Bifocal Perspectives for Machine Comprehension

Adam Abdulhamid
Dept. of Computer Science
Stanford University
adama94@stanford.edu

Andrew Lim
Dept. of Computer Science
Stanford University
alim16@stanford.edu

Pavitra Rengarajan
Dept. of Computer Science
Stanford University
pavitrar@stanford.edu

Abstract

With the introduction of end-to-end trainable neural models, several tasks across the field of natural language processing have seen enormous success, and with it several different models have been proposed and proven successful in the question answering domain. In this paper, we introduce the Bifocal Perspectives Model. It draws from several successful models, combining different aspects in novel ways, namely a bidirectional attention mechanism paired with a perspective matching layer. Overall, on the SQuAD test dataset our model achieves 52.48 F1 and 39.47 EM.

1 Introduction

Question answering (QA) has become an integral task in natural language processing over the past several years, especially with the advance of deep neural architectures and models. In the past, two types of datasets existed for training and testing QA systems. The first type were human curated and often very high in quality [1], but lacked the size to train sufficiently complex neural architectures that achieve state-of-the-art performance on many other natural language tasks today. The second type were partly synthetic, which allowed for much larger datasets. Although these datasets were much larger and could be used for training expressive neural models, they were quite different than natural language humans would produce, and therefore the models trained from these datasets had limited performance on questions using natural human language [2].

The SQuAD (Stanford Question Answering dataset) dataset from Rajpurkar et al. [3] helps bridge the gap between these previous QA datasets. The SQuAD dataset consists of 100,000+ hand curated paragraph, question, answer triplets from 500+ Wikipedia articles. The huge dataset size along with the high quality natural human language allows very complex and expressive models to be trained successfully.

In this paper we introduce the Bifocal Perspectives Model, a variant of the Multi-Perspective Context Matching (MPCM) architecture proposed by Wang et al. [6]. It incorporates a bidirectional attention mechanism inspired by that proposed in the BiDirectional Attention Flow (BiDAF) network by Seo et al. [4]. Overall, a single Bifocal Perspectives model achieves an F1 score of 52.48 and EM score of 39.47.

2 Related Work

With the release of the SQuAD dataset in 2016, there have already been several papers that achieve quite good results, many of them based on variations of the neural attention mechanism by Weston et al. [7]. There are three primary papers related to the QA task our model tackles. All of them are based on end-to-end trainable neural networks with variations on encoding, decoding, and the attention mechanism.

Xiong and Zhong [8] introduced the Dynamic Coattention Network (DCN), where they proposed a novel attention mechanism that attends to the question and context paragraph simultaneously. The DCN architecture then combines these attention mechanisms for downstream use in the overall network architecture.

Seo et al. [4] introduced the BiDAF mechanism, which proposes a similar concept of bi-directional attention. For the BiDAF network, the previously mentioned attention mechanism is extended to flow in both directions, namely from context to query, and also from query to context. Also introduced in the BiDAF network was the ability to pass the attention vector at each time step to further layers in the model, alleviating the problem of early summarization that some attention mechanisms fall subject to.

Wang et al. introduce the MPCM model [6], which compares each paragraph word to the question from multiple perspectives. The MPCM model is based off an assumption that a given span within a passage is more likely to be the correct answer if it is similar to the question. Each perspective learns to attend differently between the paragraph and question, which allows the group of collective perspectives to provide a comprehensive signal of similarity between paragraph words and the question.

3 Data

3.1 Dataset

As mentioned above, the dataset consists of 100,000+ hand curated paragraph, question, answer triplets from 500+ Wikipedia articles [3]. The paragraphs and questions are simply lists of words, but the answers are pairs of indices. The first represents the start index of the answer in the original paragraph, while the second represents the end index of the answer in the paragraph.

The very high quality nature of these paragraph, question, answer triplets along with the much higher quantity of examples than we previously had allows us to train very expressive neural models that have been shown to perform quite well. After plotting the length distributions, we note that almost all the paragraphs are less than 400 words and almost all the questions are less than 30 words.

