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

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova

Abstract

- Introduce a new language representation model called BERT (**Bidirectional Encoder Representations from Transformers**)
- It is designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers.
- It can be fine tuned with just one additional output layer
- Create models for wide range of tasks, such as question answering and language inference, without substantial task specific architecture modifications.

Introduction

- Language model pre-training has been shown to be effective for improving many natural language processing tasks.
- Two existing strategies: feature-based and fine-tuning.
- The feature-based approach include the pre-trained representations as additional features.
- The fine-tuning approach, such as the **Generative Pre-trained Transformer** (OpenAI GPT) introduces minimal task-specific parameters, and is trained on the downstream tasks by simply fine-tuning all pretrained parameters.
- The two approaches share the same objective function during pre-training, where they use unidirectional language models to learn general language representations.

Introduction (contd.)

- OpenAi GPT uses left to right architecture.
- Can be harmful for tasks like Q&A where context from both directions is required.

Contributions of the paper

- They have demonstrates the importance of bidirectional pre-training for language representations
- They show that pre-trained representations reduce the need for many heavily-engineered task specific architectures.
- BERT advances the state of the art for eleven NLP tasks.

BERT

- There are two steps in BERT framework: pre-training and fine-tuning.
- During pre-training, the model is trained on unlabeled data over different pre-training tasks.
- For fine tuning, the BERT model is first initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks
- BERT’s model architecture is a multi-layer bidirectional **Transformer** encoder based on the original implementation described in **Vaswani** et al. (2017)
- BERTBASE: L=12, H=768, A=12, Total Parameters=110M
- BERTLARGE: L=24, H=1024, A=16, Total Parameters=340M

Pre-training

The BERT pre-training phase consists of two unsupervised predictive tasks, one is the Masked Language Model and the other is Next Sentence Prediction.

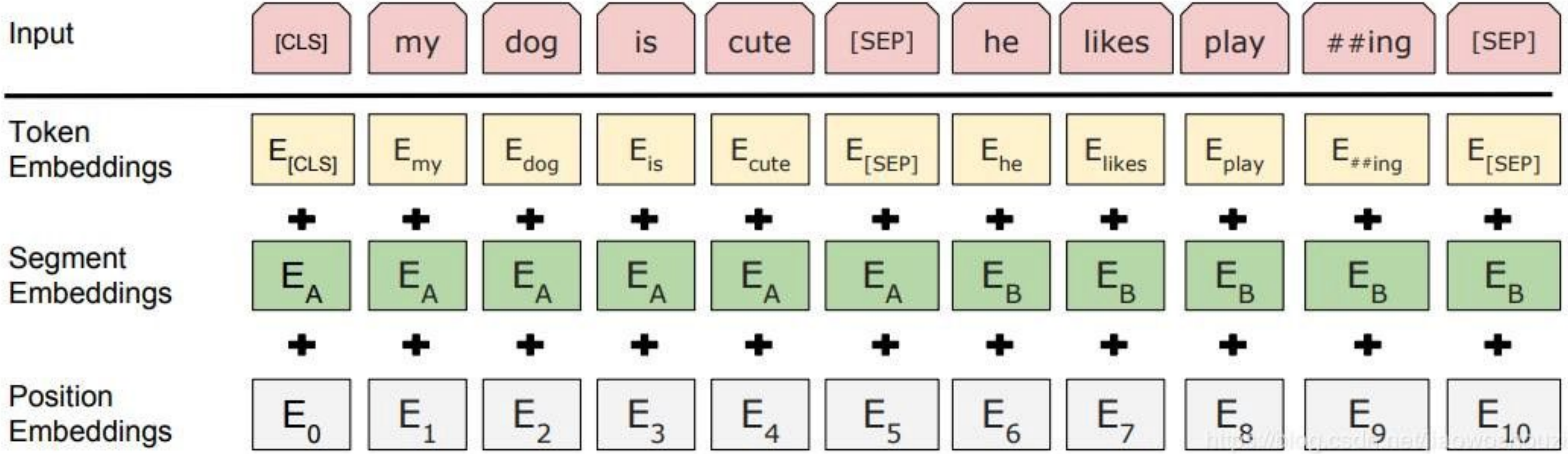
Masked Language Model

- In order to train a deep bidirectional representation, some percentage of the input tokens are simply masked at random, and then those masked tokens are predicted.
- The output token vectors are fed to softmax over the vocabulary.

