Bi-Directional Attention Flow for Question Answering

Amirpasha Ghabussi, Dhruv Kumar, Gaurav Sahu, Kashif Khan

University of Waterloo - STAT-946, Project report

Abstract

Question answering, i.e., answering questions given a context, involves learning the complex relationship between a given context and a question. The development in attention mechanisms have aided the task of QA. Attention mechanisms act as a *soft pointer* guiding the model in making more informed prediction. In this paper, Bi-Directional Attention Flow (BiDAF) mechanism has been introduced, which obtains a query-aware representation of the context. We implemented the proposed mechanism and trained it on Stanford Question Answering Dataset (SQuAD) Rajpurkar et al. [1]. We obtain an Exact Match (EM) score of 53.25 % and F1-score of 71.90 %.

1. Introduction

The task of Question Answering (QA) has become recently popular within the natural language processing community. The development of neural attention mechanisms, which enables a model to focus on specific parts of the input paragraph (thus, acting as soft-pointers), has improved the performance of the already-existing architectures. Traditionally, attention mechanisms have been used in the following ways: First, the contexts are summarized into a fixed-size vector from which the computed attention weights extract the most relevant information; Second, they are defined as a function of the attended vector so as to make them temporarily dynamic (in the text domain); Third, they are applied only in one direction wherein the query attends on the given context paragraph.

In this paper, a hierarchical multi-stage architecture, namely, Bi-Directional Attention Flow (BiDAF) network has been proposed for capturing the interactions between context and query. BiDAF obtains a query-aware context representation using a combination of character-level, word-level, and contextual embeddings along with the bi-directional attention flow. The proposed mechanism is different from what has been used traditionally for attention - it does not summarize the context to a fixed-size vector; is memory-*less* and only considers the query and the context at the current time. We conjecture that the latter characteristic allows the BiDAF network to learn the attention between query and context in a better way.

2. Model Architecture

The proposed multi-stage, hierarchical network consists of six distinct layers (Figure 1)

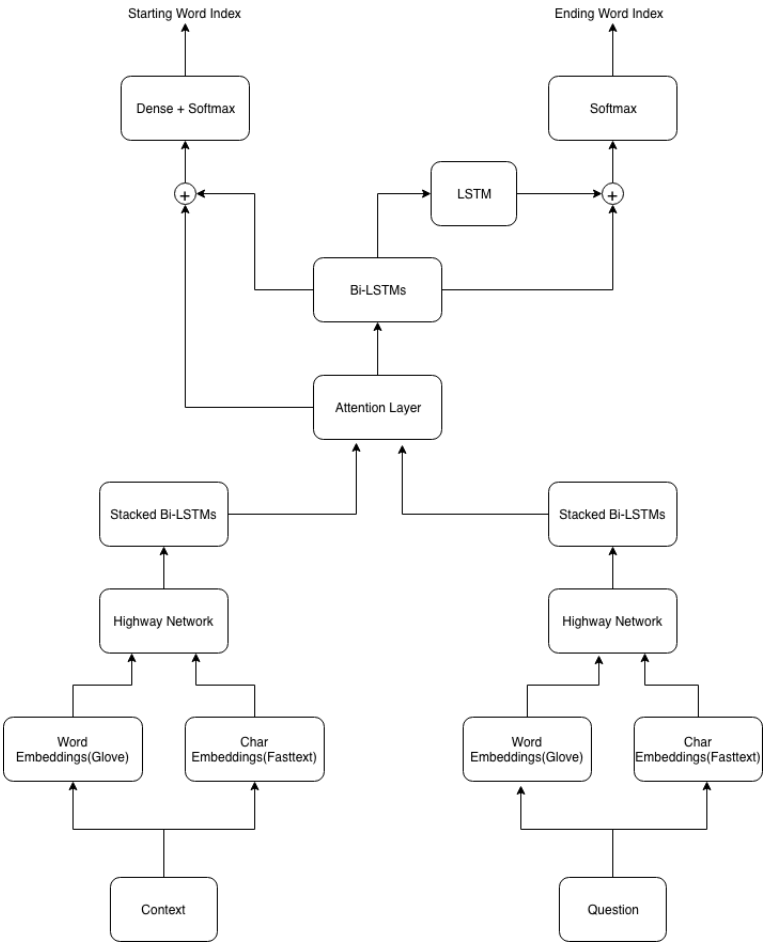

Figure 1: BiDirectional Attention Flow Model

1. **Character Embedding Layer**: If $\{x_1, ...x_T\}$ and $\{q_1, ...q_J\}$ denote the words in the input context and query, this layer maps each word to a high-dimensional vector space. We use fastText [2] to learn the character embeddings for words in our vocabulary.
2. **Word Embedding Layer**: This layer also maps each word to a high-dimensional vector space. We use the pre-trained GloVe [3] vectors to obtain the fixed word embedding. Finally, the character and the word embeddings are concatenated and passed through a two-layer highway network, which outputs two separate sets of d-dimensional matrices, namely, $\mathbf{X} \in \mathcal{R}^{dxT}$ and $\mathbf{Q} \in \mathcal{R}^{dxJ}$ for the context and the query respectively.

