Iterative Attention Network for Question Answering

Tom Henighan
Department of Physics
Stanford University
Stanford, CA 94305
henighan@stanford.edu

Abstract

In recent years deep learning models have with great success utilized attention mechanisms for the task of question answering. However, none employ multiple iterations of attention. Here I propose iterative attention networks for question answering, which allows for multiple stages of attention both from the passage to the question and vice-versa. Intutively, this is motivated by the human behavior of looking back and forth at the question and passage before producing an answer. I find a second attention iteration offers a significant performance boost over a baseline model with a single attention iteration. Further, I show that the model shifts its attention towards query words which indicate the form of the answer in the second iteration.

1 Introduction

Reading comprehension is the ability to read and understand the meaning of a passage or piece of text. Giving machines this capability, machine comprehension (MC), is a long-standing goal in artificial intelligence [1, 2]. In classrooms, humans' reading comprehension is tested by answering questions requiring interpretive-understanding of the text. The same paradigm is used in MC, thereby making desirable datasets with triplets of a passage along with questions and answers based on the passage. Previous human-annotated datasets were too small in size for effective use with state-of-the-art deep-learning architectures [3, 4]. The Stanford Question Answering Dataset (SQuAD) offers an order of magnitude more data, with $\sim$100,000 question-answer pairs on $\sim$500 passages. The answers are subsets of the reference passages, making the task more tractable [5]. Additionally, Rajpurkar et al show that, unlike previous semi-autonomously generated QA datasets, SQuAD is diverse in its answers and requires logical reasoning, even over multiple sentences [5].

Much of the recent success of deep neural networks for natural language processing has come through the use of attention mechanisms, which allows the model to refer to a targeted area in the passage. Unsurprisingly, previous work suggests this to be a promising avenue for question answering [6, 7].

Here I introduce an iterative attention network architecture for question answering trained on SQuAD, illustrated in figure 1. Conceptually, the design is meant to allow the machine to iteratively look back and forth at the query and passage before predicting the answer. It consists of an initial encoding stage, followed by a first attention stage, which is then encoded again before reaching a second attention stage before a final stage of decoding and softmax prediction. The single model obtains an F1 score of 62%, compared to the single model state of the art of 80%.

Interestingly, the model seems to pay attention to query words which are specific to the question in the first iteration, then shifts its attention towards query words which reveal the form of the answer in the second iteration.

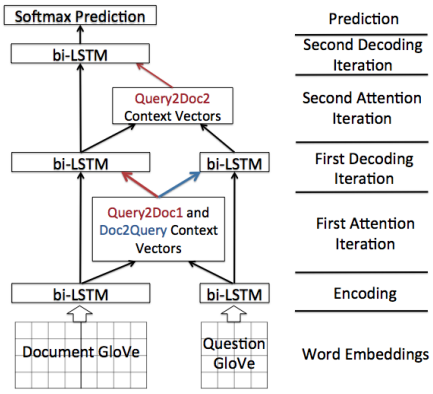

Figure 1: Illustration of iterative attention network architecture. The Attention mechanism illustrated in the first and second attention iterations are described in more detail in figure 2

2 Iterative Attention Networks

An overview of the architecture is given in figure 1. Conceptually, it is meant to mimic the behavior of humans (or at least this human) to look back at the question a second time after having found the relevant passage in the context paragraph.

2.1 encoding

The α dimensional pre-trained word vectors for each query and passage word are first retrieved. These are fed into bi-directional recurrent-neural-networks (RNNs) with long short-term memory cells (LSTM) [8] and state size β at the encoder stage. The output at each cell from each direction of the bi-RNN was concatenated to produce an output vector of length 2β (this is true for all 5 RNNs in the system). We will represent the encoded query and passage word representations as matrices Q_1 and P_1, whose columns are the encoded word representations. Let us call the query and passage lengths n and m, respectively, so $Q_1 \in \mathbb{R}^{2\beta \times n}$ and $P_1 \in \mathbb{R}^{2\beta \times m}$.

