Reading Comprehension with Coattention Encoders

Chan Lee
Stanford University
`chanlee@stanford.edu`
`Codalab ID: chanlee`

Jae Hyun Kim
Stanford University
`jakekim@stanford.edu`
`Codalab ID: chanlee`

Abstract

We present an end-to-end network architecture for question answering task given a real-world paragraph context. Using SQuAD dataset composed of 100K <question, context, answer> pairs, we train a coattention encoder to obtain a knowledge representation of the question and context data. This in turn is given as input to the decoder, which generates answer spans within the context. The prediction results are evaluated using F1 and EM scores, and we show that the coattention model performs significantly better than a simple Bi-LSTM baseline model. After sufficient training, we were able to obtain 54.2% F1 and 41.3% EM score against the test set.

1 Introduction

Answering questions is a challenging task even for deep learning computational models since it requires understanding both the nature of the language (such as noun and verb forms) and knowledge about the world (such as different meanings of a word depending on the context). And SQuAD, with real-world paragraph contexts and robust forms of questions, provides a nice benchmark for testing a model on such task of reading comprehension.

We present a neural network architecture aimed to answer questions in the SQuAD set for question answering given a real-world paragraph context. We train a coattention encoder to obtain a knowledge representation of question and context data. This, in turn, is given as input to the decoder, which generates answer spans within the context. We also train a baseline model of Bidirectional LSTM and compare the performance of two models. We then analyze some of the factors that affect the accuracy of the predictions, such as how long the prediction is and what grammatical structure the question takes. Finally, we give a suggestion as to how we can improve the performance of our model.

2 Background/Related Work

2.1 Stanford Question Answering Dataset (SQuAD)

Rajpurkar et al. (2016) [5] released SQuAD which is composed of 100K question-answer pairs and corresponding context paragraph. The context paragraphs were extracted from a set of articles on Wikipedia. Humans generated questions using the paragraph as a context, and selected a span of the context as the target answer. The SQuAD is an interesting dataset in several ways. It is orders of magnitude larger than all previous hand-annotated datasets. Also, compared to previous datasets such as Story Cloze test [4], the questions are more realistic and answers are in much more robust form. Rajpurkar et al. (2016) have shown that although the answer is confined to the context span, answers require different types of logical reasoning, including multi-sentence reasoning.

The following is an example of a data entity in SQuAD `<question, context, answer>`:

- Question: `"Why was Tesla returned to Gospic?"`

- Context paragraph:

  ```
  "On 24 March 1879, Tesla was returned to Gospic under police guard
  for not having a residence permit. On 17 April 1879, Milutin
  Tesla died at the age of 60 after contracting an unspecified illness
  (although some sources say that he died of a stroke). During that
  year, Tesla taught a lrge class of students in his old school,
  Higher Real Gymnasium, in Gospic."
  ```

- Answer: `"not having a residence permit"`

2.2 Bidirectional LSTM

Long Short Term Memory Recurrent Neural Network (LSTM) improves the basic Recurrent Neural Network (RNN) model by replacing nonlinear units in the hidden layers by memory blocks. The memory blocks can store and access information over long periods of time. This enables LSTM to avoid the vanishing gradient problem of RNN models.

Bidirectional RNNs improve RNN by processing the data in forward and backward directions using two separate hidden layers. It computes the forward hidden sequence, backward hidden sequence, and the output sequence by iterating the backward layer from t =1 to T and consequently updating the output layer. This enables Bidirectional RNNs to make use of previous and future context.

Bidirectional LSTM combines these two models and can encode long-range context in both forward and backward directions. It has been widely used in various researches such as named entity recognition in Twitter posts.

2.3 GloVe

GloVe is an unsupervised learning algorithm for obtaining vector representations for words [3]. First developed by Stanford NLP group in 2014, it uses a log-bilinear model with a weighted least-squares objective, and trains aggregated global word-word co-occurrence statistics from a corpus. The goal is to encode ratio of word-to-word co-occurrence probabilities in the model. As a result, the representation of the vocabulary embodies linear substructures of the word vector space. We use the GloVe word vectors to preprocess our vocabulary data.

2.4 Dynamic Coattention Network

Dynamic Coattention Network (Xiong at el. 2017) [1] is a state of the art model for question answering. On the SQuAD test dataset, a single DCN model achieved the highest F1 score of 75.9% compared to the previous 71.0%. The DCN ensemble obtained F1 of 80.4% compared to the previous 78.1%.

