Dynamic Coattention with Sentence Information

Alex Ruch
aruch@stanford.edu

Abstract

Machine reading comprehension involves teaching a computer to read a document and then answer questions related to that document. State of the art results on this task are dominated by Neural Networks. Several of the most successful of these have tended to use some sort of attention mechanism to capture relevant information from the document and the question at the same time. We explore some of the parameter choices made by one recent paper and attempt to extend the results by creating a hierarchical encoder based on sentence information.

1 Introduction

The recently published SQuAD dataset has become a great resource for doing machine comprehension with deep networks. It includes over 100,000 (document, question, answer) triplets which allows for fairly complex architectures to be fit. Herein we re-implement a recent model that performs well on the SQuAD dataset using an attention mechanism encoder. The attention mechanism allows for the document to be weighted by the question.

2 Model

The base model follows very closely the model outlined in Dynamic Coattention Networks for Question Answering [1]. Here we outline the base model – noting the minor changes that were made – and then discuss the proposed extension to the model. We try to follow the same notation in [1] for clarity.[1]

2.1 Base Encoder

Given a document and question represented as word vectors: $(x_1^D, x_2^D, \ldots, x_m^D)$ and $(x_1^Q, x_1^Q, \ldots, x_n^Q)$. We first encode the document using a bi-directional LSTM (BiLSTM) $d_t = [\text{LSTM}_{\text{fw}}(d_{t-1}, x_t^D); \text{LSTM}_{\text{bw}}(d_{t+1}, x_t^D)]$ to get a document matrix $D = [d_\varnothing, d_1, d_2, \ldots, d_m] \in \mathbb{R}^{2\ell \times (m+1)}$. Here are the first changes to the network with a BiLSTM in place of a LSTM and the sentinel vector at the beginning rather than end of the document. The former decision was made due to the preponderance of answers towards the beginning of the document (see Figure 5). It seems reasonable to include more context from later in the passage in the representation of the beginning vectors in the context paragraph. Including the sentinel vector at the beginning was made to ease some of the code complexity, and has no impact on the later steps of the model as long as we remove the sentinel vector before the final step in the encoding.

As in the original model we encode the question with the same BiLSTM $q_t = [\text{LSTM}_{\text{fw}}(q_{t-1}, x_t^Q); \text{LSTM}_{\text{bw}}(q_{t+1}, x_t^Q)]$ to get a question matrix $Q' = [d_\varnothing, q_1, q_2, \ldots, q_n] \in \mathbb{R}^{2\ell \times (n+1)}$. With the final question encoding being $Q = \tanh(W^{(Q)}Q' + b^{(Q)}) \in \mathbb{R}^{2\ell \times (n+1)}$.

[1] We use row vectors where the column should be

We skip the discussion for the document level coattention, C^D, because it is calculated similarly to the sentence level coattention mechanism discussed in 2.3, with S replaced by D, and it is is formulated exactly as in the original paper.

Finally, we take the original document encoding concatenated with the coattention representation $U' = [D; C^D]$ and apply a BiLSTM over that to get the final encoding for the document as in [1].

2.2 Decoder

We use the encoded representation of the document U, to predict the start and end point of the span. We implemented the same decoder as in the original paper, but with no early stopping. Every question gets 4 iterations through the Highway Maxout Network.

2.3 Sentence Level Coattention

We explored the possibility of including a sentence level encoding of the document to attempt to increase the accuracy of the model. The cleanest way to implement a sentence level encoding would be to process each sentence in isolation, and then combine them into a document. However, due to the presence of documents with many very short sentences and some with several very long sentences, creating an input flexible enough to handle the disparate sentence number and length requirements was infeasible. Instead we used a similar encoding to the original document, but with a different LSTM. Given a document of word embeddings as before, $(x_1^D, x_2^D, \ldots, x_m^D)$ we calculate $s_t^d = \text{BiLSTM}(s_{t-1}^d, s_{t+1}^d, x_t^D)$. To get the sentence level information, we first processed sentence level tokens using the nltk in Python. We then extracted the forward portion of the LSTM at the final word token of every sentence and the backwards portion of the LSTM from the first word of every sentence to get a sentence encoding $S \in \mathbb{R}^{2\ell \times k}$ where k is the number of sentences in the document (see Figure 1).

