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

Comment on this paper

- a new era of NLP has just begun a few days ago
    - Google Brain Research Scientist Thang Luong
- a milestone, ... Google's tradition of violent aesthetics
    - Baoxun Wang, Chief Scientist of Chinese AI startup Tricorn
- 충격, 공포
    - Naver Clova

Summary

- a new leanguage **representation** model
- designed to pre-train deep bidirectional representations
    - by jointingly conditioned on left and right context in all layers
- can be fine-tuned for a wide range of NLP tasks
- **SOTA on 11 NKLP tasks**
- only-but-important novelty is **bidirectional on transformer**

Introduction

Language model pre-training

- effective for improving NLP tasks such as
    - sentence-level tasks
        - natural language inference
        - para-phrasing
    - token-level tasks
        - named entity recognitiion
        - SQuAD quetion and answering

two ways of applying pretrained repr to downstream tasks

- feature-based
    - use tasks-speicific architectures with pre-trained repr as additional features
- fine-tuning
    - such as Generative Pre-trained Transformer (OpenAI GPT)
    - use minimal task-specific parameters,
    - is trained on the downstream tasks by simply fine-tuning the pretrained parameters

Our approach

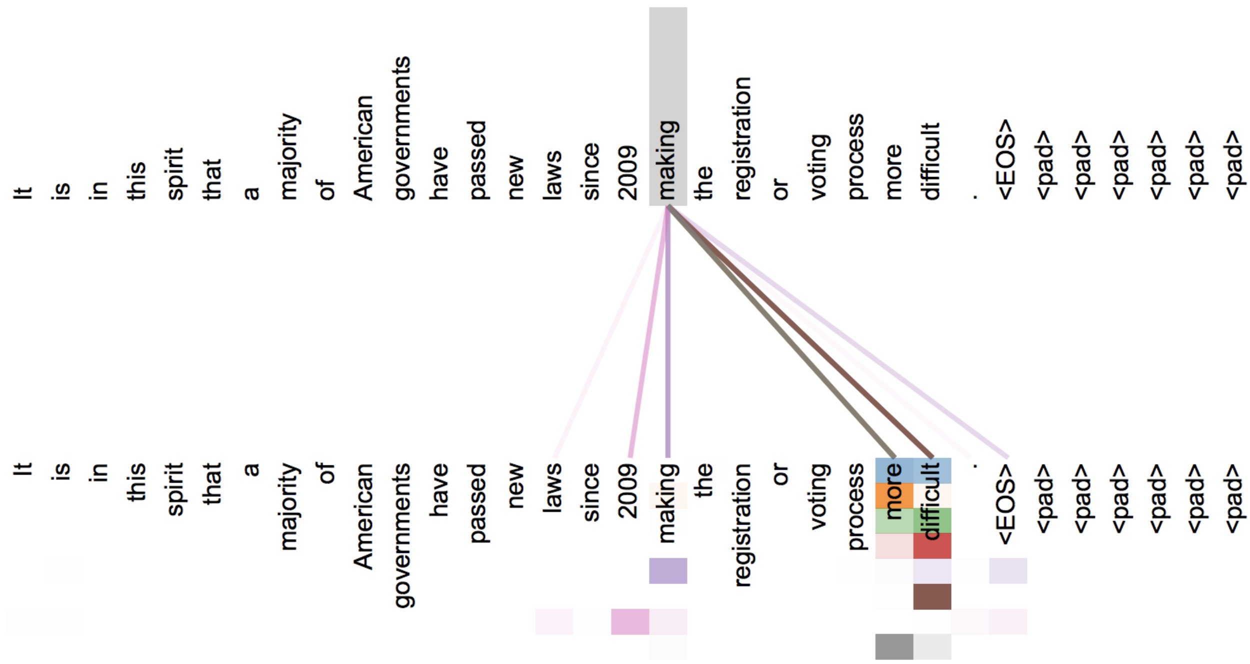
- blame on current fine-tuning method
  - only use unidirectional, left-to-right context
  - limits on attention : attend only to left context
- suggest masked language model(MLM)
- suggest next sentence prediction task