2.2 attention iterations

The procedure for obtaining the context vectors is illustrated in figure 2. First the score matrix $S_1 = P_1^T Q_1 \in \mathbb{R}^{m \times n}$ is computed, whose entries give the attention score for each query/passage word pair. The alignment weights are then obtained by normalizing the columns of S_1 to yield the Doc2Query attention weights, A_Q, and normalizing the rows then transposing to yield the Query2Doc attention weights, A_P. Normalization is done via the softmax function. Finally, the products $P_1 A_Q = C_Q \in \mathbb{R}^{2\beta \times n}$ and $Q_1 A_P = C_P \in \mathbb{R}^{2\beta \times m}$ give the Doc2Query and Query2Doc context vectors, respectively. C_P is then concatenated with P_1, and similarly C_Q with Q_1, and then fed into the next RNN stage, which also has state size β to produce the second representations of the passage and question, C_2 and Q_2. These new representations are used for a second attention iteration. In the second attention iteration, only the Query2Doc context vectors are calculated.

2.3 decoding and prediction

The result of the second attention iteration is then fed to yet another bi-LSTM decoder, also of state size β to produce a final representation of the context paragraph, $C_3 \in \mathbb{R}^{2\beta \times m}$. The start and end index were then found by taking the matrix product of C_3 with weight vectors W_{start}, $W_{end} \in \mathbb{R}^{1 \times 2\beta}$ to produce the start and end prediction vectors. These were then softmaxed to produce $\hat{\mathbf{y}}_{start}$ and $\hat{\mathbf{y}}_{end} \in \mathbb{R}^{1 \times m}$, which are predictions of the one-hot representations of the start and end word indexes, as shown in equations 1 and 2. The weights and biases of all 5 bi-

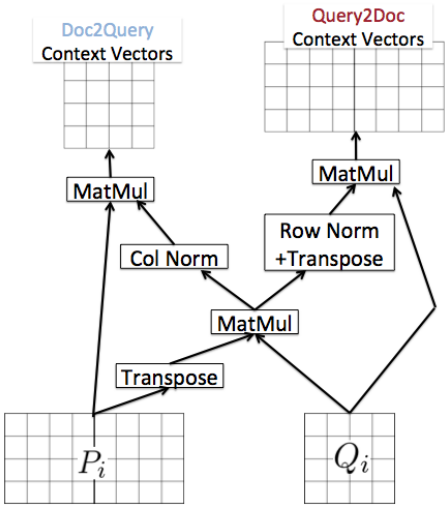

Figure 2: Illustration of Co-attention mechanism used to generate the context vectors in the first attention iteration seen in figure 1. In the second iteration, only the Doc2Query context vectors are used.

LSTM RNN's are trained separately. The outputs from each direction of the bi-LSTM RNN were concatenated in all cases.

$$\hat{\mathbf{y}}_{start} = \text{softmax}(W_{start} \cdot C_3) \tag{1}$$

$$\hat{\mathbf{y}}_{end} = \text{softmax}(W_{end} \cdot C_3) \tag{2}$$

Cross-entropy was used as the cost function during training.

3 Results and Analysis

3.1 implementation details

100-dimensional GloVe vectors pre-trained on a corpus of 6 billion words [9]. Words not present in the GloVe vocabulary are set to a vector of zeros. Word embeddings were held fixed. The state size, β for each direction of the bi-LSTMS was set to 50, leading to vectors of dimension 100 after concatenating the two directions together. All weights and biases were initialized to random values such that a norm of unity was maintained [10]. All models were implemented using tensorflow [11].

Histograms of the number of tokens in the context and question can be found in figure 3. Based on these visualizations, question and passage sequences were padded to lengths of 30 and 300, respectively, with words beyond that length being ignored.

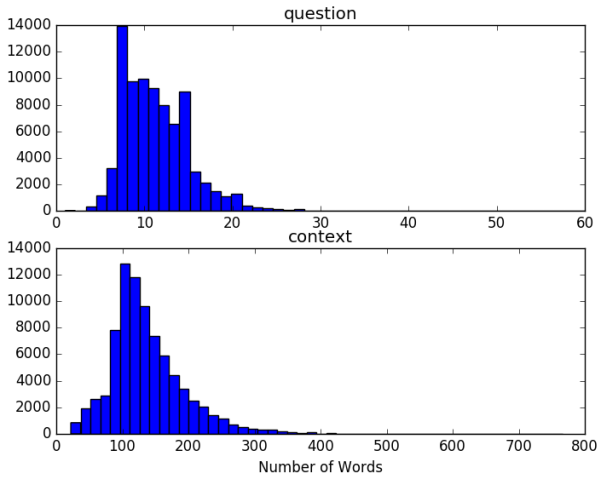

Figure 3: Histograms of number of tokens in questions and context passages over training set.

