Bidirectional Attention Flow with Self-Attention

Stanford CS224N Default Project

Yan Liu
Department of Computer Science
Stanford University
liuyan84@stanford.edu

Abstract

I extended the BiDAF model with varies optimization techniques on the SQuAD 2.0 dataset. With character embedding and multi head self attention been added to the model, my results shows an improvement of +4 point on the EM and +4 point on F1 score compared with the default project.

1 Key Information to include

- External collaborators (N/A):
- External mentor (N/A):
- Sharing project:No

2 Introduction

Question Answering system is a challenging and interesting area for NLP researches, where many uses cases can be build on a quality QA system, such as chat-bot, auto-translation etc, where those systems may save labor resources on certain works, and may even save life. There are many approaches and models build to solve such problem, and notable break through from March,2019 where the model Bert[] almost close to human performance on SqUAD 2.0 Dataset[], and sooner later the human performance has been surpassed by many other models. Bert has introduced several techniques such as Encoder, Decoder and multi head self attentions. My interest is to find out that whether using the ideas from Bert, specifically multi head self attention, as an enhancement to default project and verify it's benefits.

3 Related Works

3.1 Self Attention

Bert has been proven to be successful on machine translation[], and has already applied in many area to provide value to real world ,it introduced the use of self attention , along with Encoder and Decoder that has significantly increased the translation accuracies.

3.2 SQuAD 2.0 Dataset

SQuAD 2.0 is one the well known Question and Answering dataset which extends the SQuAD with unanswerable questions.The reason of adding unanswerable questions to the SQuAD dataset was due to that SQuAD had a design flaw where the correct answer was always present in the context. With negative samples been added to the dataset, SQuAD 2.0 is more challenging then SQuAD 1.1

and SQuAD 1.0

3.3 Bidirectional Attention Flow

Their proposed solution trains models to produce results on individual paragraphs by sampling multiple paragraphs from the documents during training, and use a shared-normalization training objective that encourages the model to produce globally correct output.[1]

4 Approach

BIDAF includes character-level, word-level, and contextual embeddings, and uses bi-directional attention flow to obtain a query-aware context representation.[2]

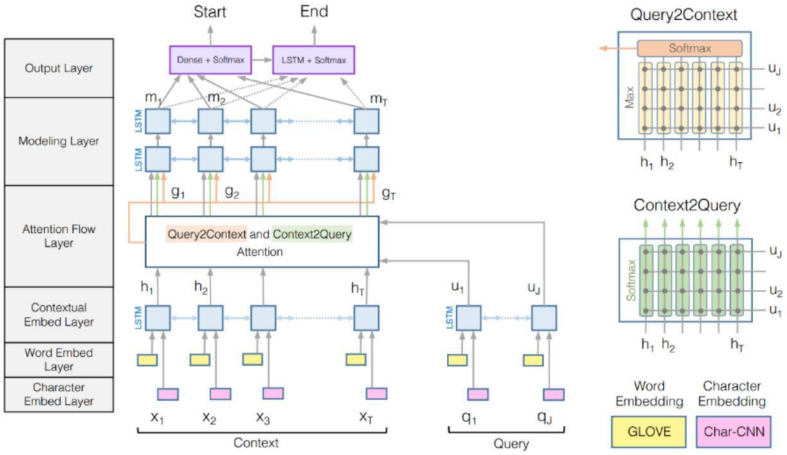

Figure 1 : BiDAF Model Architecture [1]

The first step is to added back the character embedding which was missing from the default project, since the Module has already provided the word indexes, it's very handy to attach the char embedding.

The approach to adding the char embedding is learned from Kim,2014[1], A single layer CNN which samples over each word. The idea is very simple, as due to the nature of many languages, English for example that has word may have identical sub words with suffix/prefix compositions, such as -prefix + tion or -prefix + ly, learning those features may help the Model to aware such language feature.

$$\mathbf{C} \in \mathbb{R}^{d \times l} : \text{Matrix representation of word (of length } l\text{)}$$
$$\mathbf{H} \in \mathbb{R}^{d \times w} : \text{Convolutional filter matrix}$$
$$d : \text{Dimensionality of character embeddings (e.g. 15)}$$
$$w : \text{Width of convolution filter (e.g. 1–7)}$$

1. Apply a convolution between $\mathbf{C}$ and $\mathbf{H}$ to obtain a vector $\mathbf{f} \in \mathbb{R}^{l-w+1}$

$$\mathbf{f}[i] = \langle \mathbf{C}[*, i : i + w - 1], \mathbf{H} \rangle$$

where $\langle \mathbf{A}, \mathbf{B} \rangle = \text{Tr}(\mathbf{A}\mathbf{B}^T)$ is the Frobenius inner product.

