Attention mechanisms: analyzing relevance for SQuAD models

Martín De la Fuente
Pontificia Universidad Católica de Chile
mdelafuente1@uc.cl

Ignacio Guridi
Pontificia Universidad Católica de Chile
iguridi@uc.cl

Raimundo Herrera
Pontificia Universidad Católica de Chile
rjherrera@uc.cl

Abstract

We analyze the impact of attention mechanisms in the performance of models for Machine Reading Comprehension tasks. For this, we implemented a number of models, first without attention, second with a simple attention mechanism and third with a more complex mechanism, such as Bidirectional Attention Flow (BiDAF). We explore and compare this options against the Stanford Question Answering Dataset (SQuAD), confirming the importance of attention in the performance of the models, giving insight of the reasons why it is so used in almost every state of the art model.

1 Introduction

Answering questions in natural language is an investigation area that has gained popularity in the last few years among artificial intelligence researchers. It main goal is to provide natural answers with a computer system that is able to understand human knowledge. This problem is far from being an easy task, especially because when we answer those question we do it using a lot of implicit information and based in our own experience. For this work we will be working with the Stanford Question Answering Dataset (SQuAD) in its first released version (1.1).

SQuAD is a reading comprehension dataset consisting of over 100,000 question-answer pairs on 500 articles [7]. The most important characteristic of this dataset is that every answer can be directly extracted from the paragraph associated with the question. This means we can use a neural network, feed it with a pair (context, question) and get as a prediction a pointer to the answer in the given context.

Since SQuAD was released, a lot of researchers have been trying to improve the state-of-the-art scores and many different implementations alongside new techniques have been developed. By the date this article was written there are many works rivaling human performance on the task, and some of them even surpassed humans in specific metrics (F1-Score).

In the remainder of this work we discuss some model implementation that have shifted the state of the art scores and have allowed other implementations (built on top of them) to reach the best results. In particular we examine Bidirectional Attention Flow and see how its attention mechanism produce much better results, comparing it with simple attention and not attention at all.

In Section 2 we present what attention is, how it is implemented and how BiDAF takes advantange of this mechanism. Also, attention related work from different authors are mentioned in order to have a broader idea on how we can use it for machine learning. Section 3 explains our proposed method to evaluate BiDAF attention and compares it with models using different levels of attention.

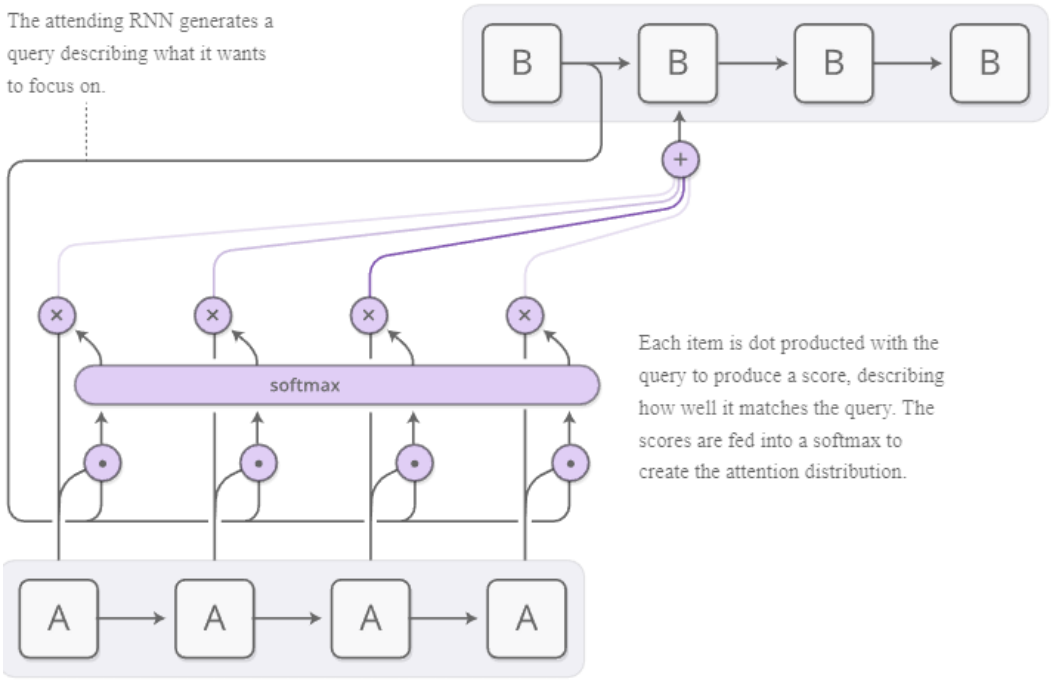

Figure 1: Attention implementation between two RNN layers. The attention distribution is calculated using softmax function over the input. Extracted from [5].