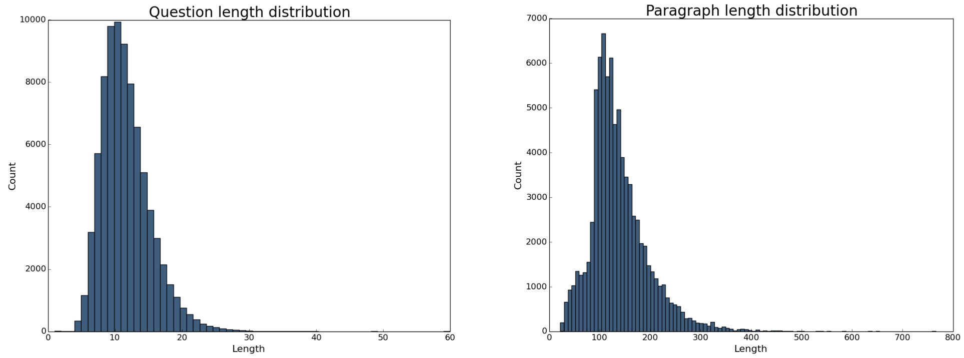

Figure 1: Distribution of (a) question lengths and (b) paragraph lengths.

3.2 Evaluation

Evaluation is scored on two primary metrics, F1 score and EM (exact match) score. EM is the fraction of answers for which the model produces a word for word match with the human responses, while F1 measures accuracy on overall words retrieved.

To ensure no bias in evaluation, the SQuAD team has kept from the public a small test set used solely for evaluation on final models. The entire model, along with model parameters, must be submitted where they will evaluate performance on the test set, and post to their public leader board. For reference, at the time of writing this paper, the best model on the public leader board obtains an F1 score of 84.006, with an EM score of 76.922.

It can be useful to note that human performance on this question answering dataset is F1 of 91.221 and EM of 82.304 [3].

4 Approach

4.1 Overview

Our bifocal perspective model is based primarily off the MPCM model but incorporates an additional, simplified variant of the BiDAF network's attention flow layer that is not present in the original MPCM model.

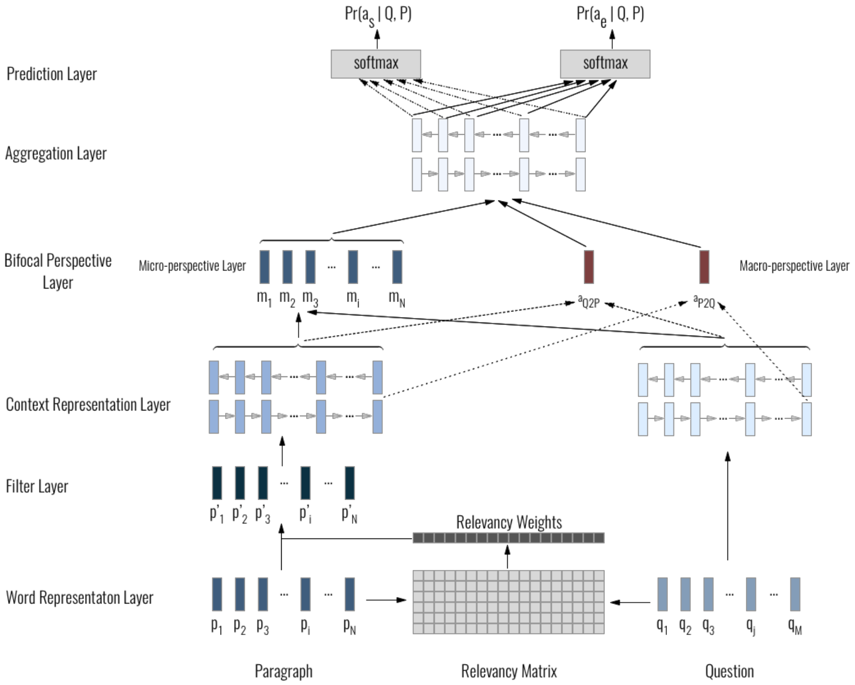

Figure 2: Architecture for Bifocal Perspective Model.

4.2 Architecture

For each question Q and corresponding paragraph P, our model predicts the answer span by maximizing the probabilities of the start index and end index of the answer, with the assumption that these probabilities are independent. We estimate the probability distributions through the following layers.

