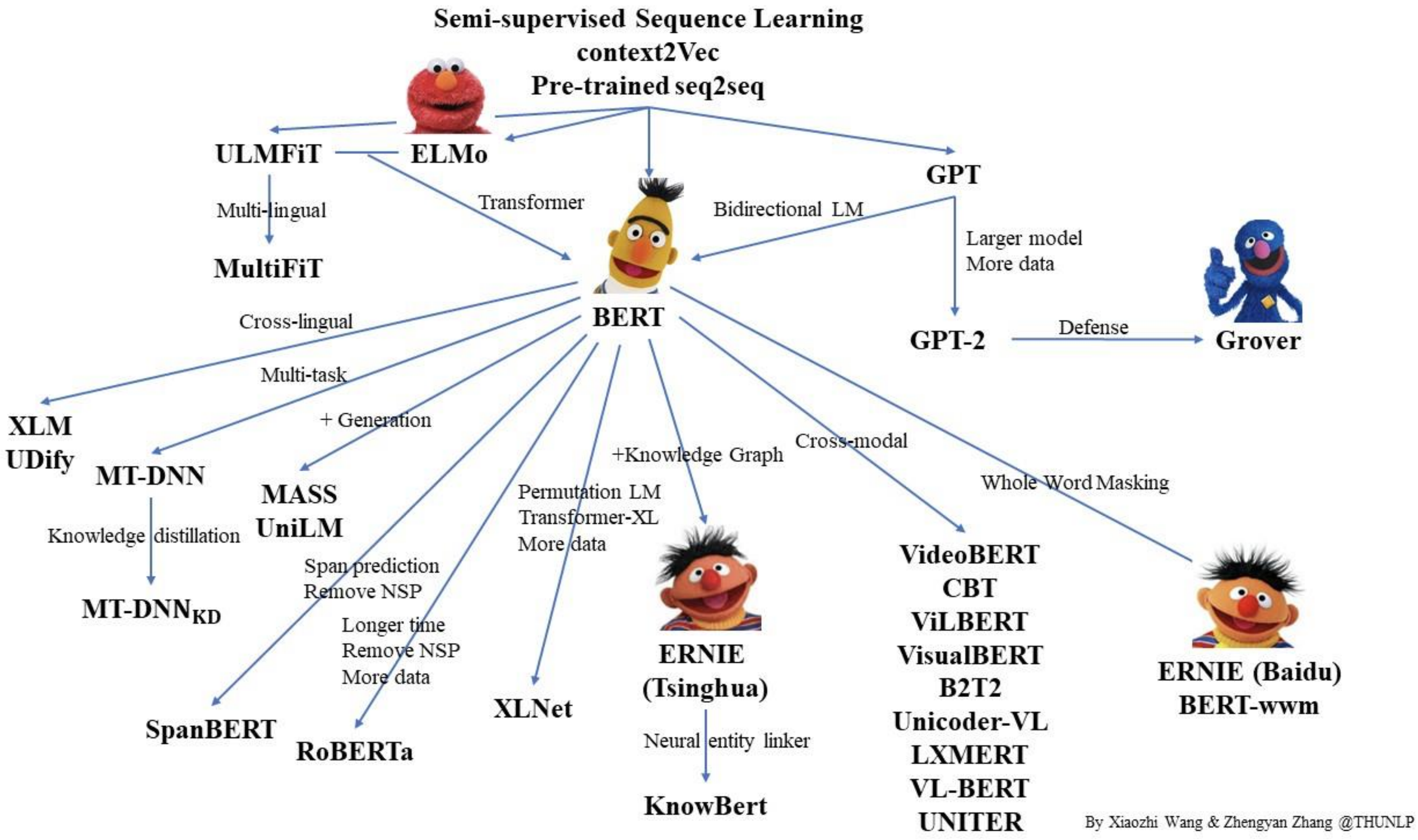
BERT

(Pre-training of Deep Bidirectional Transformers for Language Understanding)

자연어처리 텍스트마이닝


Semi-supervised Sequence Learning
context2Vec
Pre-trained seq2seq
ULMFiT
ELMo
GPT
Multi-lingual
Transformer
Bidirectional LM
MultiFiT
Larger model
More data
BERT
Cross-lingual
GPT-2
Defense
Grover
Multi-task
XLM
UDify
+ Generation
MT-DNN
MASS
UniLM
+Knowledge Graph
Cross-modal
Whole Word Masking
Permutation LM
Transformer-XL
More data
Knowledge distillation
Span prediction
Remove NSP
MT-DNN$_{KD}$
Longer time
Remove NSP
More data
ERNIE (Tsinghua)
VideoBERT
CBT
ViLBERT
VisualBERT
B2T2
Unicoder-VL
LXMERT
VL-BERT
UNITER
ERNIE (Baidu)
BERT-wwm
SpanBERT
RoBERTa
XLNet
Neural entity linker
KnowBert
By Xiaozhi Wang & Zhengyan Zhang @THUNLP

BERT

BERT (Pre-training of Deep Bidirectional Transformers for Language Understanding)

BERT Overview (1)

Pre-training of Deep Bidirectional Transformers for Language Understanding

Pre-trained model + Fine tuning	Multi layer	Bidirectional Transformer (Encoder)

BERT Overview (2)

