# BERT

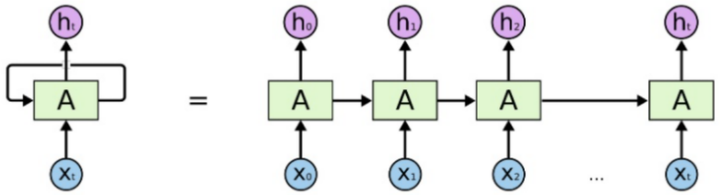
산업시스템공학과 하지수

# 자연어처리 발전흐름 - 한계를 위주로

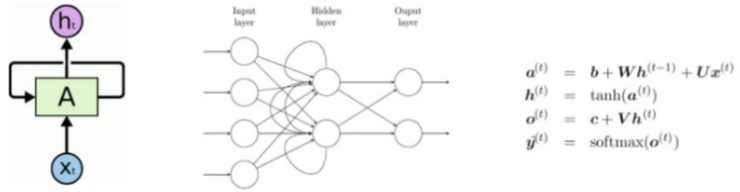
기존 신경망의 경우 연속적인 시퀀스를 처리하기 어려움

## 01. RNN (RECURRENT NEURAL NETWORK)

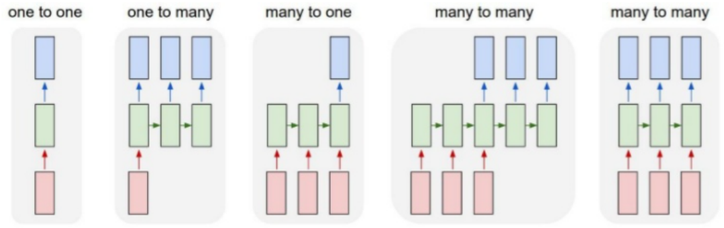
RNN은 기계번역에 주로 사용되는 신경망




이전 출력값이 현재 결과에 영향을 미침



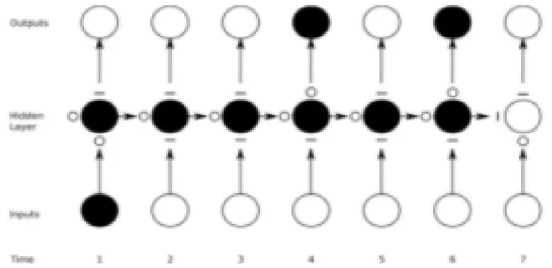

순환 W와 입력 U의 두 개의 가중치가 존재

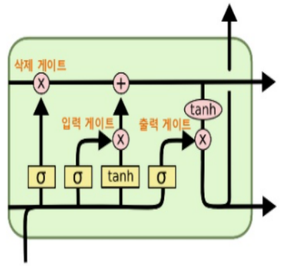



입출력이 자유로움

문장의 길이가 길어질수록 첫 단어의 의미가 끝 단어의 의미까지 반영되기 어려움

## 02. LSTM (LONG SHORT TERM MEMORY)



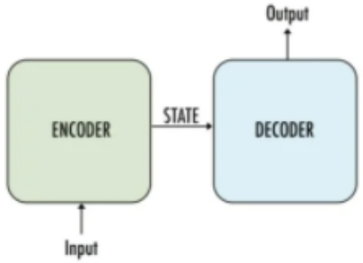

$$
\begin{aligned}
\mathbf{i}_{(t)} &= \sigma(\mathbf{W}_{xi}^{T} \cdot \mathbf{x}_{(t)} + \mathbf{W}_{hi}^{T} \cdot \mathbf{h}_{(t-1)} + \mathbf{b}_{i}) \\
\mathbf{f}_{(t)} &= \sigma(\mathbf{W}_{xf}^{T} \cdot \mathbf{x}_{(t)} + \mathbf{W}_{hf}^{T} \cdot \mathbf{h}_{(t-1)} + \mathbf{b}_{f}) \\
\mathbf{o}_{(t)} &= \sigma(\mathbf{W}_{xo}^{T} \cdot \mathbf{x}_{(t)} + \mathbf{W}_{ho}^{T} \cdot \mathbf{h}_{(t-1)} + \mathbf{b}_{o}) \\
\mathbf{g}_{(t)} &= \tanh(\mathbf{W}_{xg}^{T} \cdot \mathbf{x}_{(t)} + \mathbf{W}_{hg}^{T} \cdot \mathbf{h}_{(t-1)} + \mathbf{b}_{g}) \\
\mathbf{c}_{(t)} &= \mathbf{f}_{(t)} \otimes \mathbf{c}_{(t-1)} + \mathbf{i}_{(t)} \otimes \mathbf{g}_{(t)} \\
\mathbf{y}_{(t)} &= \mathbf{h}_{(t)} = \mathbf{o}_{(t)} \otimes \tanh(\mathbf{c}_{(t)})
\end{aligned}
$$