The SQuAD training dataset was split into a set of ∼87,000 QA pairs which were used for training and a ∼10,000 validation set. The cost is minimized using ADAM [12], with a batch size of 32. Gradient clipping was implemented, where the max gradient norm of 1.0 was chosen based on the visualization seen in figure 4, which is a histogram of the global gradient norms over the entire training dataset for random initial conditions. Note the maximum gradient norm exceeded typical values by three orders of magnitude, making gradient clipping a necessity.

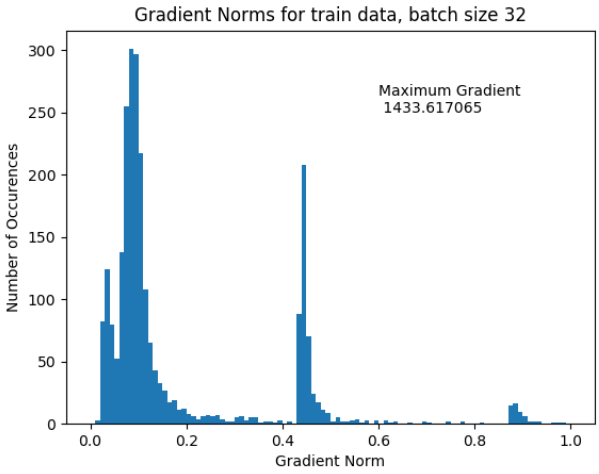

Figure 4: Histograms of global gradient norms using batch size 32 over entire training set for randomly initialized model.

3.2 results

The model was evaluated based on two metrics: exact-match and F1. Exact-match is the fraction of test examples on which the model prediction exactly matched the correct answer. F1 is the harmonic mean on recall and precision between the predicted tokens and the true-answer tokens. The F1 and EM scores on the test set were 62% and 50%, respectively.

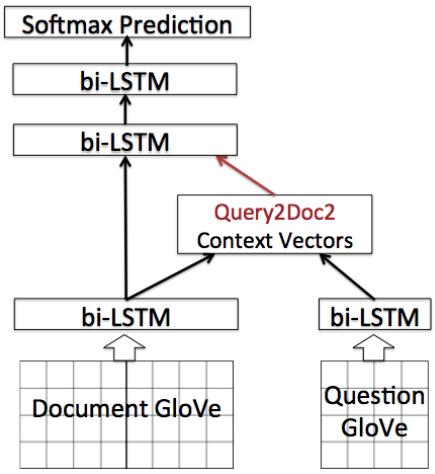

Figure 5: Illustration of a simpler architecture which retains 3 stages of RNNs, but only one iteration of attention. Labeled as "Single Attention w/ extra RNN" in 3.3.

3.3 ablation

To evaluate the benefit of two attention iterations as opposed to just one, the performance was compared to a similar architecture with a single attention stage, but that still had three stages of RNNs, illustrated in figure 5. Table 3.3 reveals that the second iteration step offered a significant performance boost over this baseline model.

Model	# Parameters	Val F1	Val EM
Single Attention w/ extra RNN	281,800	56%	39%
Double Attention	422,200	64%	46%

Table 1: Results of ablation study

3.4 analysis of double-attention

Since at both attention iterations, alignment weights over the question tokens are calculated for each token in the passage, we can easily compare what question words the model paid attention to at each step. We can then answer the question how the did the model "shift its attention." In figure 6, one finds a visualization of the Query2Doc alignment weights for each attention iteration. The staircase pattern in the top middle of the left panel shows where the corresponding phrases in the question appear almost verbatim in the passage, and indeed occurs near the correct answer (which the model correctly predicted). Thus it appears in the first iteration, the model is paying attention to words that show up in both the question and the passage, which seems a sensible strategy for finding the correct answer.

The second iteration shows a starkly different pattern, with the model paying attention to the word "many" at essentially every word in the passage. In contrast to the words most paid attention to in the first iteration, "many" reveals what *form of answer to look for*, ie, a number. While figure 6 is only a single example, one finds this behavior to be a common trend, where the first iteration shows attention paid towards many words specific to the question, while attention is sharply focused on one word which reveals the *form* of the answer in the second iteration.