As before we encode the question with the same BiLSTM $q_t = [\text{LSTM}_{\text{fw}}(q_{t-1}, x_t^Q); \text{LSTM}_{\text{bw}}(q_{t+1}, x_t^Q)]$ to get a question matrix $Q' = [d_\varnothing, q_1, q_2, \ldots, q_n] \in \mathbb{R}^{2\ell \times (n+1)}$. With the final question encoding being $Q = \tanh(W^{(Q)}Q' + b^{(Q)}) \in \mathbb{R}^{2\ell \times (n+1)}$.

Finally we can calculate a sentence level coattention through the same methodology as word level coattention: $L = S^T Q \in \mathbb{R}^{(k+1)\times(m+1)}$. We then compute the $A^Q = \text{softmax}(L)$ and $A^S = \text{softmax}(L^T)$, where the softmax is computed row-wise. Intuitively, for A^Q the softmax is indicating the most similar sentences in the document for each word in the question. Similarly A^S gives us the most relevant words in the question for each sentence in the document. We then proceed to calculate $C^Q = SA^Q$ and finally $C^S = [Q; C^Q]A^S$ to get the attention context, in light of the question, for each sentence (Figure 1. C^S is then expanded to the length of the document by giving each word in the document the sentence vector from C^S that the word is a part (i.e. words from the first sentence all get C_1^S). We implemented two methods for including C^S in the model and have two options for future research. The first method of including sentence level information in the document was just to concatenate it with D and C^D as before, so that $[D; C^D; C^S]$ is fed into the final encoding step (Figure 3). The second method we employed was to treat it as a very naive mask, and first the elementwise product of C^D and $\sigma(C^S)$. The intuition behind the second method was to see if we could help the final document encoder keep information around longer in a similar fashion as the LSTM cells themselves do.

2.4 Model Parameters

Document Length: The context paragraphs length distribution peaks around 600 and can get quite long (Figure 5). However, the answer to the questions tend to occur very early on in the document with only a few in the train/dev set occurring past the 600th token in a paragraph. Since we only need the portion of the document with the answer in it to have a chance at suggesting the correct span, 600 seems a reasonable choice for document length.

Embeddings: We used the GloVe word embeddings for the Common Crawl 840B dataset and explored initializing the embeddings of unseen words to zero or to a random vector. Intuitively using a random vector would help training learning, but should not generalize well because the random vectors are meaningless. This would cause the model to overfit to the specific random initialization