In Section 4 we display the results obtained in our comparisons and in the last section most important conclusions are given.

2 Background

2.1 Technical Aspects

Recurrent neural networks (RNN) have become very popular in the field of deep learning due to their capacity to deal with sequences. Another variants, such as LSTM and GRU are also widely used in this field because they can handle even longer sequences. Such models have been found to be very powerful, achieving remarkable results in many tasks including translation, voice recognition, and image captioning [5].

As this has happened, there are many attempts to augment RNN with new properties [5], one of them is called attention. To introduce this concept try to think in the way we humans perform complex actions. For instance, if we are translating a text we don't do it translating word by word, and either do we care about a word we wrote three paragraphs before, instead, we usually focus on the closest words or the whole sentence. In the same way, if we are recognizing faces in an image we don't focus in the background, but in the people present. This behavior is what attention tries to replicate.

The way attention is implemented in a RNN, is estimating an attention distribution from the input vector. The attention distribution is a vector that contains values between 0 and 1 describing how important is each one of the values in the input (we do this using a softmax function). To feed the next layer we dot product the input vector itself with the attention distribution, this way we are attending more to the input values that the attention distribution has determined. In the training process, the parameters in the attention layer learn where to attend and to what extent. Figure 1 describes this architecture.

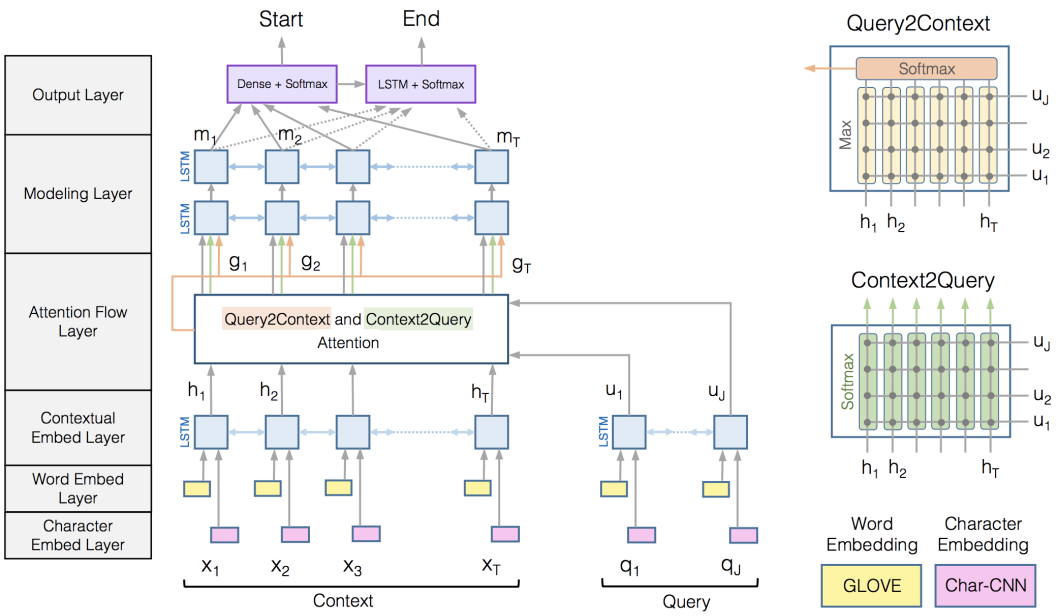

Figure 2: BiDAF model architecture, with the bidirectional context-to-query and query-to-context flows.

2.2 Related Work

A lot of researchers have developed techniques that improve the results in the field of question answering, and a many of them do so building their models on top of the basic attention mechanism and the BiDAF architecture.

For example, *Liu, Hu, et al.* [4] propose a new model using structural embedding of syntactic trees (SEST) and combine it with BiDAF. They boost the performance of algorithms for machine comprehension not using only word and char embeddings, but also taking advantage of the structured information present in written language [4].

Clark and Gardner [1] also extend the use of BiDAF to a broader problem, where they can apply QA not to single paragraphs but to entire documents. Changing a bit the way of training the network, they managed to obtain very high scores in a different dataset called TriviaQA, surpassing by far the state of the art in that dataset.

