BERT

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

Paper 소개

- Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Paper: https://arxiv.org/abs/1810.04805
- Transformer의 Encoder 부분을 이용해, Masked Language Model(MLM)과 Next sentence prediction으로 pre-train 하는 방법 제시

모델 구조

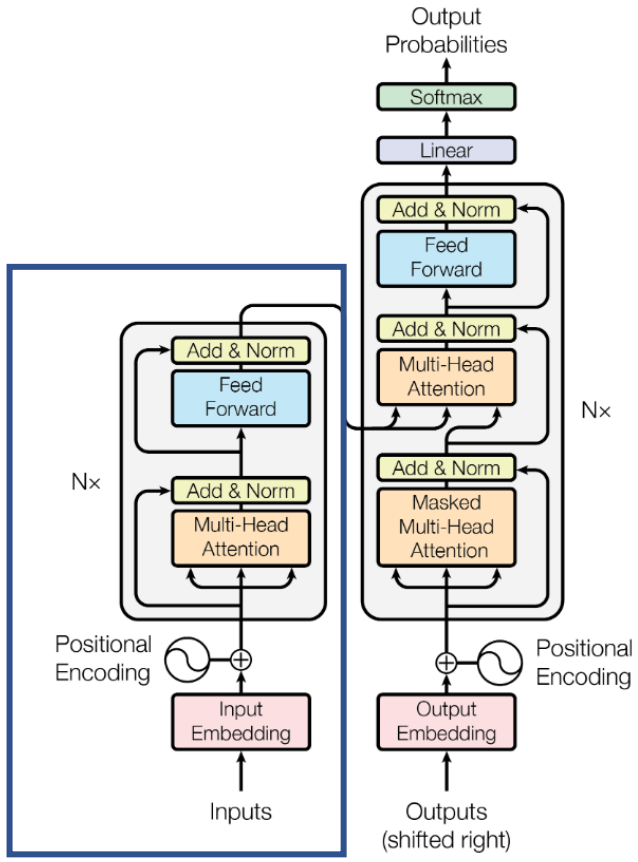
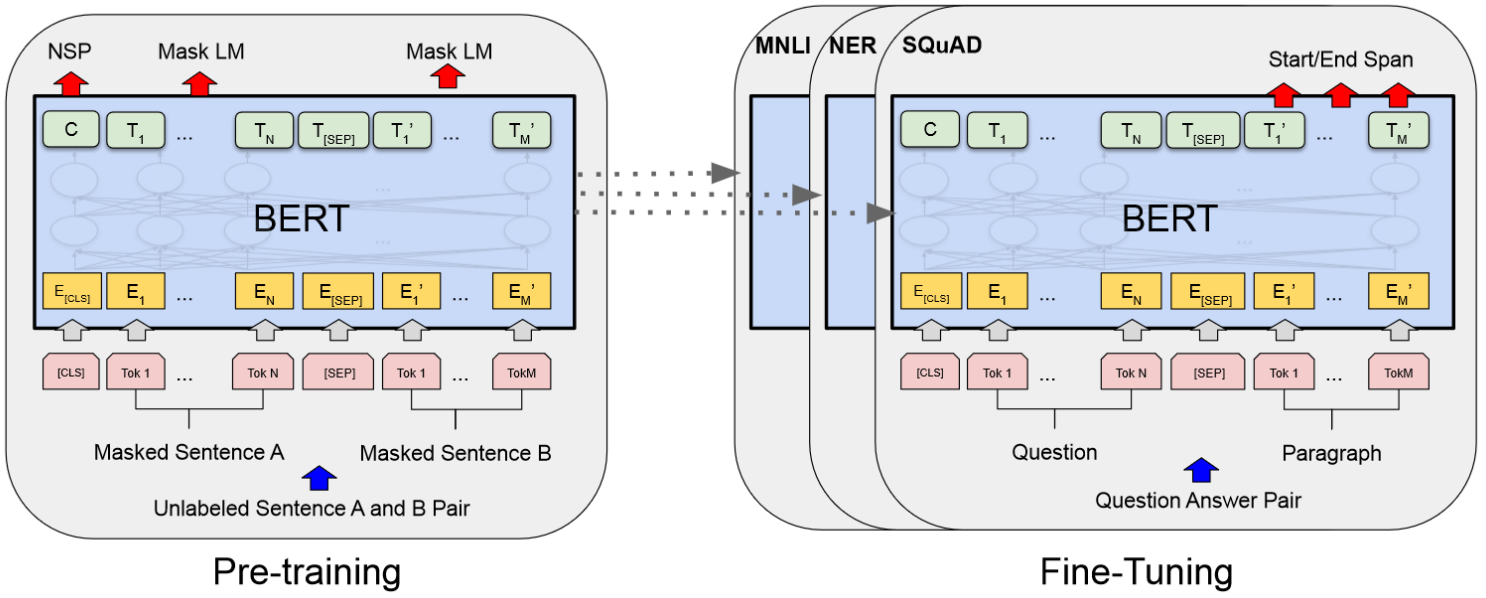


