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

Võ Tuấn Kiệt

Ngày 22 tháng 5 năm 2024

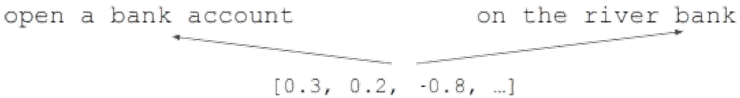
Pretraining in NLP

- Word embeddings are the basic of NLP deep learning

- Problem : Word embeddings (word2vec, GloVe) are often pre-trained on text corpus from co-occurrence statistics

Contextual Representation

- Problem : Word embeddings are applied in a context-free manner



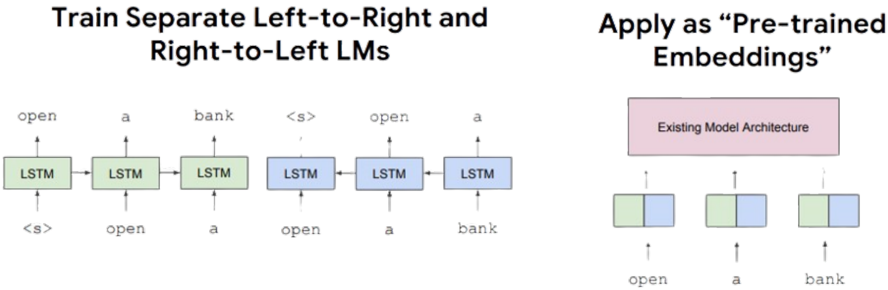
- Solution: Train contextual representations on text corpus

Apply pre-trained language respresentations

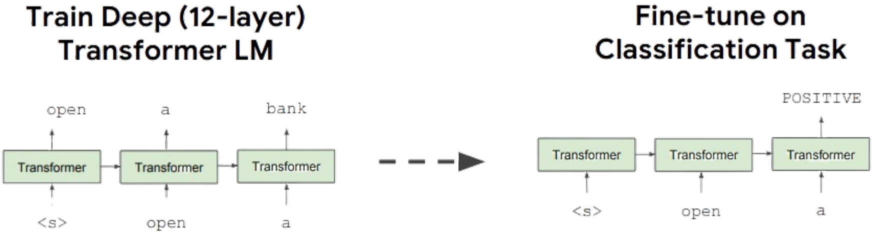
- Feature-based : include pre-trained representations as additional features (eg. ELMo)
- Fine-tunning : introduce task-specific parameters and fine-tune the pretrained parameters (eg. OpenAI GPT)

ELMo

ELMo: Deep Contextual Word Embeddings, AI2 University of Washington, 2017



Improving Language Understanding by Generative Pre-Training, OpenAI, 2018



Limitations of previous method

Standard language models are unidirectional and this limits the choice of architectures that can be used during pre-training.

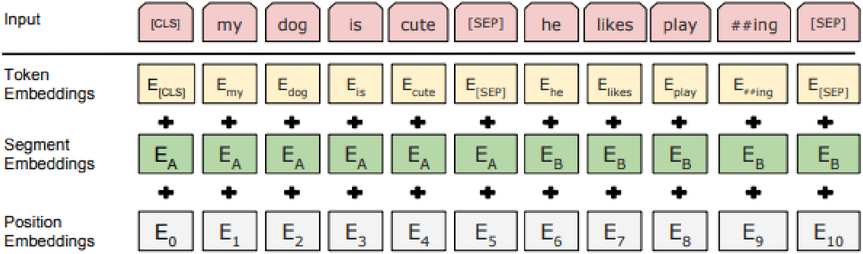
- OpenAI GPT use left-to-right architecture
- ElMo concatenates forward and backward language models

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

- Main idea :
  - Jointly condition on both left and right context in all layers.
  - Alleviates unidirectionality constraint by using :
    1. **Masked language model(MLM)** pre-training objective.
    2. **Next sentence prediction** task that jointly pretrains text-pair representation
- Advantages : BERT model can be finetuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task specific architecture modifications.