3. **Contextual Embedding Layer**: This layer consists of a forward and a backward Long Short-term Memory (LSTM) cell [4] and is placed on top of the previously generated embeddings. The final output of this layer is two matrices, $\mathbf{H} \in \mathcal{R}^{2dxT}$ and $\mathbf{U} \in \mathcal{R}^{2dxJ}$ for $\mathbf{X}$ and $\mathbf{Q}$ respectively. It is to be noted that the dimensions are $2d$ due to the concatenation of d-dimensions vectors from the forward and the backward LSTM cells.
4. **Attention Flow Layer**: This is the most important layer of the BiDAF network. As mentioned earlier, this attention mechanism differs from other prominent attention mechanisms in the literature (Rush et al. [5], Hill et al. [6], Sordoni et al. [7], Shen et al. [8]). It does not summarize the context and query to a fixed-size vector and allows the attention vector (at each time step) along with the contextual embeddings to *flow* through the subsequent modelling layer. Two types of attention are calculated in this layer:

    - **Context-to-query attention (C2Q)**: This matrix contains information about the most relevant query words given a context paragraph. If $\mathbf{a}_t \in \mathcal{R}^J$ represents the attention weights of query words with respect to the t-th context word, $\sum \mathbf{a}_{tj} = 1 \ \forall \ t$. The individual attention weights are computed by $\mathbf{a}_t = \text{softmax}(\mathbf{S}_{t:}) \in \mathbf{R}^J$, and the final attention query vector is given by $\tilde{\mathbf{U}}_{:t} = \sum_j \mathbf{a}_{tj}\mathbf{U}_{:j}$, where $\tilde{\mathbf{U}}$ is now a $2d$-by-T matrix containing the attended query vectors and $\mathbf{S} \in \mathcal{R}^{T\text{x}J}$ is a shared similarity matrix between $\mathbf{H}$ and $\mathbf{U}$, where $\mathbf{S}_{tj}$ indicates the similarity between the t-th context word and j-th query word. Overall, the matrix $\mathbf{S}$ is computed as:

    $$\mathbf{S}_{tj} = \alpha(\mathbf{H}_{:t}, \mathbf{U}_{:j}) \in \mathcal{R} \tag{1}$$

    Here, α is a function that needs to be learned within the model in order to capture the similarity between two given vectors. For this work, we have chosen $\alpha(\mathbf{h}, \mathbf{u}) = \mathbf{w}_{(S)}^T[\mathbf{h}; \mathbf{u}; \mathbf{h} \circ \mathbf{u}]$, where $\mathbf{w}_{(S)} \in \mathcal{R}^{6d}$ is a trainable weight vector, $[;]$ is a row-wise vector concatenation and $\circ$ is the element-wise multiplication.

- **Query-to-context attention (Q2C)**: This matrix contains information about the most relevant context words given a query text. The attention weights for this component are obtained as $\mathbf{b} = \text{softmax}(max_{col}(\mathbf{S})) \in \mathcal{R}^T$ and the attended context vector would then be $\tilde{\mathbf{h}} = \sum_t \mathbf{b}_t\mathbf{H}_{:t} \in \mathcal{R}_{2d}$. Since $\tilde{\mathbf{h}}$ is a row vector, it is tiled T times across the column generating

    The final output of the attention layer, denoted as $\mathbf{G}$, is calculated by combining the contextual embeddings (input to the attention layer) with the attended vectors (i.e. C2Q and Q2C) as:

    $$\mathbf{G}_{:t} = \beta(\mathbf{H}_{:t}, \tilde{U}_{:t}, \tilde{H}_{:t}) \in \mathcal{R}^{d_G} \tag{2}$$

    where $\beta(\mathbf{h}, \tilde{\mathbf{u}}, \tilde{\mathbf{h}}) = [\mathbf{h}; \tilde{\mathbf{u}}; \mathbf{h} \circ \tilde{\mathbf{u}}; \mathbf{h} \circ \tilde{\mathbf{h}}] \in \mathcal{R}^{8d \text{ x } T}$ with ; and $\circ$ as defined previously.
5. **Modelling Layer**: The modelling layer learns the conditional relationship between the context word and the query. It outputs a matrix $\mathbf{M} \in \mathcal{R}^{2d \text{ x } T}$ employing a bi-directional LSTM on $\mathbf{G}$.