To further illustrate this pattern of shifting attention towards words which reveal the *form* of the answer, I show in table 3.4 the query words for which the attention most increased and decreased over the dev set. I restricted the list to words which appeared in at least 5 questions. The attention change was found by summing the attention weights seen in figure 6 in the vertical direction, and then taking the difference between the second and first iteration.

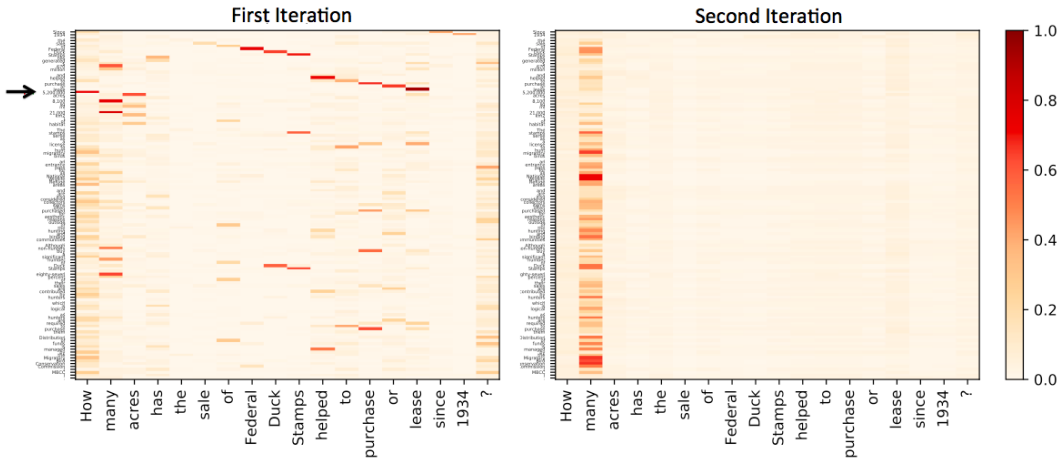

Figure 6: Visualization of Query2Doc alignment weights in first and second iteration. The arrow on the left marks the position of the answer, which is "5,200,000 acres."

Attention shifted *towards*	attention change	Attention shifted *away from*	attention change
year	35.163535	why	-12.563048
date	31.014911	how	-10.542498
many	24.178574	what	-10.041818
color	22.925476	it	-8.917352
rank	20.952049	reason	-7.802622
wrote	19.055681	?	-7.264514
country	18.509165	because	-7.016249
called	17.491110	but	-5.602870
direction	17.111677	cities	-5.421548
whose	15.951545	bison	-4.632426

Table 2: Words which attention was shifted most *towards* or *away from* from first to second attention iteration.

Indeed, the words for which the attention most increased reveal the *form* of the answer. The answer is a year (year) or a country (country) or a number (many), or a person (whose), etc. In appendix A, one will find randomly chosen questions containing each of these words to further make this point.

My understanding of why attention is shifted away from the words on the right side of the table is less clear. They do seem to be words that show up at the beginning and end of the question, which is made especially obvious by the presence of the token "?" on the list. The model does tend to pay attention to the first and last token on the first iteration, as seen in the left panel of figure 6. I also feel these words are generally useless, again with the "?" token being the flagship example.

3.5 common modes of failure

The prediction of the start and end word in the prediction step are somewhat independent. This can cause problems, as the model may have mutiple guesses for each and not make the correct pairings. Consider, for instance, figure 7 where the model gives weight to several start and end positions. How these should be paired is non-obvious, and the current algorithm just takes the argmax of $\hat{\mathbf{y}}$ for each. This can even lead to situations where the predicted start index is *after* the predicted end index (*facepalm). A more sophisticated prediction scheme could likely remedy some of these errors, for instance the highway max-out network used by Xiong et al [6].