- BERT : Bidirectional Encoder Representations form Transformer
- "Attention is all you need(Vaswani et al., 2017)"에서 소개한 Transformer 활용 Language Representation
- wiki나 book data 대용량 **unlabeled data로 모델을 미리 학습** 시킨 후, 특정 task를 가지고 있는 labeled data로 transfer learning을 하는 모델
- **Fine-tuning**을 통해 task의 state-of-the-art를 달성

SQuAD1.1 Leaderboard

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar et al. '16)	82.304	91.221
1 Oct 05, 2018	BERT (ensemble) *Google AI Language* https://arxiv.org/abs/1810.04805	**87.433**	**93.160**
2 Sep 09, 2018	nlnet (ensemble) *Microsoft Research Asia*	85.356	91.202
3 Jul 11, 2018	QANet (ensemble) *Google Brain & CMU*	84.454	90.490

https://arxiv.org/abs/1810.04805

BERT vs OpenAI GPT vs ELMo

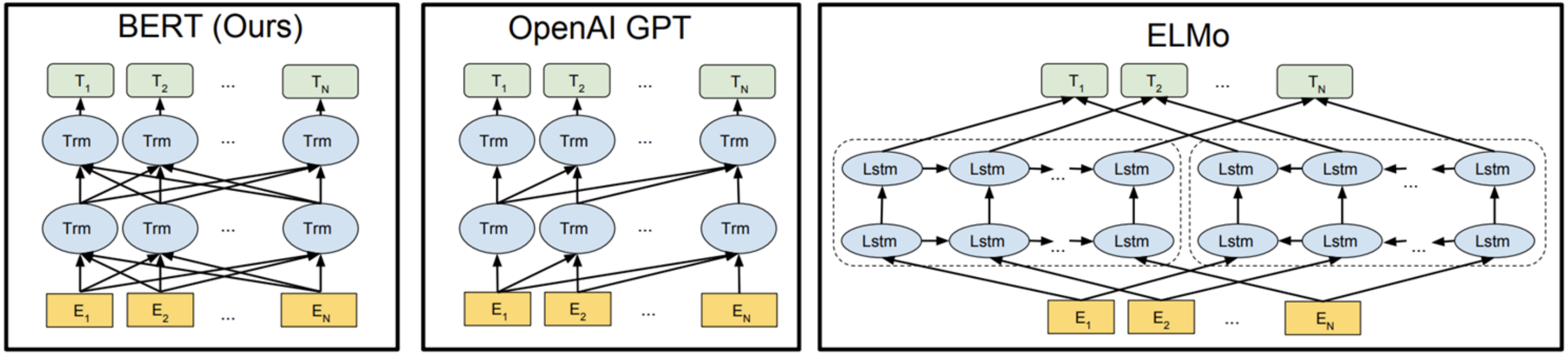


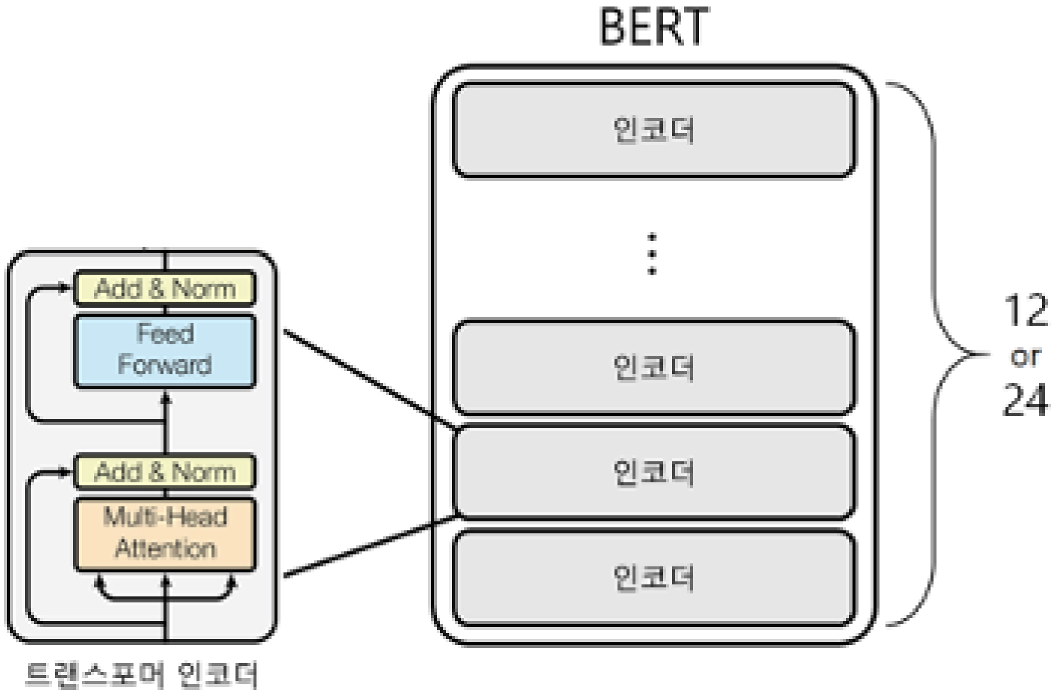
Figure 3: Differences in pre-training model architectures. BERT uses a bidirectional Transformer. OpenAI GPT uses a left-to-right Transformer. ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTMs to generate features for downstream tasks. Among the three, only BERT 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.

Evaluation

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.