3 Proposed Method

In our work we implement and test three models. The model architectures are described in the following subsection, alongside some comparison between all models and their attention mechanism approach. Also, we provide and define the metrics and discussion that will be displayed in Section 4.

3.1 Model Architectures

The BiDAF[8] model architecture is shown in Figure 2, it is mainly composed of six layers, a **character embedding layer**, which maps words to vectors using `CNNs`, a **word embedding layer**, which maps words to vectors using `word embedding`, a **contextual embedding layer** which refines the embedding of the words, a **attention flow layer**, which is the main achievement of their work, a **modeling layer**, which scans the context via `RNNs`, and finally an **output layer** to give an answer.

Our implementation of Seo, et al. (2016) [8] work, is based on the AllenNLP [2] research library, we used the BiDAF network with some details in the structure. For the word embedding layer we used GloVe[6] in its six billion tokens and 100 dimension version. The contextual layer is also bidirectional with 100 hidden neurons, mapping the embeddings to those neurons.

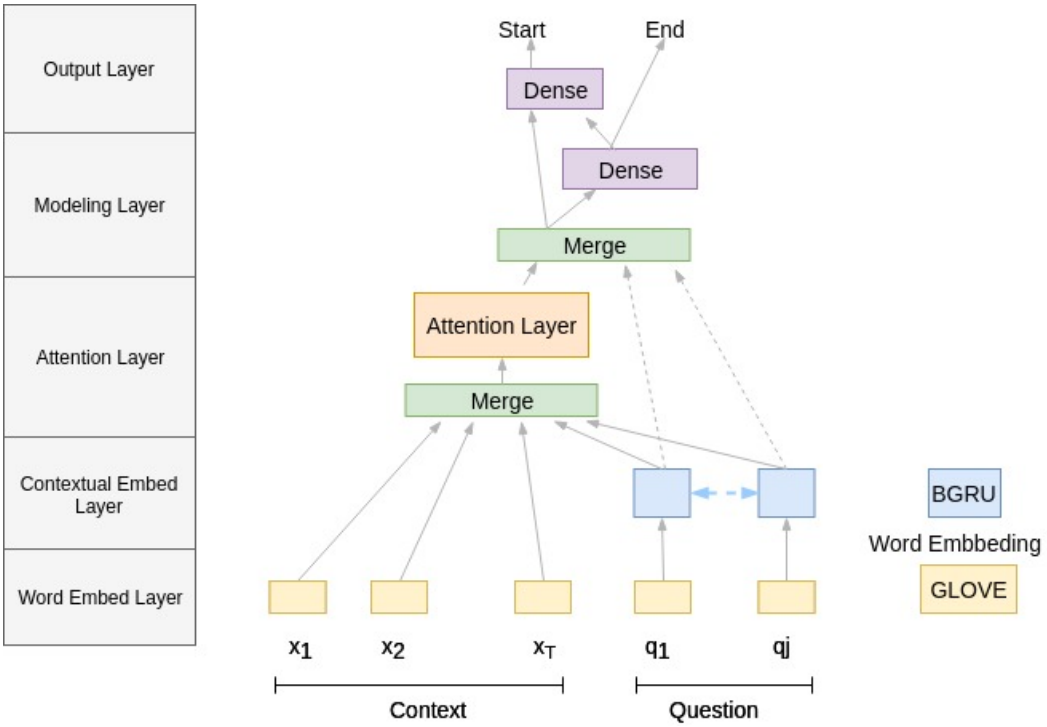

Figure 3: DRAN model architecture

The bidirectional attention flow is implemented by a similarity layer, which uses a linear function to combine the context vectors with the query ones, and vice versa. The modeling layer has also 100 hidden neurons with a dropout of 0.2 as stated in the original BiDAF work. The last layer is implemented as bidirectional LSTM with 1400 inputs and, again, 100 hidden neurons. Finally, its important to notice that the optimizer used for training time was Adam [3] instead of the originally proposed AdaDelta[9], as it has better training performance for this case.

The second model architecture is displayed in Figure 3, it is composed of seven layers. First is the encoding layer, then over the question it is a `bidirectional GRU`. then the third layer merges the contexts and the second layer output and forwards it to the `attention layer`, explained later. After that, there is a merge layer which takes the outputs of the attention layer and the second layer, this goes through a couple of dense layers, as seen in the picture 3 to get the beginning and ending of the answer.

The third model is a Non-Attentive Neural Network (NANN) model. We implemented this model as a naive approach to the QA problem, using the most simple embedding for words (Glove 50D), two bidirectional LSTM (for question and context) and merging their output to feed a dense layer. Finally, the output is generated in the last dense layer (with two units) that outputs the start and end positions in the context. This model doesn't use any attention mechanism and has about 120,000 trainable parameters, so it can be considered as a very simple approach. Figure 4 represents this model.