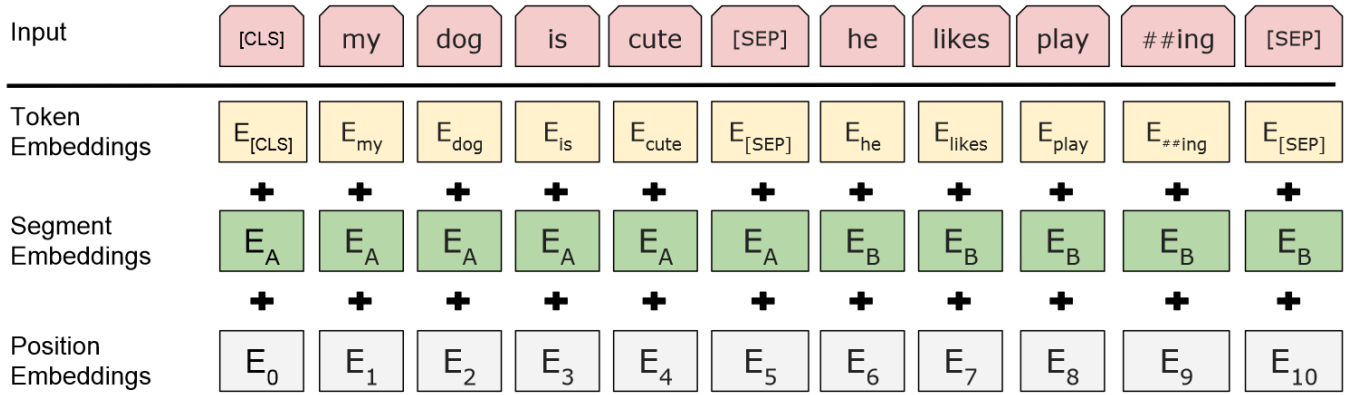
Figure 1: The Transformer - model architecture.