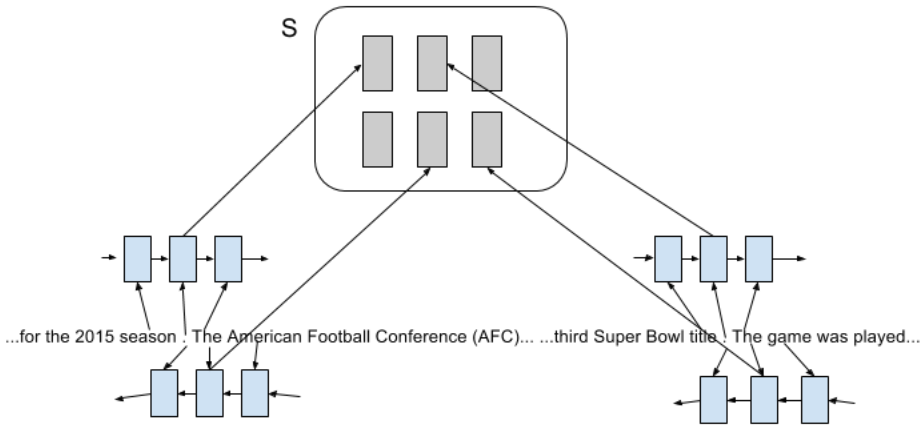

Figure 1: Sentence Encoding Mechanism

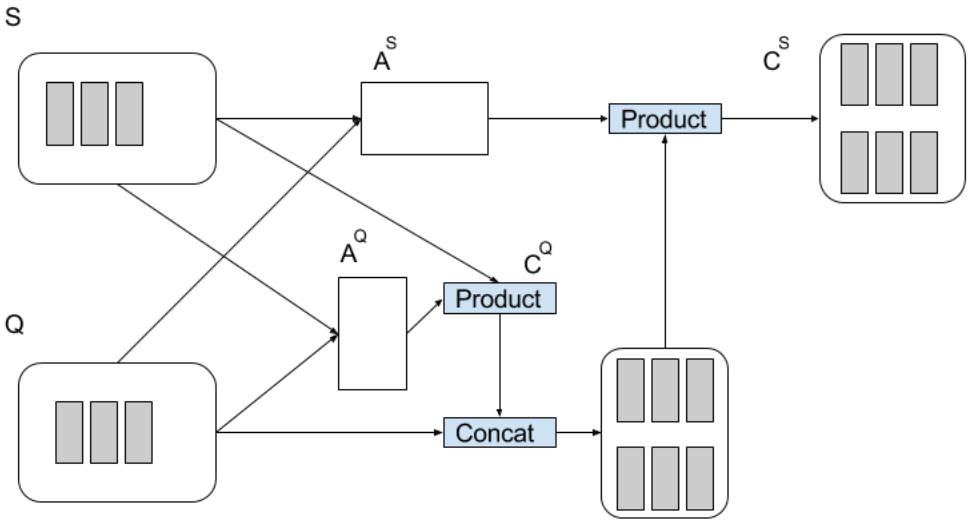

Figure 2: Coattention mechanism.

instead of learning to handle unknowns gracefully. We also observed this empirically, seeing poor development set metrics for and chose to use zero word embeddings for any tokens not found in the Common Crawl vocabulary.

Dropout Rate: While the authors in [1] investigated many model parameters there was no or limited discussion on the choice of hidden state size and dropout rate. We tested dropout rates of .15, .3, and .5 and found limited over-fitting at .3 with a state size of 200, and so used that value for the rest of the tests.

Hidden Size: We tested hidden state sizes of 150, 200, and 250 and found 200 to perform the best. A state size of 150 did not perform very well and a state size of 250 led to over-fitting very quickly (Figure

3 Results

The best model results we obtained were from the standard coattention model, unfortunately neither of the sentence based models outperformed this baseline. The naive masking approach yielded very

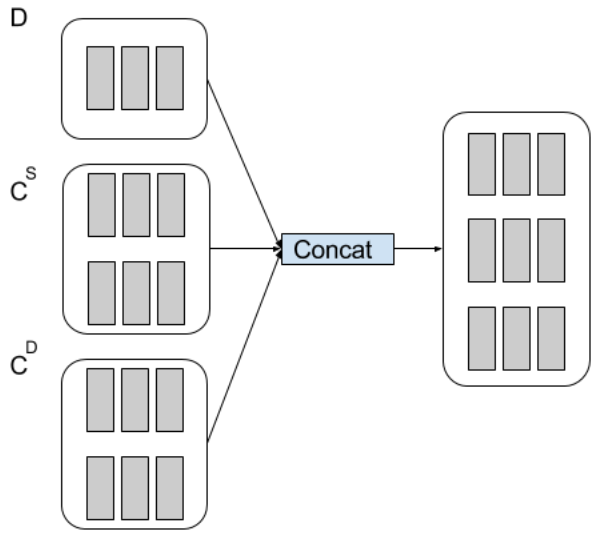

Figure 3: Scheme 1 to include sentence information. The sentences are concatenated as an additional feature.