One of the biggest advantages of DCN compared to previous models such as Bi-LSTM is that it can recover from local maxima corresponding to incorrect answers. Instead of making a single pass through the context, DCN selects an answer span by iteratively alternating between predicting the start index and predicting the end index. This is particularly effective for SQuAD dataset since many context data contain multiple intuitive answer spans, and thus multiple local maxima points.

DCN consists of two parts: a coattentive encoder and a dynamic pointing decoder. The encoder captures interactions between question and the context. The decoder alternates between predicting start index and end index.

3 Approach

We take encoder-decoder approach for reading comprehension task. The encoder summarizes the information in the question, and uses that information to further summarize the information from the context. The resulting knowledge representation of the question and context is given as input to the decoder, which uses the obtained knowledge to predict the answer. We propose two encoder models: baseline model and coattention model. The baseline model uses a bidirectional LSTM to encode the question, and uses this representation to generate a conditional encoding of the context.

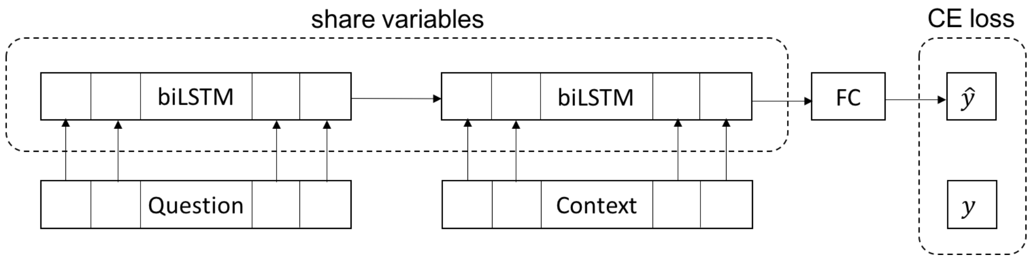

Figure 1: The baseline model uses a Bi-LSTM layer to generate conditional encoding of the context. This knowledge representation goes through a fully-connected layer to produce logits for each position of the context paragraph.

The coattention model generates an affinity matrix between the question and the context, and uses it to encode the context, similar to the attention model [6]. The resulting knowledge representation is given to a simple decoder to generate answer spans.

3.1 Problem Setup and Word Representations

Our goal is to predict the answer span $\{a_s, a_e\}$ given a question of length n, $\mathbf{q} = \{q_1, q_2, ..., q_n\}$, and a context paragraph $\mathbf{p} = \{p_1, p_2, ..., p_m\}$ of length m. While $p_1, ...p_m$ and $q_1, ..., q_n$ denote word IDs, a_s, a_e indicate the start and end position of the answer in paragraph $\mathbf{p}$.

To efficiently encoder the information in the question and the context, we represent the question and context using Glove word embeddings [cite] of dimensionality $d = 100$ and vocab size of 400k that have been pretrained on Wikipedia 2014 and Gigaword 5.

3.2 Baseline Model

The baseline model assumes that information of a question or a context paragraph can be encoded in a single hidden state. We run a Bi-LSTM over the question, concatenate the two end hidden vectors and call it the question representation. Then, we run another Bi-LSTM over the context paragraph, this time setting the initial states as the question representation we obtained earlier. Finally, we concatenate the two end hidden vectors of this Bi-LSTM and call it the knowledge representation u. u is given to a simple decoder that uses a fully-connected layer to calculate logits for each position of the context paragraph. We use two FC decoders that each predict a_s and a_e. We use softmax cross-entropy loss for training. It is mathematically presented as below.