Input Resprestentaion

- Token embeddings : Use pretrained WordPiece embeddings
- Positional embeddings : User learned positional embeddings
- Sentence embeddings : Add sentence embeddings for every tokens of each sentence
- Place the [CLS] token at the beginning of each sentence
- Seperate each sentence using [SEP] token



Model Architecture

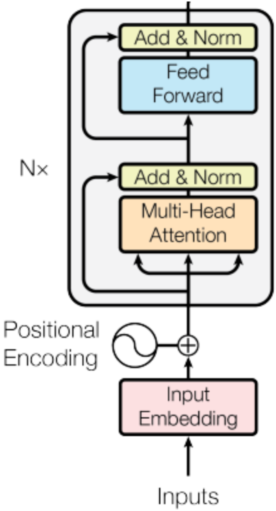
BERT model's architecture are multi-layer bidirectional Transformer encoder :

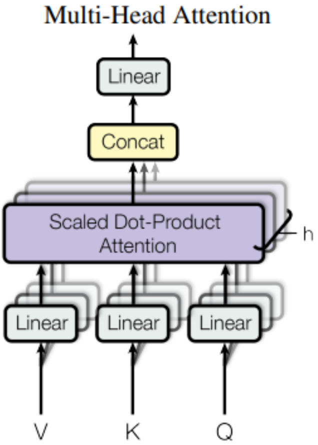
- Multi-headed self attention
- Feed-forward layers
- Layer norm and residuals
- Positional embeddings

Two model with different size were investigated :

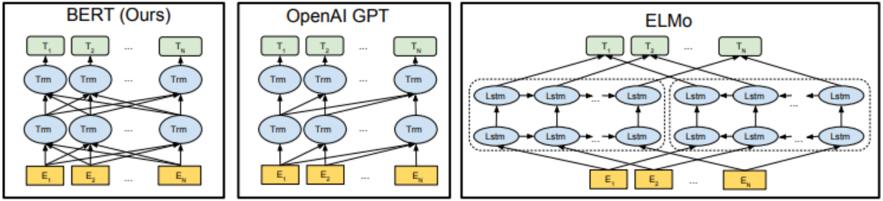
- BERT-Base : 12-layer, 768-hidden, 12-head
- BERT-Large : 24-layer, 1024-hidden, 16-head

Transformer Encoder And Multi-Head Attention





BERT, GPT, ELMo



Hình: Differences in pre-training model architectures

Training Details

- Data: Wikipedia (2.5B words) + BookCorpus (800M words)
- Batch Size: 131,072 words (1024 sequence * 128 length or 256 sequences * 512 length)
- Training Time: 1M steps (40 epochs)
- Optimizer: AdamW, 1e-4 learning rate, linear decay
- Trained on 4x4 or 8x8 TPU slice for 4 days

Pretraining BERT : Masked LM

Mask out 15% of the input words, and then predict the masked words.
Not all tokens were masked in the same way :

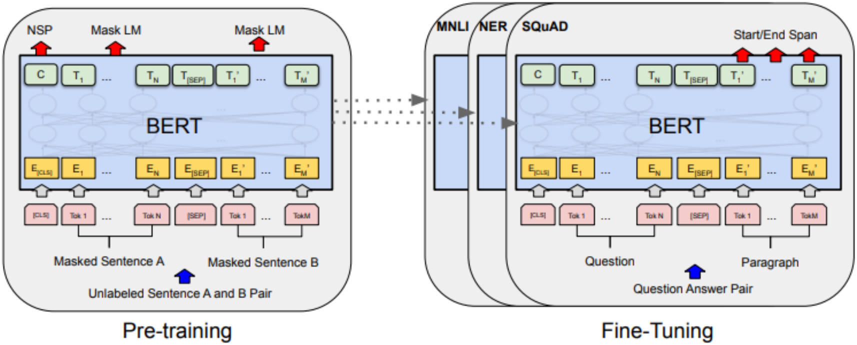
- 80% of the time: Replace the word with the [MASK] token, e.g., my dog is hairy → my dog is [MASK]
- 10% of the time: Replace the word with a random word, e.g. my dog is hairy → my dog is apple
- 10% of the time: Keep the word unchanged, e.g., my dog is hairy → my dog is hairy

Pretraining BERT : Next Sentence Prediction

To learn relationships between sentences, predict whether Sentence B is actual sentence that proceeds Sentence A, or a random sentence, e.g :