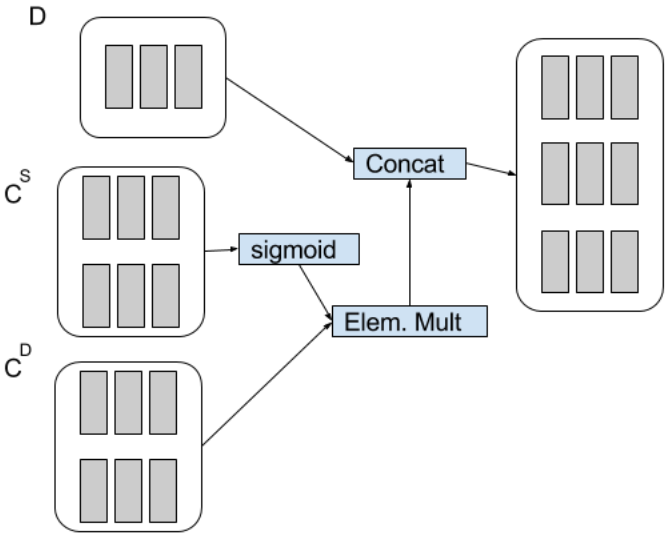

Figure 4: Scheme 2 to include sentence information. Here we use the sentence level attention as a very naive mask.

poor results with the EM score not reaching 30% after 4 epochs and the concatenation approach yielded essentially the same results as the base model but with almost 1.5x longer per epoch 7. We used a dropout rate of .3, pool size of 12, and a hidden state size of 200. From this we achieved a 65.2% F1 score and a 54% EM metric after 8 epochs of training.

The coattention model performs well across question types (Figure 8) and question lengths 9, though after 20 question tokens the model starts to lose some accuracy. Performance does noticeably degrade as the *answer* length increases 10

3.1 Example Answers and Analysis

Context: Super Bowl 50 was an American football game to determine the champion of the National Football League (NFL) for the 2015 season. The American Football Conference (AFC)