BERT 의 구조

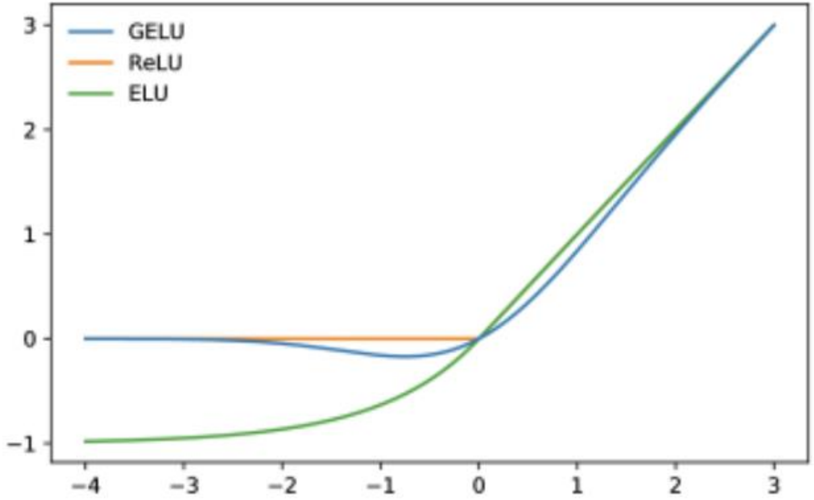

BERT
인코더
인코더
인코더
인코더
12
or
24
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
트랜스포머 인코더