6. **Output Layer**: The model derives a phrase from a given paragraph by predicting its start and end indices given a context. The probability distributions are derived as:

$$\mathbf{p}^1 = \text{softmax}({\mathbf{w}_{(\mathbf{p}^1)}}^T[\mathbf{G}; \mathbf{M}]), \tag{3}$$

$$\mathbf{p}^2 = \text{softmax}({\mathbf{w}_{(\mathbf{p}^2)}}^T[\mathbf{G}; \mathbf{M}^2]) \tag{4}$$

where $\mathbf{M}^2 \in \mathcal{R}^{2d \text{ x } T}$ is generated by passing $\mathbf{M}$ over a bi-directional LSTM layer.

3. Experiments and Results

We train the model to minimize the objective function:

$$L(\theta) = -\frac{1}{N}\sum_{i}^{N} \log({\mathbf{p}_{y_i}}^1) + \log({\mathbf{p}_{y_i}}^2), \tag{5}$$

where θ denotes the set of all parameters passed to the model. We use the same metrics as reported in the original paper, i.e., Exact Match (EM) and the F1 score, to evaluate our model.

We obtained the EM score of 53.25 and F1 score of 71.9 as compared to 67.7 and 77.3 in the original paper. In our model, we used the batch size of 64 with the Adam Optimizer, a learning rate of 0.001 and a decay of 0.9999. We also stacked 2 Bi-LSTMs in our implementation with the hidden state of 100 and 0.2 dropout. We used the 300 dimension version of Glove and Fasttext Embeddings.

We also compare our results with some implementations of BIDAF online. Most of them have a EM score in the range of 30-62 and a F1 score of 40-70. In comparison to other results, our F1 score is much better than the EM score.

4. Discussions

We have implemented the bi-directional attention flow network for the task of question answering. However, we were not able to fully replicate the original results which could be pertained to the lack of implementation details in the paper. There is considerable inconsistency observed in the model description and the provided open-source implementation. However, the performance of our implementation is significantly better than some other works such as (Beigi et al. [9], Parakkal [10], Tuason et al. [11]) which attempted to replicate the results.

We also analyzed some changes that we can do to this model to increase its performance. We could use an LSTM layer for predicting the starting index as well and further use the end state of this LSTM as the starting state of the LSTM we use in predicting the end-index. The intuition behind this is that this would help the model to learn extra information about the answer. For example, learning that starting index should be less than or equal to the end index. We also propose to experiment with other state-of-the-art embeddings like ELMo [12] and CoVe[13] in addition to Fasttext which we did in this work.

5. References

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

[2] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, T. Mikolov, Fasttext.zip: Compressing text classification models, arXiv preprint arXiv:1612.03651 (2016).

[3] J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word representation, in: Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543.

[4] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation 9 (1997) 1735–1780.

[5] A. M. Rush, S. Chopra, J. Weston, A neural attention model for abstractive sentence summarization, arXiv preprint arXiv:1509.00685 (2015).

[6] F. Hill, A. Bordes, S. Chopra, J. Weston, The goldilocks principle: Reading children's books with explicit memory representations, arXiv preprint arXiv:1511.02301 (2015).

[7] A. Sordoni, P. Bachman, A. Trischler, Y. Bengio, Iterative alternating neural attention for machine reading, arXiv preprint arXiv:1606.02245 (2016).

[8] Y. Shen, P.-S. Huang, J. Gao, W. Chen, Reasonet: Learning to stop reading in machine comprehension, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, pp. 1047–1055.

[9] B. Beigi, S. Hemmati, M. Painter, Cs224n project final report bidirectional attention flow model for reading comprehension (2017).

[10] B. Parakkal, Machine comprehension using bidaf (????).

[11] R. Tuason, D. Grazian, G. Kondo, Bidaf model for question answering (????).

[12] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer, Deep contextualized word representations, arXiv preprint arXiv:1802.05365 (2018).

[13] M. Feng, J. Liao, S. Qing, T. Li, J. Wang, Cove: Co-operative virtual network embedding for network virtualization, Journal of Network and Systems Management 26 (2018) 79–107.