$$h_0^q = 0, h_t^q = \text{Bi-LSTM}_{enc}(h_{t-1}^q, h_{t+1}^q, q_t)$$

$$h_0^c = h_n^q, h_t^c = \text{Bi-LSTM}_{enc}(h_{t-1}^c, h_{t+1}^c, p_t)$$

$$u = h_m^c$$

$$\hat{y_s} = \text{FC}_s(u), \quad \hat{y_e} = \text{FC}_e(u)$$

$$l = \text{softmax-CE}(y_s, \hat{y_s}) + \text{softmax-CE}(y_e, \hat{y_e})$$

3.3 Coattention Model

The coattention model closely follows the coattention encoder of [1]. Let $(x_1^Q, x_2^Q, ..., x_n^Q)$ denote the word vectors of the question and $x_1^C, x_2^C..., x_m^C)$ denote the word vectors of the context paragraph. An LSTM layer encodes the question and the context paragraph: $c_t = \text{LSTM}(c_{t-1}, x_t^C), q'_t = \text{LSTM}(q'_{t-1}, x_t^Q)$. The question and the context paragraph is now denoted as $C = [c_1, ..., c_m], Q' = [q'_1, ..., q'_n]$. Again, the same LSTM layer is used for both encoding. To allow the difference between question encoding and context encoding, we introduce another non-linear layer on the top of question encoding: $Q = \tanh{(W^{(Q)}Q' + b^{(Q)})}$

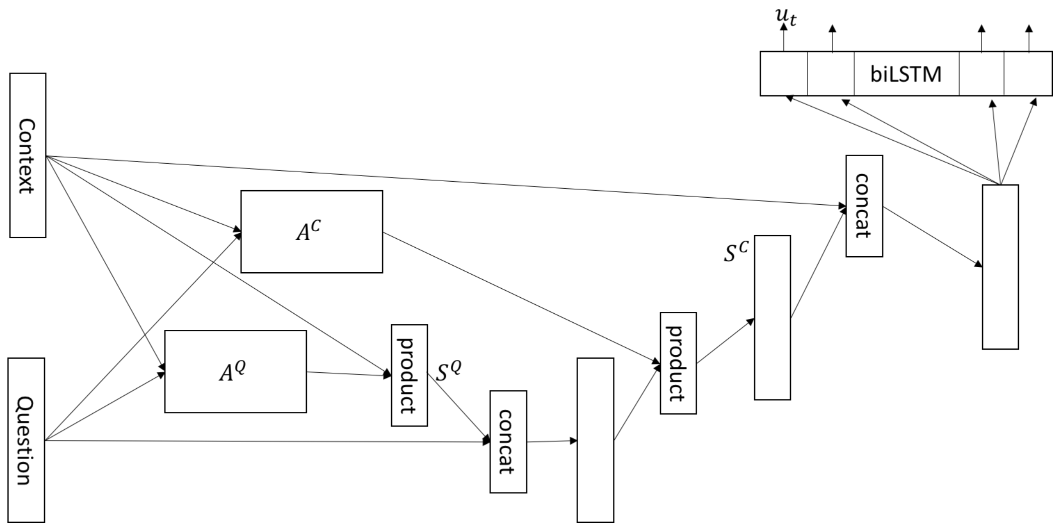

Figure 2: The coattention encoder model produces summaries of context for each word in question S_Q and summaries of question for each word in context S_C. Hence the information in the context and the question are interwoven and put into the Bi-LSTM to produce knowledge representatino u_t.

Using these encodings, we compute the affinity matrix: $L = D^TQ$. The affinity matrix is normalized using softmax row-wise and column-wise to produce attention weights for the question and the context paragraph, respectively.

$$A^Q = \text{softmax}(L) \quad \text{and} \quad A^C = \text{softmax}(L^T)$$

Next, we compute the attention contexts of document for each word of the question.

$$S^Q = CA^Q$$

Then, we compute the summaries of the question for each word of the document. Here, we use both Q and C^Q to represent the information the question.

$$S^C = [Q; S^Q]A^C$$

Finally, we fuse the temporal information by running C^D through a bidirectional LSTM.

$$u_t = \text{Bi-LSTM}(u_{t-1}, u_{t+1}, [c_t; S^Q_t]$$

We denote $U = [u_1, ..., u_m]$ as the knowledge representation, and U goes through a fully-connected layer to produce logits for each position of the context paragraph. Again, softmax cross-entropy loss is used to train the model.

3.4 Decoder

While we train the model using sparse softmax loss between logits and labels, we have an additional decoder that makes prediction for a_s, a_e using the logits. This step does not affect the learning, but aims to maximize the F1 and EM score, explained the later section.

First, all logits for positions after the length of the context paragraph are set to $-\infty$. Hence a_s, a_e are never chosen from the positions after the context paragraph ends. Then, a_s, a_e are chosen as below.

$$a_s = \text{argmax}(\hat{y_s}), \quad a_e = \text{argmax}(\hat{y_e})$$

Finally, if $a_s > a_e$, we set $a_s = a_e$ so that it predicts the word in position a_e as the answer span.

