Tech Review: Overview in BERT paper
Lingfei Yang

BERT refers to Bidirectional Encoder Representations from Transformers, which is used in pre-trained representation. It is designed to pretrain deep "bidirectional" representations from unlabeled text by jointly conditioning on "both left and right" context in all layers.

I. Overview

In the BERT paper, it compares with Elmo (Peters et al., 2018a) and GPT, i.e., Generative Pre-trained Transformer, launched by OpenAI (Radford et al., 2018). It highlights two main differences:

1. GPT uses left context only, while BERT employs both left and right contexts, which is bidirectional, to enhance the performance.

2. The framework based is different.
   - Elmo: based on RNN framework, so it requires adjustment.
   - BERT: based on transformer framework, so it only requires change in one additional output layer for a wide range of tasks.

As tested, performance of BERT is better on 11 natural language processing tasks in terms of GLUE score, MultiNLI accuracy, and SQuAD v1.1 and v2.0 question answering Tests F1.

The empirical results show that the advantage of BERT is the it enables low-resource tasks and generalizes the implementation in deep bidirectional architectures so that is is broadly used in NLP (Natural Language Processing).

There are two main tasks in NLP:

- **Sentence-level tasks** (such as natural language inference, paraphrasing) aims to predict the relationships between sentences.
- **Token-level tasks** (such as named entity recognition, question answering): models are required to produce fine-trained output at the token level.

Two strategies for pre-trained language representations to downstream tasks are:

- **Feature-based**: e.g. ELMo, in essence, it is based on RNN framework. It uses task-specific architectures that include the pre-trained representations as additional features.
- **Fine-tuning**: e.g. GPT, uses minimal task-specific parameters trained on the downstream tasks by simply fine-tuning all pertained parameters.

The common area is that both use "**undirectional**" language models. Language model is undirectional, for example, given words, the model is to predict what need to say, which is considered as undirectional.

To relieve the restrictions of undirectional model, BERT uses:

1. "**masked language model**" (MLM, Close task, Taylor, 1953) pre-trained objectives, which randomly masks some tokens from the input and predict the original word by allowing both left and right context.

2. "**next sentence prediction**" task: Given two sentences, the task is to predict if these two sentences are randomly picked, or they jointly retrains text-pair representations. The model can use this task to learn sentence level information.

The contribution of BERT:

1. Bidirectional
2. Its pre-trained representations reduce the need for many heavily-engineered task specific architectures. It is the first time that a model performs both feature-based tasks and fine-turning tasks.
3. Codes are available at https://github.com/google-research/bert.

II. Details of BERT

Two steps in the BERT framework:

- Pre-training: model is trained on unlabeled data over different pre-training tasks.
- Fine-tuning: first initialize with the pre-trained parameters, all parameters are fine-tuned using labeled data from the downstream tasks. Each downstream task has separate fine-tuned models but with the same pre-trained parameters.

Model architecture: multi-layer bidirectional Transformer encoder.

BERTbase (L=12, H=768, A=12), total parameter = 110M, the size is the same as OpenAI GPT for comparison purpose.
BERTlarge (L=24, H=1024, A=16), total parameters=340M, used to display higher performance

Here,
- L: number of layers
- H: hidden size
- A: the number of self-attention heads

Input/Output Representations: both a *single* sentence and a *pair* of sentences (e.g. <Question, Answer>) where "*sentence*" can be an arbitrary span of contiguous text. Inputs are called "**sequence**", which can be a single sentence or two sentences together. It is different from Transformers, which has both input sequence and output sequence. Instead, BERT has only one sequence. However, it can combine two sentences together.

The model uses WordPiece embeddings with a 30,000 token vocabulary. The first token of a sequence is always special classification token ([CLS]). To differentiate sentence pairs (packed as a single sequence), two methods are: (1) separate them with a special token [SEP], (2)add a learned embedding to every token to indicate whether it belongs to sentence 1 or 2.

Task #1: Masked LM

15% of all WordPiece tokens in each sequence is masked at random ([MASK]). The target is to predict the masked words. However, it leads to a downside that is a mismatch between pre-training and fine-tuning. To solve this problem, (1) 80%: [MASK], (2) 10%: random token, (3) 10%: unchanged i-th token.

Task #2: Next Sentence Prediction (NSP)

It is about the relationship between two sentences (A and B). 50% of the time B is correct next sentence of A (IsNext), and 50% is a random sentence form the corpus (NotNext).

Pre-training data: BooksCorpus (800M words), English Wikipedia (2,500M words). Document-level corpus.