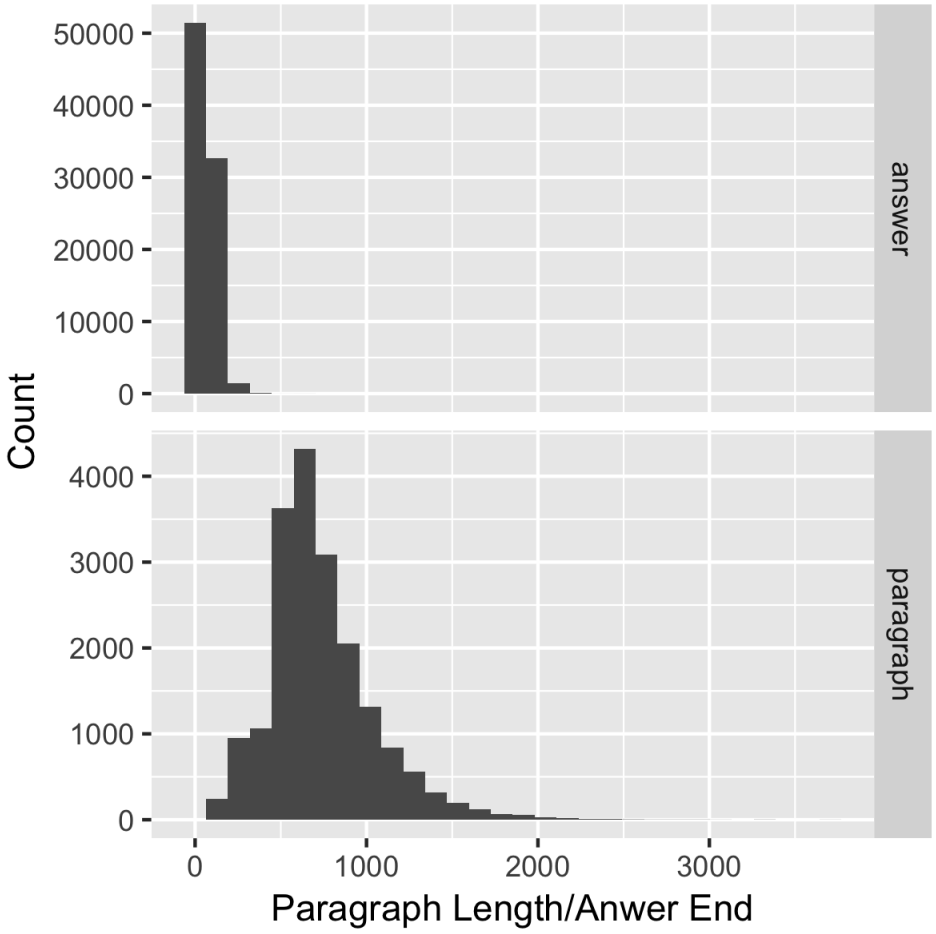

Figure 5: Length of paragraphs and endpoints of questions

champion Denver Broncos defeated the National Football Conference (NFC) champion Carolina Panthers 24Ž01310 to earn their third Super Bowl title....

Question: Which NFL team represented the AFC at Super Bowl 50?

Ground Truth: Denver Broncos

Prediction: Empty

Analysis: We forgot to include a mechanism to verify that the answer end is greater than the answer start. Most of the weight is on National Football League and it happens that the end comes before the start in this instance. This also illustrates one of the failings of the coattention mechanism, in the answer is based on similarity to the question so NFL is highly weighted even though that answer is meaningless.

Context: Other green spaces in the city include the Botanic Garden and the University Library garden. They have extensive botanical collection of rare domestic and foreign plants, while a palm house in the New Orangery displays plants of subtropics from all over the world....

Question: Where is a palm house with subtropic plants from all over the world on display?

Ground Truth: New Orangery

Prediction: New Orangery

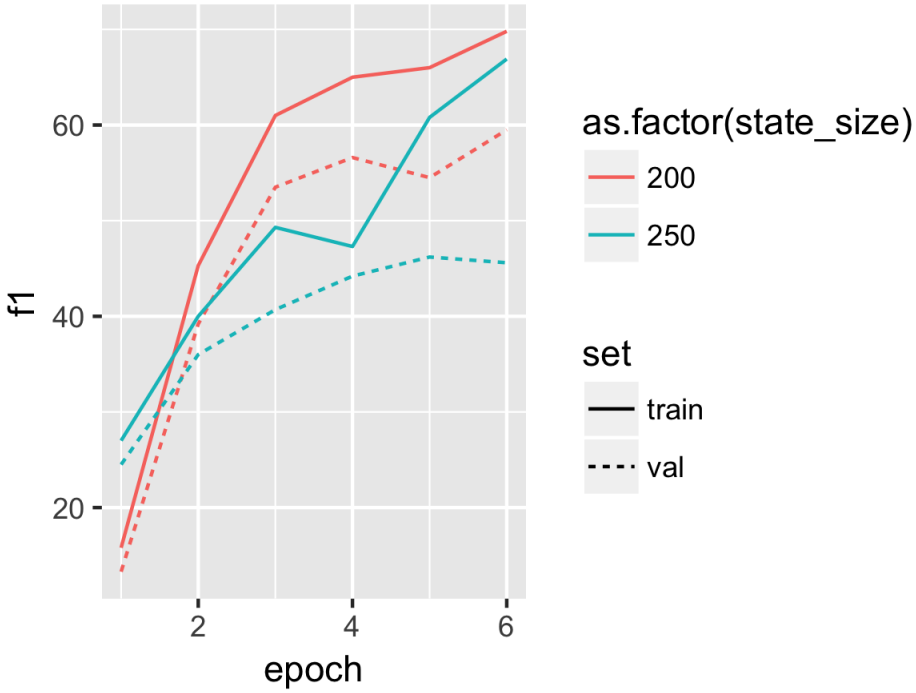

Figure 6: Comparison of state size on train and validation sets

Analysis: Here we illustrate the similarity point, but with a correct answer. The text surrounding the answer is very similar to the question, so the coattention mechanism is easily able to identify the correct answer.

