Clayton Cohn
30 Sep 2020

DePaul University
CSC594: Advanced Deep Learning

Assignment 2: BERT

1 Abstract

in 2018, the advent of bidirectional contextualization ushered in a new era in Natural Language Processing (NLP). Often referred to as the "ImageNet [1] Moment" for NLP, models like BERT [2], ELMo [3], ULMFiT [4], and GPT-2 [5] proliferated throughout the NLP landscape, as transfer learning quickly became the status quo for many NLP tasks. Of those models, BERT emerged as the front-runner in Natural Language Understanding (NLU) when its release set a new state of the art (SOTA) on 11 separate NLP benchmarks [2]. This paper seeks to present and explain BERT in a manner that is both intuitive and informative to the reader.

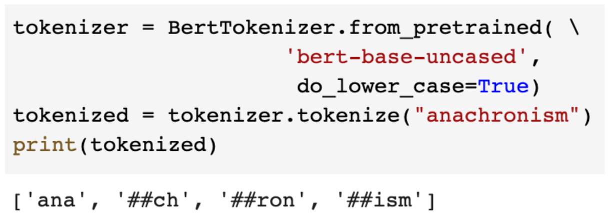
2 Components

BERT is comprised of several different components and utilizes a variety of innovations in deep NLP. This paper will dive into three of those innovations: Byte Pair Encoding (BPE) [6], Attention [7], and the Transformer [8]. Each is discussed in the subsections below.

2.1 Byte Pair Encoding

Previous word embedding models such as GloVe [9] and Word2Vec [10] map individual words to unique vectors known as "word embeddings." In the cases where a word is not present in a model's lexicon, the word is replaced by a token representing that the particular word is "out-of-vocabulary" (OOV) (e.g. **<unk>**). This is obviously an issue, as in many instances it is precisely these esoteric terms one is most concerned with (think about classifying chemical compounds, for example). BPE provides a fix for this.

The original BPE algorithm was designed for data compression. It replaced common pairs of consecutive bytes with with bytes that did not appear in the original data [11]. The BPE algorithm was later adopted for NLP and repurposed for subword tokenization by Sennrich et al. [12], whose vocabularies also included the most frequently-occurring character-level n-grams in addition to words. This enabled rare words to be represented by frequently occurring subwords instead of being replaced by a relatively useless *<unk>*. An abstruse term such as *anachronism* (something out of place in time), for example, would be broken down and represented as a combination of subwords instead of being labeled as OOV [11]. Figure 1 shows a real-world example of how BERT tokenizes the word "anachronism."



```
tokenizer = BertTokenizer.from_pretrained( \
                        'bert-base-uncased',
                         do_lower_case=True)
tokenized = tokenizer.tokenize("anachronism")
print(tokenized)
```

```
['ana', '##ch', '##ron', '##ism']
```

Figure 1: BERT's tokenized representation of the word "anachronism." The "##" characters indicate that a particular token is a subword.

2.2 Attention

While the Encoder-Decoder [13] architecture is effective on shorter sentences (less than 30-40 words), it starts to lose its efficacy as the sentences increase in length. It is difficult for neural networks to "memorize" long sentences for the purposes of contexualization, as networks have a tendency to "forget" earlier parts of the sentence. The Attention mechanism remedies this by only focusing on part of the sentence at a time. For each embedding, the Attention model computes a weight vector that specifies how much "attention" should be paid to each of the other embeddings in the sentence. From these weights, the decoder is able to know how much weight it should put on a given input submitted to the encoder when determining the proper output. [14]. Unlike the Encoder-Decoder framework, which uses a fixed-length context vector, the Attention model's context vector is filtered for each time-step in the output [15]. Figure 2 highlights where attention is placed during an example machine translation task [16].