3.6 future studies

In this work only the 100-dimensional GloVe vectors trained on 6 billion tokens was used. It may be useful to try the 300-dimensional vectors trained over a 840 billion common-crawl corpus, which

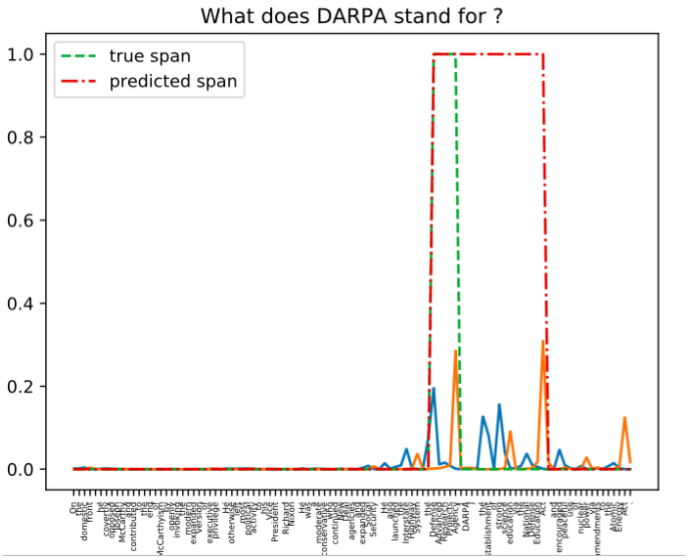

Figure 7: True answer span (dashed green line), predicted span (red dot-dash line) $\hat{\mathbf{y}}_{start}$ (solid blue) and $\hat{\mathbf{y}}_{\mathbf{end}}$ (solid orange).

has shown superior performance on other tasks [9]. The dropout regularization technique, also not used in this work, is also a promising avenue for boosting performance [13]. More careful studies of the state sizes of the hidden layers as hyperparameters is also necessary. It should be noted the optimal state size is likely to be influenced by the dropout rate and the size of the input word vectors [13].

If the above studies proved useful, it might be interesting to take this idea a step further and try more than two iterations of attention, seeing what number of iterations yields optimal performance. It is not unreasonable to imagine a system where the number of attention iterations is variable and decided at compute-time. Understanding how the attention shifts in these iterations and what question-types perform better with more iterations may be insightful for future developments in MC.

4 Conclusions

I have proposed the iterative attention network for question answering. An architecture with two attention iterations is demonstrated to offer a significant performance boost over a similar baseline model with a single attention iteration. It is also found that in the second iteration, the model focuses its attention on query words which are specific to the form of the answer. Although the demonstrated model does not achieve state-of-the-art performance, it also using relatively low-dimensional word vectors, lacks dropout regularization, has not undergone thorough hyperparameter tuning, and was trained by a sub-par AI researcher. Remedying these deficiencies, and perhaps also employing something similar to the highway max-out network used by Xiong et al [6], shows promise.

Acknowledgments

The author would like to acknowledge several people. First his roommate and fellow physicist, Scott Kravitz, for distractions and emotional support during this difficult time. He also thanks Prof. Manning, Dr. Socher and the entire CS224n TA staff. This was a wonderful class. I learned a lot, but more importantly, had a lot of fun doing it.

References

[1] Peter Norvig. Unified theory of inference for text understanding. Technical report, DTIC Document, 1986.

[2] Léon Bottou. From machine learning to machine reasoning. *Machine learning*, 94(2):133–149, 2014.

[3] 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. In *EMNLP*, 2014.

[4] Matthew Richardson, Christopher JC Burges, and Erin Renshaw. Mctest: A challenge dataset for the open-domain machine comprehension of text. In *EMNLP*, volume 3, page 4, 2013.

[5] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[6] Caiming Xiong, Victor Zhong, and Richard Socher. Dynamic coattention networks for question answering. *arXiv preprint arXiv:1611.01604*, 2016.

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

[8] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[9] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[10] David Sussillo and LF Abbott. Random walk initialization for training very deep feedforward networks. *arXiv preprint arXiv:1412.6558*, 2014.

[11] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[12] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[13] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

A Examples of Questions containing Words towards which attention was shifted

question: In what year was Cesar made pontifex maximus?
guessed answer: 63 BC
actual answer: 63 BC

question: How many shares of OTE does the Greek state own?
guessed answer: 10 %
actual answer: 10 %

question: Youtube is ranked what on the world 's list of most visited sites ?
guessed answer: third
actual answer: third

question: What country is more religious than other developed nations ?
guessed answer: United States
actual answer: United States

question: What is it called when a visible mound forms on the surface of a plate?
guessed answer: a colony
actual answer: a colony