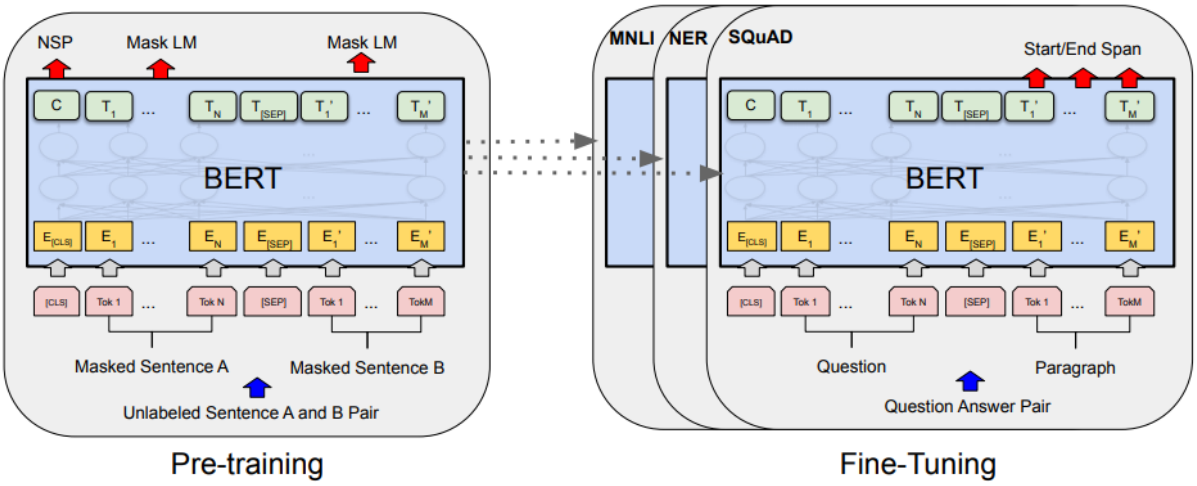
BERT 내용정리



- BERT는 절대 위치 임베딩을 사용하는 모델이므로 일반적으로 왼쪽보다는 오른쪽의 입력을 채우는 것이 좋습니다.
- BERT는 마스크 언어 모델링(MLM) 및 다음 문장 예측(NSP) 목표로 훈련되었습니다. 마스크 토큰을 예측하고 일반적으로 NLU에서 효율적이지만 텍스트 생성에는 최적이 아닙니다.
- 무작위 마스킹을 사용하여 입력을 손상시킵니다. 더 정확히 말하면 사전 학습 중에 지정된 비율의 토큰(일반적으로 15%)이 다음과 같이 마스킹됩니다.
    - 확률 0.8의 특수 마스크 토큰
    - 확률 0.1로 마스크된 것과 다른 랜덤 토큰
    - 확률 0.1의 동일한 토큰
- 모델은 원래 문장을 예측해야 하지만 두 번째 목적이 있습니다. 입력은 두 문장 A와 B(그 사이에 분리 토큰 있음)입니다. 확률 50%로 문장은 코퍼스에서 연속적이고 나머지 50%에서는 관련이 없습니다. 모델은 문장이 연속적인지 여부를 예측해야 합니다.

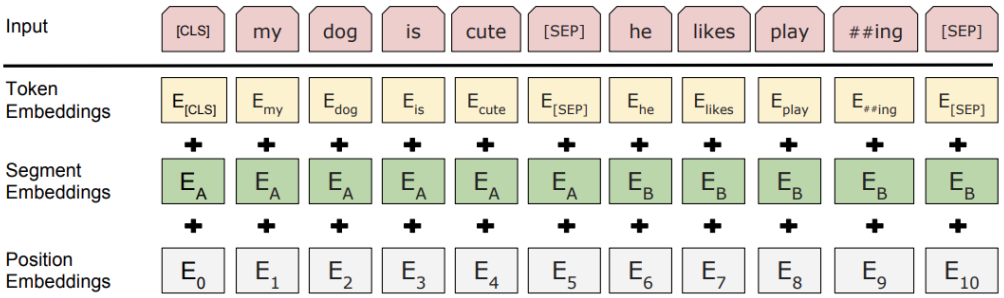


Figure 2: BERT input representation. The input embeddings is the sum of the token embeddings, the segmentation embeddings and the position embeddings.

