과학기술정보통신부
Ministry of Science and ICT

정보통신기획평가원

medici

AiDA Lab.
Artificial Intelligence & Data Analysis Laboratory

2021년 혁신성장 청년인재 집중양성 추경 사업

빅데이터 분야

자연어 처리

언어모델
(Language Model)

Seokhwan Yang

언어 모델(Language Model)

- 언어 모델이란?
  - 언어 모델이란 문장의 확률을 나타내는 모델이다.
  - 문장은 단어들로 이루어진 시퀀셜 데이터이다.
  - 따라서 언어 모델이란 단어 시퀀스에 확률을 부여하는 모델이다.

    → 단어 시퀀스를 입력받아서 해당 시퀀스가 얼마나 그럴듯한지 확률을 출력
  - 한국어 말뭉치로 학습된 언어모델은 자연스러운 한국어 문장에 높은 확률값 부여 → 한국어 언어 모델이 필요한 이유

언어 모델(Language Model)

- 다음 문장에서 어떤 것이 가장 자연스러운가?
  - 나는 버스정류장에서 방금 버스를 _____.

1. 사랑해
2. 고양이
3. 놓쳤다
4. 사고남

우리는 아주 쉽게 정답을 맞출 수 있다. 3번

그렇다면 컴퓨터는? AI는?

문제가 “버스를”이 아니라 “버스가”였다면 쉽게 4번을 선택할 것이다.

문제가 “버스를”인 경우 4번을 선택하면? 뜻은 전달된다. 어색하지만..

언어 모델(Language Model)

- 다음의 두 문장 중에서 우리가 살아가면서 더 자주 접할 문장은?

  1. 저는 어제 점심을 먹었습니다.
  2. 저는 2015년 3월 18일 점심을 먹었습니다.

1, 2번 중에서 틀린 문장은 없다.

그러나 2번 같은 문장을 살면서 쉽게 접하지는 않는다.

언어 모델(Language Model)

- 인간의 경우는…
  - 우리는 살아오면서 수많은 문장을 접해왔고
  - 머릿속에는 단어와 단어 사이의 확률이 자신도 모르게 학습되어 있다.
  - 그래서 대화 도중에 몇 단어 정도 알아듣지 못해도 대화에는 큰 지장이 없다.
  - 추가적으로 문맥 정보를 이용하는 것도 큰 도움이 된다.

언어 모델(Language Model)

- 인간과 같은 능력을 가진 언어 모델을 만들기 위하여...
  - 우리는 인터넷, 책 등 다양한 경로로 수많은 문장을 수집하고
  - 단어와 단어 사이의 출현 빈도를 세어 확률을 계산한다.
  - 특정 분야의 문장 분포를 파악하기 위해 전문 분야에 대한 코퍼스를 수집하기도 한다.
  - 이러한 과정의 궁극적인 목표는
  - 일상 생활에서 사용하는 실제 언어(문장)의 분포를 정확하게 근사하는 것이다.

언어 모델(Language Model)

- **문장의 확률 표현**
  - **문장에서 i 번째로 등장하는 단어를 w_i로 표시한다면**
  - **n개의 단어로 구성된 문장이 해당 언어에서 등장할 확률(=언어모델의 출력)**
    - $P(w_1, w_2, w_3, w_4, \dots, w_n)$
    - n개의 단어가 동시에 나타날 결합 확률
  - **예: 잘 학습된 한국어 모델이 있다면**
    - $P(\text{무모}, \text{운전}) < P(\text{난폭}, \text{운전})$
    - "난폭"이 나타난 다음에 "운전"이 나타날 확률 → $P(\text{운전}|\text{난폭}) = \frac{P(\text{난폭}, \text{운전})}{P(\text{난폭})}$: 조건부 확률

조건부 확률 표기:
결과가 되는 사건을 앞에,
조건이 되는 사건을 뒤에 표기함

언어 모델(Language Model)