3.5 Evaluation

We use two evaluation metric for the task: F1 and EM. F1 score loosely measures the overlap between the prediction and ground truth answer. It is defined as the harmonic mean of precision and recall, where precision denotes the fraction words predicted as answer that were actually answer, and recall denotes the fraction of true labels that were actually predicted as answers.

$$\text{precision} = \frac{tp}{tp + fp} \quad \text{recall} = \frac{tp}{tp + fn}$$

$$F1 = 2\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

EM is the fraction of answer spans of which every word was predicted correctly. We choose the hyperparameters based on F1 and EM scores rather than the loss.

4 Experiments

4.1 Data preprocessing & Handling out-of-vocabulary words

To save memory and time, the maximum length of questions was set to 20, and the maximum length of the context paragraph was set to 750. Any tokens after these limits were omitted before being put to the model.

Since we use a limited vocabulary size of 400k words, there are occasions where our model does not recognize the word in a question or context. By default, this maps to <unk> in our vocabulary list. This, however, led to incorrect predictions even when the model has the correct answer span. To prevent this issue, we pass the original context string instead of decoding the context using our vocabulary, with the exception of utf-8 incompatible words. As a result, we were able to achieve a significant boost in our performance, approximately $+10$ for both F1 score and EM.

4.2 Implementation Details

The Glove word embeddings were generated with dimension 100. In the baseline model, the Bi-LSTM layer that encoded the question and the context had 200 hidden units for each LSTM layer, adding up to 400 units for the Bi-LSTM layer. For the coattention model, the LSTM layer for encoding had 200 hidden units. The Bi-LSTM layer in the final stage of coattention model had 400 units in total.

The models were trained using ADAM optimizer [cite] with initial learning rate of 1e-3. The learning rate was exponentially decayed by 0.95 for every 300 batch. The batch size was set to 20, and the gradient was clipped with maximum gradient norm of 1.0. The model was trained for 4 epochs, and the parameters with the highest F1 and EM score on the validation set chosen.

4.3 Results

Baseline model (Bi-LSTM) The initial training loss was approximately 13.5, and the model reached training loss of approximately 11 after 0.5 epoch of training. After that the training loss stayed relatively stable, not increasing or decreasing. The F1 and EM score on the validation set was < 10, and hence we concluded that the baseline model does not effectively capture the information in the context or question.

Coattention encoder model The training loss constantly decreased until epoch 6, while the validation F1 and EM peaked at epoch 3 and did not show much improvement over the following epochs. After epoch 5, the performance started to decrease due to overfitting. At its peak, the model showed validation F1 of approximately 55, and EM of approximately 40. Using this model, we achieved F1 of 54.235 and EM of 41.33 on the test set. The train loss and the validation performance is shown in Figure 3.

4.4 Error Analysis

Effect of token lengths on prediction accuracy Figure 4. shows the relationship between average of EM score and the number of tokens in question, context, and answer (true label). We can observe that the average EM decreases as the true label spans get longer. Moreover, the EM score is low for

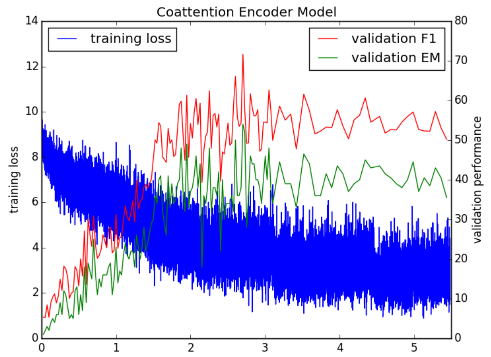

Figure 3: The coattention encoder model shows maximum validation F1 and EM of approximately 55 and 40 near epoch 3. After epoch 5, the performance deteriorates due to overfitting. Noisy training loss curve is due to a small batch size of 20.

