Improving SQuAD Performance through Model Combination

Stanford CS224N Default Project

Matt Riedman
Department of Computer Science
Stanford University
riedman@stanford.edu
TA: Sarthak

1 Abstract.

Machine comprehension on text is an important task in natural language processing. The publishing of the Stanford Question Answering Dataset (SQuAD) has provided a large number of questions and answers through crowdsourcing that allow for a common testing dataset for machine comprehension models. I leverage strengths of existing models with various implementations by combining their outputs in order to produce predictions that incorporate knowledge from all models. I find that naive combinations of output layer decrease performance, but when combining the two models prior to the final output component, both F1 and EM increase past either model on its own. Because of the limitations of any particular model construction, and in particular choosing a single attention layer, model combination appears to be a promising avenue for improving performance.

2 Background.

In the past few years, the problem of machine comprehension (MC) has gained much attention in the NLP community. Due to the vast troves of data across the Internet, in libraries, and even on particular sites such as Wikipedia, the task of automatically extracting information from text has proven valuable. In fact, Google's search function already leverages MC in their very popular feature. In 2016, Rajprurkar et al. provided a dataset, SQuAD, with over 100,000 pairs of contexts and questions answerable in the text [1]. With a highly competitive leaderboard, SQuAD provides an opportunity for new methods of MC to be compared to earlier ones on a standardized metric, meaning promising models on this dataset can be identified and applied to other areas where MC may be useful.

Because of the popularity of SQuAD, it provides an opportunity to objectively compare the performance of different MC models and layers. In particular, it means that the various techniques and ablations attempted by researchers, many of whom have fairly similar architecture, can be near effortlessly compared using the initiative's evaluation statistics EM and F1.

One of the greatest challenges in producing a successful question-answering model is identifying the most relevant words in the query and context. Conventionally, this process occurs primarily in an attention layer, whose results are then passed through an LSTM to get span probabilities.

However, any one choice of attention leaves open the possibility that certain types of queries or contexts are not amenable to that particular approach. To fix this, I train two models separately to achieve a certain standard of performance and then take softmax output to create a joint prediction that allows for efficient use of both models' strengths.

3 Related Work

The SQuAD dataset was introduced in 2016 by Rajpurkar et al. [1] as a common training and test set that would allow a standard on which question answering models could be compared. This was updated two years later with SQuAD 2.0 [2], which introduced unanswerable questions, which the researchers found significantly decreased performance of models that were then highest performing on the original SQuAD.

First, I introduce the model I use as a baseline, BiDAF, which was developed by Seo et al. [3]. Its primary contributions to the broader task of question answering lie in its use of LSTMs surrounding an attention layer, a format that has since been widely copied, and its titular bidirectional attention, in which question and context words attend to each other via encoded representations.

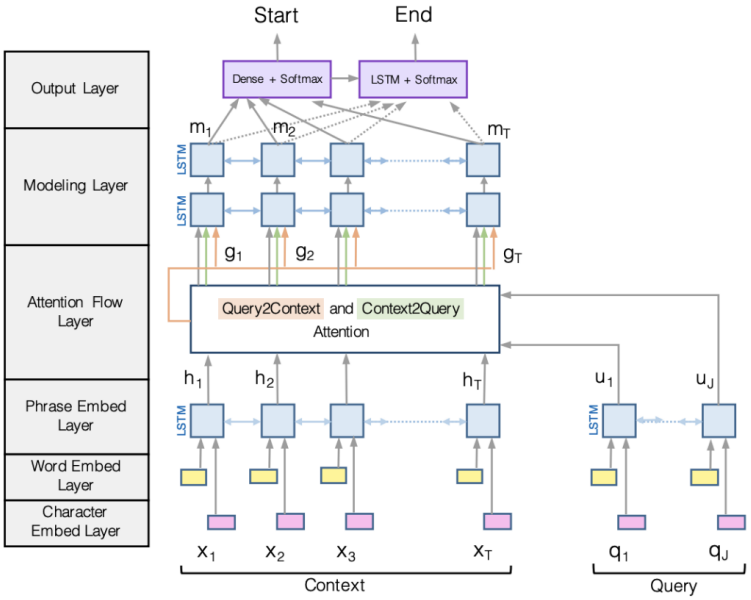
Second, in 2018, researchers Zhong et al. released their Dynamic Coattention model [4]. This model uses a similar LSTM-based structure to BiDAF, although it uses a revised attention layer. Similar to BiDAF, the Coattention encoder considers each query word in light of the context words and each context word in light of the query, which in this case takes the form of an affinity matrix. Then, after some further processing, the output is passed through a Bi-LSTM, allowing it to recover from local optima, which is particularly important in allowing the model to answer "N/A."

Importantly, because of the two models' different attention layers, they will naturally develop certain types of questions which they can answer with great reliability and others which they struggle with. Rather than trying to revamp the entire structure to create a model with both strengths, I propose combining the two somewhere after the attention layer so that *both* can have contributions to the final output.