4.2.1 Word Representation Layer

This first layer represents each word in the question and each word in the paragraph as a 300-dimensional vector word embedding. We begin with GloVe word embeddings that were pre-trained on Wikipedia 2014 and Gigaword 5, with a vocabulary size of around 400,000 lowercased words. Words with capitalized characters and other such out-of-vocab words are initialized with random values. We train these word vectors to increase performance on our reading comprehension task. This layer outputs a sequence of word vectors that corresponds to the question representation, $Q : [\boldsymbol{q}_1, ..., \boldsymbol{q}_M]$, and a sequence of word vectors that corresponds to the paragraph representation, $P : [\boldsymbol{p}_1, ..., \boldsymbol{p}_N]$.

4.2.2 Filter Layer

Although the paragraphs in our training set were anywhere up to 750 words in length, only a small segment of the paragraph was needed to answer the question, with mean answer length of 3.4. Thus, the filter layer's core purpose is to reduce the amount of redundant information in the paragraph by assigning more importance to words in the paragraph that are more relevant to the question. To this end, we calculate a relevancy matrix by computing the relevancy degree between each word in the question and each word in the paragraph. For a given question word $\boldsymbol{q}_i \in Q$ and paragraph word $\boldsymbol{p}_j \in P$, we define the relevancy degree between each word pair $r_{i,j}$ as the cosine similarity: $r_{i,j} = \frac{\boldsymbol{q}_i^\intercal \boldsymbol{p}_j}{\|\boldsymbol{q}_i\| \cdot \|\boldsymbol{p}_j\|}$. Then, the relevancy degree for each paragraph word corresponds to the maximum relevancy degree between that paragraph word and all of the question words: $r_j = \max_{i \in M} r_{i,j}$. Each paragraph word vector is filtered by this relevancy degree and then passed to the next layer, where $\boldsymbol{p'}_j = r_j \cdot \boldsymbol{p}_j$. This layer outputs a sequence of word vectors that corresponds to the question representation, $Q : [\boldsymbol{q}_1, ..., \boldsymbol{q}_M]$, and a sequence of word vectors that corresponds to the updated filtered paragraph representation, $P : [\boldsymbol{p'}_1, ..., \boldsymbol{p'}_N]$.

4.2.3 Context Representation Layer

The context representation layer serves as the core encoding layer for our model. In this layer, we incorporate the time element by using a bidirectional long short-term memory network (BiLSTM) to encode contextual embeddings for each question word. Similarly, we apply a separate BiLSTM to encode contextual embeddings for each paragraph word. With a state size of h, this layer outputs h-dimensional hidden states in the forward direction and h-dimensional hidden states in the backward direction, for each word in both the question and the paragraph. We retrieve the contextual embeddings for each question word by applying a BiLSTM as follows:

$$\begin{aligned} \overrightarrow{\boldsymbol{h}}_i^q &= \overrightarrow{LSTM}(\overrightarrow{\boldsymbol{h}}_{i-1}^q, \boldsymbol{q}_i) && \text{for } i = 1, ..., M \\ \overleftarrow{\boldsymbol{h}}_i^q &= \overleftarrow{LSTM}(\overleftarrow{\boldsymbol{h}}_{i+1}^q, \boldsymbol{q}_i) && \text{for } i = M, ..., 1 \end{aligned}$$

Similarly, we apply a BiLSTM to the paragraph as follows:

$$\begin{aligned} \overrightarrow{\boldsymbol{h}}_j^p &= \overrightarrow{LSTM}(\overrightarrow{\boldsymbol{h}}_{j-1}^p, \boldsymbol{p'}_j) && \text{for } j = 1, ..., N \\ \overleftarrow{\boldsymbol{h}}_j^p &= \overleftarrow{LSTM}(\overleftarrow{\boldsymbol{h}}_{j+1}^p, \boldsymbol{p'}_j) && \text{for } j = N, ..., 1 \end{aligned}$$