Prerequsites

Transformer

- Attention is all you need
- [https://arxiv.org/pdf/1706.03762.pdf (https://arxiv.org/pdf/1706.03762.pdf)](https://arxiv.org/pdf/1706.03762.pdf)
- [http://nlp.seas.harvard.edu/2018/04/03/attention.html (http://nlp.seas.harvard.edu/2018/04/03/attention.html)](http://nlp.seas.harvard.edu/2018/04/03/attention.html)
- Google, 2017
- 계산이 많이 드는 RNN/LSTM 대신해서 간단한 Product-based attention 기제를 가지고 LM tasks를 풀 수 있다.

Attention Visualizations



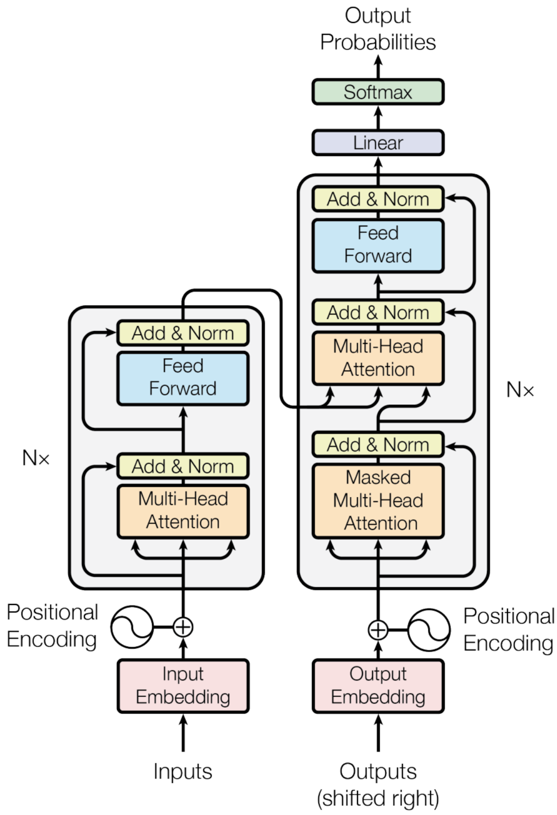


Figure 1: The Transformer - model architecture.