BERT 의 구조

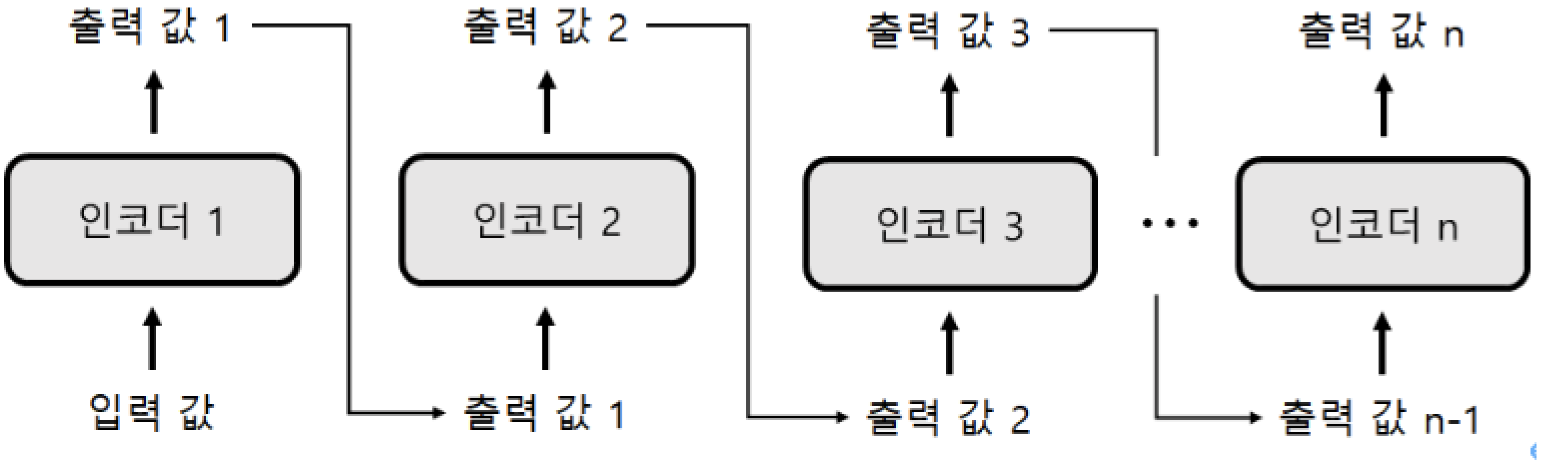
- position wise feed-forward network

$$FFN(x) = max(0, x \cdot W_1 + b_1)W_2 + b_2$$

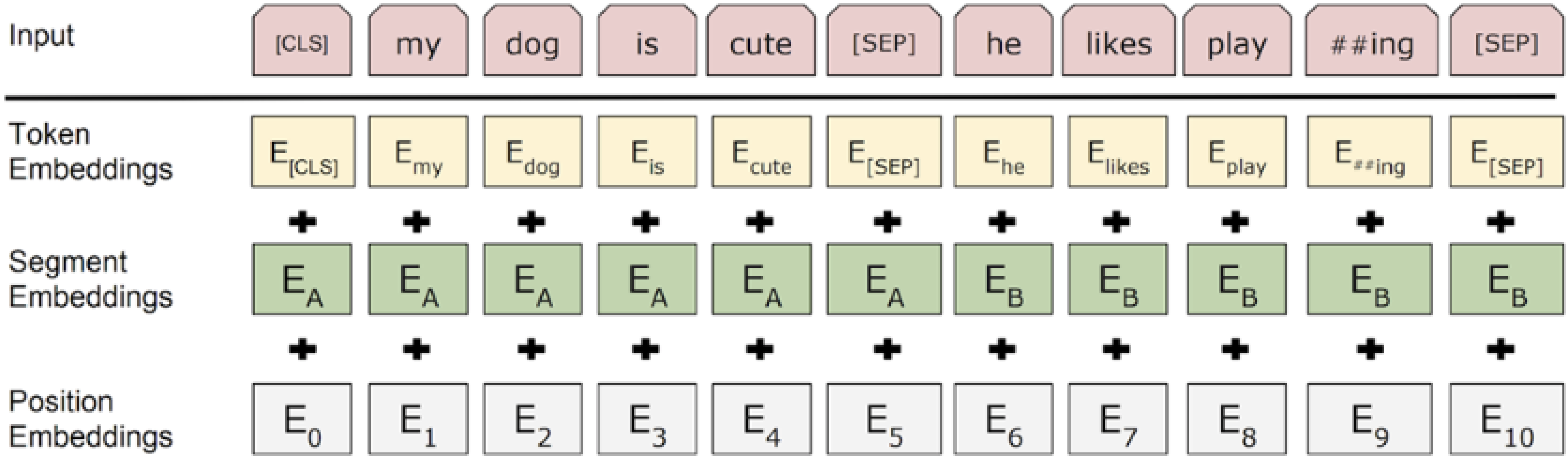
- GELU



BERT 의 구조



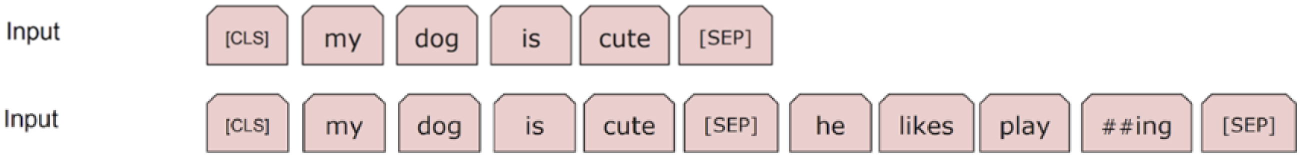
BERT 의 입력값


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}

특수 토큰

- [CLS] 토큰 (Classification Token)
  - 입력 값 제일 앞에 붙는 토큰이며, 분류 과제(Classification Task) 수행용

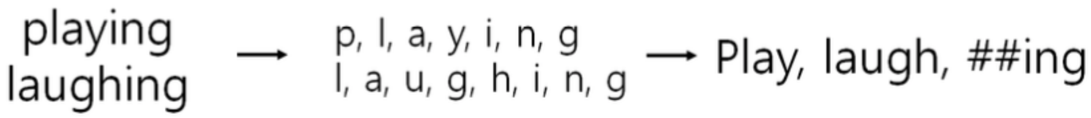
- [SEP] 토큰 (Separate Token)
  - [SEP] 토큰은 문장 구분용



- [PAD] 토큰
  - 입력 시퀀스의 길이를 맞춰주기 위해 사용하는 토큰

임베딩

- 토큰 임베딩(Token Embeddings)
  - 워드피스(WordPiece) 임베딩을 사용, 워드피스 임베딩은 BPE알고리즘에 기반



- 세그먼트 임베딩(Segment Embeddings
  - 각 문장을 구분용

- 포지션 임베딩(Position Embeddings)
  - 입력 값의 위치 정보

BPE(Byte Pair Encoding) 알고리즘

- BPE(Byte pair encoding) 알고리즘은 1994년에 제안된 데이터 압축 알고리즘
- 자연어 처리의 단어 분리 알고리즘으로 응용

문장	a a a b d a a a b a c
bigram	aa aa ab bd da aa aa ab ba ac

Frequency

aa	4
ab	2
bd	1
da	1
ba	1
ac	1

Word Piece Model (1)

- BPE(Byte pair encoding) 알고리즘을 tokenizer에 적용

문장	자연어 덮밥, 자연어 처리, 연어 덮밥

bigram

자	##연	##어	덮	##밥
자	##연	##어	처	##리
연	##어	덮	##밥	

자연	##연어	덮밥
자연	##연어	처리
연어	덮밥	

Frequency

자연	2
##연어	2
덮밥	2
연어	1
처리	1

=> 처음으로 등장한 최빈토큰을 선택

Word Piece Model(2)

- BPE(Byte pair encoding) 알고리즘을 tokenizer에 적용

문장 자연어 덮밥, 자연어 처리, 연어 덮밥

bigram

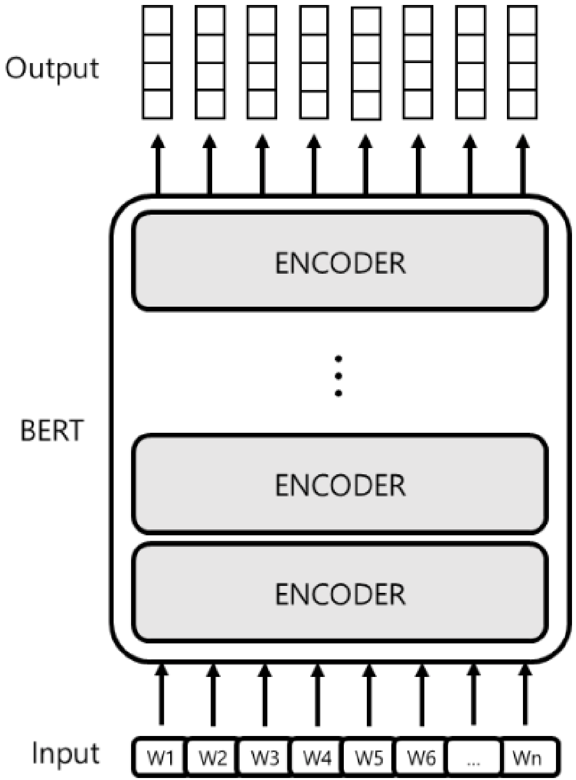
자연	##어	덮	##밥	자연어	덮밥
자연	##어	처	##리	자연어	처리
연	##어	덮	##밥	연어	덮밥

