Question Answering with Self-Matching Attention

Stanford CS224N Default Project
Track: SQuAD

Xavier Arguello
Department of Computer Science
Stanford University
xavier.arguello@stanford.edu

Ankit Patel
Department of Computer Science
Stanford University
gt3@stanford.edu

Abstract

Neural networks are state-of-the-art for question answering (QA). The basis of the success of neural architectures for QA, including Bidirectional Attention Flow (BiDAF) [1], is that question-passage similarity provides a strong signal to finding answer in the passage. Through our experiments, and without relying on pre-trained language models, we improve the baseline BiDAF model, to perform well on Stanford Question Answering Dataset (SQuAD version 2). In particular, we show how Self-Matching Attention (SMA), first used in R-Net [2], can mitigate information loss in the context-query attention mechanism, and provide considerable improvement to the baseline. In addition, we show how to integrate convolution output of character embedding with word embeddings. We analyze contributions of these techniques in the final model.

1 Key Information to include

- TA mentor: Christopher Wolff
- External collaborators: "No"
- Sharing project: "No"

2 Introduction

The goal of an end-to-end QA system is to extract information from a given source (a context) such as a passage, document, image, etc. based on the user's request (a query). The effectiveness of such a system lies in its ability to provide concise and accurate answers. Even in this simple paradigm, where an answer is generated based on the user's question, the number of use cases is quite large. For example, the question could be fact-based or it might require extensive reasoning as in a Math equation solver. Similarly, in machine translation, answers are required in a different language which makes designing of these systems quite challenging.

The intricacies of Natural Language Processing (NLP) has inspired researchers to come up with general techniques that can address a broad range of use cases. In recent years, models trained in a particular domain have shown outstanding results. For instance, we have already surpassed human performance on SQuAD QA! [1]

These models perform exceptionally well in the domain they are trained on. Although when applied to another domain or in the presence of adversarial examples, their performance is subpar as they are not able to generalize well [3]. The quest to unified multi-task learning is an active area of research.

[1] https://rajpurkar.github.io/SQuAD-explorer

The requirements for our project are much simpler. Given a context (paragraph) and a query (question), the goal is to find an answer within that text as shown in the SQuAD 2.0 example below.

Query: Why was Tesla returned to Gospic?

Context: 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 large class of students in his old school, Higher Real Gymnasium, in Gospic.

Answer: not having a residence permit

In addition to locating the answer span, the model is also required to predict if the question is unanswerable. This requires extensive reasoning power as the model needs to rule out the possibility of any answer appearing in the passage. Perhaps this is why reading comprehension is considered the testbed for evaluating how machines understand human language [4].

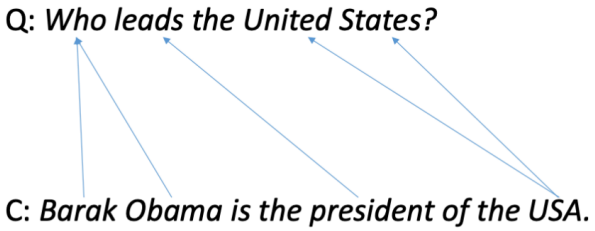


Figure 1: Relevant pairings of words $(context \rightarrow query)$

It's worth mentioning that Attention models compared to Recurrent ones have gained in popularity in recent years. A parallelizable Transformer architecture is more suited to NLP tasks. While many pre-trained language models also implement the Tranformer model, the task at hand is to implement techniques based on existing research.

To that end, we layout our approach to improving the baseline model in two key ways:

1. Implement character-level embeddings.
2. Implement Self-Matching Attention (SMA) as described in R-Net [2].

Our leaderboard scores are a reflection of the improvements arising from the two approaches mentioned above.

3 Related Work

The birth of the SQuAD dataset has ushered in vast research from the ML community towards building better Language Models. Question Answering in particular has gained momentum since that time. BiDAF [1] is a great example of a successful implementation of a recurrent model with context-query attention mechanism at its core. Even with the advent of Transformer architecture [5], a non-recurrent approach, the primary way of encoding information about the passage and question has not changed. These Attention-based approaches continue to thrive and evolve.

