Longer-term dependency learning using Transformers-XL on SQuAD 2.0

Stanford CS224N Default Project

Belinda Mo
Department of Computer Science
Stanford University
bmo98@stanford.edu

Abstract

I propose an application of the Transformer-XL attention model to the SQuAD 2.0 dataset, by first implementing a similar architecture to that of QANet, replacing the RNNs of the BIDAF model with encoders, and then changing out the self-attention layer to that of Transformer-XL [1]. In traditional transformers, there exists an upper dependency length limit equal to the length of this context. The Transformer-XL addresses these issues by caching the representations of previous segments to be reused as additional context to future segments, thus increasing the context size and allowing information to flow from one segment to the next. This longer-term dependency capture can be particularly useful when applying transformers to domains outside of natural language. Similar results are shown with the Transformer-XL / QANet combined model as for the baseline BIDAF, and some suggestions and explanations are provided in the Discussion section below.

1 Introduction

The default final project attacks the problem of question answering using the Stanford Question Answering Dataset (SQuAD2.0). The model is given a paragraph and a question about that paragraph as input, with the goal of an output that answers that question correctly. This is an interesting problem because this objective trains the model to understand text, similar to how we test students on understanding through reading comprehension passages in standardized tests. A model powerful enough to reliably understand text can be used in a variety of applications, from parsing legal documents to medical records, all the while having the advantage over human beings in being able to process much more information than we can in the same amount of time.

The baseline model provided for the default final project is the BIDAF model [2], which has the advantage of being easy to train and test, as well as reporting decent performance on the SQuAD dataset. The BIDAF model successfully harnesses the power of bidirectional RNNs to capture sequential inputs, with a simple architecture involving two recurrent layers and one bidirectional attention layer. However, in the past few years, Transformers [3] have become the deep learning standard in natural language processing, which capture long-term dependencies better than prior RNN-based models and generally see improved results upon such models. Transformers a built on self-attention and convolution, and comprise feed-forward encoders / decoders. Because they have a more easily parallelizable architecture, Transformers are able to handle larger amounts of data than RNNs with better performance.

QANet [4] is a Transformer-inspired architecture applied to question answering, replacing RNNs with self-attention and convolution. QANet is mainly based on the Encoder Block, which is similar to the Transformer in its use of positional encodings, residual connections, layer normalization, self-attention sublayers, and feed-forward sublayers. The proposed final model is based heavily on the QANet architecture.

However, there are still time-length dependency limitations to the traditional Transformer model - specifically, because it uses a fixed-length context, breaking the sequence into segments to process separately, there exists an upper dependency length limit equal to the length of this context. Additionally, does not take into account sentence boundaries; therefore, the segmentation of the sequence results in context fragmentation. Transformer-XL [1] addresses these issues by caching the representations of previous segments to be reused as additional context to future segments, thus increasing the context size and allowing information to flow from one segment to the next. By doing do, this model captures long-range dependencies better than traditional Transformers.

I modified the baseline BIDAF model with three modifications - adding character embeddings, replacing the RNNs with the Encoder blocks of QANet, and then replacing the attention layer in these Encoder blocks with the Transformer-XL attention (with the accompanying update to relative positional encodings).

2 Related Work

The proposed model is initially built upon the BIDAF [2] model, which is also the provided baseline model. BIDAF achieved state-of-the-art results when it was released on the SQuAD Question Answering dataset.

The model is also built upon QANet [4], which is similar to the Transformer in that it uses stacked convolutional sublayers and depthwise separable convolution, thus addressing local dependencies in the input sequence. QANet also saw state-of-the-art performance on the SQuAD Question Answering dataset, until the release of BERT [5].

Finally, this model modifies the attention layer of QANet with the self-attention proposed by the authors of Transformer-XL [1]. Transformer-XL reuses hidden states obtained in previous segments as memory for computing the hidden states for the current segment, therefore adding a sort of recurrence to the purely self-attentive transformer model. There is also the added contribution of a novel positional encoding scheme that takes into account relative positions as opposed to absolute ones, which generalizes better for this new model.