Frequency

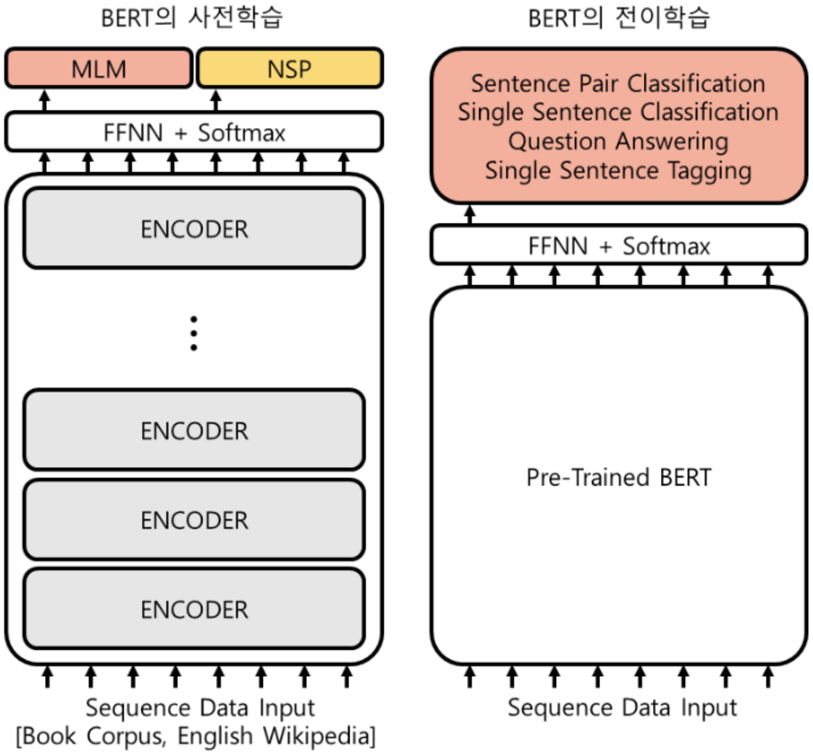
자연어	2
덮밥	2
처리	1
연어	1

=> 처음으로 등장한 최빈토큰을 선택

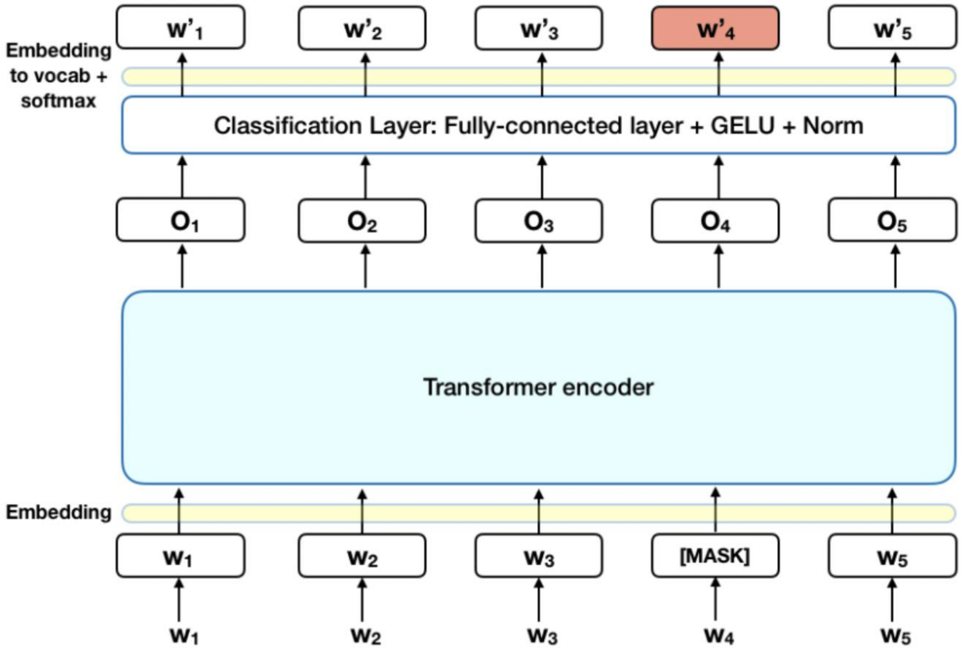
BERT 의 출력값


Output
BERT
ENCODER
ENCODER
ENCODER
Input
W1
W2
W3
W4
W5
W6
...
Wn

BERT 의 두가지 학습 구조



BERT 의 사전 학습 – MLM(Masked Language Mode)



BERT 의 사전 학습 – MLM

1. **80%의 단어는 [MASK] 토큰으로 바꾸고, BERT는 [MASK] 토큰에 들어갈 단어를 예측한다.**

   "My dog is [MASK]" → hairy

2. **10%의 단어는 무작위 단어로 바꾸고, BERT는 해당 위치에 들어갈 정답 단어를 예측한다.**

   "My dog is apple" → hairy

3. **10%의 단어는 그대로 두고, BERT는 해당 위치에 들어갈 정답 단어를 예측한다.**

   "My dog is hairy" → hairy

➡ pre training과 fine tuning 과의 불일치 문제 해결

BERT 의 사전 학습 – MLM

Masking Rates			Dev Set Results		
MASK	SAME	RND	MNLI Fine-tune	NER Fine-tune	NER 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

