirectional GRU in my experiment and leads to extremely slow training. After the bi-GRU, we obtained $H \subset R^{2d*T}$ from the context word vector X and $U \subset R^{2d*J}$ from the question word vector Q.

$$\left\{\overrightarrow{h_1}, \overleftarrow{h_1}, ..., \overrightarrow{h_T}, \overleftarrow{h_T}\right\} = biGRU(\{x_1, x_2, ..., x_T\}) \tag{4}$$

$$\{\overrightarrow{u_1}, \overleftarrow{u_1}, ..., \overrightarrow{u_J}, \overleftarrow{u_J}\} = biGRU(\{q_1, q_2, ..., q_J\}) \tag{5}$$

and

$$h_i = \left[\overrightarrow{h_i}; \overleftarrow{h_i}\right] \in R^{2d} \vee i \in \{1, 2, ..., T\} \tag{6}$$

$$u_i = [\overrightarrow{u_i}; \overleftarrow{u_i}] \in R^{2d} \vee i \in \{1, 2, ..., J\} \tag{7}$$

Since the forward and backward hidden states are stacked in the last channel, we want to get a better representation for each position i by aggregating the forward hidden state and backward hidden state at i. We use two dense layers to do that. But to get rid of gradient vanish problem, we also add residual connections where,

$$g_i = ReLU(W_1 h_i + b_1) + h_i \tag{8}$$

$$f_i = ReLU(W_2 g_i + b_1) + g_i \tag{9}$$

and we use f_i as the output at each position i for this contextual embedding layer.

Attention Flow Layer: Before this layer, context and question are processed separately, we need some ways to relate context with question to get an unified representation for the model to know what the question is asking and how to find the answer from the context according to the question. [9] proposed using attention mechanism to better capture the corelation between context and question. [5] proposed using an extra context to query attention besides the query to context attention.

The implementation of the attention layer is the same as the paper[5], we compute attentions in two directions: from context to query as well as from query to context. Both of them are from a shared similarity matrix $S \in R^{T*J}$, which contains a similarity score S_{tj} for each pair (h_t, u_i).

$$S_{tj} = \alpha\left(H_{:t}, U_{:j}\right) \in R \tag{10}$$

Different from the original paper, the α function we choose is simply the dot production between $H_{:t}$ and $U_{:j}$ because we find this will significantly reduce the memory usage without losing much performance.

Context-to-question attention: Context to question attention(C2Q) increases the weights of question words that are most relevant to context word. For i-th context word, the attention weight is computed by $a_t = softmax(S_{t:})$, and then each corresponding attended question vector is $\widetilde{U_{:t}} = \sum_j a_{tj} U_{:j}$. Thus $\widetilde{U}$ becomes a 2d-by-T matrix containing the attended query vectors for the entire context.

Question-to-context attention: Question to context attention(Q2C) signifies which context words have the closest similarity to one of the question words. For each context location $i \in \{1, 2, ..., T\}$, I take the max of the corresponding row of the similarity matrix S, $m_i = max_j S_{ij} \in R$. Then I take the softmax over the resulting vector $b = softmax\left(m\right) \in R^T$ and get an attention distribution over the context locations. Finally I use b to take a weighted sum of the context hidden states h_i, so the attended context vector is $\widetilde{h} = \sum_t b_t H_{:t} \in R^{2d}$. $\widetilde{h}$ is tiled T times across the column, thus $\widetilde{H} \in R^{2d*T}$

Finally, we combined the contextual embeddings and the attention vectors to get a complex representation for each position i.

$$G = \beta\left(H, \widetilde{U}, H \circ \widetilde{U}, H \circ \widetilde{H},\right) \tag{11}$$

For G, we also use a residual dense layer to aggregate the features as we do in the previous section to get a better representation.

Modelling Layer: This layer uses two bidirectional GRU to scan the attended representation. From our experiments, we find the modelling layer is extremely important to BiDAF. So we added two layers of bidirectional GRU (similar to the contextual embedding layer). M and N ($M \in R^{2d*T}$ and $N \in R^{2d*T}$) represent the output of these two layers respectively, capturing the interaction among the attended representation. Like what we do for previous layers, we also add residual dense layer.

Output Layer: This layer helps to provide the start and the end position of the answer to the question. We skipped the additional bidirectional LSTM for the end index prediction. And the softmax is performed on the logits from last layer to obtain the probability distribution of the start and end index over the entire paragraph.

3.4 Weighted-loss Bi-directional Attention Flow (BiDAF)

Because our goal is to get higher F1 and EM score, so I also modified the loss function to account more for those examples with lower F1 and EM. Before, our loss for a single example is

$$loss_i = CE_i^{start} + CE_i^{end} \tag{12}$$

We change the loss function to

$$Weighted_loss_i = (1 - F1 - EM) * loss_i \tag{13}$$

and

$$Weighted_loss_i = \frac{1}{N}\sum_{j=1}^{N} Weighted_loss_i \tag{14}$$

4 Results and Analysis

Table 1: Hyperparameter value

hyperparameter	value	hyperparameter	value
Learning rate	0.001 $\rightarrow$ 0.0001	Gradient clipping	5
Dropout(P_{keep})	0.8	Hidden state size	30
Fixed context size	400	Fixed question size	30
Batch size	100		

For our final experiment, the hyperparameters are listed in 1. The results are in 2 and it verifies our assumptions from the previous section. We can see by using BiDAF, the performance has a huge jump and by using our proposed loss, we get additional performance gain.

The BiDAF model performs the best on "when" questions and performs the worst on "why" question. This might be because "when" type of question have short length, "why" question have longer answer length.

And after analyzing the incorrectly predicted examples of the model, I find there're mainly four kinds of mistakes made by the model.

