Reading Comprehension using Bi-Directional Attention Network

Neelmani Singh
Stanford University
neelmani@stanford.edu

Pratik Kumar
Stanford University
pratikk@stanford.edu

Abstract

Reading comprehension, answering a query about a given context, is an important task in machine learning. This task is proven to be difficult as it involves modelling interactions between two separate pieces of information i.e. the context and the query. Attention system has been successful for reading comprehension where a small portion of the context is focused based on the query. We have used the architecture from Bidirectional Attention Flow(BiDAF) model, which was introduced by Seo et al.[1], to build a multi-layer architecture for this problem. Our model achieved F1 score of 72.925% and EM of 62.75% on Stanford Question Answering Dataset(SQuAD).

1 Introduction

Reading comprehension is the ability to process text, understand its meaning, and to integrate it with what the reader already knows. Applying machine learning to the reading comprehension task, also known as machine comprehension, has become popular. From a research perspective, this is an interesting task because it provides a measure of how well systems can 'understand' text. From a practical perspective, this task is useful in building an AI system so that you can understand any piece of text – like a class textbook, etc. Models designed for end to end machine comprehension must generate a relationship between the context and the query and should be able to pick the key words from the context that answers the query.

In this paper, we describe our approach for the machine comprehension problem. We have a baseline model[5] with GRU contextual embed layer, basic attention and fully connected ReLU network. Our architecture is inspired from the architecture used in Bidirectional Attention Flow(BiDAF)[1] and is built on top of baseline model. We experimented with various alternatives at each of the layers which is described in the subsequent sections.

2 Problem Definition – The SQuAD Challenge

Stanford Question Answering Dataset (SQuAD)[2] is a reading comprehension dataset. This means our model will be given a paragraph also called as context, as input. The goal is to answer the question correctly.

Let us formulate the problem a bit more formally. Given a context or passage which is a sequence of words of length N, $C = \{C_1, \dots C_N\}$ and a question or query which is again a sequence of words of length M, $Q = \{Q_1, \dots Q_M\}$, our model needs to predict a pair of indices $\{l_{start},\ l_{end}\}$ such that $1 \leq l_{start}, l_{end} \leq N$ which is the start index and end index of answer within the context.

3 Approach

In this section, we present our neural network architecture for end to end machine comprehension.

3.1 Architecture

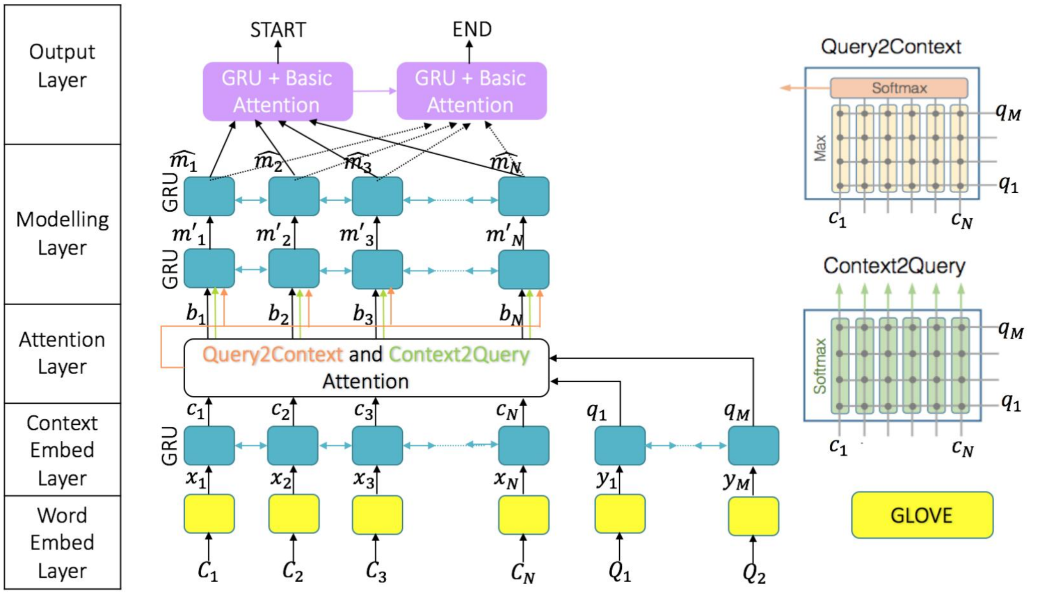

Figure 1: Architecture of our model

3.2 Architecture Details

Our reading comprehension model has a multi-layer architecture as shown in Figure. It is built on top of baseline architecture and is inspired by BiDAF architecture. It consists of the following layers:

Word Embed Layer

Word embed layer is the first layer which maps the words from the context and the query to a d-dimensional vector space. We use GloVe[3] word vectors to obtain the embedding of each word in the context and the query.

