BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding

What does it do?

BERT [1] (Bidirectional Encoder Representations Transformers): is a paper published by the team of researchers at Google AI. It has exhibited state-of-the-art results for various tasks of Natural Language Processing (NLP), such as question and answering, sentence pair classification, sentence tagging etc.
BERT is designed for the training on large corpus of text e.g. Wikipedia, so that it can be represented as generalized language model, and then it can be fine tunned on an additional layer which is specific to a target task of the language like text classification without significant changes in original architecture as shown in figure 1. It is an unsupervised, a bidirectional model for the pre-training of languages.

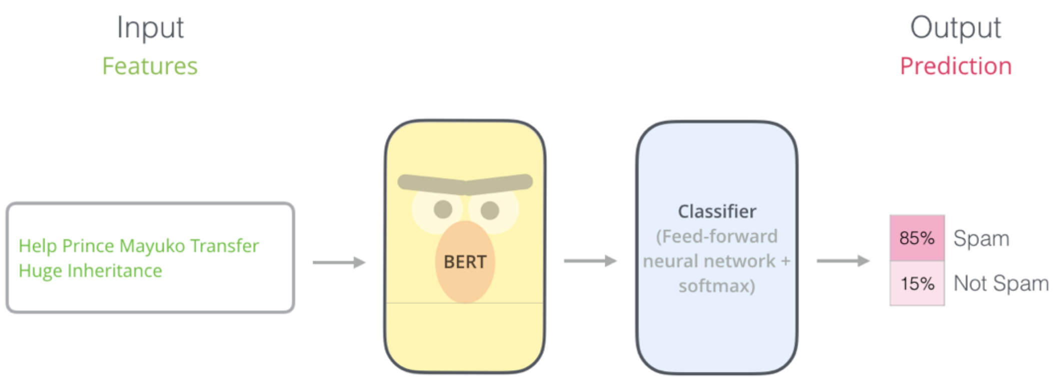
Input
Features
Output
Prediction
Help Prince Mayuko Transfer Huge Inheritance
BERT
Classifier (Feed-forward neural network + softmax)
85% Spam
15% Not Spam

Figure 1BERT text classification

How does it do it?

BERT: BERT has two steps pre-training and fine-tuning. Pre-training is performed on unlabeled data specific to a language task. After that, a fine-tuning is performed for a particular target task. For this, a BERT model is initialized with pre-trained parameters and after that each and every parameter is optimized using supervised learning specific to a target task like question and answering. Each of the target task has their own optimized model, however, they are fine-tuned with same pre-trained parameters.

The architecture of BERT's model is made up of several layers of **Transformer** [2] model as suggested by Vaswani et al. Transformer is an attention model which captures attention of word in both direction from left-to-right and from right-to-left in a sentence, therefore, it preserves context of a word in a sentence with accuracy.