BERT 의 사전 학습 – NSP(Next Sentence Prediction)

- Input : `[CLS] the man went to [MASK] store [SEP] he bought a gallon [MASK] milk [SEP]`

  label : `IsNext`

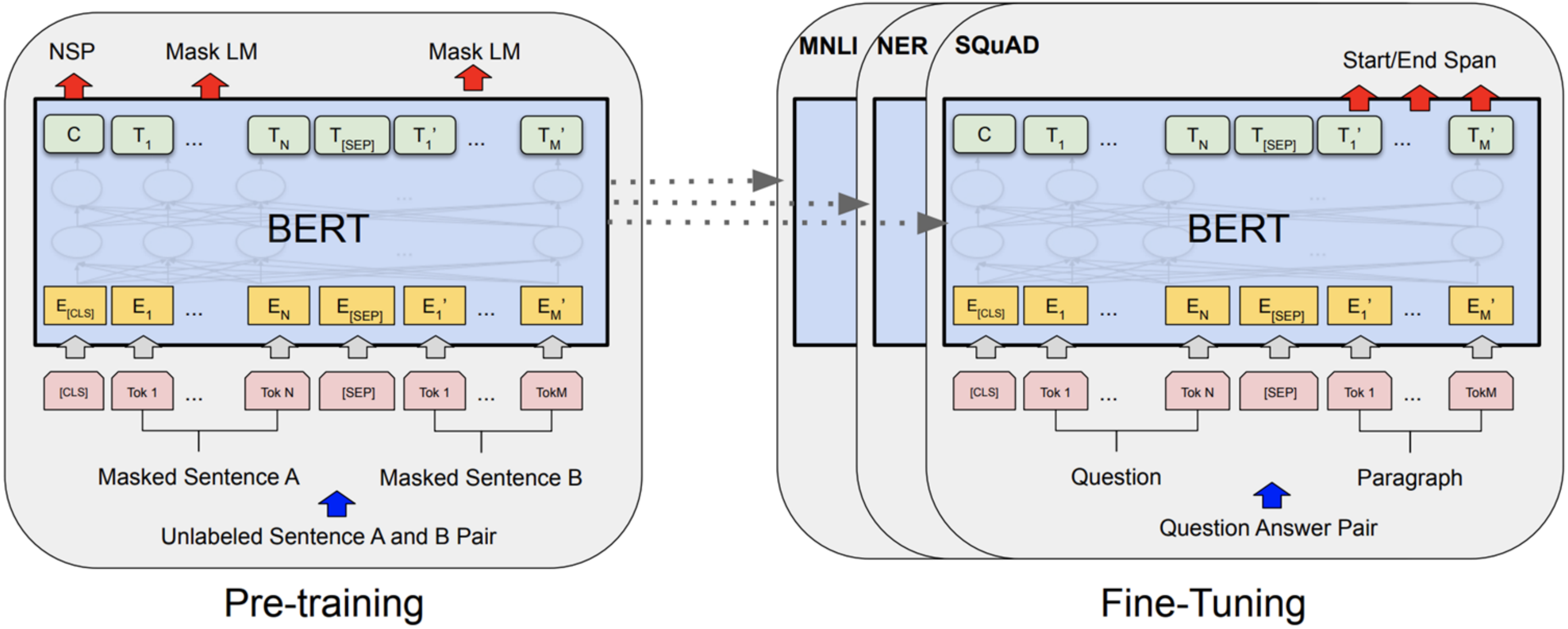
- Input : `[CLS] the man [MASK] to the store [SEP] penguin [MASK] are flight ##less birds [SEP]`

  label : `NotNext`

BERT 의 사전 학습 – NSP(Next Sentence Prediction)