For example, QANet [6] achieved much higher score on the SQuAD leaderboard[1] by combining ideas from BiDAF and Transformers. Models like QANet also show improvements in architecture can boost training performance.

The landscape seems to be changing yet again with the introduction of large pre-trained models such as BERT [7]. These models reduce the task of implementing state-of-art models to a plug-n-play approach. We'll consider the topic of applying pre-trained models out of scope for the purposes of this paper and instead focus on exploring techniques at a more foundational level. After all these large LMs have stemmed from the ground-breaking research, first explored at the grassroots-level [8]. So it pays to study them well!

GNNs on the other hand offers a fresh new perspective to this dynamic field. They can harness the expressive power of knowledge graphs that produce more explainable predictions [9, 10]. This is certainly an area of research to keep an eye on.

4 Approach

Our approach to improving the baseline model is two-fold: improvements to embeddings and attention.

4.1 Baseline

The provided starter code[1] trains on SQuAD 2.0 dataset. BiDAF model with pre-trained 300-dimensional GloVe word embeddings is already implemented.

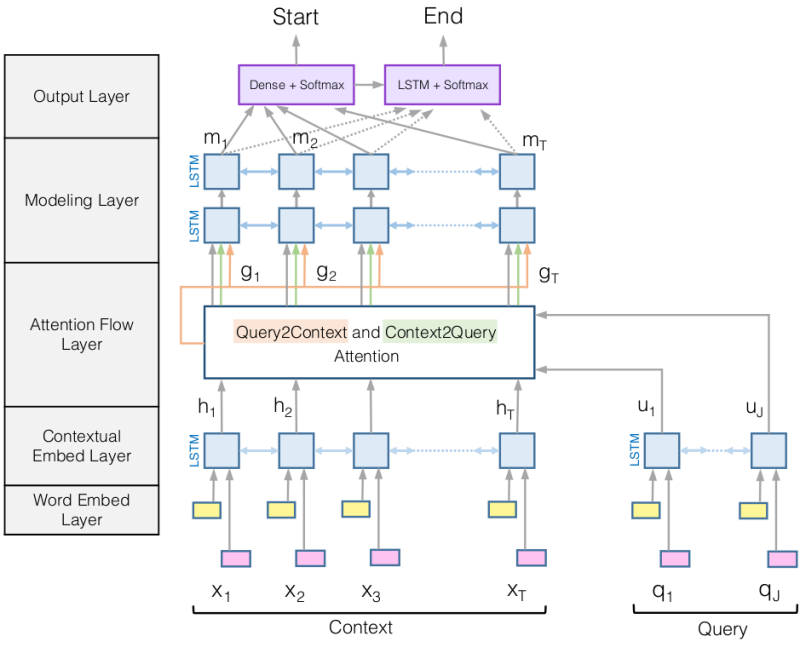


Figure 2: Bidirectional Attention Flow Model Architecture [1]

4.2 Improvements to Embeddings

The main idea is to combine word and character-level embedding for every word in the embedding layer. Word-based models cannot deal with unknown (or misspelled) words i.e. words not in vocabulary. Character embedding alleviates this issue and helps with morphology. The challenging part is in doing it efficiently since each word is made up of multiple characters. For each word we generate a fixed-length vector of predetermined size by first convolving over character embeddings and then applying max-pooling in the dimension of word length. This vector is then concatenated with word embeddings before passing through a projection and highway network f as shown below.

$$e(c_i) = f\left([\, GloVe(c_i),\ CharEmb(c_i)\,]\right) \tag{1}$$

$$e(q_i) = f\left([\, GloVe(q_i),\ CharEmb(q_i)\,]\right) \tag{2}$$

$e(c_i)$ and $e(q_i)$ are then passed through two bidirectional LSTMs separately to produce contextual embeddings for context and query [1].