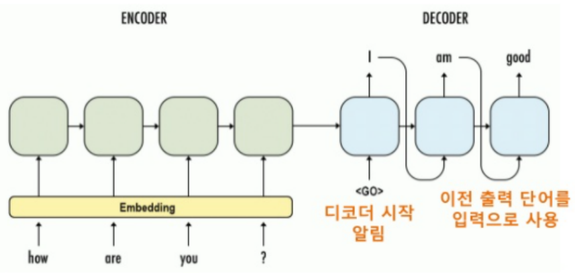
RNN의 처음 입력 정보가 뒤로 갈수록 사라지는 문제점을 보완하여 입력 중 핵심적인 정보를 잊어버리지 않고 뒤로 전달하는 목적

입력과 순환 각각 4개씩 총 8개의 가중치가 존재

## 03. SEQ2SEQ

RNN의 출력이 발 이전 입력까지만 고려해서 정확도가 떨어지고 전체 입력 문장을 반영하지 못하는 문제점을 보완




인코더와 디코더 두 개의 LSTM으로 구성

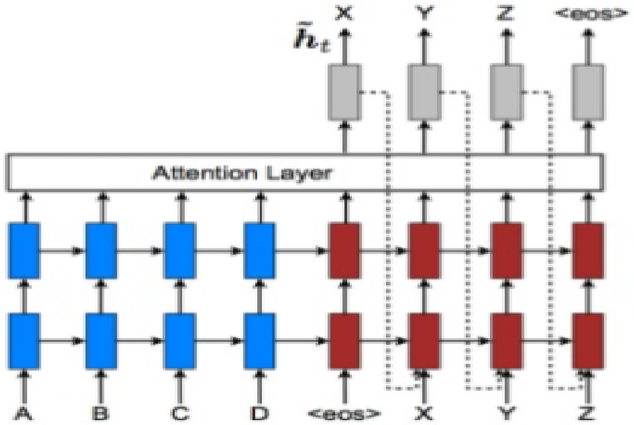
인코더로 입력 문장을 먼저 처리하고 디코더로 답변 문장 출력

# 자연어처리 발전흐름 - 한계를 위주로

기존 신경망의 경우 연속적인 시퀀스를 처리하기 어려움

## 04. ATTENTION

LSTM과 SEQ2SEQ는 정보의 흐름을 조정하는 게이트만으로 부족해서 입력 문장의 길이가 길어지면 답변의 정확도가 떨어짐

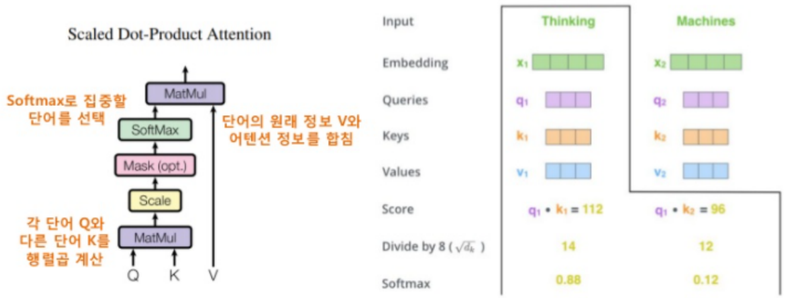



어텐션은 중요한 단어에 집중하여 디코더에 바로 전달하는 것

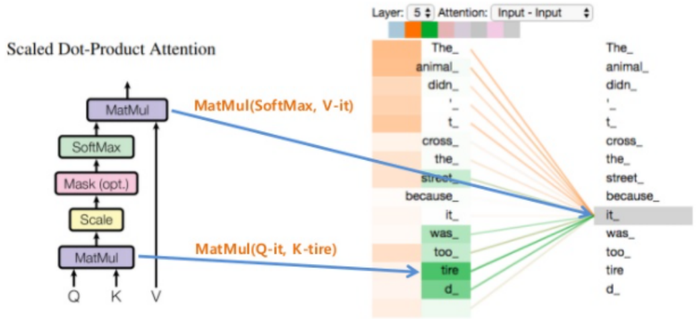
인코더의 출력값들을 모아 디코더 계산에 같이 사용하는데, 디코더에서 각 단어를 생성할 때 인코더의 어떤 단어에서 정보를 받을지 어텐션 레이어가 결정