Next sentence prediction

- To understands sentence relationships BERT also pre-trains for a binarized next sentence prediction task
- Select some sentences for A and B, where 50% of the data B is the next sentence of A, and the remaining 50% of the data B are randomly selected in the corpus, and learn the correlation.
- The purpose of adding such pre-training is that many NLP tasks such as QA and NLI need to understand the relationship between the two sentences, so that the pre-trained model can better adapt to such tasks.

BERT model input



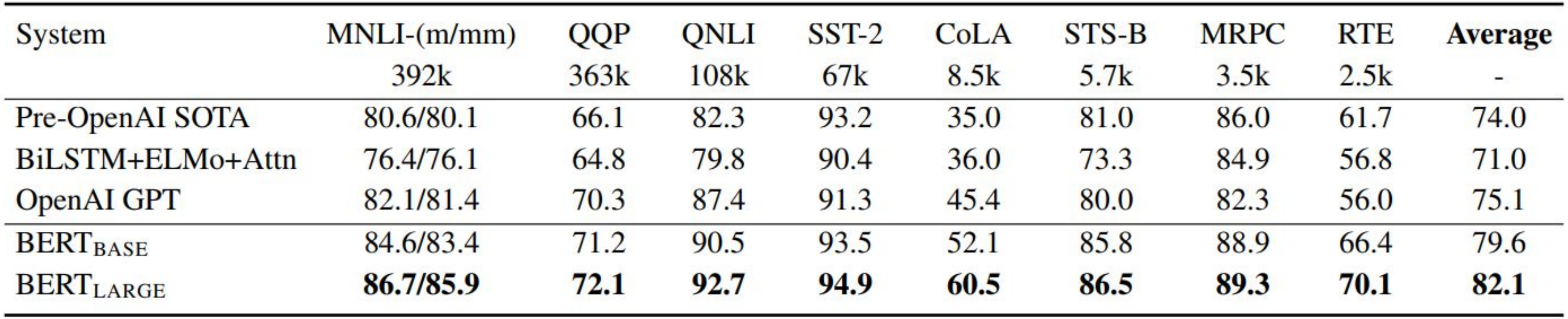
Input	[CLS]	my	dog	is	cute	[SEP]	he	likes	play	##ing	[SEP]
Token Embeddings	$E_{[CLS]}$	E_{my}	E_{dog}	E_{is}	E_{cute}	$E_{[SEP]}$	E_{he}	E_{likes}	E_{play}	$E_{\#\#ing}$	$E_{[SEP]}$
	+	+	+	+	+	+	+	+	+	+	+
Segment Embeddings	E_A	E_A	E_A	E_A	E_A	E_A	E_B	E_B	E_B	E_B	E_B
	+	+	+	+	+	+	+	+	+	+	+
Position Embeddings	E_0	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}

Img source: Breaking BERT Down

BERT Fine-Tuning for Downstream NLP Tasks

- For each downstream NLP task, we simply plug in the task specific inputs and outputs into BERT and fine-tune all the parameters end-to-end.
- Sentence A and sentence B from pre-training can be analogous to sentence pairs in paraphrasing, hypothesis-premise pairs in entailment, question-passage pairs in question answering, etc.
- Token representations are fed into an output layer for token level tasks, such as sequence tagging or question answering
- The [CLS] representation is fed into an output layer for classification, such as entailment or sentiment analysis.

Results



System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
BERT$_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT$_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

GLUE Test results, scored by the evaluation server (https://gluebenchmark.com/leaderboard). The number below each task denotes the number of training examples. The “Average” column is slightly different than the official GLUE score, since we exclude the problematic WNLI set.8 BERT and OpenAI GPT are single model, single task. F1 scores are reported for QQP and MRPC, Spearman correlations are reported for STS-B, and accuracy scores are reported for the other tasks. We exclude entries that use BERT as one of their components.