- Transformer의 encoder 구조를 그대로 사용
- Attention 기반 encoder

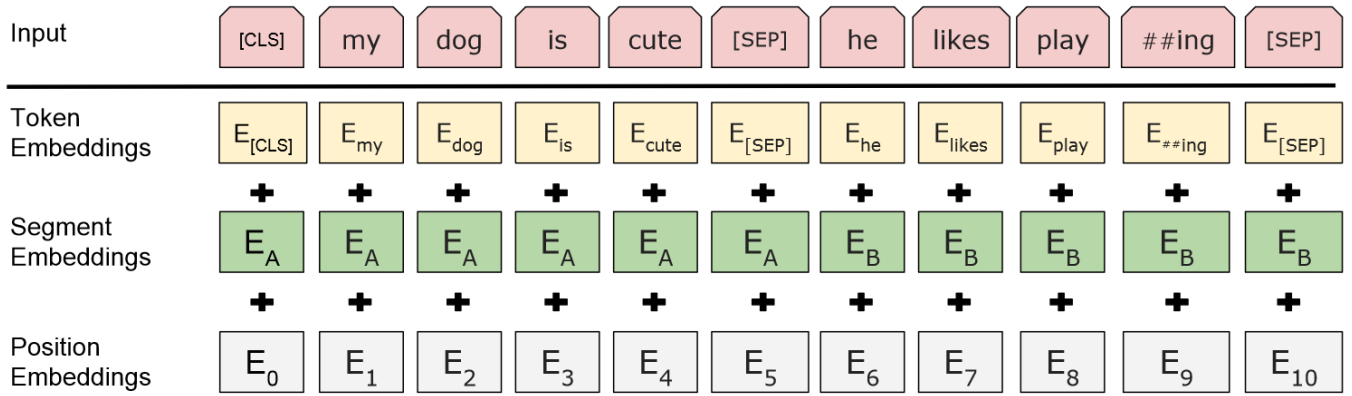
https://arxiv.org/pdf/1706.03762.pdf

Input Representation



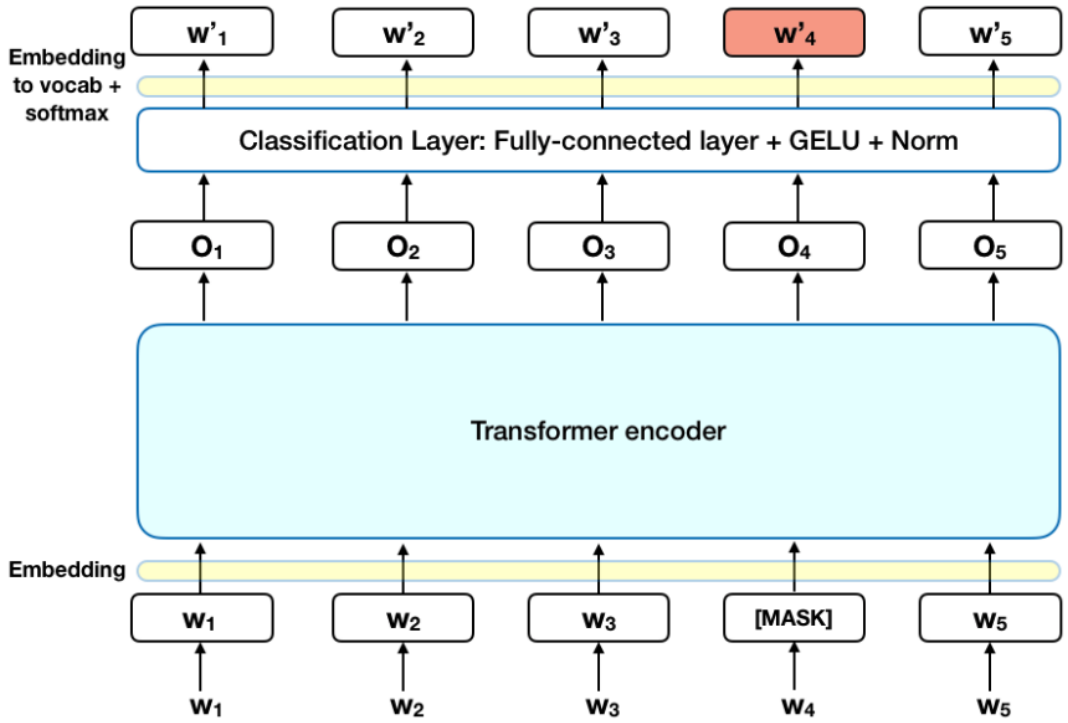
- Token Embeddings: 각 Token의 embedding을 나타낸다. Token은 WordPiece Model로 나눈다.
  - WordPiece Model: 통계학적 방법으로, 단어의 일부분 중 자주 나타나는 부분을 위주로 token을 분리. 예: 공부한다 → 공부 + #한다
  - Subword를 추출하는 방식이라서 Out-Of-Vocabulary가 발생하지 않는다.
- Segment Embeddings: 첫 번째/두 번째 문장 여부를 나타내는 임베딩.
- Position Embeddings: token의 위치를 embedding. Transformer와 동일한 embedding 사용.

Input Representation



- Input의 시작은 "[CLS]" Token이다.
  - 이 Token의 output은 input의 전반적인 의미를 embedding하게 된다.
  - classifier를 만들 때, 이 [CLS] token의 output을 embedding으로 보아 간단한 classifier를 붙이면 된다.
  - GPT와 달리, pretrain/finetuning 과정에서 모두 사용
- 두 문장 사이의 구분은 [SEP] Token이다.
  - BERT의 pre-train/fine-tuning시 두 문장을 Input으로 주어야 하는 경우가 있다.
  - 이 때 Token을 구분하기 위해 [SEP] Token을 사용한다.