## 05. TRANSFORMER

LSTM이 필요없고, ATTENTION만 이용하여 인코더 디코더 구현




TINKING이라는 INPUT이 임베딩을 통해 벡터화 한 후 Q,K,V로 변환, 어텐션을 계산




셀프어텐션을 통해 문장에서 중요한 단어들에 집중하여 각 단어의 정보를 업데이트 함

인코더와 디코더는 각각 여러 개로 중첩되어 구성됨

# BERT 란

BERT(BIDIRECTIONAL ENCODER REPRESENTATIONS FROM TRANSFORMERS)는 2018년 구글이 공개한 사전 훈련된(PRE-TRAINED) 모델

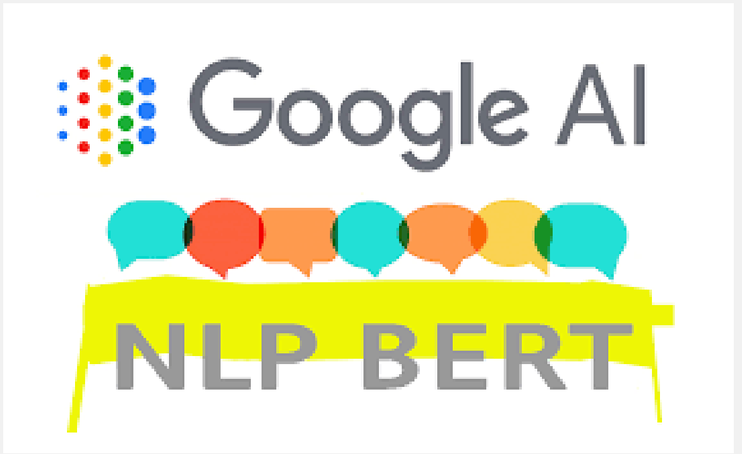
## 1) BERT의 특징

### 전이학습 모델

사전 학습된 대용량의 레이블링 되지 않는(UNLABELED) 데이터를 이용하여 언어 모델을 학습하고 이를 토대로 특정 작업(문서 분류, 질의응답, 번역 등)을 위한 신경망을 추가하는 전이 학습 방법

### 사전학습 모델

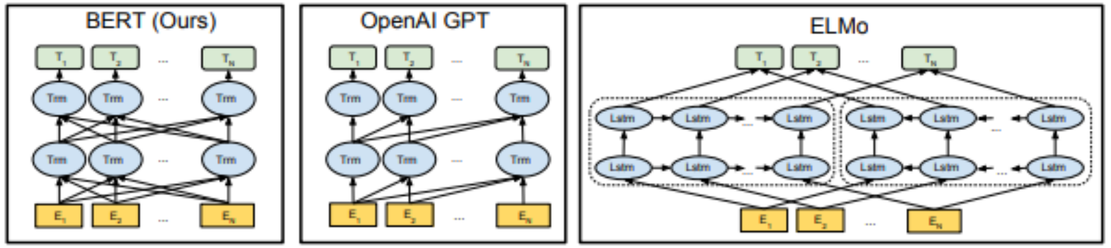
BERT 모델은 기본적으로 대량의 단어 임베딩 등에 대해 사전 학습이 되어 있는 모델을 제공하기 때문에 상대적으로 적은 자원만으로도 충분히 자연어 처리의 여러 일을 수행할 수 있음