We also define the following:

$$\begin{aligned} \boldsymbol{h}_i^q &= [\overrightarrow{\boldsymbol{h}}_i^q; \overleftarrow{\boldsymbol{h}}_i^q] \\ \boldsymbol{h}_i^p &= [\overrightarrow{\boldsymbol{h}}_i^p; \overleftarrow{\boldsymbol{h}}_i^p] \end{aligned}$$

4.2.4 Bifocal Perspective Layer

4.2.4.1 Macro-perspective Layer

The macro-perspective layer is structured similarly to the BiDAF attention flow layer [4] but employs an attention mechanism more similar to that present in the Stanford Attentive Reader presented by Chen et al. [2]. The paragraph-to-question (P2Q) attention identifies the question words which are most relevant for each paragraph, and the question-to-paragraph (Q2P) attention identifies the paragraph words that are most similar to the question and should be focused on as likely answer candidates.

More concretely, for P2Q, we obtain alignment weights for a particular question contextual embedding $\boldsymbol{h}_i^q$ and the paragraph representation $\boldsymbol{h}_N^p$, which are used to compute the P2Q attention vector $\boldsymbol{a}_{\texttt{P2Q}}$ as follows:

$$\begin{aligned} \alpha_i &= \texttt{softmax}_i({\boldsymbol{h}_N^p}^T \boldsymbol{W}^1 \boldsymbol{h}_i^q) \\ \boldsymbol{a}_{\texttt{P2Q}} &= \sum_i \alpha_i \boldsymbol{h}_i^q \end{aligned}$$

Likewise for Q2P, we obtain alignment weights for a particular paragraph contextual encoding $\boldsymbol{h}_i^p$ and the question representation $\boldsymbol{h}_M^q$, which are used to compute the Q2P attention vector $\boldsymbol{a}_{\texttt{Q2P}}$ as follows:

$$\alpha_i = \texttt{softmax}_i({\boldsymbol{h}_M^q}^T \boldsymbol{W}^2 \boldsymbol{h}_i^p)$$
$$\boldsymbol{a}_{\texttt{Q2P}} = \sum_i \alpha_i \boldsymbol{h}_i^p$$

4.2.4.2 Micro-perspective Layer

The micro-perspective layer is the main attention layer within our model. In this layer, we define micro-perspectives on the question and compare each paragraph embedding with the perspectives defined for the question. We begin by constructing an l-dimensional vector $\boldsymbol{m}$, where each element $m_k \in \boldsymbol{m}$ corresponds to a value from the k-th perspective and is calculated using a weighted cosine similarity metric f_m where $f_m(\boldsymbol{h}_s, \boldsymbol{h}_t, \boldsymbol{W})_k = \texttt{cosine}(\boldsymbol{W}_k \circ \boldsymbol{h}_s, \boldsymbol{W}_k \circ \boldsymbol{h}_t)$.

We employ the full-matching strategy of Wang et al. [6] to compare each paragraph contextual embedding with the question. Each forward embedding of the paragraph is compared with the forward representation of the question, and each backward embedding of the paragraph is compared with the backward representation of the question. Intuitively, this matching strategy proves to be particularly useful when a segment of the question matches with either the left or the right context. Formally, we represent the full-matching micro-perspectives as follows:

$$\overrightarrow{\boldsymbol{m}}_j^{full} = f_m(\overrightarrow{\boldsymbol{h}}_j^p, \overrightarrow{\boldsymbol{h}}_M^q; \boldsymbol{W}^3)$$
$$\overleftarrow{\boldsymbol{m}}_j^{full} = f_m(\overleftarrow{\boldsymbol{h}}_j^p, \overleftarrow{\boldsymbol{h}}_1^q; \boldsymbol{W}^4)$$

We concatenate these matching vectors $\boldsymbol{m}_j = [\overrightarrow{\boldsymbol{m}}_j^{full}; \overleftarrow{\boldsymbol{m}}_j^{full}]$ in order to get the matching vector for each position of the paragraph.

4.2.5 Aggregation Layer

The aggregation layer serves as the primary decoding layer for our model. We concatenate the full-matching vectors $\boldsymbol{m}_j$ from the perspective layer with $\boldsymbol{a}_{\texttt{Q2P}}$ and $\boldsymbol{a}_{\texttt{P2Q}}$ from the macro-perspective layer before running a BiLSTM on this concatenated representation to compute the aggregation vector at each time step. The aggregation layer effectively allows different paragraph words to interact with neighboring words and allows our model to capture these semantic interactions and syntactic dependencies.