- **결합 확률과 조건부 확률**
  - **3개의 단어가 동시에 등장할 결합 확률**
    - $P(w_1, w_2, w_3) = P(w_1) \times P(w_2|w_1) \times P(w_3|w_1, w_2)$
    - 다음의 3가지 사건이 동시에 발생하여야 함
      - 첫 번째 단어(w_1) 등장
      - 첫 번째 단어(w_1) 등장 후 두 번째 단어 (w_2) 등장
      - 첫 번째 단어(w_1)와 두 번째 단어 (w_2) 등장 후 세 번째 단어 (w_3) 등장
  - **조건부 확률로 다시 쓰면**
    - $P(w_1, w_2, w_3, \dots, w_n) = \prod_{i=1}^{n} P(w_i|w_1, \dots, w_{i-1})$

언어 모델(Language Model)

- 우리는 임의의 단어 시퀀스가 해당 언어에서 얼마나 자연스러운지를 이해하고 있는 언어 모델을 구축하고자 한다.
- 조건부 확률의 정의에 따라... 수식의 좌변, 우변이 같으므로

$$P(w_1, w_2, w_3, \ldots, w_n) = \prod_{i=1}^{n} P(w_i | w_1, \ldots, w_{i-1})$$

- 이전 단어(컨텍스트)들이 주어졌을 때 다음 단어를 맞히는 문제로도 목표를 달성할 수 있다.

언어 모델(Language Model)

- **순방향 언어 모델(Forward Language Model)**
  - **문장의 앞에서 뒤로, 사람이 이해하는 순서대로 계산하는 모델**
    - 어제 카페 갔었어 거기 사람 많더라: 어제→카페→갔었어→거기→사람→많더라
  - **GPT(Generative Pretrained Transformer), ELMo 등**
- **역방향 언어 모델(Backward Language Model)**
  - **문장의 뒤부터 앞으로 계산하는 모델**
    - 어제 카페 갔었어 거기 사람 많더라: 많더라→사람→거기→갔었어→카페→어제
  - **ELMo(Embeddings from Language Models) 등**

ELMo는 순방향, 역방향 모두 사용

언어 모델(Language Model)

- 넓은 의미의 언어 모델
  - 전통적인 의미의 언어 모델은 조건부확률의 정의를 따르는 수식으로 표현
  - 최근에는...

$$P(w|context)$$

    - 컨텍스트(주변 맥락 정보)가 전제된 상태에서 특정 단어(w)가 나타날 조건부 확률로 표현하기도 한다.

언어 모델(Language Model)

- **잘 학습된 언어 모델**
  - **어떤 문장이 자연스러운지 가려낼 수 있으므로 그 자체로 가치가 있다.**
  - **학습 대상 언어의 풍부한 맥락을 표현하고 있다.**

  → 기계 번역, 문법 교정, 문장 생성 등 다양한 태스크 수행 가능

    - 기계 번역: $P($? | You can't be free from death $)$
    - 문법 교정: $P($ 두 시 삼십 이분 $) > P($ 이시 서른 두분 $)$
    - 문장 생성: $P($? | 발 없는 말이 $)$

트랜스포머(Transformer)

- 트랜스포머(Transformer)란?
  - 2017년, 구글이 제안한 Sequence-to-Sequence 모델의 하나
  - 구글이 발표한 “Attention is all you need”라는 논문에서 제안된 모델이며
  - 기존의 seq2seq 구조인 Encoder-Decoder 방식을 따르면서도
  - 논문 제목처럼 Attention 만으로 구현한 모델임
  - RNN 모델을 사용하지 않고 Encoder-Decoder를 설계하였음에도 불구하고 번역 등의 성능에서 RNN 모델보다 우수한 성능을 보임

트랜스포머(Transformer)

- 기존의 seq2seq 모델의 한계
  - 기존의 seq2seq 모델은 인코더-디코더 구조로 구성
  - 인코더는 입력 시퀀스를 하나의 벡터 표현으로 압축하고, 디코더는 이 벡터 표현을 통해서 출력 시퀀스를 만들어 냄

    → 이러한 구조는
    - 인코더가 입력 시퀀스를 하나의 벡터로 압축하는 과정에서
    - 입력 시퀀스의 정보가 일부 손실된다는 단점이 발생하였고
    - 이를 보정하기 위해 어텐션(Attention) 모델이 사용됨