Pre-train #1: Masked Language Model



https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270

- 사람은 문장의 단어 중 일부가 없어도, 나올 단어를 예측할 수 있다.
- Input 중 [MASK]로 바뀐 Token이 어떤 Token이었는지 예측하는 Task
- [MASK]로 바뀐 Token만 예측한다.
- 주변의 context만 보고, 단어를 예측할 수 있도록 Transformer의 encoder를 학습한다.
- 기존의 Language Model과 달리, 앞/뒤 context를 모두 보는 것이 특징

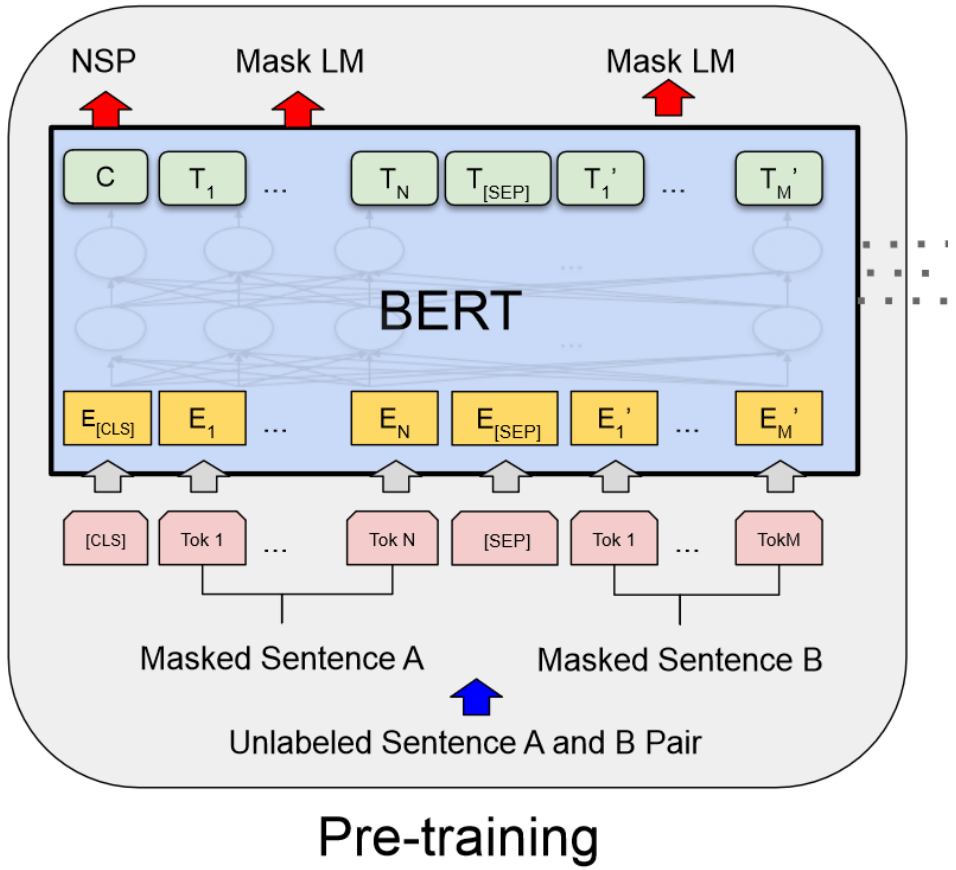
Pre-train #1: Masked Language Model

Masking Rates			Dev Set Results		
MASK	SAME	RND	MNLI	NER	
			Fine-tune	Fine-tune	Feature-based
80%	10%	10%	84.2	95.4	94.9
100%	0%	0%	84.3	94.9	94.0
80%	0%	20%	84.1	95.2	94.6
80%	20%	0%	84.4	95.2	94.7
0%	20%	80%	83.7	94.8	94.6
0%	0%	100%	83.6	94.9	94.6

Table 8: Ablation over different masking strategies.