<Embedding방법 3가지>

- Token Embedding: Word Piece 임베딩 방식 사용, 각 Char(문자) 단위로 임베딩을 하고, 자주 등장하면서 가장 긴 길이의 sub-word를 하나의 단위로 만듭니다. 자주 등장하지 않는 단어는 다시 sub-word로 만듭니다. 이는 이전에 자주 등장하지 않았던 단어를 모조리 'OOV'처리하여 모델링의 성능을 저하했던 'OOV'문제도 해결 할 수 있습니다.
- Segment Embedding: Sentence Embedding, 토큰 시킨 단어들을 다시 하나의 문장으로 만드는 작업입니다. BERT에서는 두개의 문장을 구분자([SEP])를 넣어 구분하고 그 두 문장을 하나의 Segment로 지정하여 입력합니다. BERT에서는 이 한 세그먼트를 512 sub-word 길이로 제한하는데, 한국어는 보통 20 sub-word가 한 문장을 이룬다고 하며 대부분의 문장은 60 sub-word가 넘지 않는다고 하니 BERT를 사용할 때, 하나의 세그먼트에 128로 제한하여도 충분히 학습이 가능하다고 합니다.
- Position Embedding: BERT의 저자는 이전에 Transformer 모델을 발표하였는데, Transformer란 CNN, RNN 과 같은 모델 대신 Self-Attention 이라는 모델을 사용하는 모델입니다. BERT는 Transformer의 인코더, 디코더 중 인코더만 사용합니다. Transformer Self Attention은 입력의 위치를 고려하지 않고 입력 토큰의 위치 정보를 고려합니다. 그래서 Transformer모델에서는 Sinusoid 함수를 이용하여 Positional encoding을 사용하고 BERT는 이를 따서 Position Encoding을 사용합니다.

BERT는 위 세가지 임베딩을 합치고 이에 Layer정규화와 Dropout을 적용하여 입력으로 사용합니다.

<Pre Trainning 기법>

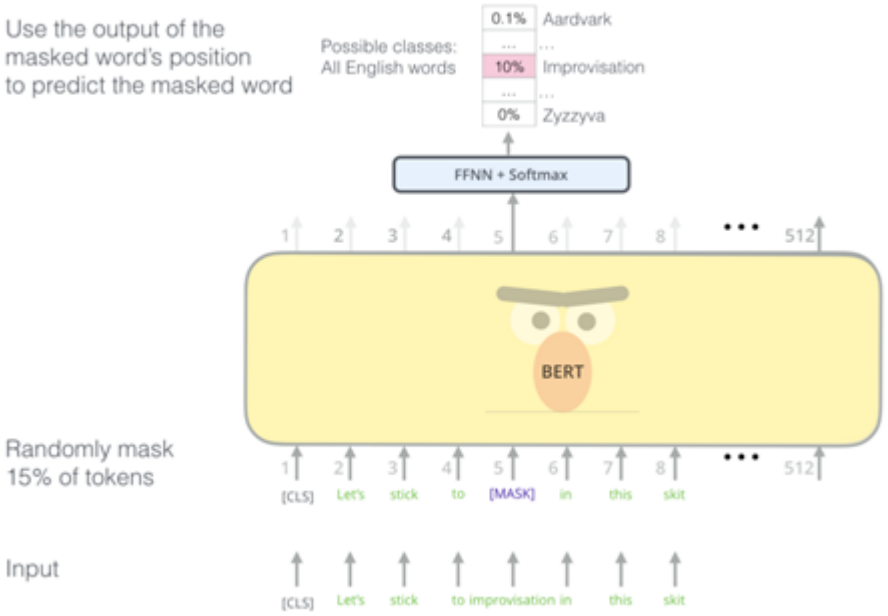
데이터들을 임베딩하여 훈련시킬 데이터를 모두 인코딩 하였으면, pre_trainning에 들어가게 됩니다. 기존의 방법들은 보통 문장을 왼쪽에서 오른쪽으로 학습하여 다음 단어를 예측

하는 방식이거나, 예측할 단어의 좌우 문맥을 고려하여 예측하는 방식을 사용합니다.
특히나 BERT는 언어의 특성을 잘 학습하도록

- MLM(Masked Language Model)
- NSP(Next Sentence Prediction)

