**연 구 용 역 보 고 서**

**페이크 뉴스 자동 검출 응용 서비스의 인공지능 표준 시스템 분석**

**수탁기관: 동의대학교 산학협력단**

한 국 전 자 통 신 연 구 원

**연 구 용 역 보 고 서**

**페이크 뉴스 자동검출 응용 서비스의 인공지능 표준 시스템 분석**

**수탁기관: 동의대학교 산학협력단**

한 국 전 자 통 신 연 구 원

**제 출 문**

**한국전자통신연구원장 귀하**

본 보고서를 “페이크 뉴스 자동검출 응용 서비스의 인공지능 표준 시스템 분석”의 최종 연구보고서로 제출합니다.

2020년 11월 08일

수 탁 기 관 : 동의대학교 산학협력단

수 탁 기 관 장 : 이 임건 (인)

연 구 책 임 자 : 김 근형

참 여 연 구 원 : 한 윤진

박 예진

이 승찬

**요 약 문**

1. **제 목**

페이크 뉴스 자동검출 응용 서비스의 인공지능 표준 시스템 분석

1. **연구의 목적 및 중요성**

**1) 연구의 목적**

**2) 연구의 필요성**

1. **연구의 내용 및 범위**
2. **연구 결과**
3. **활용에 대한 건의**



**목차**

**그림 목차**

**표 목차**

# 제 1 장 개요

## 제 1 절 연구 배경 및 목적

## 제 2 절 연구 내용

# 제 2 장 사전 학습된 언어모델 BERT

## 제 1 절 BERT(Bidirectional Encoder Representations from Transformers)

BERT는 인공지능 기반 자연어처리 분야에 등장한 구글의 강력한 언어모델이다. 기본적으로 wiki나 book data와 같은 대용량 unlabeled data로 모델을 사전 학습(pre-training)하여 문맥을 파악할 수 있도록 구축된 모델이다. 사전학습된 모델 BERT에 분류하고자 하는 task에 맞게 labeled data로 추가 학습하여 분류 성능을 높인다. BERT는 사전학습에는 Masked Language Model(MLM)과 Next Sentence Prediction(NSP) 두 가지 방법이 사용된다. MLM은 랜덤한 위치의 토큰을 마스킹 시킨 뒤, 나머지 토큰들을 기반으로 마스킹된 토큰의 단어를 예측하고, NSP는 두 개의 문장을 입력하여 두 문장이 연속된 문장인지 예측하며 학습 성능을 높인다.

BERT Encoder는 MLM과 NSP를 위해 Encoder-Decoder 아키텍처로 이루어진 Transformer의 Transformer Encoder 구조를 기반으로 한다. Transformer는 RNN(Recurrent Neural Network)과 마찬가지로 시퀀스 데이터를 처리할 수 있도록 설계된 모델이다. 하지만 RNN과 다르게 Transformer는 시퀀스 데이터를 순차적으로 처리하지 않는다. 순차적으로 처리되는 RNN은 문장이 길어지는 경우 기울기 소실 문제(Vanishing gradient problem)로 인해 초기 토큰에 대한 정보가 사라질 수 있다는 단점이 있는 반면, Transformer는 병렬처리하여 초기 토큰의 정보를 보존할 수 있다. 또한 데이터 병렬처리 방법은 학습 시간을 단축시킨다.