트랜스포머(Transformer)

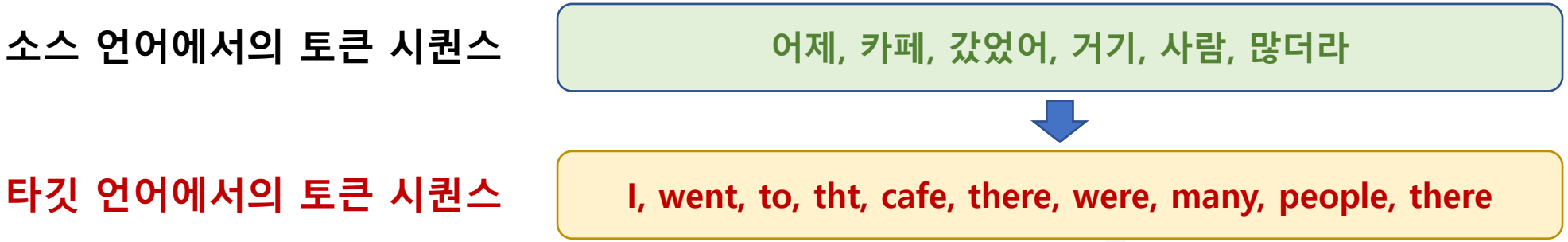
- 그런데...
  - 어텐션을 RNN의 보정을 위한 용도로서 사용하는 것이 아니라
  - 어텐션만으로 인코더와 디코더를 만들어보면 어떨까?

➔트랜스포머의 등장

- RNN에서 사용한 순환방식을 사용하지 않고 순수하게 어텐션만 사용한 모델
- 기존에 사용되었던 RNN, LSTM, GRU 등은 점차 트랜스포머로 대체되기 시작
- GPT, BERT, T5 등과 같은 다양한 자연어 처리 모델에 트랜스포머 아키텍처가 적용됨

트랜스포머(Transformer)