4.2.6 Prediction Layer

For the final layer, we run two separate feed-forward neural networks over all hidden states of the aggregation layer to get un-normalized scores for each index of a paragraph. These scores are then passed through a softmax operation to generate probability distributions over the start and end index, where we select the maximum out of each to determine the start index and end index of our predicted answer.

5 Experiments & Results

5.1 Baseline Model

We began with a simple baseline model, upon which we conducted experiments to incrementally build our sophisticated end-to-end deep neural network model. Our baseline model consisted of three main layers: an encoding layer, a unidirectional global attention layer, and a decoding layer. Our encoding process consists of a separate BiLSTM for the context and the question; the question BiLSTM outputs the two end hidden states concatenated together, and the context BiLSTM outputs all of the intermediate hidden states as well. Our unidirectional global attention layer computes Q2P attention from the paragraph to the question as described in detail above. Our decoding process consists of a linear layer over all intermediate states followed by the softmax operation to get probability distributions, from which we select the maximum of each to be the start index and the end index. Our baseline model had hidden state size of 100, minibatch size of 32, learning rate of 0.001, and included learning rate annealing with 100 decay steps and a decay rate of 0.96. After the first epoch, this model yielded an F1 of 19.63, and EM of 16.00.

5.2 Macro-perspective Layer

We use bidirectional sequence attention to improve performance of our baseline model. In addition to the Q2P attention, we also compute P2Q attention in the opposite direction. We once again consider hidden state size of 100, minibatch size of 32, learning rate of 0.001, and learning rate annealing with 100 decay steps and decay rate of 0.96. The incorporation of bidirectional attention allows us to achieve an F1 of 25.06 and EM of 15.00 after the first epoch.

5.3 Filter Layer

The key insight of the filter layer is that much of the paragraph contains redundant information, and only a small segment of the paragraph is needed to answer the question. We use the relevancy matrix to filter out redundant information from the passage, such that if a word in the paragraph is more relevant to the question, more of its information is considered in subsequent steps. With hidden state size of 100, minibatch size of 32, learning rate 0.001, and learning rate annealing with 100 decay steps and decay rate of 0.96, we see an F1 of 38.22, and EM of 24.00 after the first epoch.

5.4 Micro-perspective Layer

In addition to our macro-perspective layer, we added a micro-perspective layer with full-matching perspectives based on Wang et al. [6]. With 50 perspectives, hidden state size of 100, minibatch size of 32, learning rate 0.001, and learning rate annealing with 100 decay steps and decay rate of 0.96, we see an F1 of 40.98 and EM of 29.18 after the first epoch.

5.5 Hyperparameter Tuning

After implementing the crux of the architecture, we performed a series of experiments to tune hyperparameters including dimensionality of word vectors, method of training embeddings, learning rate, annealing rate, hidden state size, paragraph maximum length, question maximum length, minibatch size, and dropout.

Given the plots of question lengths and paragraph lengths shown in Figure 1, we decided to use cutoffs of 28 for maximum question length and 300 for maximum paragraph length.

All of the aforementioned experiments used the initial version of the starter code provided, training embeddings where OOV words were initialized with zero embeddings. However, when training embeddings using the updated starter code with random initialization of OOV words, basic experiments showed approximately a 2 percent increase in F1 and EM scores.

When training our model on 300-dimensional GloVe vectors rather than 100-dimensional GloVe vectors, we see a significant gain: with 300-dimensional embeddings, we saw an F1 of 53.8 and an EM of 40 after just one epoch, whereas with 100-dimensional embeddings we saw the F1 plateau at around 51 after 5 epochs.

6 Discussion

6.1 Quantitative Analysis

As mentioned above, we performed a sequence of experiments using models that incrementally extended each other starting from our original baseline and ending with our bifocal perspective model. The following table presents a comparison of F1 and EM scores on the validation set after one epoch for these intermediate experiments:

Model	F1	EM
Baseline	19.63	16.00
Macro-perspective Layer	25.06	15.00
Filter Layer	38.22	24.00
Micro-perspective Layer	40.98	29.18
Tuned Bifocal Perspective Model	58.39	44.42

Table 1: Comparison of F1 & EM scores on validation set.