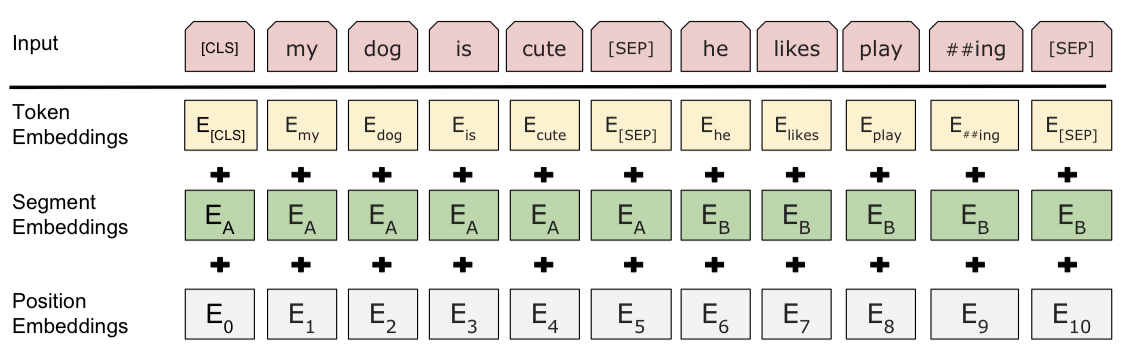


그림 1. BERT Input Data Embeddings

BERT의 입력으로는 단어 토큰화된 토큰들이 사용된다. 그림 1의 Input에서 [CLS] 토큰은 BERT Encoder의 전체 layers를 거치고 난 후 token sequence의 결합된 의미를 가지게 되어 분류에 사용되는 토큰이며, [SEP] 토큰은 문장이 두 개 들어갈 때 문장을 구분하는 것에 사용되는 토큰이다. BERT Encoder에 입력하기 위해선 Embedding 과정을 거쳐야 하는데, Token embeddings과 position embeddings이 사용되는 Transformer Encoder과는 다르게, BERT Encoder에서는 토큰이 어느 문장에 속하는지 알 수 있도록 segment embeddings까지 더하여 입력 데이터를 생성한다.

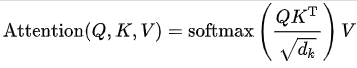


그림 2. Attention result

BERT Encoder는 Multi-Head Attention이 사용되는데 이는 임베딩된 입력 데이터의 차원을 Head 수만큼 나누어 각각 Attention을 계산하고 추후에 합치는 과정이다. Attention 은 모든 토큰 간의 상관관계에 따라 가중치를 계산하는 메커니즘이다. 주어진 토큰 벡터 Query에 대해서 다른 토큰 Key와의 관계를 계산하여 가중치를 구하게 되는데 이를 Value에 적용시켜 출력값을 생성한다. 모든 토큰에 대한 attention 계산은 다음과 같이 최적화된 행렬 계산으로 표현될 수 있다

설정한 layers 수만큼 BERT Encoder가 반복된 후 [CLS] 토큰은 token sequence의 결합된 의미를 가지게 되는데, 여기에 분류 모델을 추가되면 문장 분류 학습이 가능하게 된다. BERT는 다양한 task에 사용될 수 있으며, 입력 데이터와 분류 모델을 어떻게 Fine-tuning 하느냐에 따라 다른 유형으로 변형 가능하다.