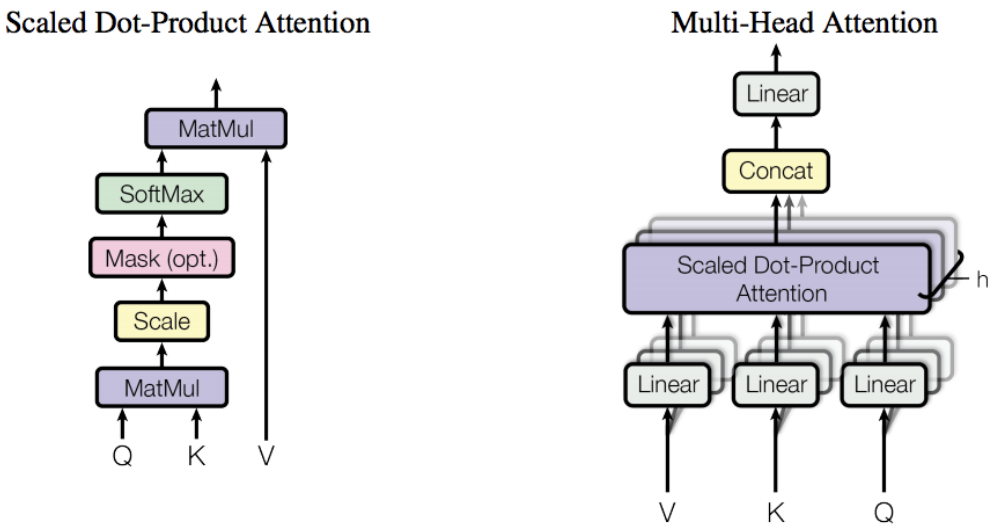


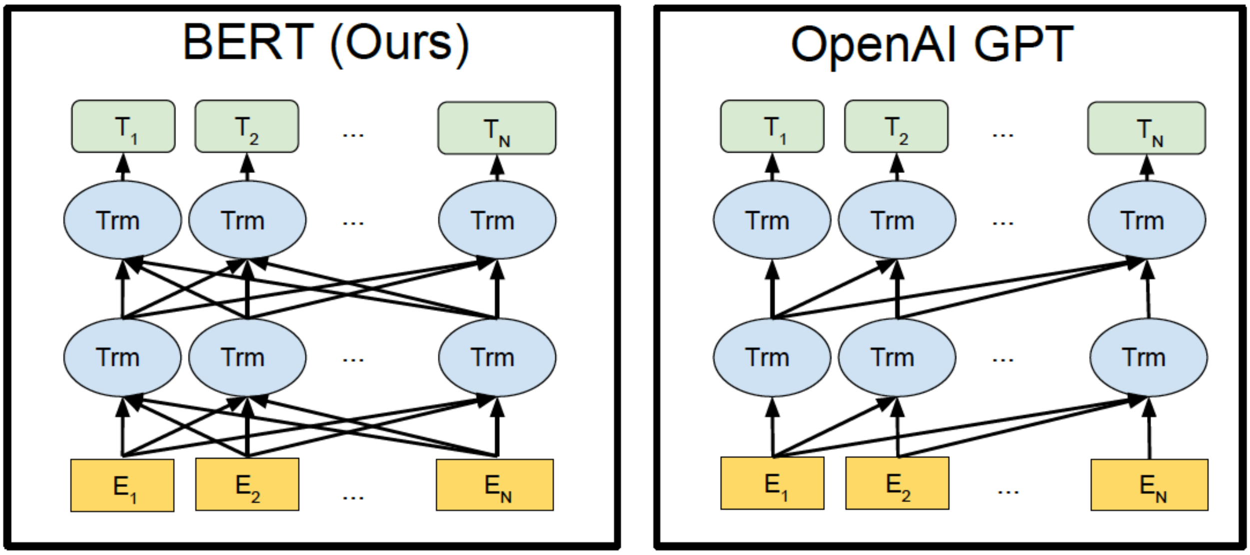
Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

OpenAI Transformer

- https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf (https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf)
- OpenAI, Radfold, 2018
- Transformer를 pre-training 할 때 사용