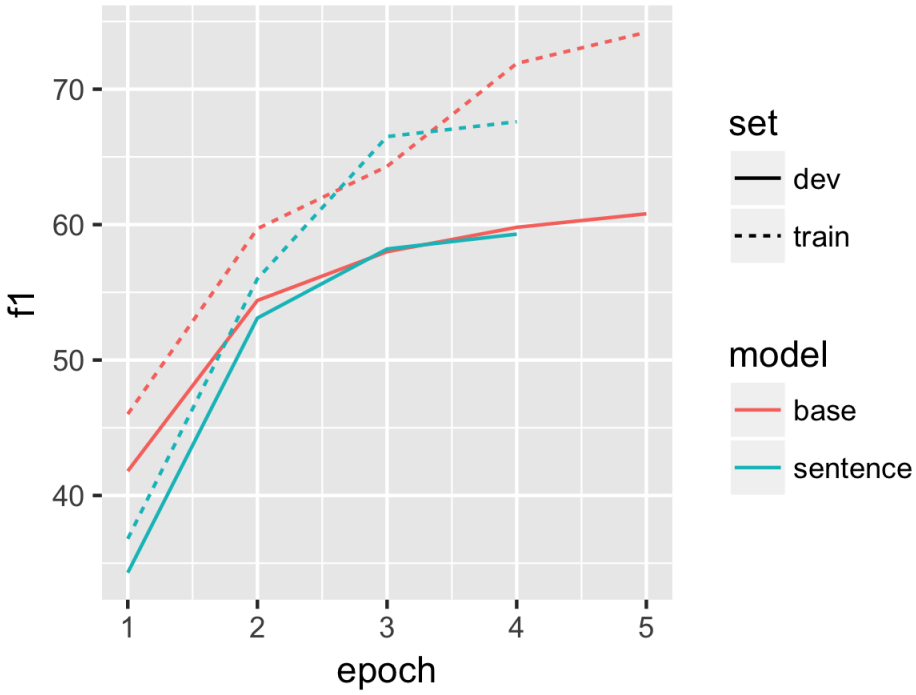

Figure 7: Dynamic Coattention vs Coattention with Sentences

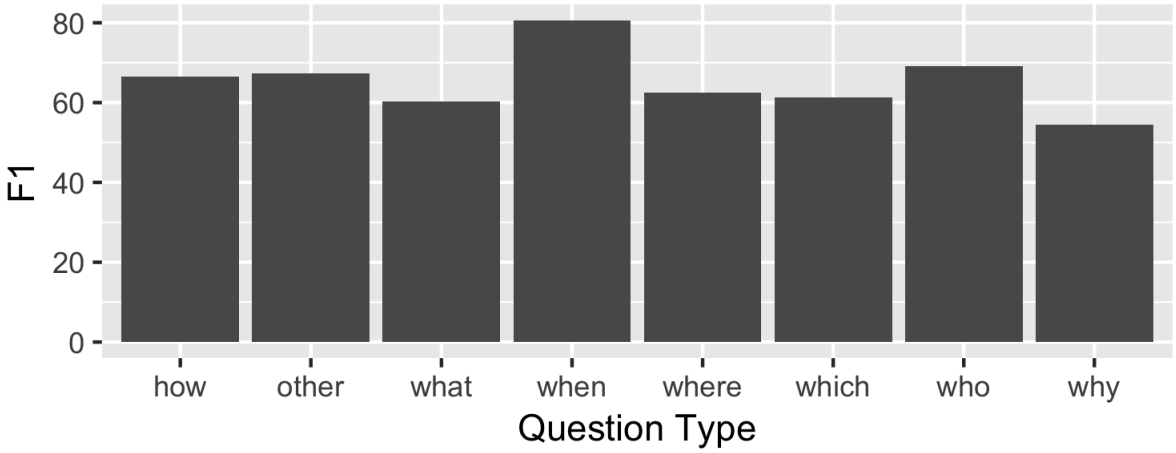

Figure 8: Question Type Performance

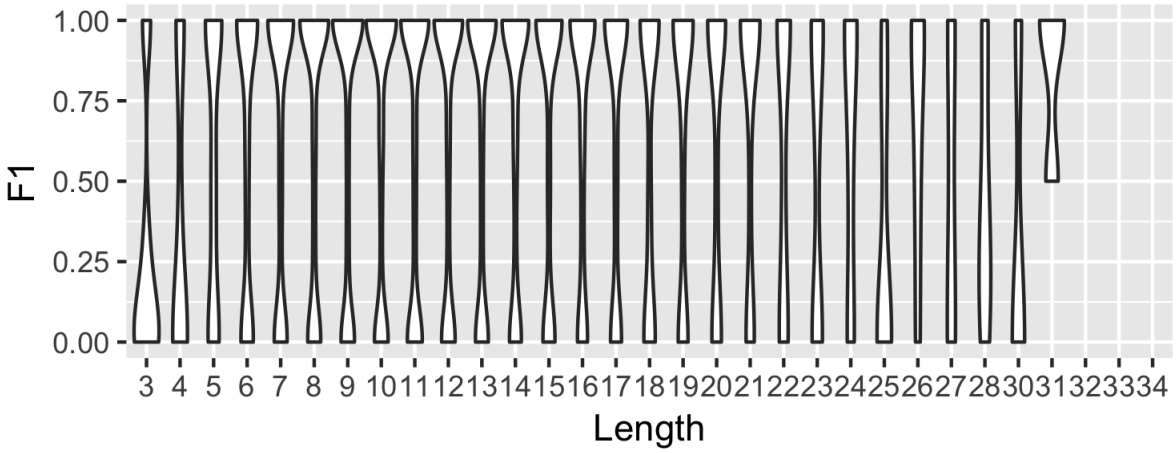