### 양방향성

이전의 대부분 모델은 문장이 존재하면 왼쪽에서 오른쪽으로 진행하여 문맥을 파악하는 방법

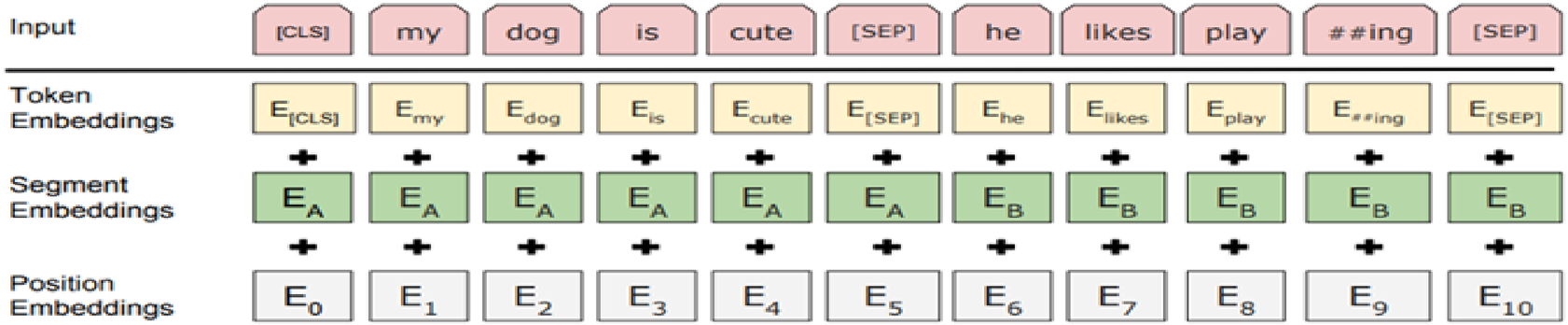
기존 단방향성 모델은 성능 향상, 문맥 파악에 한계점이 존재했었고, 이를 해결하기 위해 양방향성을 띄는 모델을 제안하는 방향으로 진행됨



# BERT의 구조

## BERT의 INPUT REPRESENTATION

BERT의 input representation은 그림과 같이 세 가지 임베딩 값의 합으로 구성됨



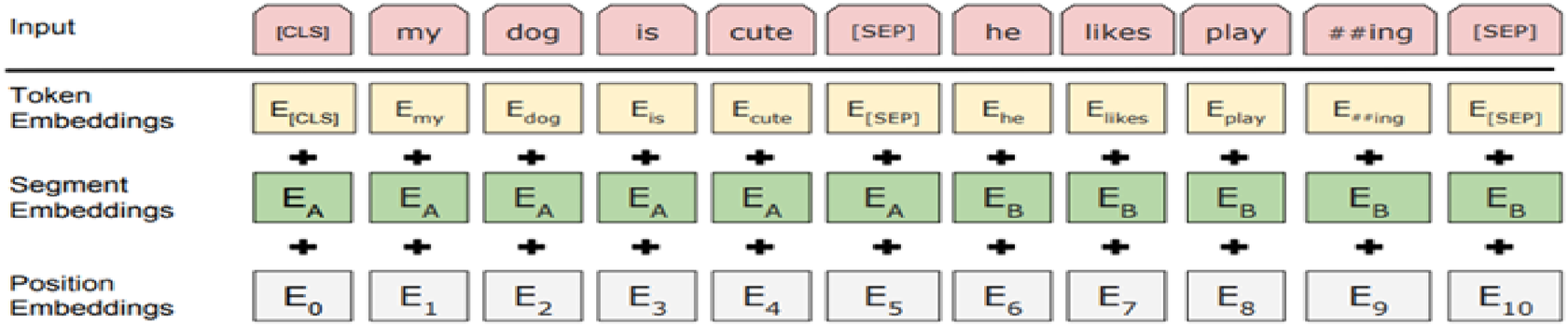

### Token Embeddings

- Word piece 임베딩 방식 사용
- 자주 등장하는 단어(sub-word)는 그 자체가 단위가 되고, 자주 등장하지 않는 단어(rare word)는 더 작은 sub-word로 쪼개어 짐
- 이전에 자주 등장하지 않은 단어를 전부 Out-of-vocabulary(OOV)로 처리하여 모델링의 성능을 저하했던 문제를 해결
- 입력 받은 모든 문장의 시작으로 [CLS] 토큰(special classification token)이 주어지며 이 [CLS] 토큰은 모델의 전체 계층을 다 거친 후 토큰 시퀀스의 결합된 의미를 가지게 됨
- 여기에 간단한 classifier을 붙이면 단일 문장, 또는 연속된 문장을 분류할 수 있고 만약 분류 작업이 아니라면 이 토큰을 무시
- 문자의 구분을 위해 끝에 [SEP] 토큰을 사용