3 Approach

The model modifies the BiDAF model in three steps:

1. Character embeddings were added to the Input Embedding layer. This is done by maintaining the word embeddings provided by the baseline model and concatenating them with character embeddings (which are found by passing the character vectors through a Convolutional layer and then the provided Highway Encoder). I am using a character dropout rate of 0.05, similar to that of [4].
2. The first RNN in the BiDAF baseline model was replaced with an Embedding Encoder, and the second one with a Model Encoder (similar to QANet). These are comprised of encoder block(s) consisting of a [convolution-layer × #] + self-attention-layer + feed-forward-layer, where the embedding encoder has 1 of these blocks with a convolutional kernel size of 7 and 4 convolutional layers while the model encoder has 7 blocks with a convolutional kernel size of 5 and 2 convolutional layers (using the same parameters specified in the QANet research).
3. The self-attention layer of the Embedding Encoder was replaced by the Transformer-XL attention. Because of a lack of time, I used fewer heads as the 8 referenced in the paper. from [1] - that is, 4 heads, a hidden dimension of 128, and a dropout of 0.2.

Therefore, the final model architecture is as follows:

1. **Embedding Layer:** The baseline BIDAF model included word embeddings. I took these word embeddings and concatenated the character embeddings as detailed above. The output of this layer is of size (batch size, length, 2 * hidden size), when compared with the word embedding-only output of (batch size, length, hidden size), due to the concatenation.
    - Word Dropout = 0.1
    - Character Dropout = 0.05.

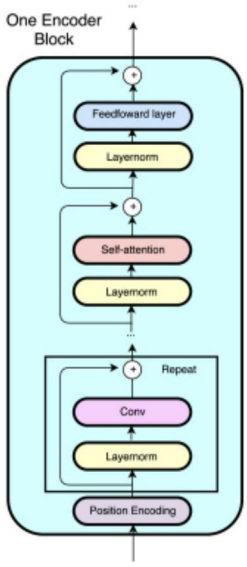

Figure 1: The Encoder Architecture. Note that "Position Encoding" uses the relative position embeddings and "Self-attention" uses the Transformer-XL attention.

2. **Embedding Encoder Layer (Fig 1):** This is the same as the Encoder block used in QANet, with the self-attention swapped out for the Transformer-XL attention. The output of this layer is of size (batch size, length, 2 * hidden size). The architecture is as follows:
   - *Positional Embedding Layer:* Relative positioning embeddings are used as in Transformer-XL [1], which are fixed sinusoidal embeddings consisting of sin and cos functions at varying wavelengths.
   - *Store Residual*
   - *Layer Norm*
   - *Depthwise Separable Convolutional Layers:* Depthwise separable convolutions are used rather than traditional ones, as [4] observed that it is memory efficient and has better generalization.
     - # Convolutional layers = 4
     - Convolutional kernel size = 7
   - *Add Residual*
   - *Store Residual*
   - *Layer Norm*
   - *Multihead Transformer-XL Attention:* This is similar to a Multihead Self-Attention, with the difference being that this attention layer keeps a memory component from the previous segment to then calculate the position-based attention term. The content-based attention (the standard Transformer attention) is added to the position-based attention, and then softmaxed and projected back to the input dimension and added to the residual connection. By maintaining memory, the Transformer-XL is able to capture some recurrence while still benefiting from the Transformer self-attention structure.
     - # Attention heads = 4
     - Hidden dimension = 64
     - Dropout = 0.2
   - *Add Residual*
   - *Store Residual*

- *Layer Norm*
- *Feed-Forward Layer:* A linear feed-forward layer.
- *Add Residual*

3. **Attention Layer:** This is the same as the BIDAF bidirectional attention layer as provided in the default baseline model. The output of this layer is of size (batch size, length, 8 * hidden size)
   - Dropout = 0.2
4. **Model Encoder Layer:** This is the same as the Embedding Encoder Layer, only differing in the number of encoder blocks, convolutional layers, and convolutional kernel size.
   - Total # of Encoder blocks = 7
   - # Convolutional layers = 2
   - Convolutional kernel size = 5
5. **Output Layer:** This is the same as the BIDAF output layer as provided in the default baseline model. The output of this layer is are two tensors (the logits), each of size (batch size, length).
   - Dropout = 0.2

4 Experiments

4.1 Data

For training and testing, I used the SQuAD 2.0 dataset provided by the class. The SQuAD dataset contains (context paragraph, question, answer) triples and is pre-divided into training / development / and test sets. The paragraph contexts are from Wikipedia and the questions and answers crowd-sourced using Amazon Mechanical Turk. There are 150,000 questions, for which about half are unanswerable questions. Additionally, in the development and test sets, every answerable question has three answers (which don't always agree), each from a different crowd worker.