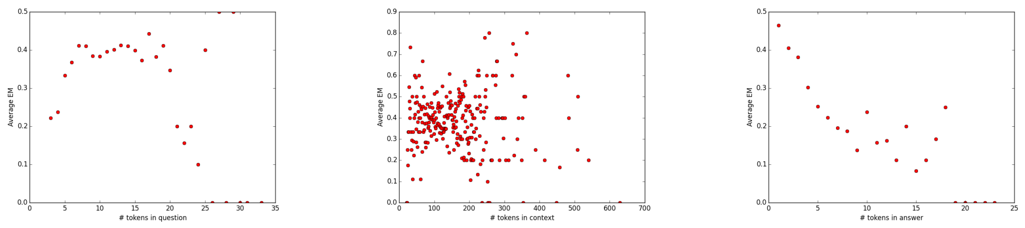

Figure 4: Average EM decreases as the true label spans get longer. The EM is low for short questions with less than five words. The relationship between EM and the context length is obscure. F1 scores show a similar behavior.

short questions with less than five words. The length of the context paragraph and the EM score is not significantly correlated. Hence, our model, while accurately predicting short answer spans that consist of less than five words, has difficulty predicting longer answers.

Error classification The coattention model has strengths and weaknesses. In general, the model is capable of predicting an approximate location of the answer span. However, it often ends up predicting a wrong span that is either too long or too short. Regarding this property, we classify the erroneous predictions into four categories: misunderstanding of the semantic structure, misunderstanding of meaning, too-narrow prediction, and too-wide prediction. The first two errors occur when the model fails to locate the answer span. The last two occur when the model locates the answer span but selects a wrong scope.

The first type of error, misunderstanding of semantic structure, shows predictions that have different part-of-speech from the true labels. For example, in one question, while the true label is `"New Guinea"`, the model predicted `"proceeded smoothly"`. This shows that for this particular case, the model was unable to recognize the fact that the answer should be a noun.

Misunderstanding of meaning shows predictions that have same part-of-speech as true labels but different meanings. For example, in one question, while the true label is `"Wilson's geographer"`, the model predicted `"American Empire"`. Although the model figured out that the question was asking for a noun, it chose a wrong noun.

Too-narrow prediction refers to predictions that capture only a part of the true labels. For example, the model predicts `"Bowman"` while the true label is `"Isiah Bowman"`. Too-wide prediction denotes predictions that capture more than the true labels. For example, the model predicted `"Asia and the Middle East"` when the true label was `"Middle East"`. This shows that the model was unable to identify the fact that "Asia" was not part of the answer.

These two errors imply that the model is not good at predicting the accurate length of the answer span. This is explained by the fact that the predictions of a_s and a_e are made independently of each other in the decoder. Hence, the decoder's decision of a_e is not affected by its decision of a_s, and vice versa. This is not desirable, because the probability distribution of a_e changes as a_s changes and vice versa. Hence, allowing the decoder to make decisions of a_e and a_s interdependently has a great potential to enhance the decoder and should be investigated further.

5 Conclusion and Future Directions

The coattention encoder model successfully captured the co-dependent representations of the question and the context, achieving 54.2% F1 and 41.3% EM on the SQuAD dataset. However, it makes decisions for a_s and a_e independently and consequently fails to capture the fact that the probability distribution of a_e is dependent on the choice of a_s and vice versa. Therefore, a decoder that can capture the dependency between them may significantly enhance the prediction.

One possible way to achieve this is to sequentially make the decisions. In this model, the decoder would first select a_s. Then, conditioning on the chosen a_s, the decoder recalculates the logits for a_e and chooses the a_e that maximizes the conditional probability. This approach can be further extended by allowing the decoder to iteratively make decisions, choosing a_s and a_e back and forth. This allows the model to escape from local minima and reach the optimum.

References

[1] Xiong, C., Zhong, V., & Socher, R. (2016). Dynamic Coattention Networks For Question Answering. arXiv preprint arXiv:1611.01604.

[2] Shuohang Wang and Jing Jiang. Machine comprehension using match-LSTM and answer pointer. arXiv preprint arXiv:1608.07905, 2016b.

[3] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In EMNLP, volume 14, pp. 1532–43, 2014.

[4] Mostafazadeh, N., Chambers, N., He, X., Parikh, D., Batra, D., Vanderwende, L., ... & Allen, J. (2016). A corpus and evaluation framework for deeper understanding of commonsense stories. arXiv preprint arXiv:1604.01696.

[5] Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016). Squad: 100,000+ questions for machine comprehension of text. arXiv preprint arXiv:1606.05250.

[6] Mnih, V., Heess, N., & Graves, A. (2014). Recurrent models of visual attention. In Advances in neural information processing systems (pp. 2204-2212).