2. Take the *max-over-time* (with bias and nonlinearity)

$$y = \tanh(\max\{\mathbf{f}[i]\} + b)$$

Figure 2: Char Embedding [5]

The activation function was using tanh in [5] ,and I have chosen to use Relu over tanh from the conclusion of [6], where Relu normally trains much faster than tanh. In my model,the implementation was based on the contribution from [15]

However, such methodologies may or may not be particularly useful for single character languages, like Chinese. Although that we may come up with a vector of separating the composition of different subcomponent of a character, but learning those features seems extraordinarily complex [3]

The second step is to add the the self-attention layer after the attention layer , which was introduced from []. The general idea of self-attention was trying to find inter-relationships with each word for a given sentence. Learning the features could provide information on better predict the next word for a given context.

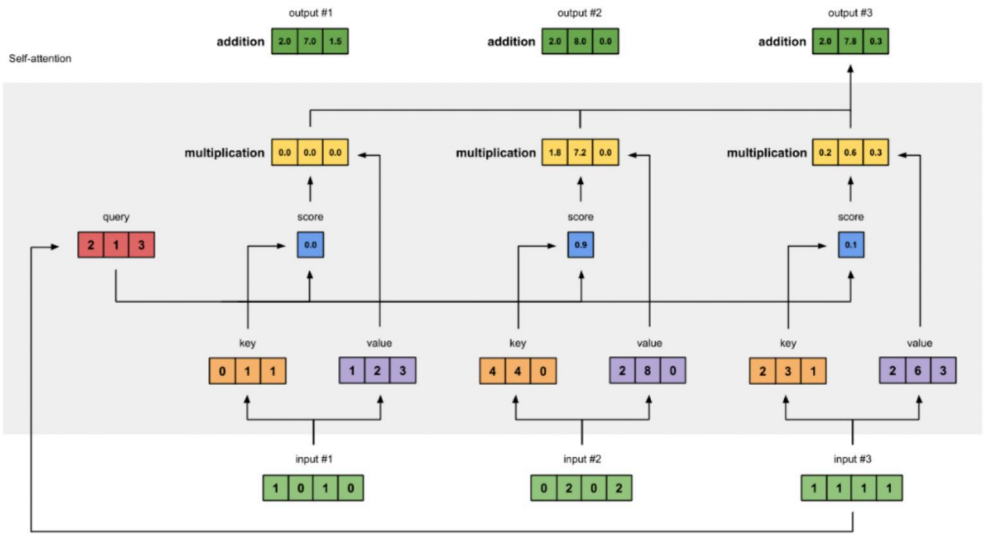

Figure 3 : Attention calculation path

The Q , K and V in the above are query , key and value matrixs which was random initialized and learned during the training. Q,K,V are of the same shape from the input for self attention.

The self-attention have several variants, but most widely used is multi head self-attention by repeating the computation of single head self attention for multiple times, and concatenating the results to a larger matrix followed by using another weight matrix W0 to correct the size of the matrix for the downstream to processing :

$$\text{MultiHead}(Q,K,V) = \text{Concat}(\text{head}_1,...,\text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Where attention is computed from :

$$\text{Attention}(Q,K,V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Adding the self attention layer into the BiDAF model was inspired by[7], the layer is placed after the Bi-Attention, and my implementation is using the standard library[16] with minor fixes.

Figure 4: Self Attention Layer in the BiDAF Model

5 Experiments

- **Data**: The Data will be based on SQuAD 2.0, which extends the SQuAD with unanswerable questions. SQuAD 2.0 combines the 100,000 questions in SQuAD1.1 with over 50,000 unanswerable questions. Those 50,000 unanswerable questions were designed to look similar to answerable ones[11]

  There are overlaps on the official SQuAD 2.0 dev set verses the dev set used for this project, we can only use the dev set provided for this particular project to perform training and evaluate.

  The Dataset provided for this project splits into train/dev/test with 129941/6078/5919 examples respectively. All of the examples are organized into (context,question,answer), where the context is taken from Wikipedia. A Sample of the data from the training set is shown below

  {"title": "Beyonc\u00e9", "paragraphs": [{"qas": [{"question": "When did Beyonce start becoming popular?", "id": "56be85543aeaaa14008c9063", "answers": [{"text": "in the late 1990s","answer_start": 269}], "is_impossible": false}

  Preprocessing will be done by on the JSON format data and convert it to our input.Describe the dataset(s) you are using along with references. Make sure the task associated with the dataset is

clearly described.

- **Evaluation method**: I am using Em and F1 metrics to evaluate our model, by definition from the SQuAD dataset it means:

  EM: This metric measures the percentage of predictions that match any one of the ground truth answers exactly.

  F1 score: This metric measures the average overlap between the prediction and ground truth answer.

  .

- **Experimental details**: I am using the default param setup for the training, with Batch size 64 and 30 epochs. Learning rate of 0.5 and drop out probability at 0.2. For char CNN I am using 1D conv net with stride = 1 and kernel = 5. I'm using 8 heads for multi head attention to compute the attention score. Although the learning rate seems a bit large, and kernel size for the char embedding could be tuned between 1-7 [5] , but finding the optimal hyper-parameters seems not allowed due to the time constraint.