4 Approach.

5 BiDAF and Dynamic Coattention

For my experiments, I chose to draw on the BiDAF and Coattention models described earlier. I chose these models because I believed their overall structural similarities would result in greater compatibility, particularly in my experiment with a shared output layer. However, because they had distinct attention layers, I assumed the strengths and weaknesses of the two models would be different, meaning there would be opportunity for improvement if their knowledge was combined well.



(Seo et al., 2017): Bidirectional Attention Flow for Machine Comprehension

Above is a diagram displaying the layers of the BiDAF model. It uses both character and word embeddings on the context and query that are then passed through an LSTM layer to produce the attention input.

Of special relevance to this paper is BiDAF's attention layer, which takes as input question hidden states $c_1, \ldots, c_N \in \mathbb{R}^h$. and query hidden states $q_1, \ldots, q_M \in \mathbb{R}^h$. These are then used to compute a similarity matrix S where

$$S_{ij} = w_{\text{sim}}^\top [c_i; q_j; c_i \circ q_j]$$

where brackets denote concatenation and $\circ$ denotes elementwise product. This similarity matrix is the softmaxed in both directions to produce

$$\overline{S_{i,:}} = \text{softmax}(S_{i,:}) \in \mathbb{R}^M, \quad \overline{\overline{S_{:,j}}} = \text{softmax}(S_{:,j}) \in \mathbb{R}^N$$

for $1 \leq i \leq N$ and $1 \leq j \leq M$.

These are used to produce

$$a_i = \sum_{j=}^{M} \overline{S}_{i,j} q_j \in \mathbb{R}^h$$

$$b_i = \sum_{j=1}^{N} \overline{S}\overline{\overline{S}}^\top c_j \in \mathbb{R}^h$$

giving the context-to-question outputs a_i and the question-to-context output b_i. These are then used to produce the attention layer output

$$g_i = [c_i; a_i; c_i \circ a_i; c_i \circ b_i] \in \mathbb{R}^{4h}$$

for $1 \leq i \leq N$, which are then passed into the final LSTMs to give the model output.

Similar to BiDAF attention, the Coattention attention layer uses question vectors q_i corresponding to words in the question and context vectors c_j for words in the document. These are stacked into matrices Q' and C, where the rows of Q' are the q_i and the rows of C are the context vectors. Then Q' is passed through a linear layer with $\tanh$ nonlinearity to produce

$$Q = \tanh(W^{(Q)} Q' + b^{(Q)})$$

with trainable $W^{(Q)}$, $b^{(Q)}$. They are then multiplied to produce an affinity matrix $L = C^\top Q$ which, ideally, would correspond to weights dictating the likelihood a certain question word corresponds with an answer word. These are then softmaxed in both directions to produce

$$A^Q = \text{softmax}(L), \quad A^C = \text{softmax}(L^\top).$$

Then, the attention contexts for each question word are defined to be

$$C^Q = C A^Q$$

and the representation of the question and context together is

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

where the brackets represent a concatenation of matrices.

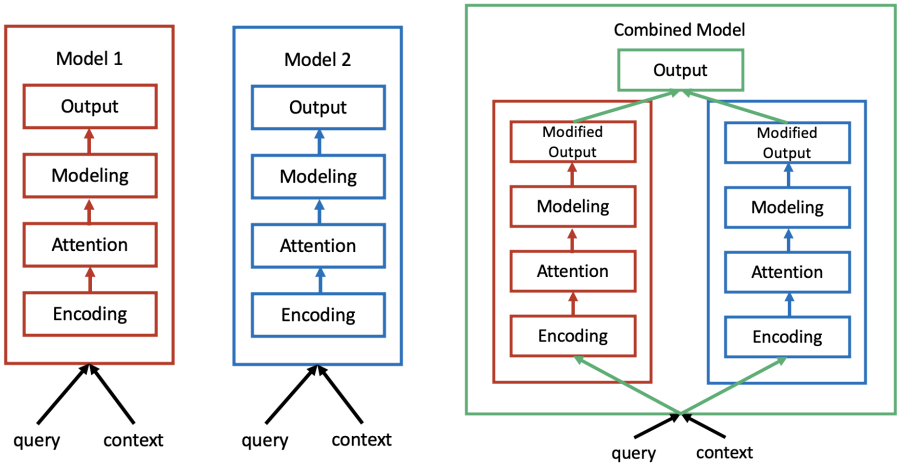
The rows of this matrix are then concatenated with the context encodings to make

$$[C; C^*]$$

and fed into another bidirectional LSTM to produce vectors u_t which can then be used to select the optimal span, mirroring the general process in the diagram for BiDAF.

5.1 Combined Model Approaches

My primary original contribution is the development of three types of combined models. The general method for constructing a combined model is depicted below:



At a high level, two models (in my case, BiDAF and Dynamic Coattention) are trained separately. Then, their outputs are modified and that modified output is passed through a new layer that yields predictions.

First Combined Output A standard SQuAD model's output consists of probability vectors p^s and p^s derived from a softmax that indicate the probability that the span including the answer to the question starts and ends at p_i^s and p_i^e, respectively, for all context indices i. For this approach, I did not modify the original output layer for either model. Thus, the input to the combined output layer is p^{s1} and p^{e1} from the BiDAF model and p^{s2} and p^{e2} from Dynamic Coattention.