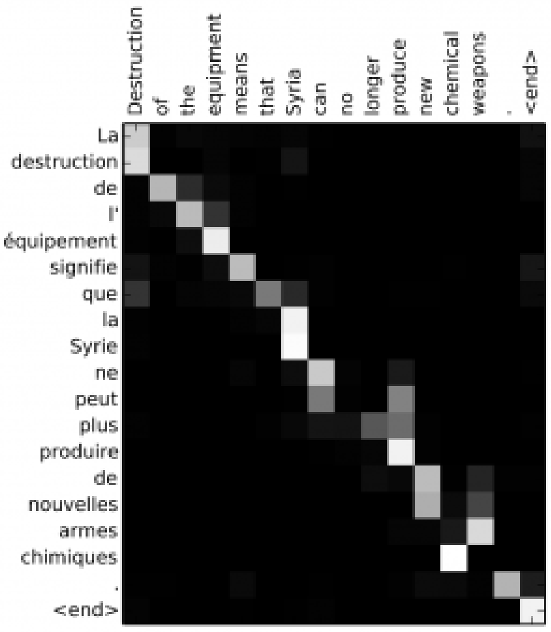


Figure 2: Attention during an example machine translation task.

When using attention, a layer takes into account the activations from other layers in addition to its own. *Self-attention*, conversely, is only concerned with activations in the same layer in which the attention is being applied (hence the "self" prefix). While attention mechanisms are often used to transfer information from an encoder to a decoder, self-attention is only applied within a single layer (in the encoder, in BERT's case). Self-attention is good at modeling dependencies between different parts of the same sequence, as opposed to the dependencies between two different sequences [17]. Transformers, for example, utilize self-attention for dependency modeling. Figure 3 illustrates a Transformer layer's self-attention mechanism.

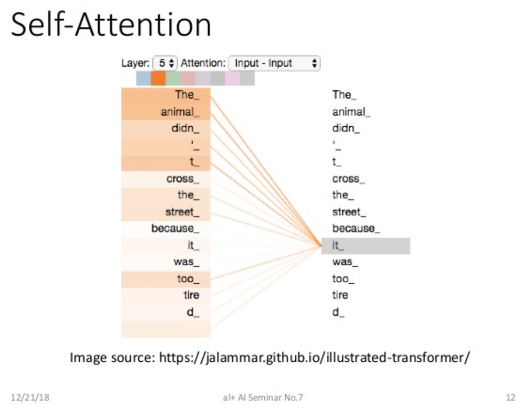


Figure 3: Self-attention of a Transformer layer.

2.3 Transformer

Transformers [8] eschew recurrence and convolution altogether and rely solely on self-attention to formulate dependencies between inputs and outputs [7]. Recurrent nets' sequential nature generally precludes them from working in parallel, as each time-step depends on the hidden state of the last time-step. This can become very computationally expensive, especially with longer sentences. Additionally, long-term dependencies tend to get lost, as the model forgets what it previously learned (especially in the earliest parts of the sequence). The Long Short-Term Memory (LSTM) architecture tried to remedy this, but it still encountered similar issues. The Transformer was created to mitigate these issues.

Transformers consist of 6 encoders and 6 decoders. Each encoder has two layers: a self-attention layer and a feed-forward neural network layer. Each decoder consists of three layers: a self-attention layer, an attention layer for the encodings, and a feed-forward neural network layer [8]. The encoder generates encodings that indicate the relevant parts of the input (where the attention should be paid), and the decoder takes these encodings and uses them to generate an output. Figure 4 illustrates the Transformer's architecture [8].

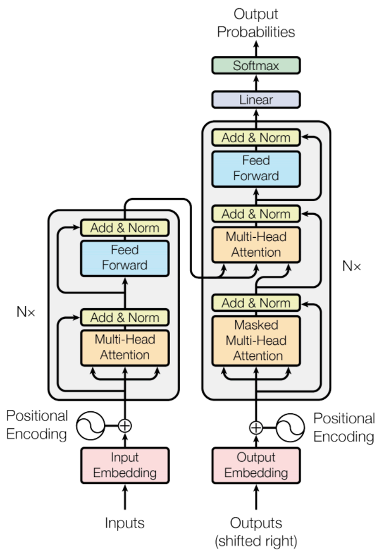


Figure 4: Transformer architecture.