3.2 Model Comparison

Our work focuses on the comparison of the three models described previously, BiDAF[8], DRAN[7] and our simpler and naive implementation NANN. The main difference between the three implementations is the way in which attention is approached.

In the first one, they propose an attention flow [8], which not only considers the context information but also the query information. The vectors produced are *queryaware* and at the same time *contextaware*. The proposed method is called bidirectional because of that, the efforts to include information of both sides affecting one another seamlessly. To achieve the latter, the intermediate attention layer has two main types of attention, a context-to-query one and a query-to-context one.

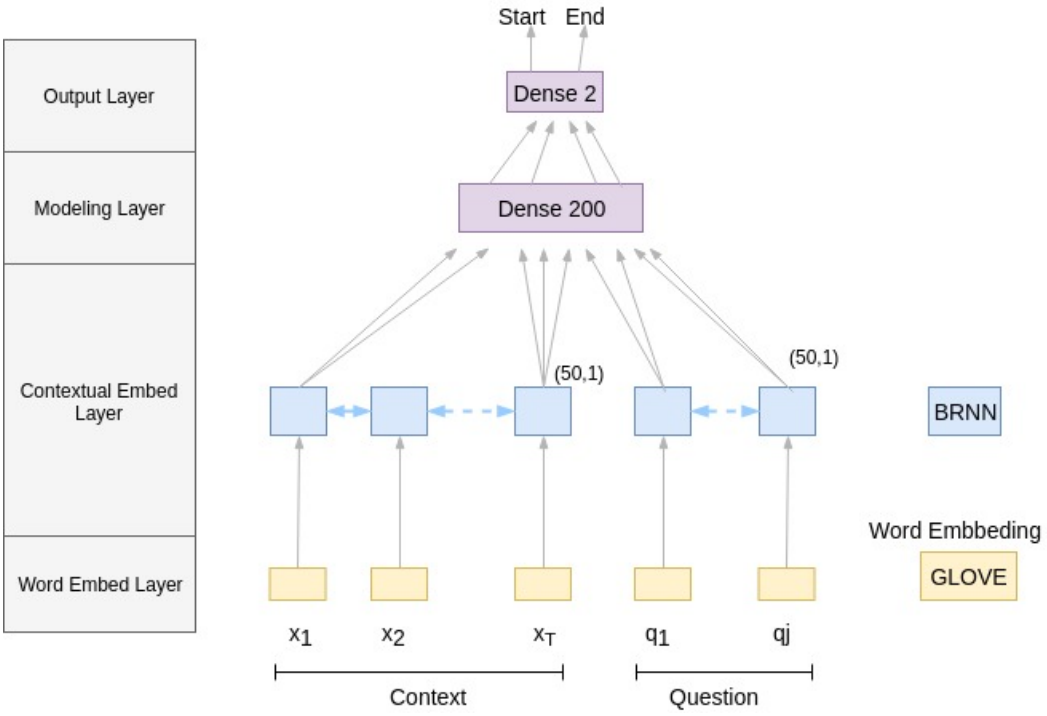

Figure 4: NANN model architecture

This flow method ensures that the model will consider (1) the most relevant query words for each context word and (2) the most similar context words for each query word. This is achieved using simple summations and operating with `softmax` functions over each word vector in each direction and finally combining both directions in a common matrix.

In the DRAN model the attention is a very simple implementation, which uses dot products between the `softmax` of the previous layer output, and the original input vector. This determines which vector component is important i.e. which vector the model pays attention to.

In the NANN model, there is no attention model, which we will discuss as one of the main weaknesses of the model.

We ran the three models for 10 hours, and checked for the scores, F-1 and exact match, after the amount of epochs that finished in the elapsed time. We compare our implementations of the above models and the implementations found in their papers, with their results.

4 Results

Our work unveiled that attentive models perform significantly better than non-attentive ones. Our scores, displayed in Table 1, show that this models require long seasons of training, as 10 ours gave poor improvement from random results. The DRAN model displayed in their work, performs way better than what we achieved in 10 hours of training, and our NANN model also performs poorly with that amount of training.