On our final model submission, we used the following hyperparameters: word embedding size of 300, hidden state size of 100, batch size of 32, dropout rate of 0.2, question maximum length of 28, paragraph maximum length of 300, learning rate of 0.001, and annealing with 100 decay steps and decay rate of 0.96.

With this model trained for 5 epochs, on the dev set we achieve an F1 score of 51.41 and EM score of 37.88. On the test set, we achieve an F1 score of 52.48 and EM score of 39.47.

6.2 Qualitative Analysis

We conduct further analysis of our model's performance on the validation set. Figure 3(a) shows the performance of our model on various types of questions. We sampled different questions using a regular expression match over various common question words. In total, our sampling coverage is 80.18% of the 4284 total questions. We see that our model performs best on ”when” and ”where” questions, likely due to the fact that it is easier to identify temporal expressions and locations based on boundary words such as ”in,” ”at,” and ”during.” Our model, however, performs worse on ”what” and ”why” questions, which are more difficult since their answers are often longer phrases with more variety of structure. Figure 3(b) shows that performance in terms of F1 and EM both drop as the answer length increases, which is not surprising as we expect longer answers to be more difficult to determine exactly. We also note that our model performs best on questions with answers of length 4, which is understandable given that the average answer length in the training set is 3.4 words. Thus, our model has learned during training that a common correct answer span is 4 words in length.

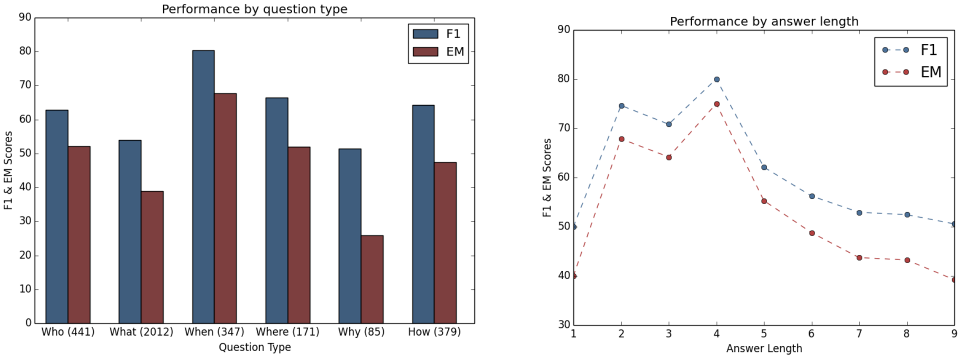

Figure 3: Performance (a) by question type and (b) by answer length.

Figures 4 and 5 provide some visual plots of the prediction distributions over start and end words, as well as visualization of our attention mechanism for the question and paragraph. In Figure 4(a), we see that our predictions are peaked on the correct words, namely the years corresponding to the correct answer. In Figure 5(a), we see that the P2Q attention indicates that the word ”When” is very important, which is intuitive given that the question word often specifies the type of answer we are looking for. In the Q2P attention, we see there is attention on the word ”president”, likely because this word occurs in the question as well, and also on the years themselves which indicates that the attention mechanism is effectively focusing on salient words.