# BERT의 구조

## BERT의 INPUT REPRESENTATION

BERT의 input representation은 그림과 같이 세 가지 임베딩 값의 합으로 구성됨




### Segment Embeddings

토큰으로 나누어진 단어들을 다시 하나의 문장으로 만들고, 첫 번째 [SEP] 토크까지는 0으로 그 이후 [SEP] 토큰까지는 1 값으로 마스크를 만들어 각 문장들을 구분함

### Position Embeddings

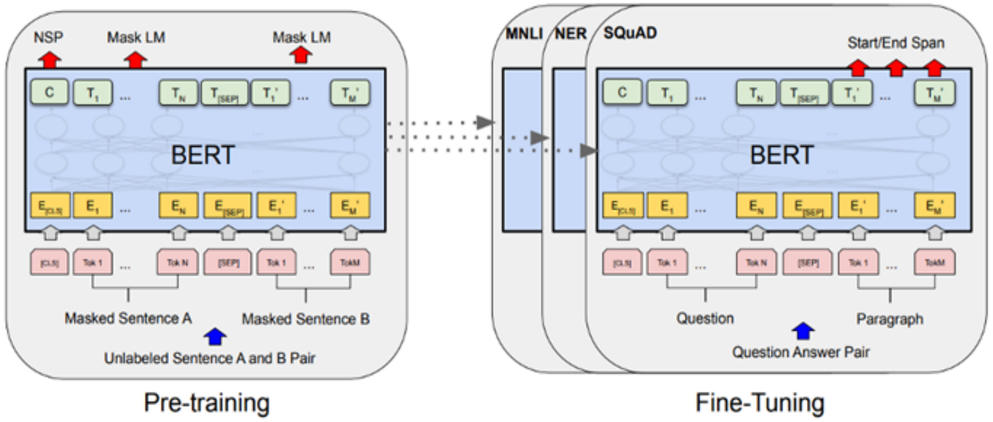
- 토큰의 순서를 인코딩 함

- BERT는 Transformer의 encoder를 사용하는데 Transformer는 Self-Attention모델을 사용

- Self-Attention 모델은 입력의 위치에 대해 고려하지 못하므로 입력 토큰의 위치 정보를 주어야 하기 때문에 Transformer에서는 Sigmoid 함수를 이용한 Positional encoding을 사용하였고, 이를 변형하여 Position Encodings를 사용

# BERT의 구조

## BERT의 PRE-TRAINING 과 FINE-TUNING




BERT를 이용한 자연어 처리는 2단계로 진행

- 거대 Encoder가 입력 문장들을 임베딩하여 언어를 모델링하는 Pre-training
(엄청난 수의 Wikipedia와 BooksCorpus 단어를 학습한 BERT를 다양한 task에 적용)

- 이를 Fine-Tuning하여 여러 자연어 처리 Task를 수행하는 과정
(다른 작업에 대해 파라미터 재조정을 위한 추가 훈련 과정)

## BERT의 적용 사례


1) Question and Answering
- 주어진 질문에 적합하게 대답
- KoSQuAD, Visual QA etc.

