Simple Dynamic Coattention Networks

Wenqi Wu
Stanford Center for Professional Development
wenqiwu@stanford.edu

Abstract

Reading comprehension (RC), or the capability to process document texts and answer questions about them is a difficult task for machines, as human language understanding and real-world knowledge are needed [4]. This can serve a wide range of applications, from simplifying information retrieval processes to building more robust artificial intelligence. Previously, most natural language processing was done with classical probabilistic models. With the recent progress in deep learning, more researchers are switching to using neural networks as they are proven to produce better results.

1 Introduction

In this project, a suitable deep learning model is devised to enable the generation of answers from document text when posed with a relevant question.

Given a document text of m words $(w_1^D, w_2^D, w_3^D, ..., w_m^D)$ and a question of n words $(w_1^Q, w_2^Q, w_3^Q, ..., w_n^Q)$, an answer would be generated. The answer would be a subset sequence of the document text $(w_i^D, w_{i+1}^D, w_{i+2}^D, ..., w_{i+j}^D)$, where $0 < i \leq m$ and $i + j \leq m$ [1].

The performance of the model is determined by the F1 and exact-match scores it achieves from evaluation of the Stanford Question Answering Dataset (SQuAD).

2 Dataset

The Stanford Question Answering Dataset (SQuAD) will be trained on and validated by our learning model. SQuAD is a reading comprehension dataset, comprising questions on document texts from Wikipedia, where each question's answer is a subset sequence or span from the corresponding text[4]. For this project, 81381 and 4284 SQuAD entries make up the training and validation set respectively.

After sufficient training, the learning model will be evaluated on a test set of 9733 data entries.

Single SQuAD data entry
Document text: 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) champion Denver Broncos defeated the National Football Conference (NFC) champion Carolina Panthers to earn their third Super Bowl title.
Question: Which NFL team represented the AFC at Super Bowl 50?
Answer: Denver Broncos

Table 1: Example of a single SQuAD data entry

As there may be several satisfactory answers to a question, 3 answers are assigned to each question. The prediction is considered correct as long as it matches at least one of the answers.

3 Method

3.1 Dynamic Coattention Networks

Dynamic coattention networks offer good performance on the SQuAD dataset. More importantly, they have all the important characteristics of a learning model which is adaptive and robust. The coattentive encoder is able to generate a good representation of the inputs, taking into account both document text and question, while the dynamic pointing decoder is capable of breaking out of local maxima when generating predictions[1].

3.2 Method

Initially, the dynamic coattention networks model was implemented and evaluated. However, it was found that this model is relatively computationally expensive and trains slowly. As such, I decided to implement a simplified version of dynamic coattention networks that trains faster and offers decent performance.

3.3 Coattention Encoder

For the encoder, we follow the exact implementation of dynamic coattention networks. Let $(x_1^D, x_2^D, ..., x_m^D)$ be the word vector sequence corresponding to words in a document text and $(x_1^Q, x_2^Q, ..., x_n^Q)$ be the word vector sequence corresponding to words in a question.

First, we evaluate for D and Q, word vector representations of the document text and question respectively.

$$D = [d_1 \quad . \quad . \quad . \quad d_m \quad d_\emptyset] \text{ where } d_t = \text{LSTM}_{enc}(d_{t-1}, x_t^D) \tag{1}$$

$$Q' = [q_1 \quad . \quad . \quad . \quad q_n \quad q_\emptyset] \text{ where } q_t = \text{LSTM}_{enc}(q_{t-1}, x_t^Q) \tag{2}$$

$$Q = \tanh(W^Q Q' + b^{(Q)}) \tag{3}$$

We compute the affinity matrix, L.

$$L = D^\top Q \tag{4}$$

We then find A^Q (A^D), the attention weights across the document text (question) for each word in the question (document text), by normalizing the affinity matrix, L row-wise (column-wise).

$$A^Q = \text{softmax}(L) \text{ and } A^D = \text{softmax}(L^\top) \tag{5}$$

We compute C^D, the shared representation of both the document text and question.

$$C^Q = DA^Q \text{ and } C^D = [Q; C^Q]A^D \tag{6}$$

Finally, we get U, which is passed on to the decoder to do boundary prediction of answers.

$$U = [u_1 \quad . \quad . \quad . \quad u_m] \text{ where } u_t = \text{Bi-LSTM}(u_{t-1}, u_{t+1}, [d_t; c_t^D]) \tag{7}$$

3.4 Dynamic Pointing Decoder

In question answering, the answer is always going to be a subset sequence of the document text[6]. As such, it is intuitive to use pointer networks in the decoder to make predictions of discrete tokens that correspond to word positions in the document text. In dynamic coattention networks, pointer networks are run for several iterations to allow the model to break free from local maxima and achieve better overall predictions[1]. Running the pointer network for several iterations results in a deep architecture. The inclusion of a highway network coupled with maxout activation function for optimization allows information to flow across several layers without attenuation and makes training significantly easier[2].

