Project Proposal - Question Answering on SQuAD 2.0

Stanford CS224N Default Project

Thibaud Bruyelle
Department of Computational and Mathematical Engineering
Stanford University
thibaudb@stanford.edu

Tom Morvan
Department of Computational and Mathematical Engineering
Stanford University
tommrvn@stanford.edu

1 Introduction

Question Answering (QA) is growing field of NLP showing increasingly good results and with countless applications. The conception of AI by children is best embodied by an "all-knowing" machine able to answer each and every one of their questions. Although this naive approach is far from a thorough representation of Artificial Intelligence, it highlights a key challenge that has yet to be solved. Coming up with an efficient QA mechanism would be a major breakthrough for many professions that rely on large and complex data, such as doctors or lawyers, by saving them a great amount of time and energy. A recent development was the release of a novel language model by Google AI named BERT, which allowed a better understanding of word contexts. Based on this milestone, will we try and outperform current methods for question answering on the famous SQuAD 2.0 dataset.

2 Research paper summary : BERT

2.1 Transformers

Before diving into the depths of the BERT model, we would like to outline the recent advancements in NLP that have led to the following architectures.

Introduced in 2014, the RNN-based sequence to sequence model was an amazing step-forward in NLP since it could handle a lot of tasks such as Machine Translation or Questions-Answering. Furthermore, the model was reinforced when the attention mechanism was added to it in 2015. However, one of the main issue with this model is dealing with long-range dependencies. Hopefully, a totally novel architecture that could handle this kind of issue came out : the transformer. Transformers were proposed for the first time in [1] and rely on a self-attention mechanism which enables the understanding of contextual meaning of the input sentence with a lot of precision. Transformers are composed of an encoder and a decoder. BERT builds upon the Transformers' encoder architecture, and we will therefore focus on the latter. Thanks to the attention layer, a Transformer is able to better understand a word in the input sentence by looking up all the other words and computing a weighted contribution score. Indeed, each input word is matched to three vectors: q is the query vector, k the key vector (dimension d_k) and v the value vector. By packing all these vectors into matrices, we are able to compute the attention of all the input words with respect to each other:

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

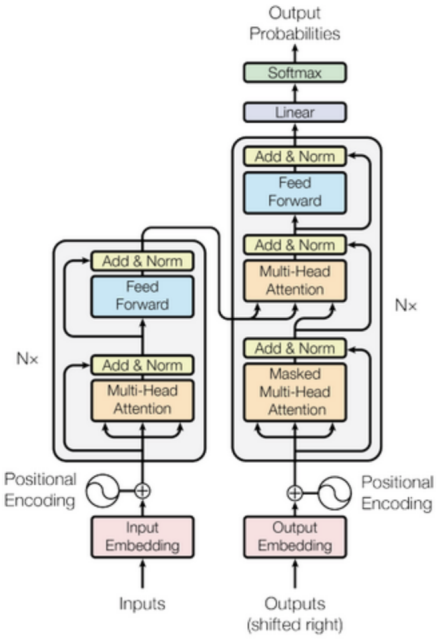

Figure 1: Transformer's architecture

Thanks to these attention scores, we are now provided with a new kind of word embedding that is strongly dependant on the context.

2.2 BERT

The BERT framework, a new language representation model described in [2], uses pre-training and fine-tuning to create state-of-the-art models for a wide range of tasks. BERT uses a multi-layer bidirectional Transformer encoder. Its self-attention layer performs self-attention in both directions, which is the main specificity of BERT model when compared to other Transformer based models. Figure 2 shows a visual representation of its architecture. BERT model is pre-trained on Masked Language Modelling and Next Sentence Prediction. We will discuss it in the next sections.