- 기계 번역에서의 seq2seq 작업 예시

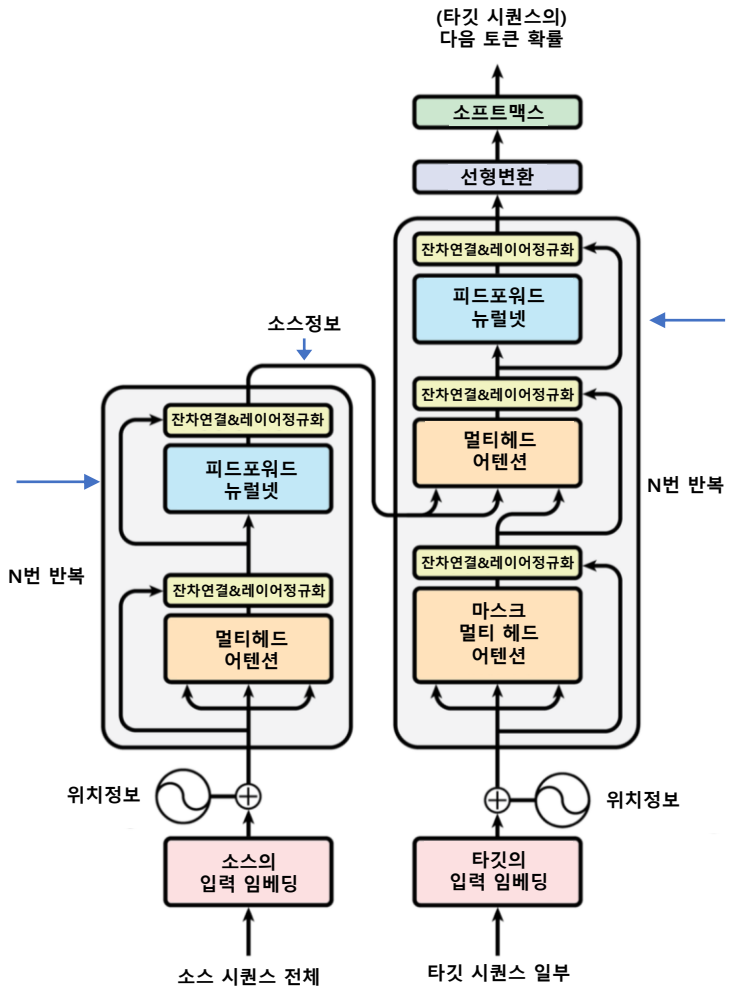


  - 소스 시퀀스의 길이(토큰 6개)와 타깃 시퀀스의 길이(10개)가 다르다.
  - 그러나 기존 seq2seq에서는 인코딩 길이는 고정 → Attention으로 극복

  ➔ 제대로 된 seq2seq는 소스, 타깃의 길이가 달라도 과제 수행에 문제가 없어야 함

트랜스포머(Transformer)

- 트랜스포머의 인코더-디코더 구조
  - 인코더의 입력: 소스 시퀀스
  - 디코더의 입력: 타깃 시퀀스의 일부



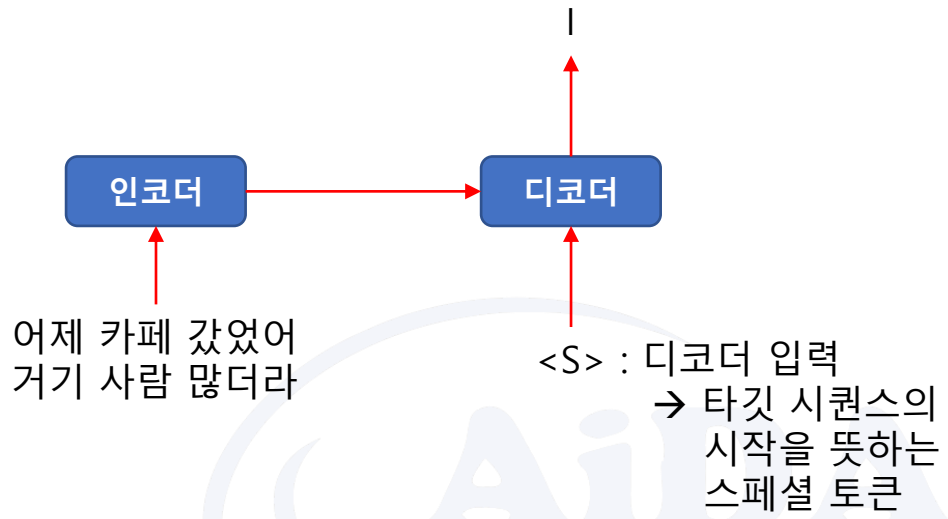
트랜스포머의 구조

트랜스포머(Transformer)

- **트랜스포머의 학습 방법**

  1. **'i'를 맞히는 학습**

     - 인코더: 소스 시퀀스를 압출하여 디코더로 보내고
     - 디코더: 인코더에서 보내온 정보와 현재 디코더 입력을 모두 고려하여 토큰(I)를 맞힘
     - 디코더의 최종 출력: 타깃 언어의 어휘 수만큼의 차원으로 구성된 벡터
       - 이 벡터는 요소의 값이 모두 확률 값



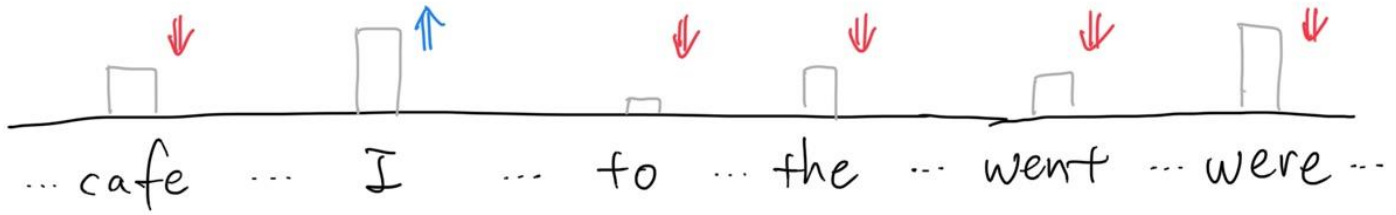
트랜스포머(Transformer)

2. 트랜스포머의 학습 진행

- 인코더와 디코더의 입력이 주어졌을 때, 정답에 해당하는 단어의 확률값을 높이는 방식으로 학습함

  → 그림에서...