3.4.1 Method

It was observed early on that dynamic coattention networks take a long time to train. This is primarily due to the maxout activation function, which led to significantly greater number of parameters. A maxout layer will require minimally twice as many parameters as a standard layer. To increase training speed, I decided to remove maxout and fix the number of iterations to 3.

We evaluate h_i, the hidden state from the i^{th} iteration.

$$h_i = \text{LSTM}_{dec}(h_{i-1}, [u_{s_{i-1}}; u_{e_{i-1}}]) \tag{8}$$

We evaluate s_i and e_i, the predictions for the start and end indices respectively.

$$s_i = \text{argmax}_t(\alpha_i, ..., \alpha_m) \tag{9}$$
$$e_i = \text{argmax}_t(\beta_i, ..., \beta_m) \tag{10}$$

α and β are the probability distributions for the most likely start word and end word respectively.

$$\alpha_t = \text{HIGHWAY}_{start}(u_t, h_i, u_{s_{i-1}}, u_{e_{i-1}}) \tag{11}$$
$$\beta_t = \text{HIGHWAY}_{end}(u_t, h_i, u_{s_{i-1}}, u_{e_{i-1}}) \tag{12}$$

The highway network in our model has the maxout activation layer removed. A dropout with keep probability of 0.9 is performed on all parameters in the highway network, except for the last layer.

$$\text{HIGHWAY} = W^{(3)}[m_t^{(1)}; m_t^{(2)}] + b^{(3)} \tag{13}$$
$$r = \tanh(W^{(D)}[h_i; u_{s_{i-1}}; u_{e_{i-1}}]) \tag{14}$$
$$m_t^{(1)} = W^{(1)}[u_t; r] + b^{(1)} \tag{15}$$
$$m_t^{(2)} = W^{(2)}m_t^{(1)} + b^{(2)} \tag{16}$$

3.5 Loss

The loss for each index (start and end) is expressed as the mean of cross-entropy loss of each prediction.

$$\text{loss}_s = \frac{1}{N}\sum_{n=1}^{N} \text{CE}(y_{s_n}, \hat{y}_{s_n}) \tag{17}$$

$$\text{loss}_e = \frac{1}{N}\sum_{n=1}^{N} \text{CE}(y_{e_n}, \hat{y}_{e_n}) \tag{18}$$

The overall loss is simply the sum of the losses for each index.

$$\text{loss} = \text{loss}_s + \text{loss}_e \tag{19}$$

3.6 Optimization

To have better annealing, learning rate of our model is set to decrease exponentially over trainings. Adam is used here as the stochastic gradient descent algorithm.

4 Results

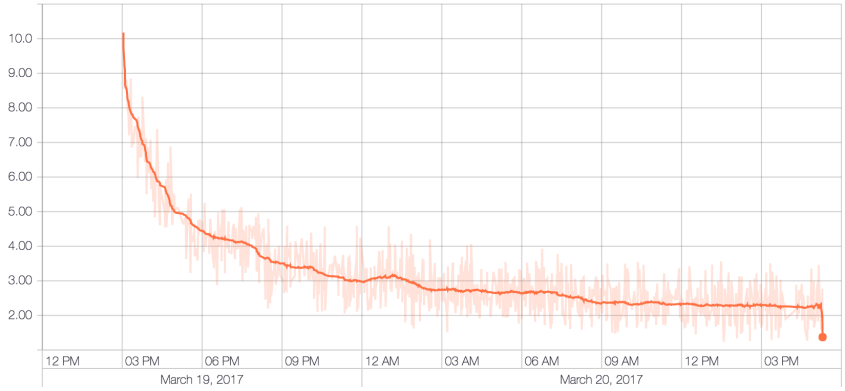

Figure 1: Loss of model over time

Scores	Dev Set	Test Set
F1	51.219	52.184
Exact match	36.868	38.75

Table 2: Performance scores of model on dev set and test set

Though our model performs distinctly better than the baseline model, it's F1 and exact-match score could not match up to the original implementation of dynamic coattention networks (80.383% and 71.625% respectively). This suggests that the absence of maxout activation function as a universal approximator and the reduction in number of iterations could impede the performance of our model.