- BERT로 Masked LM을 학습할 때 15%의 Token을 선택하여 아래와 같은 처리를 한다.
- 80%는 [MASK] token으로 바꾼다.
  - 예: my dog is hairy → my dog is [MASK]
- 10%는 전혀 임의의 단어로 바꾼다.
  - my dog is hairy → my dog is apple
- 10% 정도는 변화 없이 둔다.
  - My dog is hairy → my dog is hairy
- 이 비율은 좌측의 실험 결과에 따라 결정되었다.

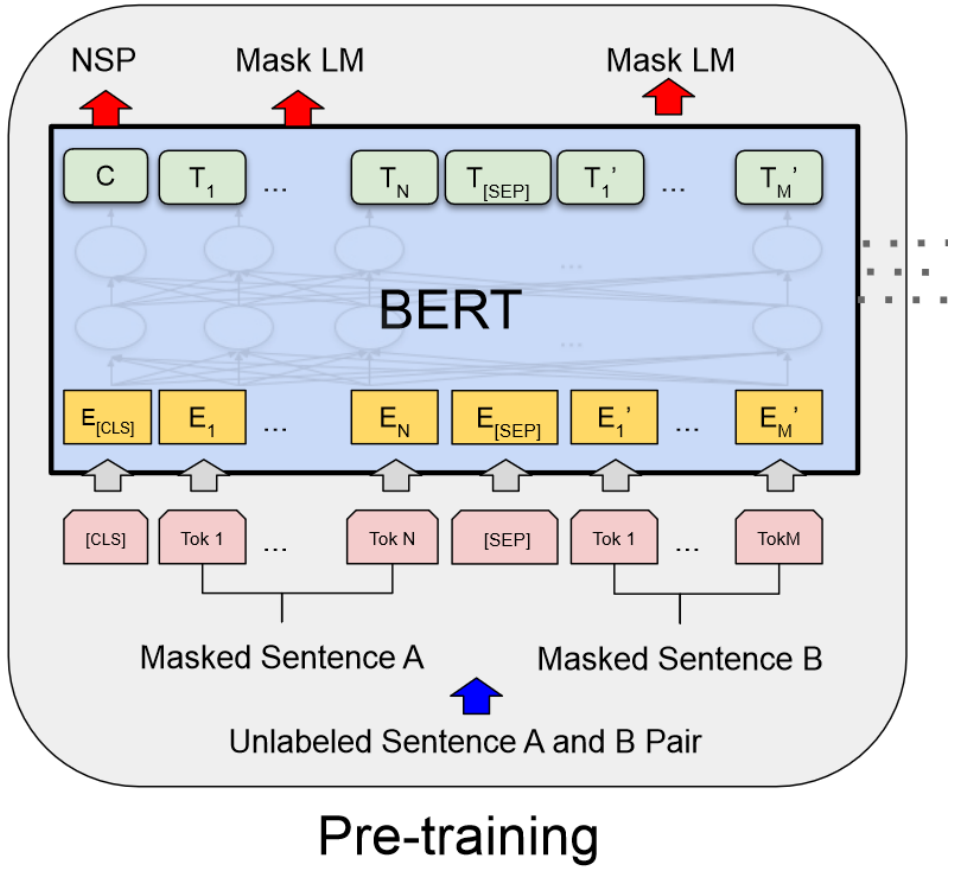
Pre-train #2: Next Sentence prediction



Pre-training

- 두 문장이 문맥상 이어지는 문장인지 아닌지 예측하는 Task
- IsNext: dataset에 있는 문장을 순서대로 뽑은 case
- NotNext: 두 문장을 랜덤하게 뽑은 case
- Pre-train Task 1과 마찬가지로, 일부 token을 [MASK]로 대체한다.

Pre-train #2: Next Sentence prediction



Pre-training

- IsNext 예제
  - Input=[CLS] the man went to [MASK] store [SEP] he bought a gallon [MASK] milk [SEP]
- NotNext 예제
  - Input=[CLS] the man [MASK] to the store [SEP] penguin [MASK] are flight ##less birds [SEP]
- IsNext 50%, NotNext 50%로 학습셋을 구성하였다.