[1] `https://github.com/michiyasunaga/squad.git`

4.3 Improvements to Attention

One limitation of the attention mechanism, in the BiDAF model, is that the query-aware condensed representation of the context has limited information of the context (3), (4). Running inference on this condensed representation to get an answer is less than ideal. Important cues from the context stay hidden from the answer candidate.

$$\alpha_{i,j} = softmax_j(S_{i,j}) \qquad a_i = \sum_{j=1}^{M} \alpha_{i,j} q_j \tag{3}$$

$$\beta_i = softmax_i(max_{j=1}^{M}(S_{i,j})) \qquad b = \sum_{i=1}^{N} \beta_i c_i \tag{4}$$

Our approach, as described in [2], shows that when an answer candidate has sufficient context, it improves the quality of inference. Specifically we implement Self-Matching Attention (SMA) to mitigate the information loss and derive an *aggregate* context representation that extracts evidence from the *entire* context w.r.t. current context word and query.

We first construct c_t as the attention-pooling vector of the entire context v^P.

$\mathrm{s}_j^t = v^T tanh(W_v^P v_j^P + W_v^{\bar{P}} v_t^P)$
$a_i^t = \frac{\exp(s_i^t)}{\sum_{j=1}^{n} \exp(s_j^t)}$
$c_t = \sum_{i=1}^{n} a_i^t v_i^P$

Therefore c_t can be succinctly expressed as:

$\mathrm{c}_t = att(v^C, v_t^C)$

Now we adaptively control the input $[v_t^P, c_t]$ by passing it through a gated attention-based RNN. Thus the final context representation h_t^P dyanmically collects signal from the entire context for words in context and encodes it to the current context word and its matching query.

$$h_t^P = BiLSTM(h_{t-1}^P, [v_t^P, c_t]) \tag{5}$$

5 Experiments

5.1 Data

We'll be using the SQuAD 2.0 dataset with custom dev and test sets. The official test set is unknown and reserved for final evaluation.

- **train**: 129,941 examples
- **dev**: 6078 examples
- **test**: 5915 examples

The dataset contains records of $(context, question, answer)$ triples of both answerable and unanswerable questions. The training set has one answer per question whereas the dev set has three answers for every question. In addition, 300 dimensional GloVe word embeddings and 64 dimensional character embeddings are provided.

5.2 Evaluation method

We use the SQuAD official Exact Match (EM) and F1 metrics for quantitative evaluation of our model.

EM score measures whether the predicted answer span exactly matches the ground truth. F1 score is the harmonic mean of precision and recall. Precision (p) is calculated as the number of correct words divided by length of predicted answer. Recall (r) is calculated as number of correct words divided by length of ground truth.

$$F1 = 2\,\frac{r\,p}{r+p} \tag{6}$$

To track the classification accuracy of no-answer predictions, we use the recommended Answer vs. No Answer (AvNA) metric. It simply states the percentage of correct predictions.

5.3 Experimental details

We used the default configuration to train the baseline model:

- character embedding size: 64
- characters in a word: 16 maximum

After applying character-level embeddings, we trained the model with the following hyper-parameters:

- hidden state size: 100
- batch size: 32
- learning rate: 0.5
- word dropout rate: 0.2
- char dropout rate: 0.05
- L2 weight decay: 0.0005

The training time was ~ 15 minutes for each epoch on Tesla V100. All models were trained for a maximum of 30 epochs.

5.4 Results

Results of model evaluation on dev and test sets are summarized in Table 3, and also reflected on their respective leaderboards.

Model	F1	EM	AvNA	F1 (Test)	EM (Test)
Baseline	61.52	58.27	68.27		
Char Embedding	63.84	60.46	70.24		
SMA	66.92	63.62	72.14	65.67	62.47

Figure 3: Performance scores on SQuAD 2.0 dev and test sets

We see modest improvements for both enhancements. Integrating learnable character embeddings leads to considerable improvement to the baseline. This improvement is attributable to the enhanced ability of the model to receive extra bit of signal to learn word meanings, and thus match words and infer answer spans better.