4.2 Evaluation method

Performance was measured via the Exact Match (EM) score and the F1 score. Because these are the same scores used in on the SQuAD leaderboard, these metrics allow me to compare my model's performance. The Exact Match score is a strict metric of whether the output is an exact match to the ground truth. The F1 score is a more forgiving metric of (2 x precision x recall)/(precision + recall), the harmonic mean of the two.

4.3 Experimental details

I used the model configurations provided, that is a 0.5 learning rate, 50000 steps between successive evaluations, and run for 30 epochs. The Transformer-XL model took multiple times longer to train than the baseline model - on the NCS Version 3 machine, the baseline model took 4 hours in total to train, whereas my built model took over 20 hours. Unfortunately, I ran out of time to finish the project before I was able to find the cause of this excessively long training time, much of it due to having to reduce the batch size to 16 (as opposed to 64 for the baseline model) in order to train the new model. I also suspect that there may be possible improvements on the structure of my model's code to take advantage of the parallelizable nature of Transformer models and to more efficiently process the data.

4.4 Results

My best model obtained an EM / F1 score of 58.698 / 62.392 on the test leaderboard for the IID SQuAD Track (Figure 5). As can be seen (Figures 2-4), my Transformer-XL model performed slightly better than the baseline; however, it took an extremely long time to train. Additionally, when compared to the results of the baseline + character embeddings scores, this model was not more performant. I was expecting better results, given that the Transformer-XL was shown in the paper to display even better scores than vanilla transformer models, even for shorter text. However, because of memory and time constraints, I did not use the same exact structure that was proposed in [1], and

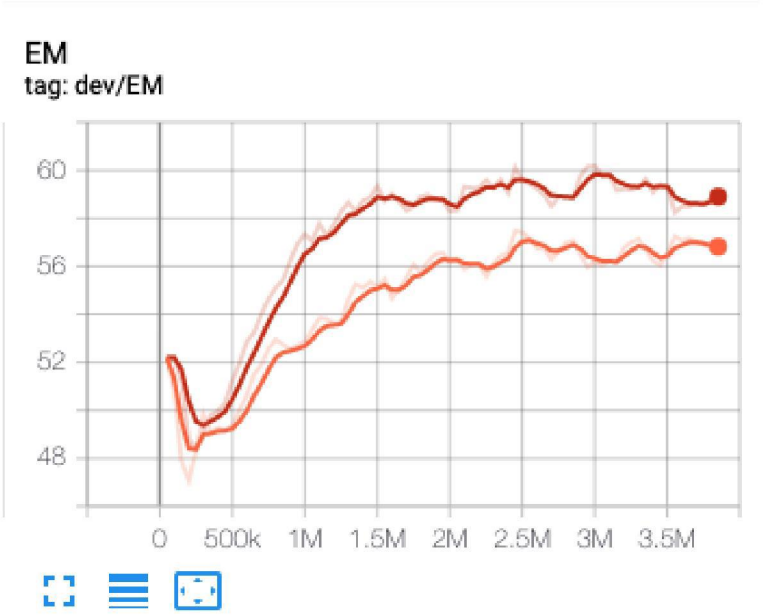

Figure 2: Comparing EM results between the baseline model (orange), and the Transformer-XL model (red).