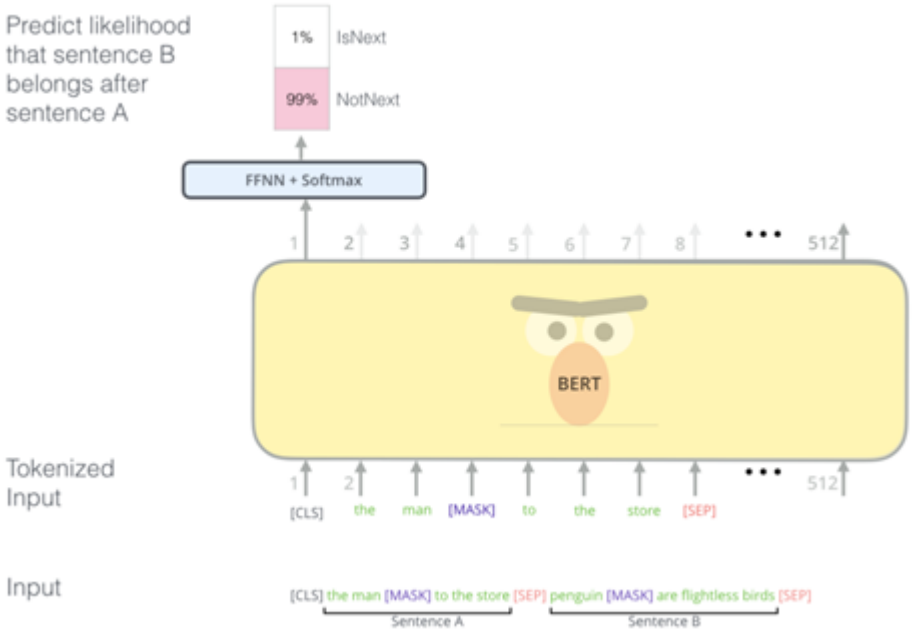
위 두가지 방식을 사용합니다.

MLM(Masked Language Model_ 마스킹된 언어 모델링)



: 입력 문장에서 임의로 토큰을 버리고(Mask), 그 토큰을 맞추는 방식으로 학습을 진행합니다.

NSP(Next Sentence Prediction_다음 문장 예측)