5 Analysis

Document text: In 1881, Tesla moved to Budapest to work under Ferenc Pusk's at a telegraph company, the Budapest Telephone Exchange. Upon arrival, Tesla realized that the company, then under construction, was not functional, so he worked as a draftsman in the Central Telegraph Office instead. Within a few months, the Budapest Telephone Exchange became functional and Tesla was allocated the **chief electrician** position.
Question: What position did Tesla accept at the exchange?
Correct answer: chief electrician
Predicted answer: chief ⟨unk⟩ position

Table 3: Sample prediction from dev evaluation

The GloVe vectors used were trained on Wikipedia 2014 and Gigaword 5 datasets, comprising 6 billion tokens and 400,000 vocabulary in total. The small vocabulary size resulted in a significant portion of words to be tokenized as ⟨unk⟩. This affected the accuracy of predicted answers, as seen from Table 3. To reduced the number of unknown words, the Common Crawl GloVe vectors, which has a larger vocabulary, should be used instead.

Document text: The First British Empire was based on mercantilism, and involved colonies and holdings primarily in North America, the Caribbean, and India. Its growth was reversed by the loss of the American colonies in **1776**. Britain made compensating gains in India, Australia, and in constructing an informal economic empire through control of trade and finance in Latin America after the independence of Spanish and Portuguese colonies about 1820.
Question: When did Great Britain lose its colonies in North America??
Correct answer: 1776
Predicted answer: 1776 . Britain made compensating gains in India , Australia , and in constructing an informal economic empire through control of trade and finance in Latin America after the independence of Spanish and Portuguese colonies about 1820

Table 4: Sample prediction from dev evaluation

From the document text in Table 4, "Its growth was reversed by the loss of the American colonies in 1776" seems ambiguous if we are not provided with sufficient context to infer that the American colonies here refers to British colonies in America. Likewise, if the learning model is unable to produce an accurate co-dependent representation of the question and document text and there is ambiguity, it will face difficulty predicting the correct answer.

6 Conclusion

In this project, I proposed a simpler version of dynamic coattention networks. On the SQuAD dataset, this model achieves 52.184% F1 and 38.75% exact match, outperforming the baseline model significantly. However, it loses out to the original version of dynamic coattention networks.

In hindsight, several poor decisions have been made during this project. Firstly, the dropout keep probability is set to too high a value (0.9). A more efficient value will be somewhere around 0.5. Secondly, instead of removing maxout completely, we can limit the number of maxout units to 2. It has been proven that just 2 maxout units can approximate any function [9]. This would likely bring about a performance boost for our model.

7 Future Directions

The proposed learning model limits the input of document texts and questions to a maximum sequence length. More research can be conducted to design a model that can receive inputs of any

length.

Recurrent neural networks are sequential in nature and difficult to parallelize. Other architectures, such as convolutional neural networks, can be considered for encoding the document texts and questions.

8 Acknowledgements

I cannot express enough gratitude to Professor Chris Manning, Professor Richard Socher, and all the teaching assistants for their guidance and support, without whom this project would not have been possible.

References

[1] Caiming Xiong, Victor Zhong, Richard Socher
Dynamic Coattention Networks for Question Answering Conference paper at ICLR, 2017

[2] Rupesh Kumar Srivastava, Klaus Greff, Jürgen Schmidhuber
Highway Networks arXiv, 2015

[3] Oriol Vinyals, Meire Fortunato, Navdeep Jaitly
Pointer Networks arXiv, 2017

[4] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, Percy Liang
SQuAD: 100,000+ Questions for Machine Comprehension of Text arXiv, 2016

[5] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, Hananneh Hajishirzi
Bi-Directional Attention Flow for Machine Comprehension arXiv 2016

[6] Shuohang Wang, Jing Jiang
Machine Comprehension using Match-LSTM and Answer Pointer arXiv 2016

[7] Zhiguo Wang, Haitao Mi, Wael Hamza, Radu Florian
Multi-Perspective Context Matching or Machine Comprehension arXiv 2016

[8] Ilya Sutskever, Oriol Vinyals, Quoc V. Le
Sequence to Sequence Learning with Neural Networks arXiv 2016

[9] Ian J. Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, Yoshua Bengio
Maxout Networks