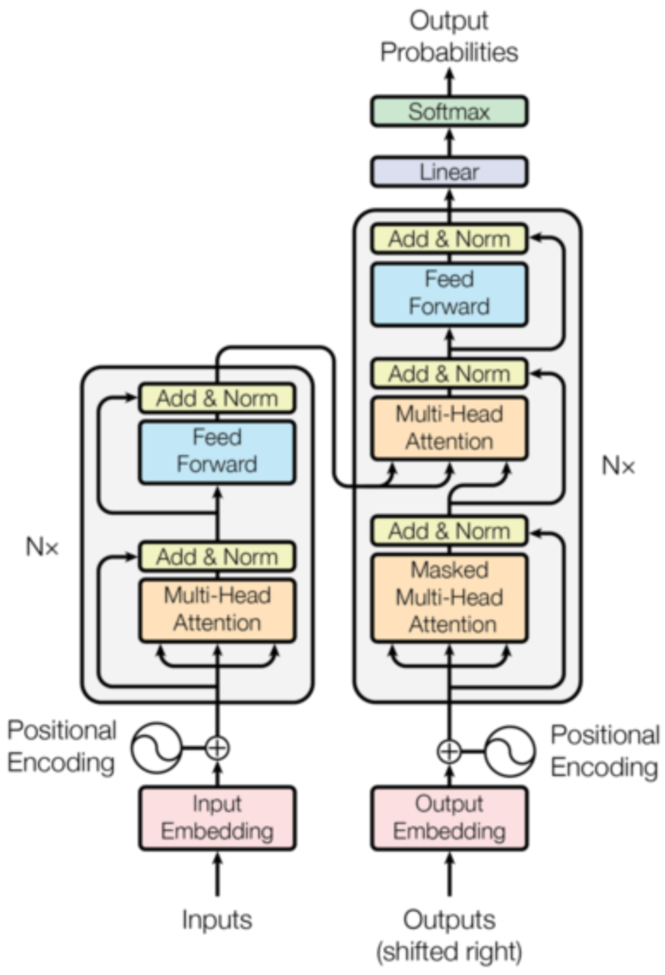

Figure 2: The Transformer – model architecture.

As shown in figure 2, Transformer is made up of two parts Encoder (left box) and Decoder (right box). Encoder is composed of 6 layers. One of the layers of Encoder is shown on left side. Each layer has two sub-parts. Multi-Head Self Attention is the first part, and position-wise fully connected feed-forward network is the second part. Similar to Encoder, Decoder is made up of 6 similar layers. Apart two sub-parts similar Encoder, it has one more layer as sub-part which perform multi-head attention with output of the Encoder.

Scale Dot-Product Attention: In attention part which is Scale Dot-Product Attention is defined as:

$$\text{Attention (Q, K, V)} = \text{SoftMax}\left(\frac{QK^T}{\sqrt{dk}}\right)V \qquad \text{Eq. 1}$$

where Q is group on queries as input, K is set of input keys and V is group of values, they all packed together into matrices and Attention is calculated from equation 1. To illustrate it further, analogy can be taken from database system where query (Q) is the statement use to fetch data, key (K) is association with the value (V). So, Q is group of embedded vectors of words that will pass to Encoder. K and V are previous embedded vectors that were passed to the Encoder and represented as memory vectors. Therefore, a query or current word is passed and look for similar words by traversing keys that are saved into memory vectors. Finally, to represent distribution of attention of current word with respect to other existing words, a SoftMax function is used.

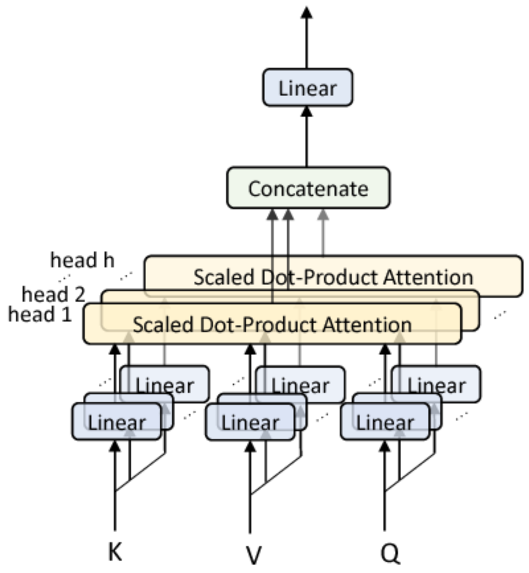

Figure 3: Multi-Head Attention

Multi-Head Attention: When an attention is performed in parallel is called multi-head attention, it takes one of the nth model's query, key and value and projected linearly to vectors corresponding to Q, K and V. All of these connected and projected parallelly to a final value as shown in equation 2.

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \ldots, \text{head}_h)$$
$$\text{where head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \qquad \text{Eq. 2}$$

BERT paper mentioned two flavors of it, BERT$_{base}$ and BERT$_{large.}$ BERT$_{base}$ is smaller version which can be compared with GPT and has 12 layers of Transformer, 768 hidden states and 12 self-attention head while BERT$_{large}$ has 24 layers of Transformer, 1024 hidden states and 16 self-attention head.