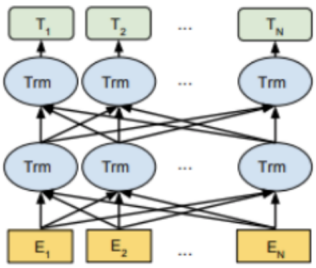

Figure 2: BERT's architecture

2.3 Input Layer

The inputs are first tokenized and then embedded using pre-trained token embeddings (WordPiece with 30,000 token vocabulary is used in [2]). As the transformer encoder (and thus BERT) relies only on attention mechanisms, we need to add positional information of the input (which are usually dealt with by the recurrent and convolutional layers). This is why positional encodings are added to the token embeddings. The positional encodings are numbers between -1 and 1 which hold information about each word's absolute position, as well as relative position. [2] gives several ways to compute positional encoding.

There are two more tweaks to the input representation in the BERT model compared to the transformer input. To allow inputs in the form of pair of sentences (which is essential for our problem with

question/context inputs), sentences are separated by the <SEP> token and sentence embeddings are also added to the input embeddings. Finally, the first token of each input is the <CLS> token, which is only used in classification tasks. For classification, the last hidden state of this token is the aggregate sequence representation, which is later passed through a linear and softmax layer. For question answering we will use a <OOV> token instead of the <CLS> token to be able to predict an unanswerable question.

2.4 Pre-training task

In order to pre-train the model, BERT starts by a *masked language model* (MLM) objective. Some tokens of the input sentence are randomly masked, then these tokens must be predicted based solely on the context of the input sentence. Such a pre-training task enables a deep bidirectional representation. Secondly, BERT performs a *next sentence prediction* (NSP) pre-training task. Indeed, most of NLP tasks such as Question Answering use the relationship between two sentences. Thus, the task aims to predict the labels {*IsNext*, *NotNext*} associated with a pair of input sentences.

2.5 Fine-tuning

Once the BERT language representational model is pre-trained, the final step lies in adapting the model to the wanted output. There exists two main strategies for applying pre-trained language representation models to different kinds of tasks : *featured-based* methods and *fine-tuning* methods. The latter introduces minimal task-specific parameters. Then the model is trained by *fine-tuning* the pre-trained parameters. As an example, Figure 3 shows the usual *fine-tuning* procedure for a Question Answering task with BERT. The *pre-training* has already been done and the pre-trained parameters are used in this QA task.

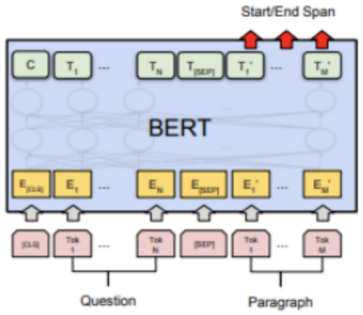

Figure 3: *Fine-tuning* procedure for BERT

3 Project description

Our main goal in this project is to use the contextual embeddings provided by BERT to efficiently tackle the SQuAD 2.0 dataset. We will most likely start with BERT's default Question Answering (QA) Output layer, before trying to outperform its results by adding Deep Learning architectures on top of BERT's encoders. Ultimately, our ambition is to secure a a top ranking on the leader-board.

Although we will be seeking to beat BERT's default QA layer, we will be using a Bidirectional Attention Flow (BiDAF) model as first baseline. This model that we will build and train will allow us to measure the efficiency of Transformers as attention mechanisms compared to Bidirectional Attention.

The results obtained through the simple BERT Output layer will then constitute our second baseline which we will try to improve. We will start by identifying its key caveats to spotlight the features that must be primarily addressed, before tending to enhance overall accuracy. We plan on using previously studied models such as CNNs, LSTMs and full Transformers (Encoders + Decoders) on top of BERT's encoders to increase the performance metrics of our pipeline. Depending on the outcome of our different methods, we will be conducting extensive research in the fields that seem the most promising.

In order to evaluate our models, we will be using EM and F1 scores as stated in the project handout.

References

[1] ParmarN. UszkoreitJ. JonesL. GomezA.N. KaiserL. Polosukhin I. VaswaniA., ShazeerN. *Attention Is All You Need. Advances in Neural Information Processing Systems, 5998–6008*. 2017.

[2] Google AI Language. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019.