Figure 9: Question Length Performance

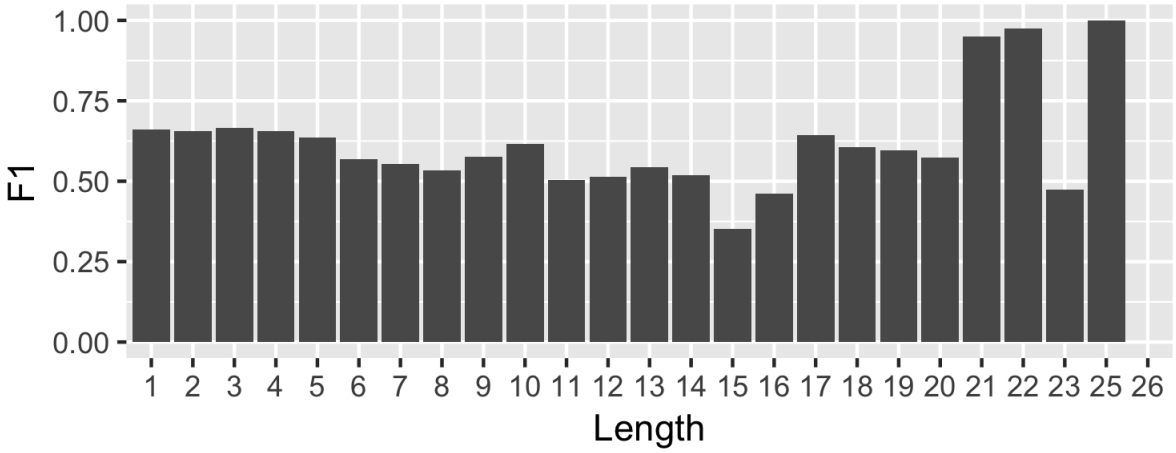

Figure 10: Answer Length Performance

References

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