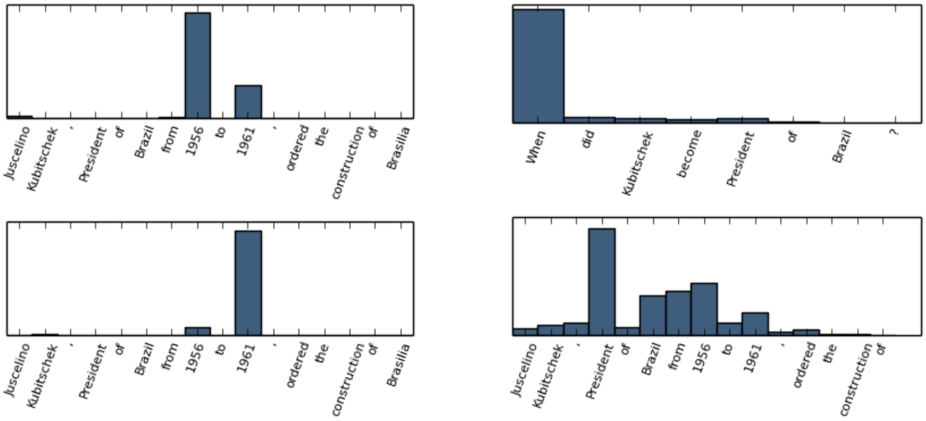

Figure 4: Well-learned example (a) prediction distributions and (b) attention vectors.

Below is an example where our model's predictions are incorrect, possibly due to the confusing nature of the question. As seen in Figure 5(b), the first word is ”Where”, but the answer is not a location, but rather an idea about where a group of people had placed their focus. We see in Figure 5(a) that our probability distributions are quite scattered, and in Figure 5(b) that our attention mechanism fails to identify important words in the paragraph.

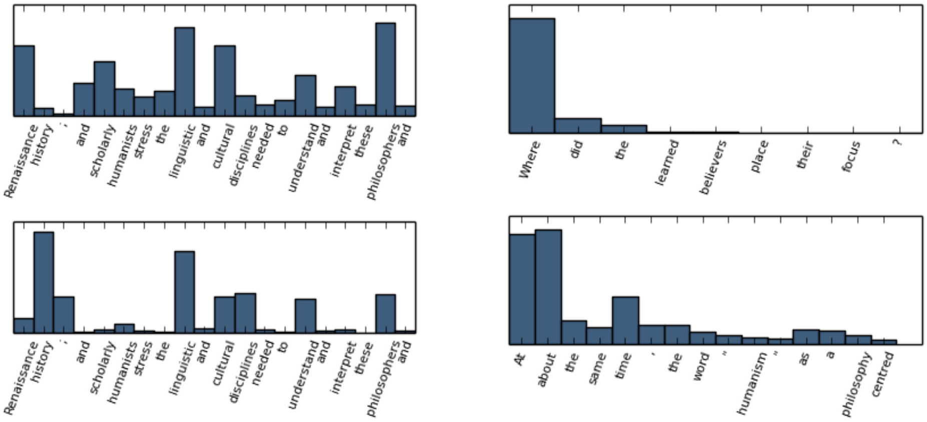

Figure 5: Poorly-learned example (a) prediction distributions and (b) attention vectors.

7 Future Work

In this paper, we considered a variety of factors that altered performance of our model. One of the most important of these was our attention mechanism, which combined bidirectional sequence attention with full matching perspectives. When we included an additional matching technique from the MPCM model, namely max-pooling perspectives, we did not see significant increases in F1/EM, but it would be interesting to see whether additional tuning would improve its performance. Wang et al. propose the Bilateral Multi-Perspective Matching (BiMPM) model [5], which is similar to MPCM but incorporates additional matching techniques. Additional work could be done to compare the relative impact of our macro-perspective layer as compared to BiMPM's attentive-matching technique. Additionally, we hypothesize that we could ensemble our model and could likely increase our F1 and EM scores by a couple percentage points.

References

[1] Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D. Manning. Modeling biological processes for reading comprehension. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[2] Danqi Chen, Jason Bolton, and Christopher D. Manning. A thorough examination of the cnn/daily mail reading comprehension task. *CoRR*, abs/1606.02858, 2016.

[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] Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *CoRR*, abs/1611.01603, 2016.

[5] Zhiguo Wang, Wael Hamza, and Radu Florian. Bilateral Multi-Perspective Matching for Natural Language Sentences. *ArXiv e-prints*, February 2017.

[6] Zhiguo Wang, Haitao Mi, Wael Hamza, and Radu Florian. Multi-perspective context matching for machine comprehension. *CoRR*, abs/1612.04211, 2016.

[7] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *CoRR*, abs/1410.3916, 2014.

[8] Caiming Xiong, Victor Zhong, and Richard Socher. Dynamic coattention networks for question answering. *CoRR*, abs/1611.01604, 2016.