Fine-tuning

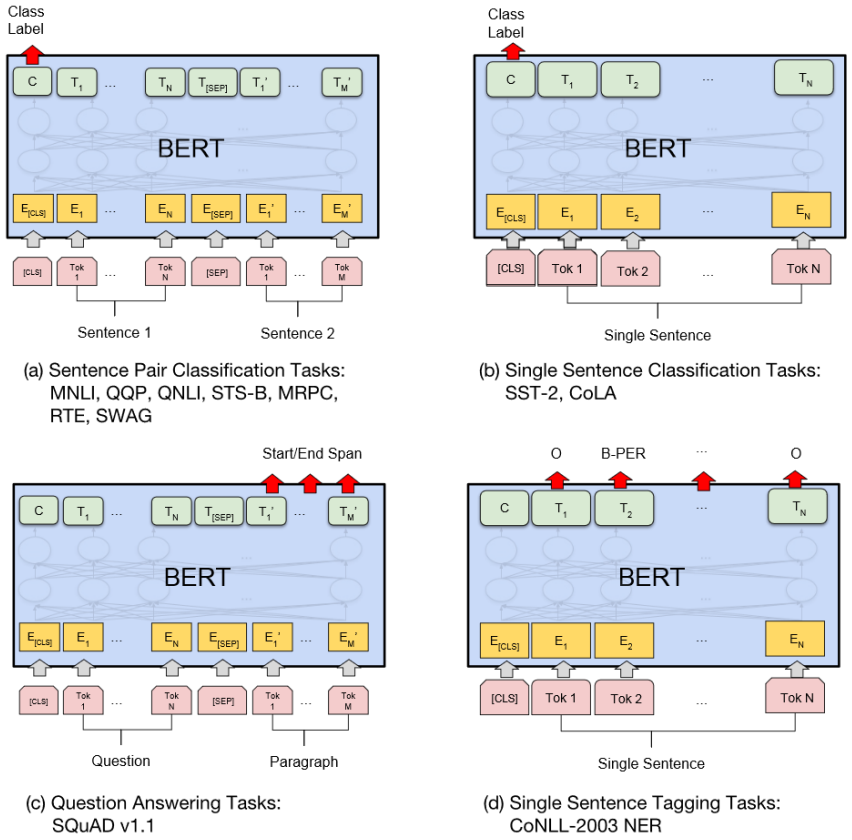


Figure 4: Illustrations of Fine-tuning BERT on Different Tasks.

- 기본적으로 Transformer Encoder의 output 값을 활용한다.
- 좌측 그림의 화살표는 아래 수식을 의미한다.

$$P = softmax(CW^T)$$

(C: [CLS]의 output, W: class 개수만큼의 layer)

- 즉, output 값에 activation function이 softmax인 layer를 한 층 쌓으면 된다.
- Fine-tuning시 모델 전체의 파라미터를 학습한다. (Transformer encoder 부분의 parameter도 갱신됨)

Hyperparameter

Pre-training

- Sequence당 Token 수: 512
- Batch당 sequence 수: 256
- Step 수: 1,000,000 (약 40 epoch)
- Learning rate (Adam): 1e-4
- Activation function: gelu
- BERT_base : L=12, H=768, A=12, Total Parameters = 110M
  - OpenAI GPT와 동일한 파라미터수
- BERT_large : L=24, H=1024, A=16, Total Parameters = 340M

Fine-tuning

- Batch size: 16, 32
- Learning rate (Adam): 5e-5, 3e-5, 2e-5
- Epoch: 2, 3, 4

Results

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
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.

- 11개 Task에 대해서 SOTA이다.

Results

System	Dev		Test	
	EM	F1	EM	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**

Table 2: SQuAD 1.1 results. The BERT ensemble is 7x systems which use different pre-training checkpoints and fine-tuning seeds.

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				
$\text{BERT}_{\text{LARGE}}$ (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.