Nonetheless, we confirmed watching the training, that even though loss was decreasing and accuracy improving, the F-1 score and the exact match -the metrics used to evaluate-, did not improve accordingly. This is mainly due to the difference between metrics employed to train and metrics employed to evaluate. In the SQuAD[7] dataset definition, they propose the two metrics we have discussed, but in training time other metrics are used, this makes overfitting not *useful* for any model, as memorizing the whole dataset will not help to achieve better results as the results are evaluated (1) in a different dataset, (2) with different metrics. We think that, as the datasets are similar, the latter reason is more important than the former one.

Table 1: Metrics comparison between models

Model	EM	F1
NANN	1.72	1.89
DRAN	10.89	15.87
BiDAF	63.13	72.11
Fully trained DRAN	60.8	55.6
Fully trained BiDAF	68.37	77.85

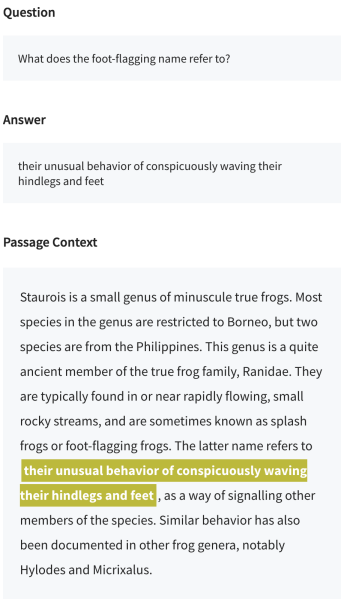

Figure 5: Fully trained BiDAF answer (correct).

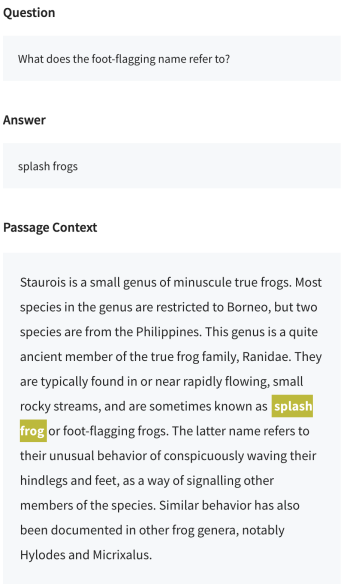

Figure 6: DRAN answer (incorrect).

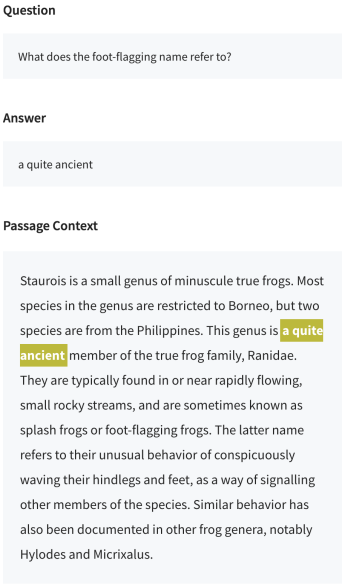

Figure 7: NANN answer (incorrect & backwards).

We also include a question-answer experiment. We sampled one animal description, and asked in Figure 5 the BiDAF fully trained model with web obtained weights, and got the correct answer. We did the same experiment with our trained implementation of the DRAN model in Figure 6, which failed, but it's answer is a plausible one, this shows that with more training, this result could be improved. Finally we asked the same question to our NANN model in Figure 7, and discovered an unusual behavior, the ending position for the answer was placed before the starting position. Nevertheless, once we corrected that behavior for this sample, switching those positions, the answer didn't make sense either, which show how poorly models without attention perform.

5 Conclusions

We made an analysis of the impact of attention mechanism in Machine Reading Comprehension tasks. According to the results, its existence it's crucial for acceptable results in the SQUAD dataset. Also, we observed that this method has been a big step in Machine Comprehension, being a must have for many subsequent projects in this area, who have developed and improved this method and will continue to do so. Also, we stumbled against the fact that the metrics used in training aren't the same as the used when evaluating. This means that excessive training generally produces over-fitting and doesn't make the model more robust. This calls the need to create better metrics to solve this problem.

References

[1] Christopher Clark and Matt Gardner. Simple and effective multi-paragraph reading comprehension. *CoRR*, abs/1710.10723, 2017.

[2] Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson H S Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. A deep semantic natural language processing platform. 2017.

[3] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[4] Rui Liu, Junjie Hu, Wei Wei, Zi Yang, and Eric Nyberg. Structural embedding of syntactic trees for machine comprehension. *CoRR*, abs/1703.00572, 2017.

[5] Chris Olah and Shan Carter. Attention and augmented recurrent neural networks. *Distill*, 2016.

[6] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[7] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250, 2016.

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

[9] Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.