- 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

Fine-Tuning Procedure



Fine-Tuning Procedure

- Sequence-level classification : Use the final hidden state of [CLS] token C $\in \mathbb{R}^H$, add a classification layer and use softmax to calculate label probabilities.
- Token tagging task : Feed the final hidden $T_i \in \mathbb{R}^H$ for each token i into a classification layer for the tagset.
- Span-level task : Represent the input question and paragraph as single packed sequence. Learn the start vector $S \in \mathbb{R}^H$ and end vector $E \in \mathbb{R}^H$ by calculating the probability of word i being the start of the anwser span $P_i = \frac{e^{S \times T_i}}{\sum_j e^{S \times T_j}}$

Experiments : GLUE

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
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Table 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.

Experiments : SQuaAD v1.1

System	Dev EM	Dev F1	Test EM	Test F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
Published				
BiDAF+ELMo (Single)	-	85.6	-	85.8
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
$\text{BERT}_{\text{BASE}}$ (Single)	80.8	88.5	-	-
$\text{BERT}_{\text{LARGE}}$ (Single)	84.1	90.9	-	-
$\text{BERT}_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
$\text{BERT}_{\text{LARGE}}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
$\text{BERT}_{\text{LARGE}}$ (Ens.+TriviaQA)	**86.2**	**92.2**	**87.4**	**93.2**

Experiments : SQaAD v2.0

System	Dev		Test	
	EM	F1	EM	F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	86.3	89.0	86.9	89.5
#1 Single - MIR-MRC (F-Net)	-	-	74.8	78.0
#2 Single - nlnet	-	-	74.2	77.1
Published				
unet (Ensemble)	-	-	71.4	74.9
SLQA+ (Single)	-		71.4	74.4
Ours				
BERT$_{\text{LARGE}}$ (Single)	78.7	81.9	80.0	83.1

Table 3: SQuAD 2.0 results. We exclude entries that use BERT as one of their components.

Experiments : SWAG

System	Dev	Test
ESIM+GloVe	51.9	52.7
ESIM+ELMo	59.1	59.2
OpenAI GPT	-	78.0
$\text{BERT}_{\text{BASE}}$	81.6	-
$\text{BERT}_{\text{LARGE}}$	**86.6**	**86.3**
Human (expert)†	-	85.0
Human (5 annotations)†	-	88.0

Table 4: SWAG Dev and Test accuracies. †Human performance is measured with 100 samples, as reported in the SWAG paper.

Experiments : Name Entity Recognition

System	Dev F1	Test F1
ELMo (Peters et al., 2018a)	95.7	92.2
CVT (Clark et al., 2018)	-	92.6
CSE (Akbik et al., 2018)	-	**93.1**
Fine-tuning approach		
$\text{BERT}_{\text{LARGE}}$	96.6	92.8
$\text{BERT}_{\text{BASE}}$	96.4	92.4
Feature-based approach ($\text{BERT}_{\text{BASE}}$)		
Embeddings	91.0	-
Second-to-Last Hidden	95.6	-
Last Hidden	94.9	-
Weighted Sum Last Four Hidden	95.9	-
Concat Last Four Hidden	96.1	-
Weighted Sum All 12 Layers	95.5	-

Table 7: CoNLL-2003 Named Entity Recognition results. Hyperparameters were selected using the Dev set. The reported Dev and Test scores are averaged over 5 random restarts using those hyperparameters.

Ablation Studies : Effect Of Pre-training Tasks

Tasks	Dev Set MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
$BERT_{BASE}$	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
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

Table 5: Ablation over the pre-training tasks using the $BERT_{BASE}$ architecture. "No NSP" is trained without the next sentence prediction task. "LTR & No NSP" is trained as a left-to-right LM without the next sentence prediction, like OpenAI GPT. "+ BiLSTM" adds a randomly initialized BiLSTM on top of the "LTR + No NSP" model during fine-tuning.

Ablation Studies : Effect Of Model Size

Hyperparams				Dev Set Accuracy		
#L	#H	#A	LM (ppl)	MNLI-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

Table 6: Ablation over BERT model size. #L = the number of layers; #H = hidden size; #A = number of attention heads. "LM (ppl)" is the masked LM perplexity of held-out training data.