  - 모델은 이번 시점의 정답인 I에 해당하는 확률은 높이고, 나머지 단어의 확률은 낮아지도록 모델 전체를 갱신

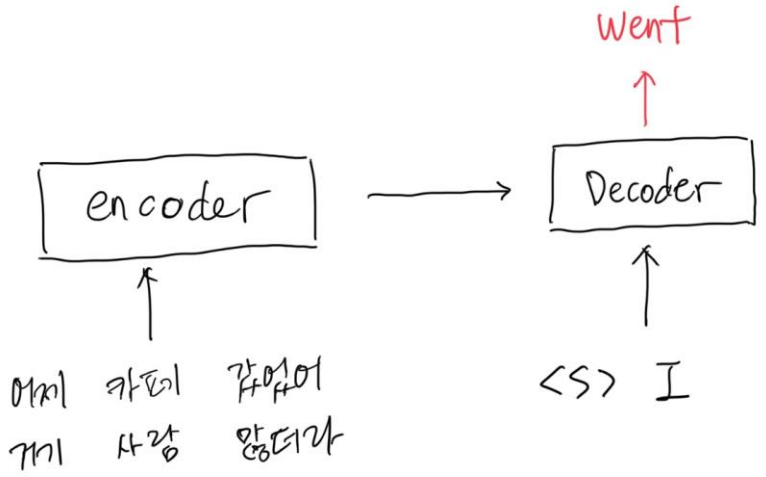


i 확률 높이기

트랜스포머(Transformer)

3. ‘went’를 맞힐 차례

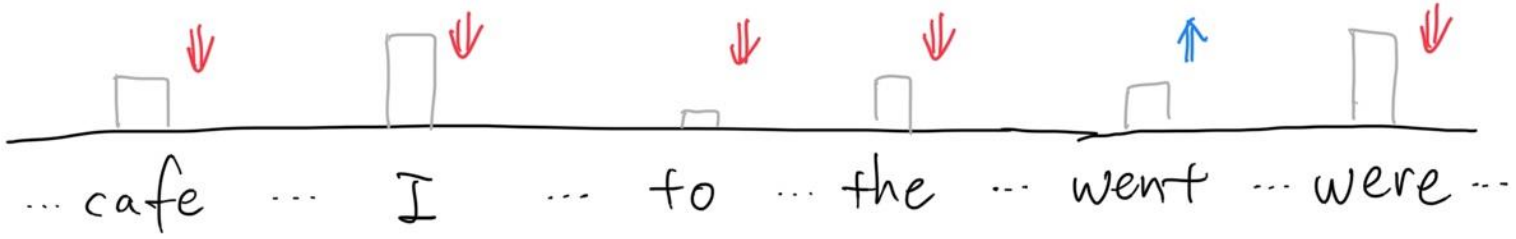
- 인코더 입력: 소스 시퀀스 전체
- 디코더 입력: <S> I
- 특이 사항
  - 학습 중의 디코더 입력과 학습을 마친 후
    모델을 실제 기계번역에 사용할 때(인퍼런스)의
    디코더 입력이 다름
    - 학습 중: 디코더 입력에 맞혀야 할 단어(went) 이전의 정답 타깃 시퀀스(<s> I)를 넣어줌
    - 학습 후(인퍼런스 때): 현재 디코더 입력에 직전 디코딩 결과를 사용



트랜스포머(Transformer)