Let $Q = \{Q_1, \dots Q_M\}$ represent the query words and $C = \{C_1, \dots C_N\}$ represent the context words. Let $E \in \mathbb{R}^{|V|\, X\, d}$ be the GloVe embedding matrix. We get the word vectors $y = \{y_i, \dots y_M\}$ for the query words Q and $x = \{x_1, \dots x_N\}$ for the context words C using the embedding matrix E.

Context Embed Layer

The contextual embed layer refines the word embedding to encode the sequence or temporal information using the context from other words. This is done both for the context and the query. The embeddings from the previous layer are fed into a 1-layer bidirectional GRU which is shared between the context and the query.

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

The bidirectional GRU produces a sequence of forward hidden states ($\overrightarrow{c_i} \in \mathbb{R}^h$ for the context and $\overrightarrow{q_j} \in \mathbb{R}^h$ for the query) and a sequence of backward hidden states ($\overleftarrow{c_i} \in \mathbb{R}^h$ and $\overleftarrow{q_j} \in \mathbb{R}^h$). We concatenate the forward and backward hidden states to obtain context hidden states c_i and the question hidden states q_j respectively:

$$c_i = [\overrightarrow{c_i}; \overleftarrow{c_i}] \in \mathbb{R}^{2h}\ \forall\ i \in \{1, \dots, N\} \text{ and } q_j = [\overrightarrow{q_j}; \overleftarrow{q_j}] \in \mathbb{R}^{2h}\ \forall\ j \in \{1, \dots, M\}$$

Attention Layer

The attention layer creates a query aware feature vector for each word in the context. We use bidirectional attention in this layer. The main idea is that the attention should flow both ways i.e. from the context to the query and from the query to the context. We compute a similarity matrix $S \in \mathbb{R}^{NXM}$ which contains a similarity score S_{ij} for each pair (c_i, q_j) of context and query hidden states.

$$S_{ij} = w_{sim}^T \left[c_i; q_j; c_i \circ q_j\right] \in \mathbb{R}$$

Here $c_i \circ q_j$ is an elementwise product and $w_{sim} \in \mathbb{R}^{6h}$ is a weight vector.

Context to Query(C2Q) Attention:

$$\alpha_i = softmax\left(S_{i,:}\right) \in \mathbb{R}^M \ \forall\, i \in \{1, \dots, N\}$$

$$a_i = \sum_{j=1}^{M} \alpha_j^i q_j \in \ \mathbb{R}^{2h} \ \forall\, i \in \{1, \dots, M\}$$

Query to Context(Q2C) Attention:

$$m_i = \max_j S_{ij} \in \mathbb{R} \ \forall\, i \in \{1, \dots, N\}$$

$$\beta = softmax(m) \in \mathbb{R}^N$$

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

For each context location $i \in \{1, \dots, N\}$ we obtain the output b_i of the bidirectional attention layer by combining the context hidden state c_i, the C2Q attention output a_i and the Q2C attention output c' as $b_i = [c_i; a_i; c_i \circ a_i; c_i \circ c'] \in \mathbb{R}^{8h} \ \forall\, i \in \{1, \dots, N\}$

Modelling Layer

The modelling layer refines the query aware representations of context words by capturing the interaction among the context words conditioned on the query. It is different from context embed layer which just captures the interaction of context words independent of the query. We use 2 layers of bidirectional GRU.

$$\left\{\overrightarrow{m'_1}, \overleftarrow{m'_1}, \dots, \overrightarrow{m'_N}, \overleftarrow{m'_N}\right\} = biGRU(\{b_1, \dots, b_N\})$$

$$\{\overrightarrow{m_1}, \overleftarrow{m_1}, \dots, \overrightarrow{m_N}, \overleftarrow{m_N}\} = biGRU\left(\left\{[\overrightarrow{m'_1}; \overleftarrow{m'_1}], \dots, [\overrightarrow{m'_N}; \overleftarrow{m'_N}]\right\}\right)$$