BERT INPUT: Input of BERT to be consistent to work on different kind of task so, it handles both single sentence and pair of sentences (question, answer). For the purpose of illustration, we would take WordPiece as embedding vectors. It contains 30,000 tokens for its vocabulary. For every sentence or sequence, first token is ([CLS]) which is used specially for the classification. Final hidden state is used as aggregator for classification task. Paired sentences are fed together and demarked by special token ([SEP]) and learning embedding added to each token to differentiate one sentence with another sentence.

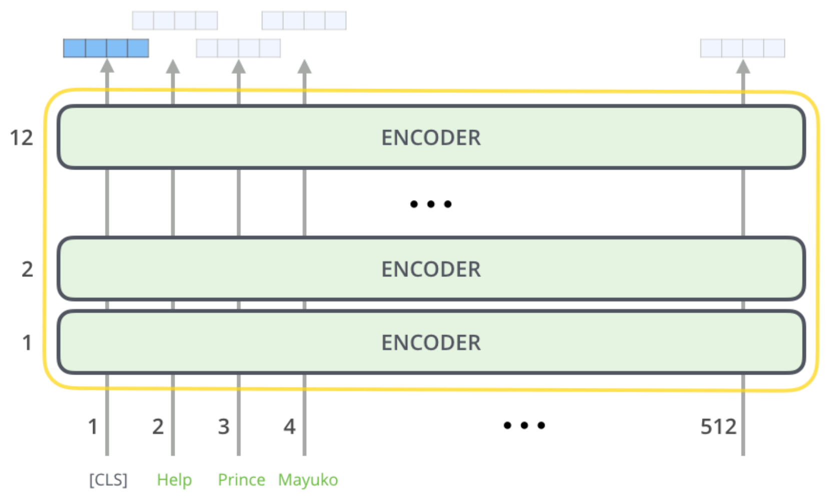

Figure 4 BERT-Input-Output of Sequence.

Let's take an example of classification, BERT takes sequence of words as input in parallel and that will be passed next layer of Transformer which then adds self-attention and passes data to next layer through feed-forward neural network and so on until data reaches to last layer. At last layer, each position emits a vector whose size is equal to hidden state that is 768 for the BASE$_{\text{base}}$. Since, it is a classification task therefore, we will consider only the output at first position output. This output or vector will be used to feed to single layer of neural network classifier, please refer the figure 3 and 4.

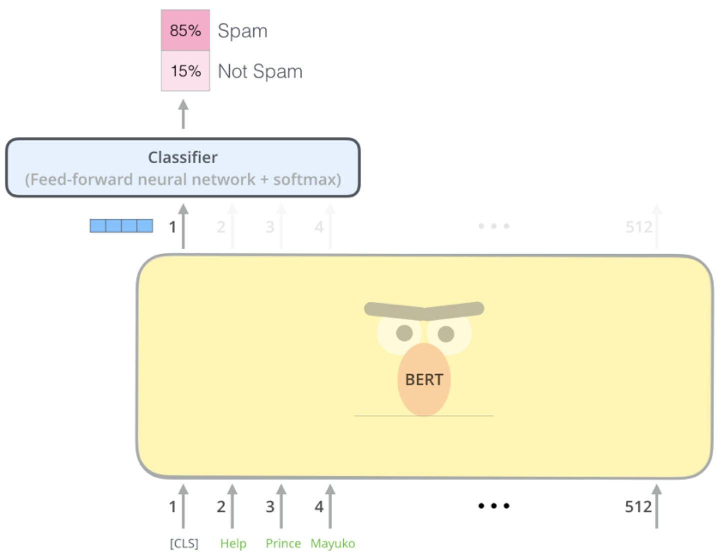

Figure 5 BERT- Text classification at fine-tuned layer.

Pre-training BERT: Next, we will show how to train BERT bidirectionally using unsupervised data.

Masked LM: To train a BERT model bidirectionally, we would mask few input tokens in random order and then predicts those tokens who are masked. Final hidden output vectors of masked token are then fed into SoftMax function using vocabulary. Generally, 15% of WordPiece would keep as masked and only masked tokens are predicted. Even thought, it creates a mismatch at the time of fine-tuning if masked tokens are replaced with words. Therefore, fine-tunning does not change masked worlds.