The results also highlight the effectiveness of Self-Matching Attention as described in [2]. Scanning the entire context and aggregating signal relevant to the current context word and query, limits the information loss and produces better predictions.

Plots of the evaluation results are shown in Appendix A.

6 Analysis

Included below is our observation on few interesting examples from dev set evaluation.

6.1 Baseline Sample

Query: How is worst-case time complexity written as an expression?

Context: If the input size is n, the time taken can be expressed as a function of n. Since the time taken on different inputs of the same size can be different, the worst-case time complexity T(n) is defined to be the maximum time taken over all inputs of size n. If T(n) is a polynomial in n, then the algorithm is said to be a polynomial time algorithm. Cobham's thesis says that a problem can be solved with a feasible amount of resources if it admits a polynomial time algorithm.

Answer: T(n)

Prediction: N/A

Word embeddings do not match mathematical expressions well.

6.2 Char Embedding Sample

Query: What president eliminated the Christian position in the curriculum?

Context: Charles W. Eliot, president 1869–1909, eliminated the favored position of Christianity from the curriculum while opening it to student self-direction. While Eliot was the most crucial figure in the secularization of American higher education, he was motivated not by a desire to secularize education, but by Transcendentalist Unitarian convictions. Derived from William Ellery Channing and Ralph Waldo Emerson, these convictions were focused on the dignity and worth of human nature, the right and ability of each person to perceive truth, and the indwelling God in each person.

Answer: Charles W. Eliot

Prediction: Charles W. Eliot

Possible example of a win for learnable character embeddings for entity name recognition.

6.3 SMA Sample

Query: What cells do plants and animals both have?

Context: Unlike animals, plants lack phagocytic cells, but many plant immune responses involve systemic chemical signals that are sent through a plant. Individual plant cells respond to molecules associated with pathogens known as Pathogen-associated molecular patterns or PAMPs. When a part of a plant becomes infected, the plant produces a localized hypersensitive response, whereby cells at the site of infection undergo rapid apoptosis to prevent the spread of the disease to other parts of the plant. Systemic acquired resistance (SAR) is a type of defensive response used by plants that renders the entire plant resistant to a particular infectious agent. RNA silencing mechanisms are particularly important in this systemic response as they can block virus replication.

Answer: N/A

Prediction: N/A

This is an interesting example where the model was not thrown off by the complex reasoning and sentence structure: "Unlike animals, plants lack phagocytic cells"; instead predicting "no answer" correctly.

7 Conclusion

Through our work, we highlighted the importance of deep reasoning in Question Answering. We demonstrated how to attain a better representation for question and passage using self-matching atten-

tion. Our model achieved considerable improvement to the baseline BiDAF on all three metrics. One of the primary challenges these models face is the frequent occurrence of "made up" answers. This has severe negative implications for user-facing systems. It shows that despite recent success of large pre-trained language models, the field of reading comprehension has much to learn. Understanding of human language is still considered somewhat of a mystery.

References

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

[2] Natural Language Computing Group. R-net: Machine reading comprehension with self-matching networks. May 2017.

[3] Priyanka Sen and Amir Saffari. What do models learn from question answering datasets? *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

[4] Chenglei Si, Ziqing Yang, Yiming Cui, Wentao Ma, Ting Liu, and Shijin Wang. Benchmarking robustness of machine reading comprehension models. *CoRR*, abs/2004.14004, 2020.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[6] Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. Qanet: Combining local convolution with global self-attention for reading comprehension. *CoRR*, abs/1804.09541, 2018.