$$\widehat{m_\iota} = [\overrightarrow{m_1}; \overleftarrow{m_1}]$$

Output Layer

The output layer gives the probability distribution for start and end pointers for the answer. Our output layer model is based on Answer Pointer component of 'Match-LSTM with Answer Pointer'[4]. This model helps condition the end pointer on start pointer of answer. M is matrix of output from Modelling layer where $M = [\widehat{m_1}, \widehat{m_2}, \dots, \widehat{m_N}]$

$$H_s = GRU(M) \ where \ H_s = [h_1^s, h_2^s, \dots, h_N^s]$$

$$a_s, \beta_s = BasicAttention(M, h_N^s)$$

$$H_e = GRU([a_s, M]) \ where \ H_e = [h_0^e, h_1^e, h_2^e, \dots, h_N^e]$$

$$a_e, \beta_e = BasicAttention(M, h_N^e)$$

Where $p^{start} = \beta_s$, $p^{end} = \beta_e$, $p_{predicted}^{start} = argmax(\beta_s)$ and $p_{predicted}^{end} = argmax(\beta_e)$.

4 Experiments

This section describes the details about dataset, various experiments carried out, training decisions based on the experiments, etc.

4.1 Dataset

SQuAD contains around 100k (question, context, answer) triplets. The context is extracted from Wikipedia and the answers are generated by humans. The length of question goes up to a length of 60 and context length up to 766. Context length has 99 percentile of 325.

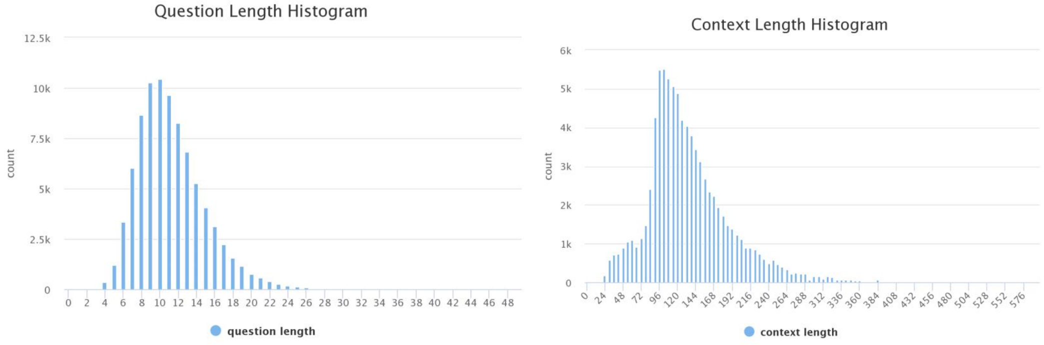

Figure 2: Question Length Histogram and Context Length Histogram

4.2 Training Details

We started with training the baseline model with defaults flags. We then tried various parameters and different network architectures as follows -

Embedding Size

We tried embedding size of 100, 200 and 300 on a small training set. We get a jump in performance from 100 to 200 and a slight jump from 200 to 300. Based on this, we used embedding size of 200 for most of our experiments. We used 300 for our final model.

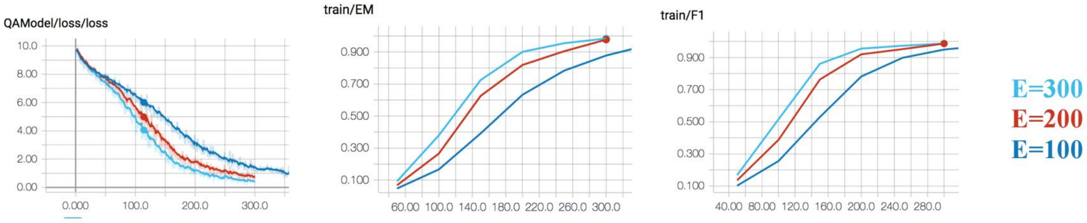

Figure 3: Comparison of performance of models with different embedding size

LSTM vs GRU

We tried changing the GRU cell with LSTM cell. The performance with LSTM cell was almost similar to what we got GRU. Since the training time increased by 10% for LSTM, we chose to go with GRU both in contextual embed layer and modelling layer.