4.1 Incorrect Start Position

From figure 6, we could see the prediction of the start position is incorrect but the model almost gets the right answer. When calculating the start index distribution, we should have taken more phrases with similar meaning into account.

Table 2: Model result on dev set

MODEL	F1	EM
Residual RNN	44.365	35.402
Stack RNN	46.601	37.096
BiDAF	72.73	62.337
BiDAF(with residual)	73.578	63.311
weighted-loss BiDAF(with residual)	74.767	64.816
ensemble BiDAF(20500, 24000, 26500, 27500, 28000, 35500, 37500, 38500)	76.004	66.263
ensemble weighted-loss BiDAF		

```
          QUESTION: what type of materials inside the cabin were removed
to help prevent more fire hazards in the future ?
        TRUE ANSWER: flammable cabin and space suit materials
   PREDICTED ANSWER: space suit materials
    F1 SCORE ANSWER: 0.667
          EM SCORE: False
```

Figure 6: Incorrect start postion

4.2 Incorrect End Position

From figure 7, we could see the model often has a hard time finding the end position of the answer although the model can get the correct start position. And as shown in figure 2, we could see most of the answer length are short and concise(less than 10), so these redundant words might greatly impair the model performance. So we should constrain the length of answers and maybe use our model to also predict the length of answers. And we also need to build up a model that the probability distribution of the end location is conditioned on the start location probability distribution. I've tried using an additional bidirectional LSTM to predict the end position by also incorporating logits of start position(similar to the original paper[5]), but skip it for now due to time constraint to tune the model.

```
          QUESTION: site conditions , local regulations , economies of
scale and the availability of skilled tradespeople all affect what ?
        TRUE ANSWER: cost of construction
   PREDICTED ANSWER: cost of construction on a per square meter ( or per
 square foot ) basis for houses can vary dramatically based on site condi
tions , local regulations , economies of scale ( custom designed homes ar
e often more expensive to build
    F1 SCORE ANSWER: 0.158
          EM SCORE: False
```

Figure 7: Incorrect end postion

4.3 Incorrect Boundary

From figure 8, we could see the model almost has the correct answer but has a hard time handling the boundary of the answer(mistakenly omitted or included). Overall, the incorrect start/end position and incorrect boundary can be formalized as a problem that it's hard for the model to distinguish phrases that have similar meaning. And it's actually also a hard problem for human being. So the ground trught answers might also have inconsistency.

```
            QUESTION: how much larger would cicada predator populations be if cicad
a outbreaks occurred at 14 and 15 year intervals ?
          TRUE ANSWER: up to 2 % higher
    PREDICTED ANSWER: 2 %
     F1 SCORE ANSWER: 0.400
            EM SCORE: False
```

Figure 8: Incorrect boundary

4.4 Wrong prediction

From figure 9, we could see the prediction is completely wrong. There might be several reasons. One might be wrong attention. The true answer "independence day" and the predicted answer "secret life of pets" might be both associated with "pay for", but attention layer is not strong enough to remember the early part information "fox" of the question. So the problem here is attention mechanism is not strong enough and in the future exploration, we could try to build a mix attention layer by combining BiDAF, co-attention and self-attention together. And the second reason might be bad word embeddings. The pretrained embedding of each word might be wrong and leads to wrong understanding of each word. We could use char CNN or contextualized word embedding to alleviate this problem.

```
CONTEXT: (green text is true answer, magenta background is predicted start, red bac
kground is predicted end, _underscores_ are unknown tokens). Length: 108
20th century fox , lionsgate , paramount pictures , universal studios and walt disn
ey studios paid for movie trailers to be aired during the super bowl . fox paid for
 deadpool , x-men : apocalypse , independence day : resurgence and eddie the eagle
, lionsgate paid for gods of egypt , paramount paid for teenage mutant ninja turtle
s : out of the shadows and 10 cloverfield lane , universal paid for the secret life
 of pets and the debut trailer for jason bourne and disney paid for captain america
 : civil war , the jungle book and alice through the looking glass . [ citation nee
ded ]
            QUESTION: what famous july fourth holiday movie did fox pay to advertis
e a sequel of during the super bowl ?
          TRUE ANSWER: independence day
    PREDICTED ANSWER: secret life of pets
     F1 SCORE ANSWER: 0.000
            EM SCORE: False
```

Figure 9: Wrong prediction

5 Discussion and Future Work

We started from the baseline model, adding many basic modules like bi-attention flow layer, bi-GRU layers and residual dense layers to better capture the relation between the question representation and context representation. From our experiments, we find: 1. features do matter and good representations are extremely useful for prediction. 2. residual connection solves the problem of degeneration problem if the model is very deep. 3. weighted loss and/or focal loss can help get further improvement. Our future work includes: 1. adding conditional prediction of end position. 2. combining co-attention, biDAF and self-attention to get a mix attention representation. And we can do attention ensemble.

Acknowledgments

I would like to thank CS224N TAs for their insightful advice and timely feedback about improving the models. Also really thank you for the teaching team to promptly solve so many problems concerning codaLab to make such task still a really great experience. I also want to thank the classmates of CS224N 17-18 winter for creating such a competitive and exciting leader board.

References

[1] R-NET: Machine Reading Comprehension with Self-matching Networks*Microsoft Research* May 8, 2017

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

[3] DCN+: Mixed Objective and Deep Residual Coattention for Question Answering *arXiv:* 1711.00106

[4] Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016). Squad: 100,000+ questions for machine comprehension of text. *arXiv e-prints* , Jun. 2016.

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

[6] Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Yoshua Bengio. How to Construct Deep Recurrent Neural Networks. *arXiv*:1312.6026

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Deep Residual Learning for Image Recognition. *arXiv*:1512.03385

[8] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. *In EMNLP*, 2014

[9] Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *In ICLR2015*