The Transformer's ability to facilitate training in parallel enabled pretrained models to be generated from massive corpora and is why transfer learning is not only pervasive in NLP today, but is the status quo for many NLP tasks [18].

3 BERT

BERT was born out of the desire for bidirectionality in pretrained models. Previously, models had to either be trained in one direction, either from front-to-back or back-to-front (ELMo achieved pseudo-bidirectionality by doing both and then concatenating the results) [2]. With the advent of self-attention and the Transformer, the BERT authors were able to create an architecture that allowed for the bidirectional contextualization of words in a vector-space. Its vocabulary consists of 30,000 WordPiece [19] tokens that were generated by a BPE algorithm similar to the one originally created for data compression. These tokens consist of words, subwords, and characters. Subwords that are not prefixes are identified with '##,' as shown in Figure 1.

BERT was pretrained via unsupervised learning on Wikipedia[20] and BookCorpus [21]. Wikipedia consists of over 2.5 billion words, and BookCorpus consists of 800 million words. The authors employed two different methods during pretraining: masked language modeling (MLM) and next sentence prediciton (NSP). MLM is the process of randomly omitting a word (or words) from a sentence and training the model to predict the missing words from its vocabulary based on the words' context. NSP jointly trains text-pairs by inputting two sentences and then training the model to discern whether or not the second sentence is a valid continuation of the first [2]. MLM was done with a masking rate of 15%, and NSP was done with the correct sentence being present 50% of the time. The model was trained with a batch size of 256 sequences * 512 tokens per sequence (128k tokens per batch). Training was done over 1,000,000 steps, which equates to roughly 40 epochs over 3.3 billion words. Training took 4 days to complete on 4 cloud TPUs (16 TPU chips) [2]. The pretraining diagram from the original BERT paper can be seen below in Figure 5.

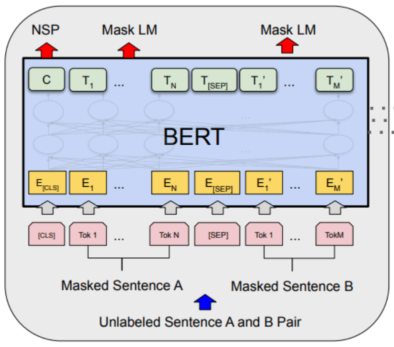


Figure 5: Pretraining BERT.

BERT was released with two versions: BERT-base and BERT-large, and each has a cased and uncased iteration (there is also a Chinese BERT for Chinese and a Multilingual BERT that was trained on 102 different languages). For the purposes of this paper, I will be discussing BERT-base, which is the most widely utilized model, as BERT-large occupies a significantly greater amount of memory and has not been shown to offer a proportional increase in efficacy. BERT-base has 12 layers (Transformer blocks), each with 12 self-attention heads and 768 hidden neurons. BERT-base consists of approximately 110 million parameters: 24 million from the embeddings, 85 million from the transformers, and 1 million from the pooler. It is roughly 450 MB, and requires at least 12 GB of ram when fine-tuning (BERT-large requires at least 16 GB), according to the authors [2].

BERT was designed to be a pretrained model that could be used out-of-the-box for any NLU task with a minimal amount of fine-tuning. When BERT was released in October 2018, it obtained a new SOTA on 11 separate NLP tasks including GLUE [22], MultiNLI [23], SQuAD 1.1, and SQuAD 2.2 [24]. This was significant, as it was the first instance that a task-agnostic pretrained model was able to outperform its deep-learning peers on a wide array of tasks. Furthermore, it led credence to the idea that transfer learning models could supplant deep learning ones just as had been the case in computer vision five years earlier with ImageNet. As a result, transfer learning has emerged as the method of choice for several different NLP tasks. The fine-tuning diagram from the original BERT paper is depicted below in Figure 6.