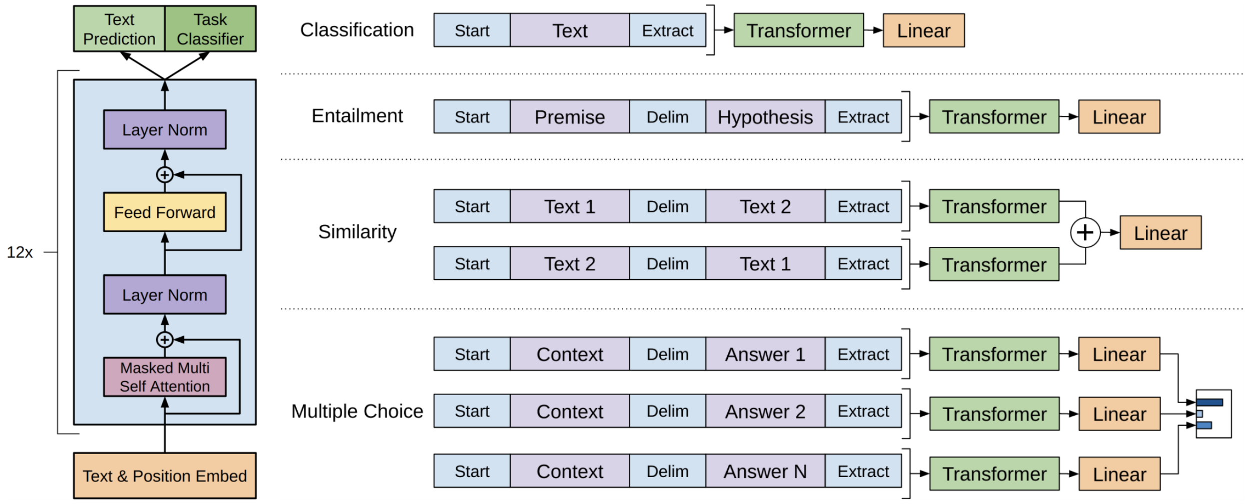


Figure 1: **(left)** Transformer architecture and training objectives used in this work. **(right)** Input transformations for fine-tuning on different tasks. We convert all structured inputs into token sequences to be processed by our pre-trained model, followed by a linear+softmax layer.

BERT

- multi-layer bidirectional transformer encoder
  - same as vaswani, 2017, tensor2tensor
  - https://github.com/tensorflow/tensor2tensor (https://github.com/tensorflow/tensor2tensor)
  - http://aclweb.org/anthology/W18-1819 (http://aclweb.org/anthology/W18-1819)

- **BERT**$_{\textbf{BASE}}$: L=12, H=768, A=12, Total Parameters=110M

- **BERT**$_{\textbf{LARGE}}$: L=24, H=1024, A=16, Total Parameters=340M

- Input representation
  - able to handle single sentence or pair of sentences
    - ◦ CLS - first sentence - SEP - second sentence - SEP
  - use WordPiece embedding with 30,000 token vocabulary
  - learned positional embeddings

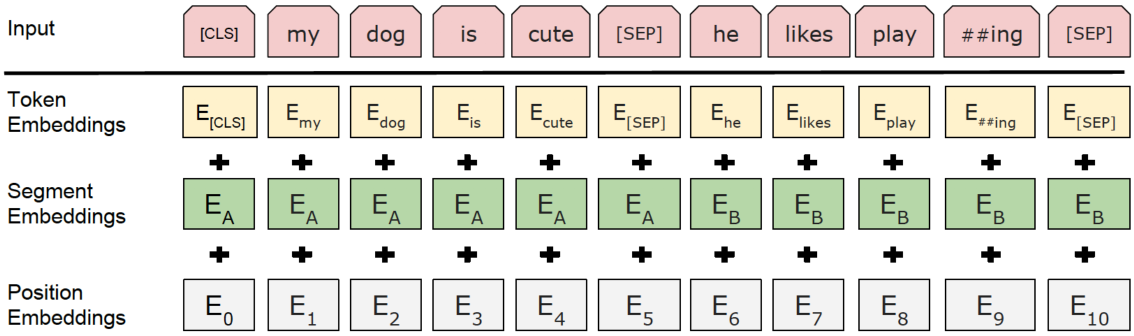


Figure 2: BERT input representation. The input embeddings is the sum of the token embeddings, the segmentation embeddings and the position embeddings.

Pretrained tasks

Task 1: Masked LM

- predict randomly masked words from single sentence, or pair of sentences
- train set : 256 batch, 15% randomly masked

```
[CLS] the man went to [MASK] store [SEP]
```

Task 2: Predict next sentence

- good for Question and Answering, Natural Language Inference(NLI)
- train set generation : positive example from corpus, negative example generation randomly