- **Results**:

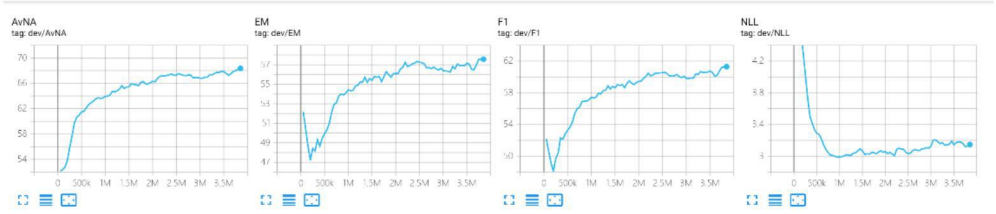

Figure 5: Em and F1 Score using the default project model

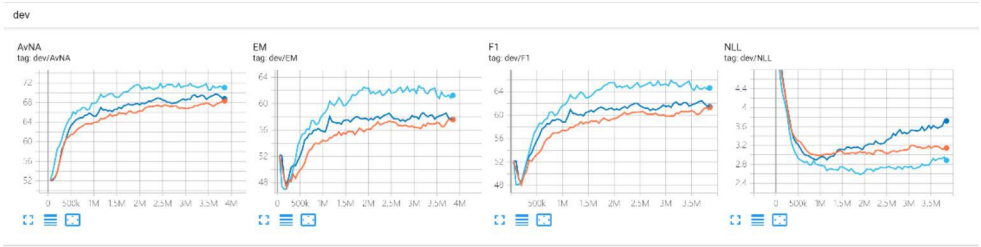

Figure 6: Em and F1 Score using the default project model + char embedding + self attention (in blue line)

The default project model provided F1:61.3 and EM:57.57 as shown on Figure 5.With the char embedding and self attention been added for the BiDAF model gained performance on the F1 and EM score, which are F1:65.293 and EM: 62.012 and better than the BiDAF+Self Attention (single model) from the official leaderboard which has F1: 62.305 and EM: 59.332

6 Analysis

The performance is as expected, but there are also rooms for improvements. One notable finding is I could also generate a masking for each word while training to force the attention computation not focus on the current word but other words of the given inputs. However, given the workload might be too much for a single person project, I decided to not implement such masking matrix for current word.

Model	F1	EM
Default Project	61.3	57.57
BiDAF+Self Attention (single model) From Squad Leaderboard	62.305	59.332
Our Model	**65.293**	**62.012**
BiDAF + Self Attention + ELMo (single model)	66.251	63.372

Figure 7 : Model Performance Analysis

7 Conclusion

Recently, a research paper from google reveals interesting findings that shows Self-Attention is not that helpful in the original Transformer paper[14] from the bias and rank collapse point of view. and adding path decomposition helps on preventing such effects. However, from our training results, adding self attention to the BiDAF model do improved our test result, and there are more rooms for improvements after that if above paper[14] is true. I will continue to explore the possibilities on further fine-tuning the model with recent updates in later time.

References

[1] Yoon Kim *Convolutional Neural Networks for Sentence Classification* 2014.

[2] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, Hannaneh Hajishirzi. *Bidirectional Attention Flow for Machine Comprehension.* 2016

[3] Jinxing Yu Xun Jian Hao Xin Yangqiu Song . *Joint Embeddings of Chinese Words, Characters, and Fine-grained Subcharacter Components.* 2017

[4] Meraldo Antonio. *Word Embedding, Character Embedding and Contextual Embedding in BiDAF — an Illustrated Guide.* 2017. https://towardsdatascience.com/the-definitive-guide-to-bidaf-part-2-word-embedding-character-embedding-and-contextual-c151fc4f05bb

[5] Yoon Kim Yacine Jernite David Sontag Alexander Rush. *Character-Aware Neural Language Models* https://nlp.seas.harvard.edu/slides/aaai16.pdf

[6] Alex Krizhevsky,Ilya Sutskever,Geoffrey E. Hinton *ImageNet Classification with Deep Convolutional Neural Networks* . 2017

[7] Christopher Clark and Matt Gardner. 2017. Simple and effective multi-paragraph reading comprehension. In ACL

[8] Natural Language Computing Group, Microsoft Research Asia†. 2017. R-NET: MACHINE READING COMPREHENSION WITH SELF-MATCHING NETWORKS

[9] Pranav Rajpurkar, Robin Jia, Percy Liang. 2018. Know What You Don't Know: Unanswerable Questions for SquAD

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N.

Gomez, Lukasz Kaiser, Illia Polosukhin. 2017. Attention Is All You Need

[11] 2021. CS 224N Default Final Project:Building a QA system (IID SQuAD track)

[12] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, Luke Zettlemoyer. 2018. Deep contextualized word representations

[13] Kyunghyun Cho, Bart Van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078 .

[14] Yihe Dong, Jean-Baptiste Cordonnier, Andreas Loukas. 2021. *Attention is not all you need: pure attention loses rank doubly exponentially with depth*

[15] *https://github.com/Oceanland-428/Improved-BiDAF-with-Self-Attention*

[16] *https://pytorch.org/docs/stable/generated/torch.nn.MultiheadAttention.html*

[17] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text