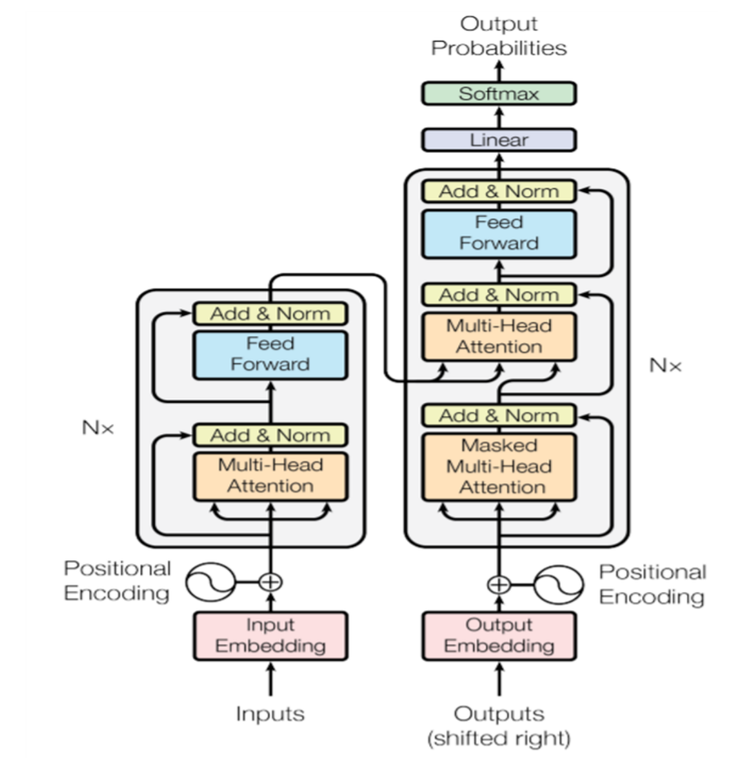
2) Machine Translation
- 구글번역기, 네이버 파파고

3) 문장 주제 찾기 또는 분류하기

4) 사람처럼 대화하기

# 사전학습 방법

## TRANSFORMER 기반의 BERT

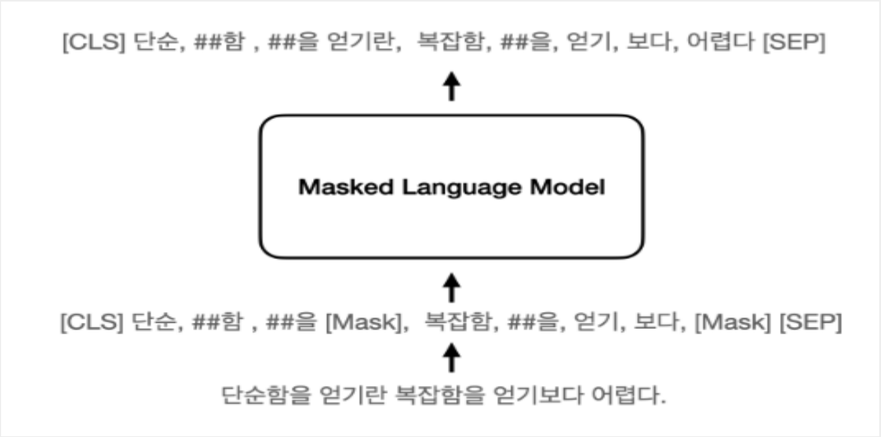



- BERT는 MLM과 NSP를 위해 Transformer 기반으로 구성됨

- 위의 그림은 트랜스포머 모델 구조를 가진 인코더-디코더 모델로 번역 영역에서 최고의 성능을 기록했고 기존 모델과 달리 CNN 및 RNN을 사용하지 않고 self-attention 개념을 도입함

- BERT는 Transformer의 인코더-디코더 중 인코더만 사용함

## 1) | MLM(MASKED LANGUAGE MODEL)




- 일련의 단어가 주어지면 그 단어를 예측하는 작업

- 입력에서 무작위 하게 몇 개의 토큰을 마스킹하고 이를 TRANSFORMER 구조에 넣어 주변 단어의 맥락으로 마스킹된 토큰만 예측

- 좌-우, 혹은 우-좌를 통하여 문장 전체를 예측하는 사전학습 언어 모델 방법과는 달리, [MASK] 토큰만을 예측하는 PRE-TRAINING 작업을 수행

- [MASK] 토큰은 PRE-TRAINING에만 사용되고, FINE-TUNING시에는 사용되지 않음

- MLM을 수행하며 BERT는 문맥을 파악하는 능력을 길러내게 됨



## 사전학습 방법

### 2) | 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

- 두 문장의 관계를 이해하기 위해 BERT의 학습 과정에서 두 번째 문장이 첫 번째 문장의 바로 다음에 오는 문장인지 예측하는 방식

- 이러한 종류의 이해를 갖춘 사전 학습 모델은 질문 답변과 같은 작업이 가능

- 학습 중에 모델에 입력으로 두 개의 문장이 동시에 제공됨

- 50%의 경우 실제 두 번째 문장이 첫 번째 문장 뒤에 오고 50%는 전체 말뭉치에서 나오는 임의의 문장

- 임의의 문장이 첫 번째 문장에서 분리된다는 가정 하에 두 번째 문장이 임의의 문장인 여부를 예측

- 완전한 입력 시퀀스는 TRANSFORMER 기반 모델을 거치며, [CLS] 토큰의 출력은 간단한 분류 계층을 사용하여 2X1 모양의 벡터로 변환됨

- ISNEXT-LABEL은 SOFTMAX를 사용하여 할당됨

- BERT는 손실 함수를 최소화하기 위해 MLM과 NSP을 함께 학습함