참고: GLUE benchmark

Dataset	Description	Data example	Metric
CoLA	Is the sentence grammatical or ungrammatical?	"This building is than that one." = **Ungrammatical**	Matthews
SST-2	Is the movie review positive, negative, or neutral?	"The movie is funny , smart , visually inventive , and most of all , alive ." = **.93056 (Very Positive)**	Accuracy
MRPC	Is the sentence B a paraphrase of sentence A?	A) "Yesterday , Taiwan reported 35 new infections , bringing the total number of cases to 418 ." B) "The island reported another 35 probable cases yesterday , taking its total to 418 ." = **A Paraphrase**	Accuracy / F1
STS-B	How similar are sentences A and B?	A) "Elephants are walking down a trail." B) "A herd of elephants are walking along a trail." = **4.6 (Very Similar)**	Pearson / Spearman
QQP	Are the two questions similar?	A) "How can I increase the speed of my internet connection while using a VPN?" B) "How can Internet speed be increased by hacking through DNS?" = **Not Similar**	Accuracy / F1
MNLI-mm	Does sentence A entail or contradict sentence B?	A) "Tourist Information offices can be very helpful." B) "Tourist Information offices are never of any help." = **Contradiction**	Accuracy
QNLI	Does sentence B contain the answer to the question in sentence A?	A) "What is essential for the mating of the elements that create radio waves?" B) "Antennas are required by any radio receiver or transmitter to couple its electrical connection to the electromagnetic field." = **Answerable**	Accuracy
RTE	Does sentence A entail sentence B?	A) "In 2003, Yunus brought the microcredit revolution to the streets of Bangladesh to support more than 50,000 beggars, whom the Grameen Bank respectfully calls Struggling Members." B) "Yunus supported more than 50,000 Struggling Members." = **Entailed**	Accuracy
WNLI	Sentence B replaces sentence A's ambiguous pronoun with one of the nouns - is this the correct noun?	A) "Lily spoke to Donna, breaking her concentration." B) "Lily spoke to Donna, breaking Lily's concentration." = **Incorrect Referent**	Accuracy

- 다양한 NLP Task들의 성능을 평가하는 benchmark

Differences in pre-training model

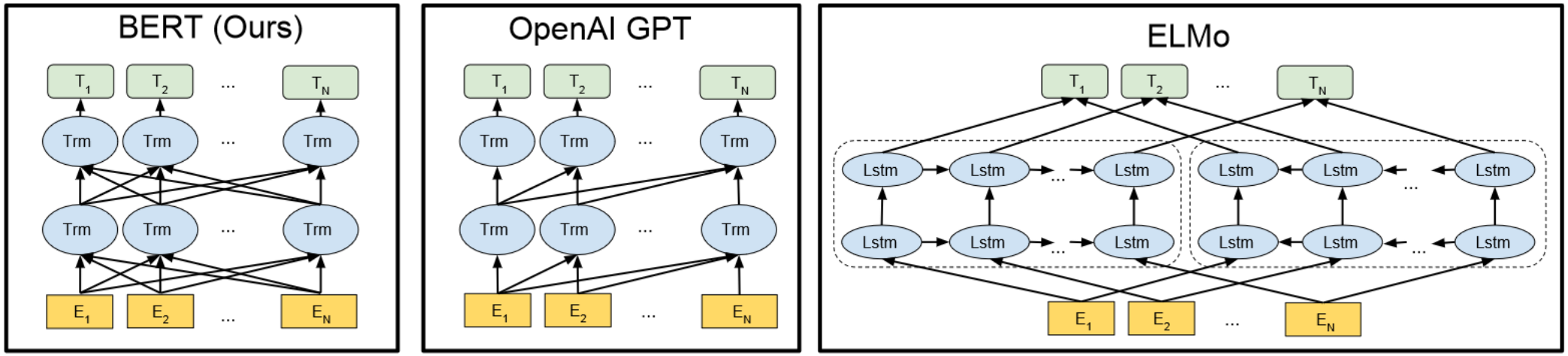
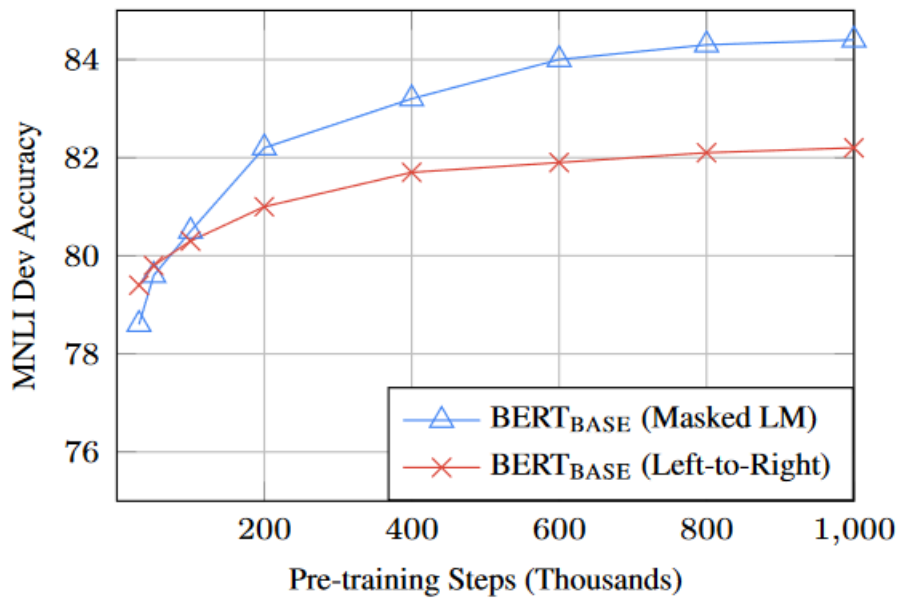