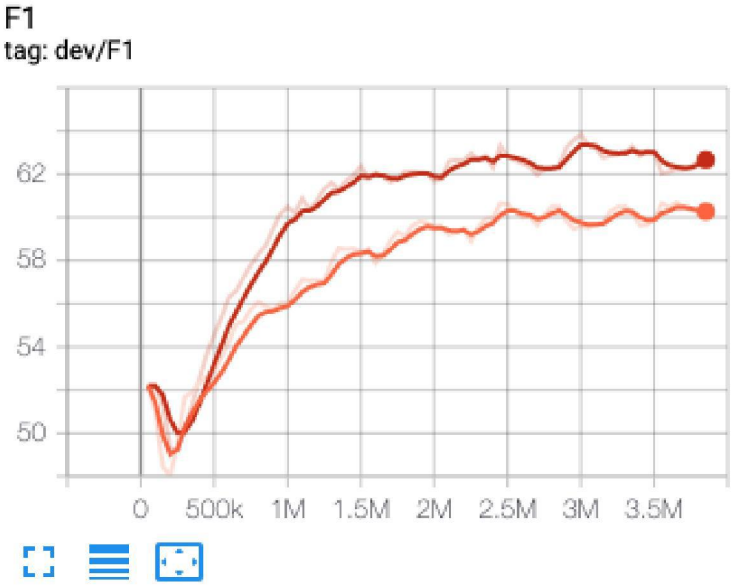

Figure 3: Comparing F1 results between the baseline model (orange), and the Transformer-XL model (red).

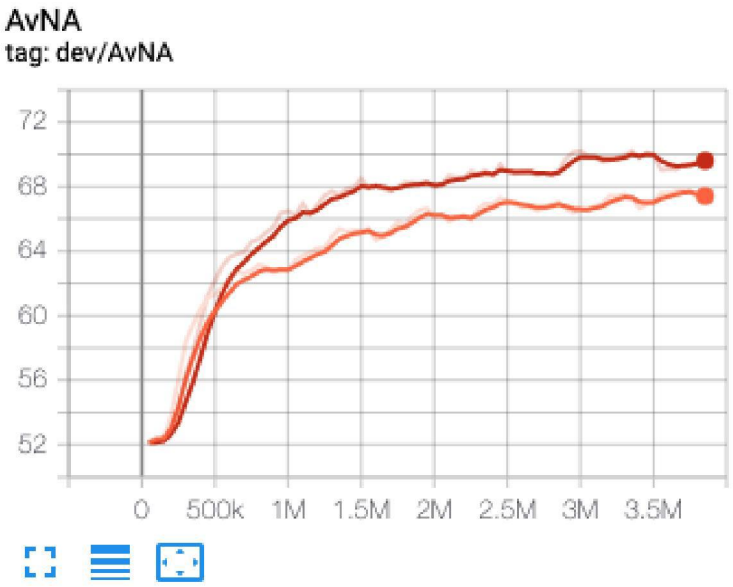

Figure 4: Comparing AvNA results between the baseline model (orange), and the Transformer-XL model (red).

bmo	58.698	62.382

Figure 5: EM / F1 score of the IID SQuAD test leaderboard

only used a 4 layer attention model versus the 12 that they used, as well as less attention heads, which is likely why this model shows lower-than-expected performance.

5 Analysis

A characteristic erroneous output of the implemented Transformer-XL model can be seen in (Figure 6), versus I was not able to spot either the baseline or the baseline + character embeddings models making such an error. Basically, the model would look "too far" for the answer and would actually skip over the correct response that is much closer in position. It is possible that this is because, by encoding the position and increasing the attention span of the model, the Transformer-XL has given incorrect weight to certain positions that are far away.

6 Conclusion

The performance of the model would likely benefit from more memory and time to run the experiments and finetune the parameters. However, I am very proud of having implemented the QANet + Transformer-XL model, having it run successfully, and have achieved my goal of gaining a deeper understanding of transformers / self-attention from the implementation level. Although it was

- **Question:** When was Montreal captured?
- **Context:** In Europe, the North American theater of the Seven Years' War usually is not given a separate name. The entire international conflict is known as the Seven Years' War. "Seven Years" refers to events in Europe, from the official declaration of war in 1756 to the signing of the peace treaty in 1763. These dates do not correspond with the fighting on mainland North America, where the fighting between the two colonial powers was largely concluded in six years, from the Battle of Jumonville Glen in 1754 to the capture of Montreal in 1760.
- **Answer:** 1760
- **Prediction:** 1756

Figure 6: An erroneous output typical of the Transformer-XL model.

made clear in the lectures that most work nowadays is done on pre-trained models, this lower level comprehension will help in applying transformers to non natural language domains, which is my near-term goal. As for future work upon this specific project, the priority would be finetuning the parameters and tweaking the code such that it can be more performant with respect to memory and time.

References

[1] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy, July 2019. Association for Computational Linguistics.

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

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

[4] 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, 2018.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.