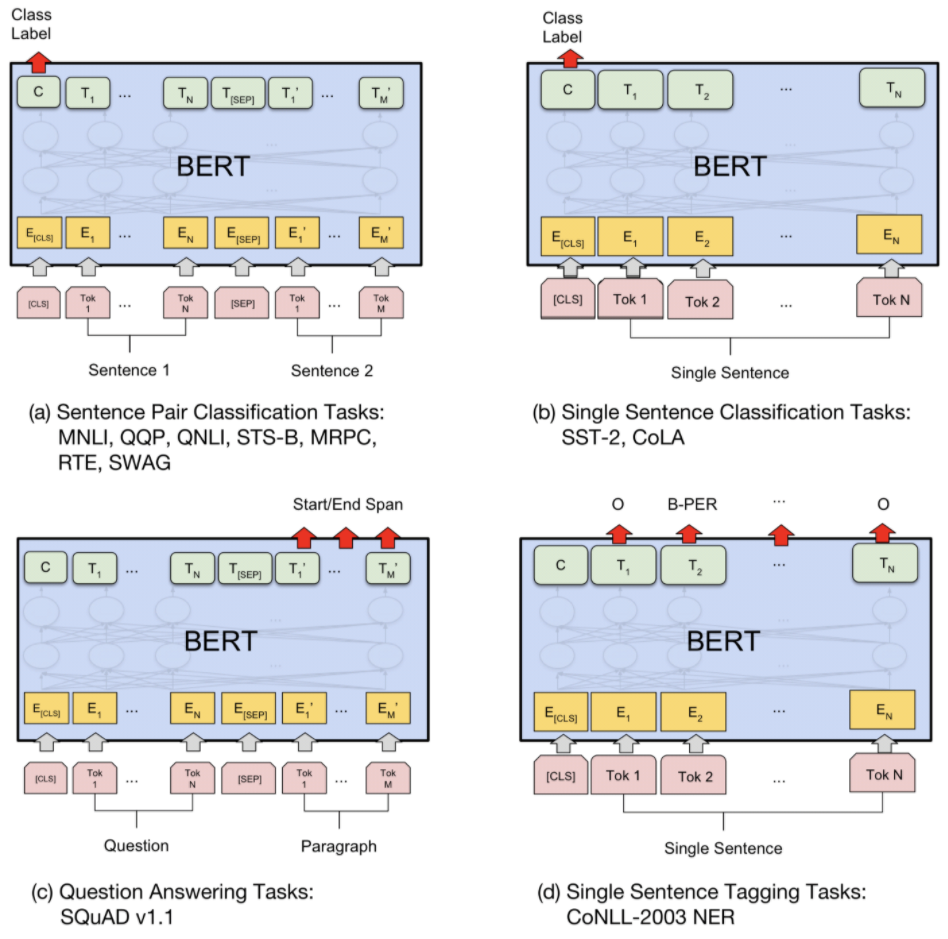


그림 3. BERT experiments result

그림 3의 (a)와 (b)와 같은 경우는 분류하고 싶은 개수에 따라 분류 모델을 조정하고, 문장의 의미를 가지는 [CLS] 토큰의 벡터를 사용하여 class를 분류하는 유형이다. (c)와 같은 Question Answering(QA) task 경우엔 Question와 Paragraph를 [SEP] 토큰으로 분류하고, Question에 정답을 Paragraph에서 찾아내는 유형이다. (d)와 같은 경우는 Named Entity Recognition(NER)이나 형태소 분석처럼 각 토큰이 어떤 class를 가지는지 각각을 분류하는 유형이다.

BERT는 위의 내용처럼 downstream task 학습을 통해 사전학습된 parameter들을 변경하며 진행되는 Fine-tuning 접근법뿐만 아니라 Feature-based 접근법으로도 사용 가능하다. Feature-based 접근법은 BERT를 통해 진행된 Feature extraction을 특정 분류를 위한 추가 모델에 입력하여 학습시키는 형태이다. Feature-based 접근법은 다른 모델인 ELMo(Embeddings from Language Model)에서도 사용되는 접근법으로, 사용하는 이유는 Fine-tuning 접근법을 사용했을 때 사용되는 pooler는 분류 모델에 들어가기 전에 [CLS] 토큰을 처리하는 과정인데, pooler의 출력 값은 분류에는 도움을 주지만 문장의 의미를 표현하기엔 좋은 값이 아니기 때문이다.