Next Sentence Prediction: Most of tasks of NLP are linked with relationships between two sentences, such as question and answer or next sentence prediction. Therefore, pre-training step consists of determining wheatear a given sentence A is following or not. For this, 50% of the time sentences are labeled as IsNext and 50% as NotNext.

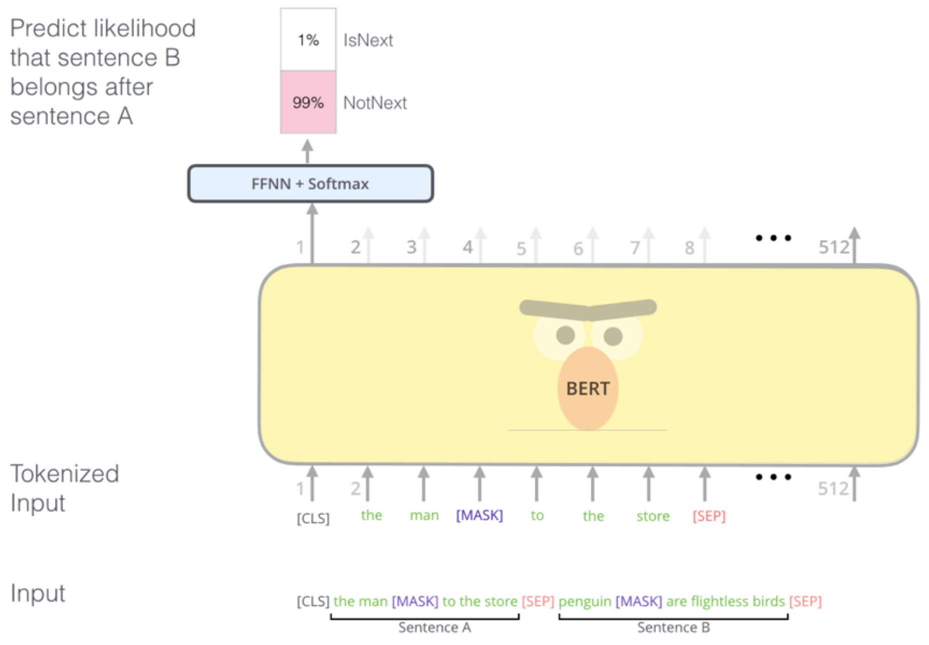
Predict likelihood that sentence B belongs after sentence A
1%
IsNext
99%
NotNext
FFNN + Softmax
1
2
3
4
5
6
7
8
512
BERT
Tokenized Input
[CLS]
the
man
[MASK]
to
the
store
[SEP]
Input
[CLS] the man [MASK] to the store [SEP] penguin [MASK] are flightless birds [SEP]
Sentence A
Sentence B

Figure 6: BERT- Next Sentence Prediction (NSP)

BERT – Feature Extraction: BERT can be used as feature extraction in the form of word embedding with context. After that those features can be used in exiting models, its more or like name-entity recognition.

Does it work?

Bert is simple and powerful model. It shows state-of-the-work on 11 NLP tasks, such as GLUE (which is a collection of several different kinds of natural language task) score. GLUE includes MNLI (Multi-Genre Natural Language), QQP (Quora Question Pairs), QNLI (Question Natural Language Inference), SST-2(The Stanford Sentiment Treebank), CoLA (Corpus of Linguistic Acceptability).

SQuAD v1.1 stands for The Stanford Question Answering Dataset contains 100, 000 question-answer pairs that were crowd sourced. Task is that for a give question from a Wikipedia at article, it has to find answer and it's starting point. BERT outperform best result F1 score available from ensemble system. However, SQuAD 2.0 task is an improvement over its earlier predecessor. It is more realistic version of question and answer system, it could predict long answer besides short answer and has +5.1 F1 score improvement.