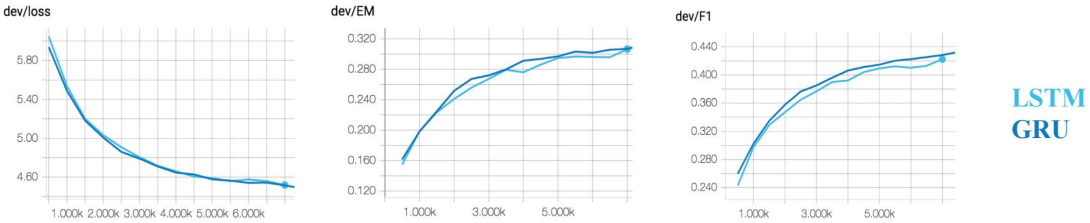

Figure 4: Comparison of performance of models with LSTM and GRU

Modelling Layer

We tried evaluating the performance of our model with 2 layers of bidirectional GRU vs 3 layers of bidirectional GRU. We do get slight improvement with an additional layer but since the training time went up by 30%, we planned to stick with 2 layers only.

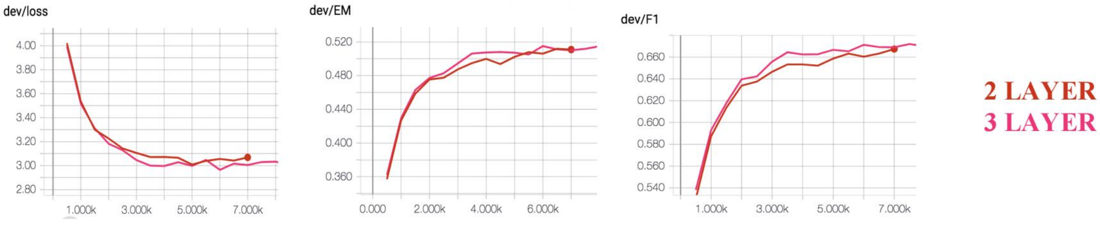

Figure 5: Comparison of performance of models with #layers in modelling layer

Based on the above experiments, we finalized on our model architecture and the various parameters for the model.

5 Results and Analysis

This section will describe the results of our models, metrics used to analyze them and some error analysis of our model

5.1 Metrics

Here are the different metrics that we have evaluated our results on:

Exact Match(EM) is the case where the system output matches the ground truth. **F1 Score** is the harmonic mean of precision and recall. **Incorrect Span** is the case where start position of answer is greater than end position. **No Match** is the case where none of the words match i.e. F1 is 0**. No Exact Match** is the case where some of the words don't match, i.e. EM is 0.

5.2 Results

Model	Exact Match	F1 Score	Incorrect Span	No Match	No Exact Match
Baseline	34.418	43.447	1752	4491	7341
BiDirectional Attention(BiDAF)	39.092	49.296	1474	4019	6944
BiDAF + Answer Pointer	45.08	56.367	1363	3338	6409
BiDAF + GRU Modelling Layer	60.757	71.650	324	2174	5037
BiDAF + GRU Modelling Layer + Answer Pointer	61.145	72.146	270	2200	5041

Table 1: Results on dev set for various experiments/models

Model	Dev EM	Dev F1	Test EM	Test F1
Baseline*	34.418	43.447	34.784	44.225
BiDAF(ours)	61.145	72.146	62.75	72.925
BiDAF(reference)[1]	67.7	77.3	68.0	77.3

Table 2: Comparison between baseline, our implementation and reference implementation

5.3 Error Analysis

To better understand our system, we used the metrics described in Section 5.1 on dev dataset apart from EM and F1 score. These metrics gave us some insight on how our model reacts on changes we made to the model.

The **incorrect span** selection reduced significantly after we replaced fully connected network with bidirectional GRU in the modelling layer. This shows that significant amount of information was lost or not learned in the fully connected network. This happens primarily because in fully connected network weights are not shared and hence it is not able to learn interactions among the word of the context, which is not the case for RNNs.

Addition of Bi Directional Attention mostly helped improving incorrect span selection. BiDAF helps prevent early summarization which enables the modeling layer learn richer interaction among words of context, hence improving the span selection.

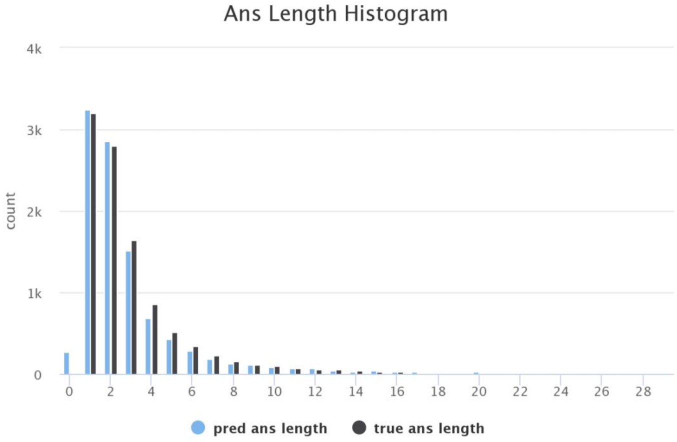

Figure 6: Comparison of answer length for predicted vs ground truth

The answer length histogram shows that most of the predicted answers are smaller in size. A value of 0 indicates incorrect span.