Figure 3: Differences in pre-training model architectures. BERT uses a bidirectional Transformer. OpenAI GPT uses a left-to-right Transformer. ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTMs to generate features for downstream tasks. Among the three, only BERT representations are jointly conditioned on both left and right context in all layers. In addition to the architecture differences, BERT and OpenAI GPT are fine-tuning approaches, while ELMo is a feature-based approach.

Pretrain



- Step 수가 많을수록 모델의 성능이 향상된다.
  - 500k step일 때보다 1,000k step일 때 1%의 accuracy를 추가로 향상시킬 수 있다.
- Step 수가 적을 땐 MLM 방식이 LTR 방식보다 성능이 떨어진다. 그러나 학습이 진행되면서 MLM 방식의 성능이 더 좋게 나온다.
  - Batch당 15%의 단어만 predict되므로

Pre-train Task 비교

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
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

Table 5: Ablation over the pre-training tasks using the $\text{BERT}_{\text{BASE}}$ 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.

- No NSP: MLM만 사용한 모델
- LTR & No NSP: 둘 다 사용 X. OpenAI GPT.
  - LTR: 왼쪽 Token만 보고 오른쪽 Token을 예측하는 모델
- 특히, NSP Task를 수행하지 않으면 QA/NLI Task에 대한 성능이 크게 하락한다.
- 또한, MLM까지 수행하지 않으면, 성능은 더더욱 하락하게 된다.
- 양쪽 문맥을 모두 확인하는 것이 성능 향상에 도움을 준다는 결론

모델 크기에 따른 성능 비교

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.

- 모델 크기가 클수록 성능이 향상되는 것을 확인할 수 있다.
- MRPC 데이터셋은 label된 데이터가 3,600건 밖에 없지만, 모델이 커질수록 성능 향상을 보였다.

Feature-based Approach with BERT

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.

- Feature-based approach:
  이미 pretrain된 BERT 모델에서 feature를 추출하여 NER Task 수행
- 추출된 feature에 softmax layer만 붙여서 가볍게 학습하여 실험 가능
- BERT_large의 성능이 가장 좋았다.
- 그러나, Transformer encoder 부분의 Top 4 hidden layer를 concat하여 사용하는 것만으로도 거의 비슷한 성능을 얻을 수 있었음
- 즉, BERT는 fine-tuning, feature-based approach 둘 다 효과적임을 나타낸다.

Summary

- Transformer의 encoder 부분을 활용한 Language representation 학습
- Masked Language Model(MLM)과 Next sentence prediction으로 pre-train, language representation 학습
- 양방향 문맥을 모두 볼 수 있는, 범용적으로 사용할 수 있는 모델을 학습해 둔 뒤, Transfer Learning으로 원하는 Task를 수행하도록 하는 방법 제시
- AlBERT, RoBERT, DocBERT등 BERT를 응용한 다양한 모델이 나와 있다.

reference

- https://arxiv.org/abs/1810.04805
- https://mino-park7.github.io/nlp/2018/12/12/bert-%EB%85%BC%EB%AC%B8%EC%A0%95%EB%A6%AC/
- http://docs.likejazz.com/bert/
- https://huffon.github.io/2019/11/16/glue/