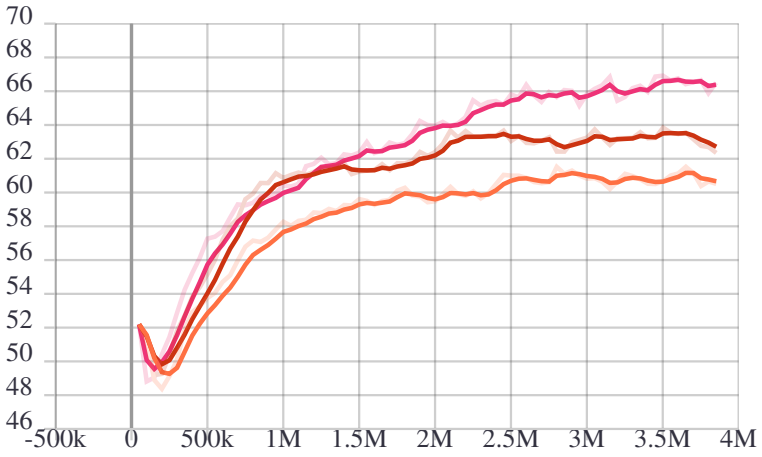
[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[8] Nelson F. Liu, Tony Lee, Robin Jia, and Percy Liang. Can small and synthetic benchmarks drive modeling innovation? A retrospective study of question answering modeling approaches. *CoRR*, abs/2102.01065, 2021.

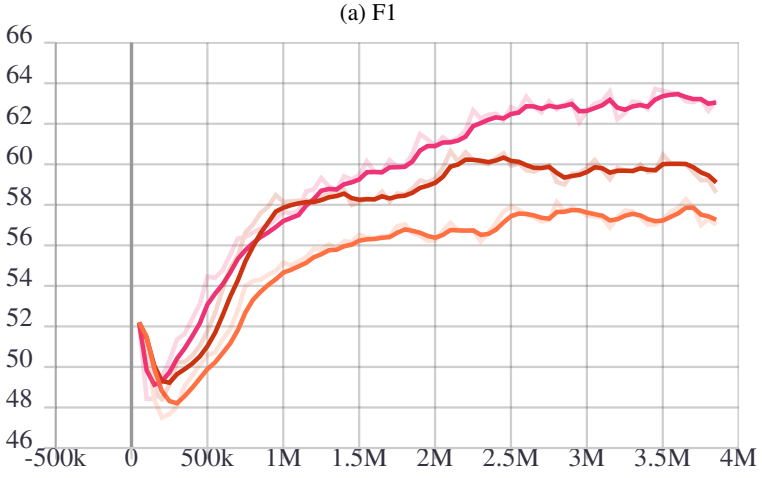
[9] Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D. Manning, and Jure Leskovec. Greaselm: Graph reasoning enhanced language models for question answering. *CoRR*, abs/2201.08860, 2022.

[10] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. QA-GNN: reasoning with language models and knowledge graphs for question answering. *CoRR*, abs/2104.06378, 2021.

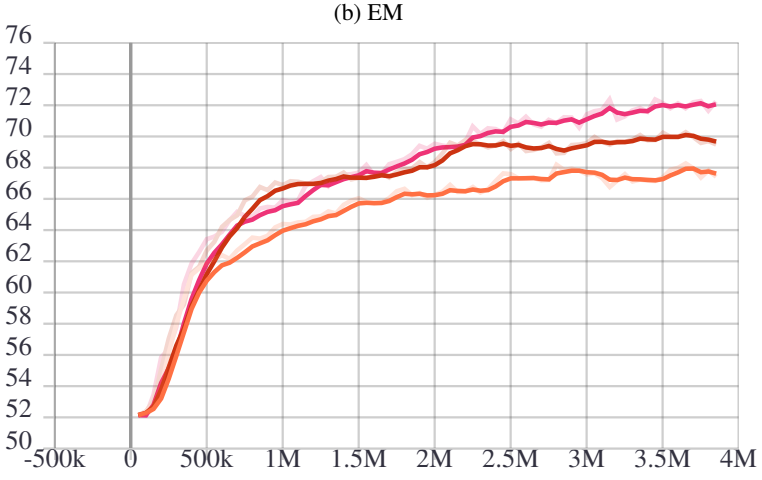
A Appendix



(a) F1



(b) EM



(c) AvNA

Figure 4: Dev set evaluation: Baseline (orange), Char Embedding (burgundy), SMA (magenta)