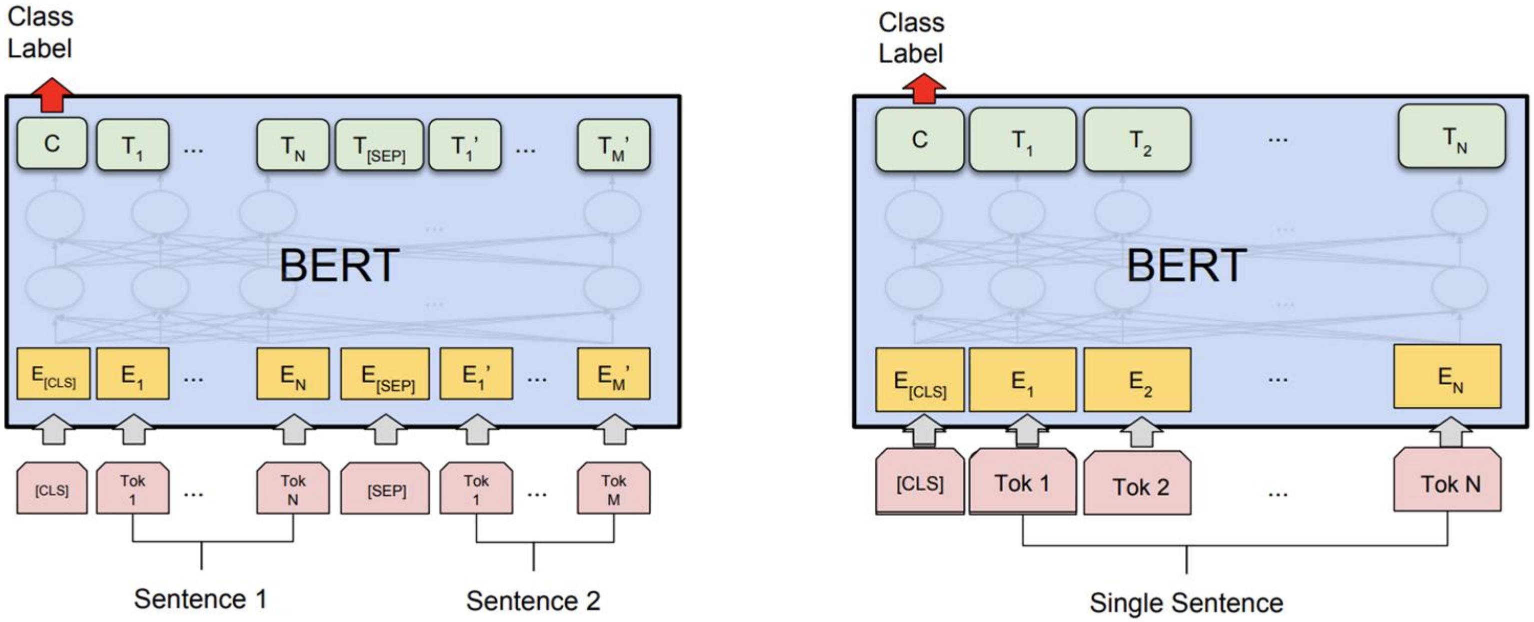
Tasks	MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
	Dev Set				
$BERT_{BASE}$	84.4	88.4	86.7	92.7	88.5
No NSP	83.9	84.9	86.5	92.6	87.9
LTR & No NSP	82.1	84.3	77.5	92.1	77.8
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

Fine-Tuning (1)



Pre-trained model를 기반으로 Fine-Tuning 하여 task 해결

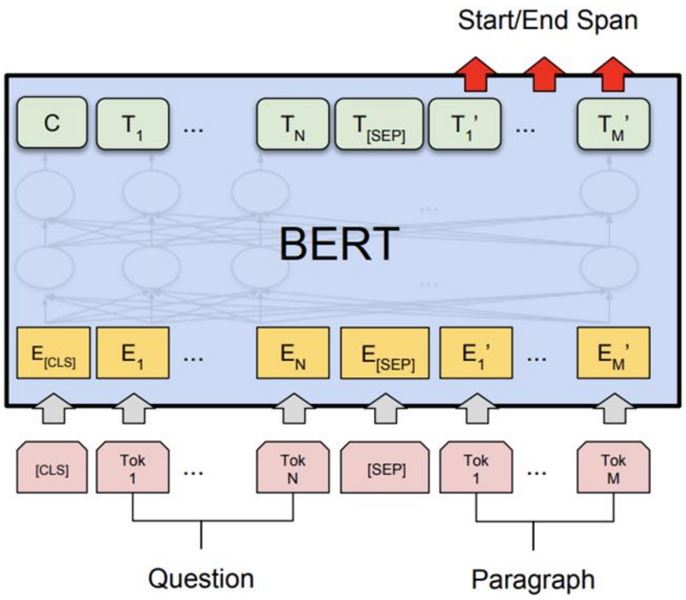
Fine-Tuning – Sequence Level



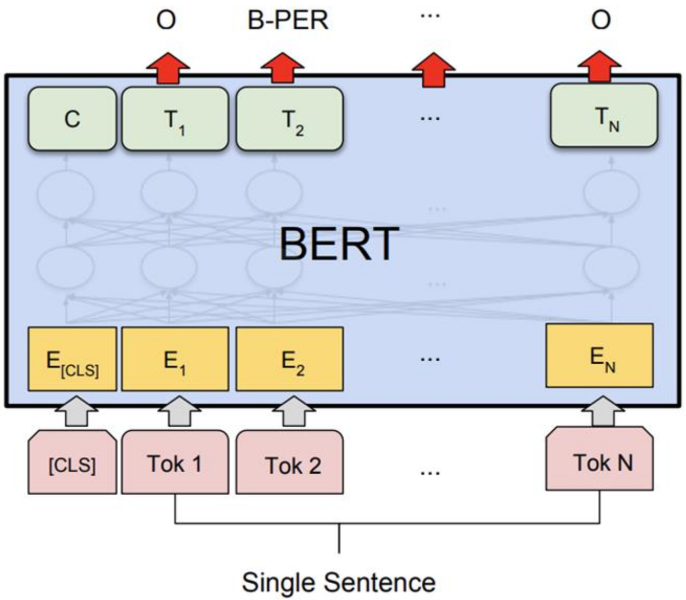
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