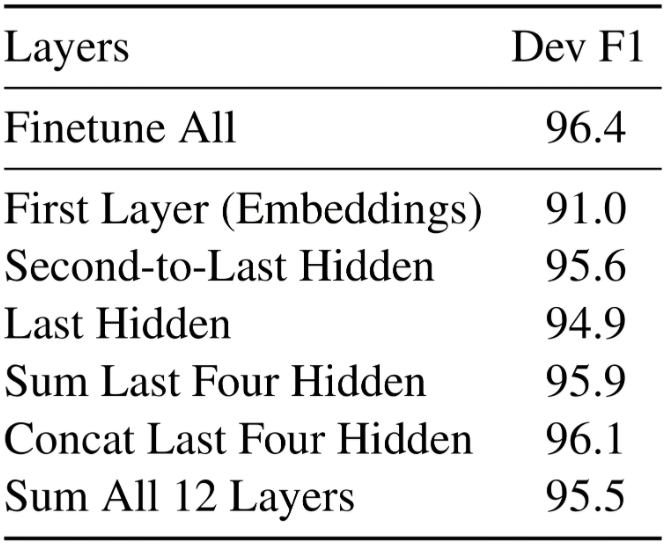
* (This output is usually not a good summary of the semantic content of the input, you’re often better with averaging or pooling the sequence of hidden-states for the whole input sequence.) [출처]
* 

그림 4. BERT with a feature-based approach

그림 4는 BERT 논문에서 Feature-based 접근법들을 사용하여 실험한 결과를 정리한 표다. 그림 4에서 볼 수 있듯이, Concat Last Four Hidden 값을 사용하면 Fine-tuning 접근법의 성능과의 차이가 크게 나지 않는다. 이를 보아, BERT는 Feature-based에서도 효과적이라고 볼 수 있다.

## 제 2절 BERT 세부 기능 분석

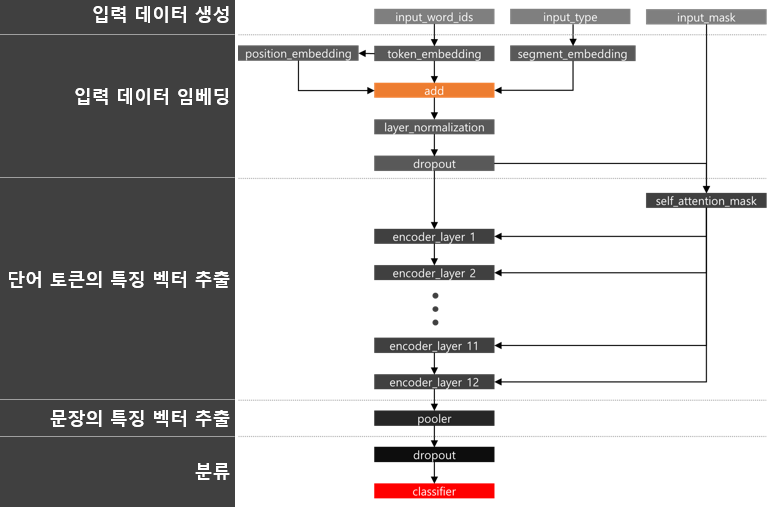


그림 5. Block diagram of the BERT

1) 입력 데이터 생성

- input\_word\_ids : 토큰화된 토큰의 index

- input\_type : 토큰의 소속 문장

- input\_mask : 토큰과 패딩의 분류

2) 입력 데이터 임베딩

- token\_embedding : 토큰 임베딩

- position\_embedding : 문장에서의 토큰 위치 임베딩

- segment\_embedding : 토큰의 소속 임베딩

3) 단어 토큰의 특징 벡터 추출

- encoder\_layer : Transformer Encoder 기반의 BERT Encoder

- Self-Attention : 각 단어 사이의 가중치를 계산하여 문맥을 파악할 수 있도록 함

- Multi-Head Attention : 여러 Head에서 계산된 Self-Attetion을 합침

- Feed-Forward Network : 여러 Head의 Self-Attention 값이 한 쪽으로 치우치지 않게 균등하게 계산함

4) 문장의 특징 벡터 추출

- pooler : Bert Encoder의 출력 중 첫 번째 토큰을 입력으로 받아 계산됨

5) 분류

- classifier : CNN, LSTM 등 분류에 사용되는 모델 추가 가능

# 제 3 장

# 제 4장 결론

**참고문헌**

[1]