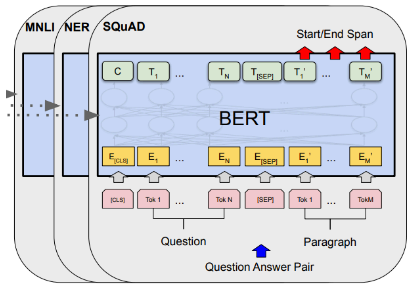


Figure 6: Fine-tuning BERT.

Fine-tuning BERT has become a popular approach for modeling many downstream tasks such as named entity recognition (NER), question answering (QA), sentiment analysis, relation extraction, intent classification, coreference resolution, and many more. Google has even started incorporating BERT in its search engine. While some tasks still remain best-attacked with deep learning, many benchmarks continue to experience new SOTA results with BERT and transfer learning.

BERT has since evolved into further pretrained, domain-specific models that are designed to model specific subject matter. BioBERT [25] and SciBERT [26], for example, were further pretrained with clinical and medical corpora to give BERT a better understanding of scientific and medical terminology. Many of the words that pervade scientific and medical corpora are abstruse and not likely to be well-represented in canonical texts such as Wikipedia or BookCorpus. This additional pretraining allows BERT to improve the contextual vectorization of terms that it was previously unfamiliar with. While BioBERT chose to utilize BERT's lexicon, SciBERT opted to create its own based on the most frequently occurring words and subwords in scientific research papers. The overlap between BERT's vocabulary and SciBERT's vocabulary is only about 40%, illustrating the drastic difference in the frequency of words in scientific versus canonical corpora [26].

BioBERT and SciBERT both generally outperform "vanilla" BERT when applied to scientific and medical corpora, indicating that BERT can be further improved if pretrained on corpora consisting of similar subject matter as the dataset(s) in the downstream task. However, this often comes at a cost, as BERT becomes less adept at modeling domain-agnostic tasks the more domain-specific the model becomes. Still, the addition of domain-specific pretraining has been shown to improve BERT's efficacy when applied to datasets in the same domain.

4 Implications

The implications of BERT are far-reaching and continue to transform the NLP landscape. Before BERT, the notion that a transfer learning model could perform as well as a deep learning model in NLP was still a question. BERT has demonstrated that not only are transfer learning models able to compete with deep learning ones, but that they are actually *more* effective in many instances. With effective transfer learning models that can produce SOTA results, NLP researchers now have a reliable, computationally cheap manner in which to train smaller datasets that are often at risk of being overfit by their deep learning counterparts. Furthermore, models such as BioBERT and SciBERT paved the way for other domains to create additional domain-specific models. Work is already being done on adapting BERT to legal texts, and I surmise that it is only a matter of time before BERT is trained to infer meaning from computer code as well. Stay tuned.

5 References

[1] J. Deng et al. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition.* 2009, pp. 248–255. URL: `https://ieeexplore.ieee.org/document/5206848`.

[2] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: (2018). arXiv: `1810.04805`. URL: `https://arxiv.org/pdf/1810.04805.pdf`.

[3] Matthew E. Peters et al. "Deep contextualized word representations". In: (2018). arXiv: `1802.05365`. URL: `https://arxiv.org/pdf/1802.05365.pdf`.

[4] Jeremy Howard and Sebastian Ruder. "Universal Language Model Fine-tuning for Text Classification". In: (2018). arXiv: `1801.06146`. URL: `https://arxiv.org/pdf/1801.06146.pdf`.

[5] Alec Radford et al. "Language Models are Unsupervised Multitask Learners". In: (2019).

[6] Philip Gage. "A New Algorithm for Data Compression". In: *The C User Journal* (1994). URL: `http://www.pennelynn.com/Documents/CUJ/HTML/94HTML/19940045.HTM`.

[7] Andrea Galassi, Marco Lippi, and Paolo Torroni. “Attention in Natural Language Processing”. In: (2019). DOI: 10.1109/TNNLS.2020.3019893. arXiv: 1902.02181. URL: https://arxiv.org/pdf/1902.02181.pdf.