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

Fine-Tuning – Token Level



(c) Question Answering Tasks:
SQuAD v1.1



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

Experiments

model 사이즈 비교

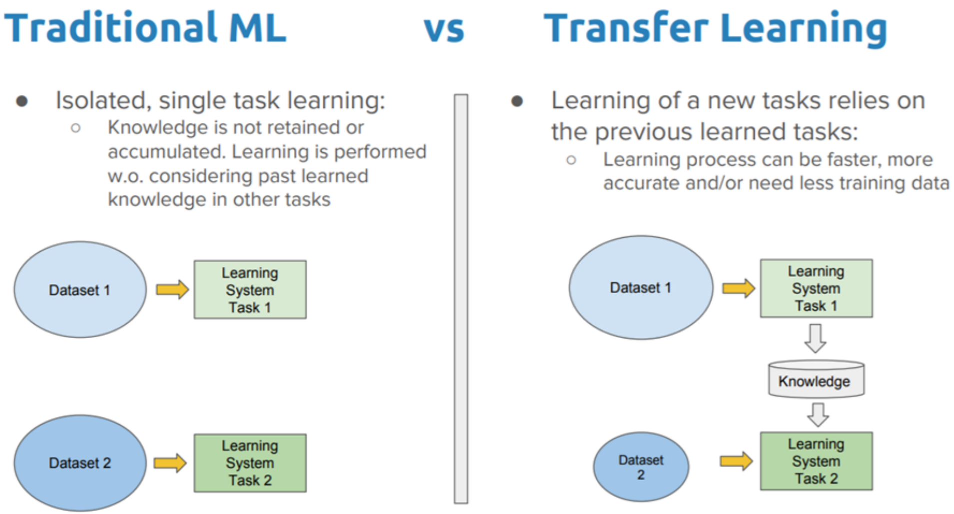
Hyperparams				Dev Set Accuracy		
#L	#H	#A	LM (ppl)	MNLI-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

Feature based approach

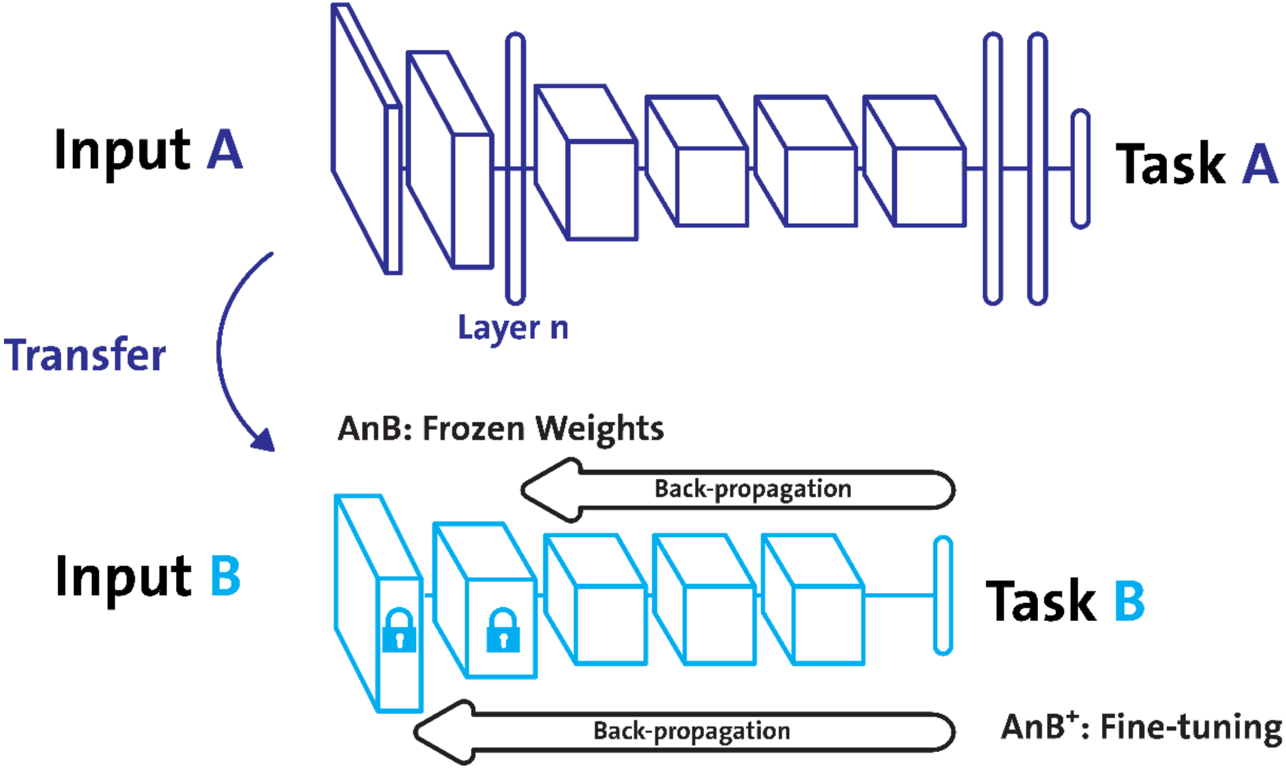
System	Dev F1	Test F1
ELMo (Peters et al., 2018a)	95.7	92.2
CVT (Clark et al., 2018)	-	92.6
CSE (Akbik et al., 2018)	-	**93.1**
Fine-tuning approach		
$\text{BERT}_{\text{LARGE}}$	96.6	92.8
$\text{BERT}_{\text{BASE}}$	96.4	92.4
Feature-based approach ($\text{BERT}_{\text{BASE}}$)		
Embeddings	91.0	-
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	-

Transfer Learning

- Transfer learning : 전이 학습은 특정 환경에서 만들어진 AI 알고리즘을 다른 비슷한 분야에 적용
- Fine-Tuning : 사전에 학습된 모델의 파라미터를 task에 맞추어 정교하게 조정하여 활용



Transfer Learning



Transfer Learning