Input = `[CLS] the man went to [MASK] store [SEP]`

`he bought a gallon [MASK] milk [SEP]`

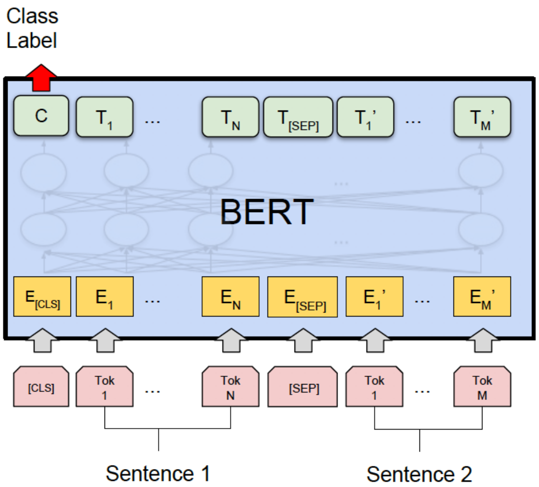
Label = `IsNext`

Input = `[CLS] the man [MASK] to the store [SEP]`

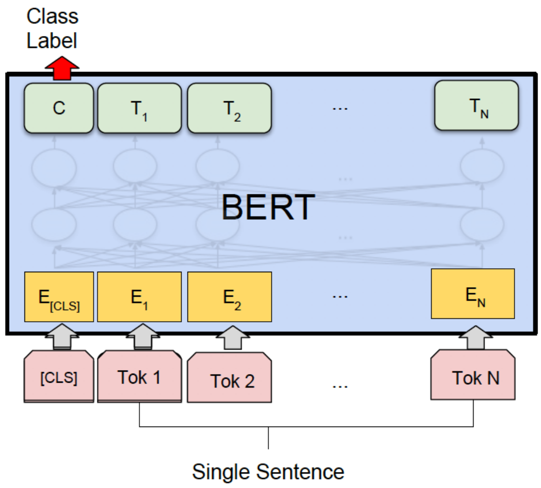
`penguin [MASK] are flight ##less birds [SEP]`

Label = `NotNext`

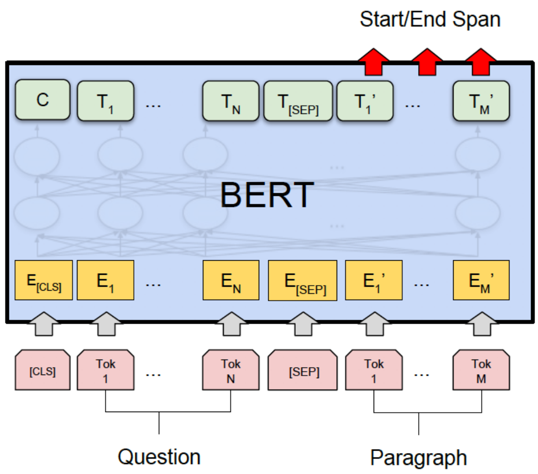
Usage of BERT for downstream tasks



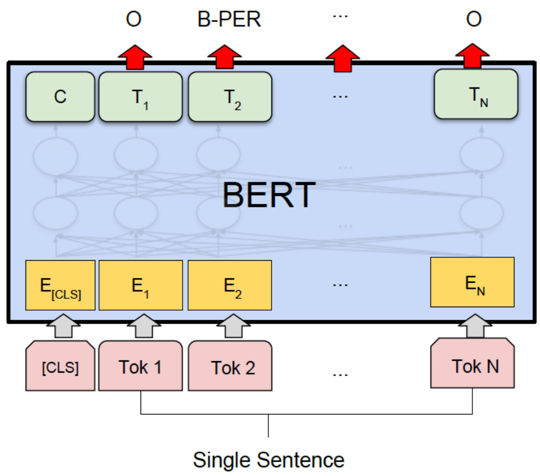
(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG



(b) Single Sentence Classification Tasks:
SST-2, CoLA



(c) Question Answering Tasks:
SQuAD v1.1



(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

Experiments

SOTA on various tasks

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	79.9	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	88.1	91.3	45.4	80.0	82.3	56.0	75.2
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

Ablation Studies

- No Next Sentence Prediction
  - but bidirectional, with masked LM
  - NSP 효과 측정
- Left-to-right, NO NSP

- bidirectional 기여 측정

Tasks	Dev Set MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
$\text{BERT}_{\text{BASE}}$	84.4	88.4	86.7	92.7	88.5
No NSP	83.9	84.9	86.5	92.6	87.9
LTR & No NSP	82.1	84.3	77.5	92.1	77.8

BERT is slower than ...