[8] Ashish Vaswani et al. “Attention Is All You Need”. In: (2017). arXiv: 1706.03762. URL: https://arxiv.org/abs/1706.03762.

[9] Christopher D. Manning Jeffrey Pennington Richard Socher. “GloVe: Global Vectors for Word Representation”. In: (Sept. 2014). URL: https://www.aclweb.org/anthology/D14-1162.pdf.

[10] Tomas Mikolov et al. “Efficient Estimation of Word Representations in Vector Space”. In: (2013). arXiv: 1301.3781. URL: https://arxiv.org/pdf/1301.3781.pdf.

[11] Akashdeep Singh Jaswal. *Byte Pair Encoding — The Dark Horse of Modern NLP*. Nov. 2019. URL: https://towardsdatascience.com/byte-pair-encoding-the-dark-horse-of-modern-nlp-eb36c7df4f10.

[12] Rico Sennrich, Barry Haddow, and Alexandra Birch. “Neural Machine Translation of Rare Words with Subword Units”. In: (2015). arXiv: 1508.07909. URL: https://arxiv.org/pdf/1508.07909.pdf.

[13] Mu Li Aston Zhang Zachary C. Lipton and Alexander J. Smola. “Dive into Deep Learning”. In: (Sept. 2020). URL: https://d2l.ai/chapter_recurrent-modern/encoder-decoder.html.

[14] Andrew Ng. *C5W3L07 Attention Model Intuition*. Feb. 2018. URL: https://www.youtube.com/watch?v=SysgYptB198.

[15] Jason Brownlee. *How Does Attention Work in Encoder-Decoder Recurrent Neural Networks*. Oct. 2017. URL: https://machinelearningmastery.com/how-does-attention-work-in-encoder-decoder-recurrent-neural-networks/.

[16] Andrew Ng. *C5W3L08 Attention Model*. Feb. 2018. URL: youtube.com/watch?v=quoGRI-1l0A.

[17] artoby. *What’s the difference between Attention vs Self-Attention? What problems does each other solve that the other can’t?* Jan. 2013. URL: https://datascience.stackexchange.com/questions/49468/whats-the-difference-between-attention-vs-self-attention-what-problems-does-ea.

[18] Wikipedia. *Transformer (machine learning model)*. July 2020. URL: https://en.wikipedia.org/wiki/Transformer_(machine_learning_model).

[19] Mike Shuster and Kaisuke Nakajima. “Japanese and Korean Voice Search”. In: (2012). URL: https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/37842.pdf.

[20] Wikipedia. *Wikipedia*. Jan. 2001. URL: https://en.wikipedia.org/wiki/Main_Page.

[21] Yukun Zhu et al. “Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books”. In: (2015). arXiv: 1506.06724. URL: https://arxiv.org/pdf/1506.06724.pdf.

[22] Alex Wang et al. “GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding”. In: (2018). arXiv: 1804.

07461. URL: https://arxiv.org/abs/1804.07461.

[23] Adina Williams, Nikita Nangia, and Samuel R. Bowman. "A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference". In: (2018). URL: https://cims.nyu.edu/~sbowman/multinli/paper.pdf.

[24] Pranav Rajpurkar et al. "SQuAD: 100,000+ Questions for Machine Comprehension of Text". In: (2016). arXiv: 1606.05250. URL: https://arxiv.org/abs/1606.05250.

[25] Jinhyuk Lee et al. "BioBERT: a pre-trained biomedical language representation model for biomedical text mining". In: (2019). DOI: 10.1093/bioinformatics/btz682. eprint: arXiv:1901.08746. URL: https://arxiv.org/pdf/1901.08746.pdf.

[26] Iz Beltagy, Kyle Lo, and Arman Cohan. "SciBERT: A Pretrained Language Model for Scientific Text". In: (2019). eprint: arXiv:1903.10676. URL: https://arxiv.org/pdf/1903.10676.pdf.