5.4 Qualitative Error Analysis

Attention attending to incorrect context

- **Context:** abc currently holds the broadcast rights to the academy awards , emmy awards (which are rotated across all four major networks on a year-to-year basis) , american music awards , disney parks christmas day parade , tournament of roses parade , country music association awards and the cma music festival . since 2000 , abc has also owned the television rights to most of the peanuts television specials , having acquired the broadcast rights from cbs , which originated the specials in 1965 with the debut of a charlie brown christmas (other peanuts specials broadcast annually by abc , including a charlie brown christmas , include it 's the great pumpkin , charlie brown and a charlie brown
- **Question:** what peanuts special is halloween-themed ?
- **Ground Truth:** it 's the great pumpkin
- **Prediction:** peanuts television specials

Over here the question has a word peanuts and therefore it attends more to the section of context having peanuts rather than the actual answer.

Special characters/out of vocabulary character not handled

- **Context:** wealth concentration is a theoretical [according to whom ?] process by which , under certain conditions , newly created wealth concentrates in the possession of _already-wealthy_ individuals or entities . according to this theory , those who already hold wealth have the means to invest in new sources of creating wealth or to otherwise leverage the accumulation of wealth , thus are the beneficiaries of the new wealth . over time , wealth condensation can significantly contribute to the persistence of inequality within society . thomas piketty in his book capital in the twenty-first century argues that the fundamental force for divergence is the usually greater return of capital (r) than economic growth (g) , and that larger fortunes generate higher returns [pp . 384 table 12.2 , u.s. university endowment size vs. real annual rate of return]
- **Question:** what do larger fortunes generate ?

- **Ground Truth:** higher returns
- **Predicted:** higher returns [pp

Over here the context around the actual answer has a special character ***[*** *which is not handled by our model.*

Long Answers are not handled correctly

- **Context:** british settlers outnumbered the french 20 to 1 with a population of about 1.5 million ranged along the eastern coast of the continent , from nova scotia and newfoundland in the north , to georgia in the south . many of the older colonies had land claims that extended arbitrarily far to the west , as the extent of the continent was unknown at the time their provincial charters were granted . while their population centers were along the coast , the settlements were growing into the interior . nova scotia , which had been captured from france in 1713 , still had a significant french-speaking population . britain also claimed rupert 's land , where the hudson 's bay company traded for furs with local tribes .
- **Question:** where did british settlers live ?
- **Ground Truth:** where did british settlers live ?
- **Prediction:** where did british settlers live ?

This example actually asserts what we have seen in Figure 6, which depicts that most of predicted answer spans are shorter in length. Here the answer has been cut short.

6 Future Work

Character Embedding can be used along along with word embedding. It should be able to help with out of vocabulary and special characters that we have seen in the error analysis.

Additional Features like POS tags can be used.

Dynamic Programming based span selection can be used where we can enforce that end pointer is greater than

Other Attention techniques like self-attention, co-attention can be used.

Acknowledgments

We would like to thanks the instructor and teaching staff for the incredible support with the default project and quickly solving any problems with codalab, leaderboards, etc.

References

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

[2] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. *Squad: 100,000+ questions for machine comprehension of text. In Empirical Methods in Natural Language Processing* (EMNLP), 2016.

[3] Pennington, Jeffrey, Richard Socher, and Christopher D Manning. 2014. *Glove: Global vectors for word representation. Proceedings of the Empiricial Methods in Natural Language Processing* (EMNLP 2014) 12.

[4] Shuohang Wang and Jing Jiang. 2016. *Machine comprehension using match-lstm and answer pointer*. CoRR, abs/1608.07905.

[5] Default Project Handout, CS224N - *http://web.stanford.edu/class/cs224n/default_project/default_project_v2.pdf*

[6] Natural Language Computing Group, Microsoft Research Asia. *R-Net: Machine Reading Comprehension with Self-Matching Networks.*