SWAG (Situations With Adversarial Generations) contains 113, 000 sentence pair completion examples, it evaluates how does common inference could be drawn from it. For instance, a sentence is given, and we need to find the best answer from the give four answers. A $\text{BERT}_{\text{large}}$ model with fine-tuned parameters, such as 3 epochs, 2e-5 learning rate and batch of size 16 has outperformed author's baseline model by 27.1% and GPT by 8.3%.

Google published an article [3] about uses of BERT for improving their SEO results. Google process billions of searches daily out of which approximately 15% are new, but they process them. A user might type or spell a word in different ways or they might combine them in any order, in order to make query to search an item on Google. Therefore, understanding of language is very core to their operation. With the advent of NLP technologies like BERT has improved their query results and snippets in search results. For an instance, query contains "to" and "from" can change the meaning of query by placement of "to" and "from". They improved the result of query, when a traveler wants to search "2019 brazil traveler to usa need a visa". Previously, Google could not able to understand the context and returned the results relating to US citizens searching for visa, they totally ignored the context, but now, with the help of BERT they could show correct results.

Where/how does it fail?

Even though BERT is good at sentence completion tasks, however, it lacks the common sense that human possess. An example, that was generated by BERT is "uses a sharpener to smooth the stone using the knife".

BERT has two limitations: It is required to have inputs to be masked but in real life this is not the case, and secondly it neglects dependency among masked tokens.

Another place where BERT fails when we deploy on commercial dynamic environments. BERT works well in static environment where we have pre-trained model, however, on production environment which is very dynamic i.e. new terms, vocabulary or text added rapidly. Therefore, it is needed to retrain model to get optimal result, since, BERT model is complex and required a huge amount of computation power, resources (Cloud TPU) and time. In a regular infrastructure it is performed on a regular basic, but with BERT, this is impractical and hence yield poor performance.

How could it be improved?

With right design choices and correct uses of hyper parameters of BERT for pre-training of model could improve the accuracy of the model as suggested in paper RoBERTa [4]. They found BERT is undertrained, therefore, they used dataset, like CC-NEWS(76GB), OpenWebText(38GB). Instead of static making they used dynamic masking. They have also changed the objectives for training, instead of NSP, they used variant, such as SEGMENT-PAIR, SENTENCE-PAIR, FULL-SENTENCES and DOC-SENTECES.

Similar to RoBERTa, there is another paper XLNET [5] which has outperformed the BERT model. It has rectified the problems associated with BERT model by using auto-regressive models, therefore, it predicts tokens in a sequence.

A recent paper on LISA (Linguistically-Informed Self-Attention) [6] model that uses embedding syntactic dependency parsing could be used for commercial propose without re-training the base model.

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” *arXiv:1810.04805 [cs]*, May 2019, Accessed: Sep. 11, 2020. [Online]. Available: http://arxiv.org/abs/1810.04805.

[2] A. Vaswani *et al.*, “Attention Is All You Need,” *arXiv:1706.03762 [cs]*, Dec. 2017, Accessed: Sep. 11, 2020. [Online]. Available: http://arxiv.org/abs/1706.03762.

[3] “Understanding searches better than ever before,” *Google*, Oct. 25, 2019. https://blog.google/products/search/search-language-understanding-bert/ (accessed Sep. 12, 2020).

[4] Y. Liu *et al.*, “RoBERTa: A Robustly Optimized BERT Pretraining Approach,” *arXiv:1907.11692 [cs]*, Jul. 2019, Accessed: Sep. 12, 2020. [Online]. Available: http://arxiv.org/abs/1907.11692.

[5] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, “XLNet: Generalized Autoregressive Pretraining for Language Understanding,” in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 5753–5763.

[6] E. Strubell, P. Verga, D. Andor, D. Weiss, and A. McCallum, “Linguistically-Informed Self-Attention for Semantic Role Labeling,” *arXiv:1804.08199 [cs]*, Nov. 2018, Accessed: Sep. 13, 2020. [Online]. Available: http://arxiv.org/abs/1804.08199.

[3] “Understanding searches better than ever before,” *Google*, Oct. 25, 2019.

https://blog.google/products/search/search-language-understanding-bert/ (accessed Sep. 12,

2020).