Then, I calculated

$$p^s = \frac{p^{s1} \circ p^{s2}}{||p^{s1} \circ p^{s2}||}, \quad p^e = \frac{p^{e1} \circ p^{e2}}{||p^{e1} \circ p^{e2}||}$$

to give the output of the whole model. In effect, I assumed the probabilities given by softmax were independent and accurate representations of the actual probabilities given the model knowledge.

Second Combined Output Second, I anticipated the possibility that although a higher assigned probability by a model corresponds to a greater likelihood that the span starts/ends at a given index, the probability itself is not accurate. That is, it is possible that for a given model, when $p_i^s = 0.1$, there is actually a 5% chance the span starts at the location.

Suppose a model outputs a probability vector $p \in \mathbb{R}^n$. Let $i^* \in \{1, \ldots, n\}$ be the index of the correct answer–that is, the actual starting index for p^s or ending index for p^e. I first calculated probabilities

$$a_m = \mathbb{P}(i^* = i | m \leq p_i < m + 0.01)$$

for each $m \in \{0, 0.01, 0.02, \ldots, 1\}$ based on results from the SQuAD training set. This then gave a lookup table from which I could get the actual probabilities implied by the softmax output. I then fed these probabilities to the first combined output layer to get the final predictions.

Third Combined Output Finally, I attempted to combine the two models' hidden layers immediately prior to the output segment, meaning that the modified outputs in the diagram above were effectively identity mappings. In their place, I concatenated the modeling layer outputs from the two models and passed that through a linear layer and then a softmax. I designed it so that, should the initial linear layer weight all components from one model as 0, it would be identical to the output layer of the other model. This way, either of the models on their own would end up in the combined model's solution space, putting a strong lower bound on performance.

6 Experiments.

6.1 Dataset and Evaluation Metrics

In this project, I am using the SQuAD 2.0 dataset [2]. Like the original SQuAD, it contains over 100,000 questions to test MC using contexts found on Wikipedia and questions that were crowdsourced and developed by humans. Unlike the original, it comes with the addition of over 50,000 unanswerable questions that the model correctly identify as such.

SQuAD has two official evaluation metrics: EM and F1. EM, standing for exact match, gives 100% if the model returns one of the human-defined correct answers and 0% otherwise, making it very strict. Meanwhile, F1 is the harmonic mean of precision and recall, making it more forgiving.

For each of my experiments, I used one BiDAF model and one Dynamic Coattention model. On both models, I used versions with 100-dimensional encoded vectors trained for 2M iterations, with the effect that they had similar performance and thus any additive effect would be most clear. (Dynamic Coattention is capable of up to 66.2 EM and 75.1 F1 per its authors [4], but this requires larger hidden layers and more iterations.) On the validation set, the BiDAF model achieved an EM of 56.38 and an F1 of 55.97. Meanwhile, the Dynamic Coattention model used had an EM of 51.92 and an F1 of 55.17.

6.2 Experiment Details

In my first experiment, I implemented the models as described in the Approach section with no training necessary. For the second experiment, I calculated the actual probabilities using a pass for each model through the training set.

For the third experiment, I trained the combined model for 1.5M iterations on the default settings, also allowing the parameters of the underlying models to train (perhaps allowing each to specialize on certain questions).

6.3 Results

Model	Dev EM	Dev F1
BiDAF	56.66	59.98
Coattention	51.92	55.17
Exp 1	45.29	45.47
Exp 2	46.54	47.22
Exp 3	58.38	62.24

Additionally, Experiment 3 yielded an EM of **58.34** and an F1 of **62.01** on the test set.

7 Analysis

Surprisingly, neither of the first two experiments yielded a result that reached baseline performance. I hypothesize this is because when one model misidentifies the start or end position, it overrides the other. Specifically, I suspect that often, one model will assign very low weights to places where the other assigns most of its weight. This means that the combined model is left to choose from many low-probability options, most of which are irrelevant. The hypothesis is supported by the fact that both models do frequently select irrelevant spans, a behavior not observed in either component model.

The third experiment, however, improved by several points upon both of its component models in EM and F1, showing that the intent to combine strengths worked, at least to some degree. Further, examination of its predictions show it is not prone to selecting random spans in the same manner as each of the first two experiments.

8 Conclusion

By combining two models of similar structure and performance, I was able to construct a third with better performance than either component without introducing and dramatically different components. However, doing so proved to be more difficult than anticipated, as naively multiplying together the probability outputs instead amplified each model's weaknesses. I have learned about several approaches to the task of question-answering work and how, beyond quantitative metrics, high performance on certain subsets of data can be exploited as seen in my third experiment. As a next step, I hope to use less similar models to see if these trends still hold.

References

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

[2] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Association for Computational Linguistics (ACL)*, 2018.

[3] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension, 2018.

[4] Caiming Xiong, Victor Zhong, and Richard Socher. Dynamic coattention networks for question answering, 2018.