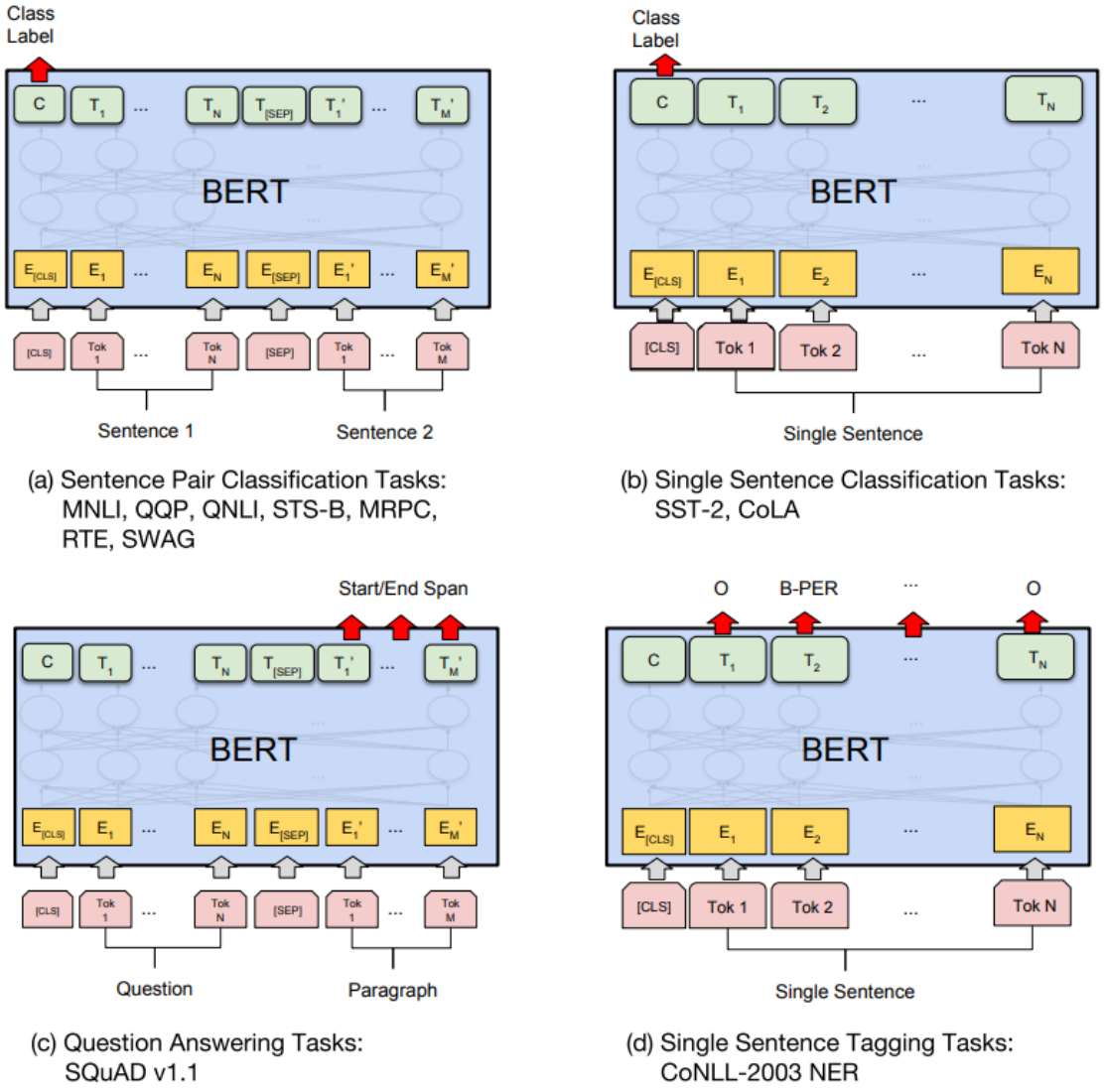


: 두 문장이 주어졌을 때, 두 문장의 순서를 예측하는 방식입니다. 두 문장 간 관련이 고려되야 하는 NLI와 QA의 파인 튜닝을 위해 두 문장의 연관을 맞추는 학습을 진행합니다.

- **MLM**은 BERT가 단일 문장 내에서 마스킹된 단어를 예측하여 문장 내 단어들의 문맥적 의미를 이해하도록 돕습니다.
- **NSP**는 BERT가 두 문장 간의 관계를 이해하게 하여 문장 사이의 논리적 흐름을 학습합니다.

<Transfer Learning>

: 학습된 언어모델을 전이학습시켜 실제 NLP Task를 수행하는 과정입니다. 실질적으로 성능이 관찰되는 부분이기도 합니다.



(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

이 그림은 BERT(Bidirectional Encoder Representations from Transformers)가 다양한 자연어 처리(NLP) 작업에서 어떻게 전이 학습을 사용하는지를 시각적으로 설명합니다.

(a) Sentence Pair Classification Tasks (문장 쌍 분류 작업):

- **작업 예시**: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG
- **설명**: 두 문장을 입력으로 받아서 두 문장 간의 관계(예: 논리적 일치 여부, 유사성 등)를 예측하는 작업입니다.
    - 두 개의 문장(Sentence 1, Sentence 2)을 각각 [CLS] 토큰과 [SEP] 토큰으로 구분하고, BERT는 이 두 문장에 대한 정보를 추출합니다.
    - [CLS] 토큰에서 최종 출력된 벡터는 문장 쌍에 대한 최종 분류(classification)를 위한 벡터로 사용됩니다.

(b) Single Sentence Classification Tasks (단일 문장 분류 작업):

- **작업 예시**: SST-2, CoLA
- **설명**: 하나의 문장을 입력으로 받아 문장의 성격을 분류하는 작업입니다.
    - 단일 문장이 [CLS] 토큰과 함께 BERT에 입력됩니다.
    - [CLS] 토큰에서 출력된 벡터는 문장의 범주를 예측하는 데 사용됩니다. 예를 들어, 문장의 감정(긍정/부정)을 예측하는 작업이 이에 해당합니다.

(c) Question Answering Tasks (질문 답변 작업):

- **작업 예시**: SQuAD v1.1
- **설명**: 주어진 문단에서 질문에 대한 답을 찾아내는 작업입니다.
    - 질문과 문단이 [CLS](시작토큰)와 [SEP](끝토큰) 토큰으로 구분된 상태로 BERT에 입력됩니다.
    - BERT는 문단 내에서 질문에 대한 답변의 시작점과 끝점을 예측합니다. 이를 통해 문단 내에서 답변을 정확히 찾아낼 수 있습니다.

(d) Single Sentence Tagging Tasks (단일 문장 태깅 작업):

- **작업 예시**: CoNLL-2003 NER
- **설명**: 하나의 문장에서 각 단어에 대해 태그를 예측하는 작업입니다. 이는 보통 개체명 인식(NER) 같은 작업에 사용됩니다.
    - 문장 내 각 토큰이 BERT에 입력되며, 각 토큰에 대한 태그(예: 인물(B-PER), 장소(O), 조직 등)를 예측합니다.
    - 각 토큰에 대한 벡터 출력이 토큰의 태그를 결정하는 데 사용됩니다.

이 네 가지 방식 모두 BERT의 출력층에서 [CLS] 또는 각각의 토큰 벡터를 사용하여 다양한 자연어 처리 작업을 수행합니다. BERT는 문장의 의미를 양방향으로 이해하기 때문에 이러한 작업에서 뛰어난 성능을 발휘합니다.