4. ‘went’확률 높이기

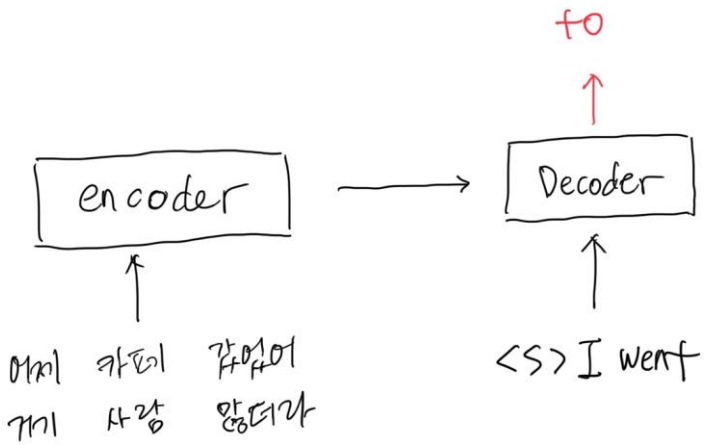
- 학습 과정 중 인코더, 디코더의 입력이 아래 그림과 같으면…
- 모델은 이번 시점의 정답인 went에 해당하는 확률은 높이고
- 나머지 단어의 확률은 낮아지도록 모델 전체를 갱신함



트랜스포머(Transformer)

5. ‘to’ 맞히기

- 인코더 입력은 소스 시퀀스 전체
- 디코더 입력은 정답인 <s> I went
- 인퍼런스 할 때, 디코더 입력은 직전 디코딩 결과



트랜스포머(Transformer)

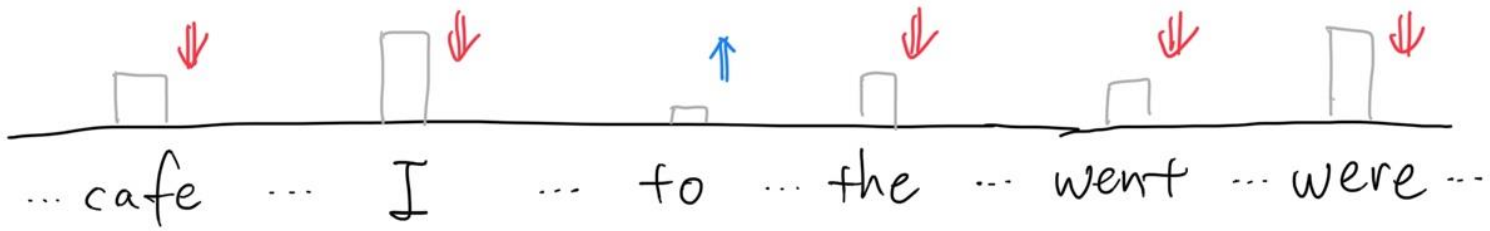
5. ‘to’ 맞히기

- 인코더 입력은 소스 시퀀스 전체
- 디코더 입력은 정답인 <s> I went
- 인퍼런스 할 때, 디코더 입력은 직전 디코딩 결과

이상의 방식으로 말뭉치 전체를 반복해서 학습

→ 기계 번역 성공

- 이번 시점의 정답인 to에 해당하는 확률은 높이고
- 나머지 단어의 확률은 낮아지도록 모델 전체를 갱신함



트랜스포머(Transformer)

- **셀프 어텐션**
  - **트랜스포머 구조에서 “멀티헤드 어텐션”은 “셀프 어텐션“ 이라고 부름**
  - **트랜스포머 경쟁력의 원천으로 평가받음**
  - **어텐션**
    - 시퀀스 입력에 수행하는 기계학습 방법의 일종
    - 시퀀스 요소 가운데 중요한 요소에 집중하고, 그렇지 않은 요소는 무시함으로써 태스크 수행 성능을 끌어올림
  - **셀프 어텐션: 디코딩 시 소스언어의 단어 시퀀스 중에서 디코딩에 도움이 되는 단어 위주로 취사 선택(가장 중요한 것만 추려서), 번역 품질을 끌어올림**

트랜스포머(Transformer)

- **셀프 어텐션 vs 합성곱 신경망(CNN)**
  - **CNN: 합성곱 필터를 이용해 시퀀스의 지역적 특징을 잡아낼 수 있다**
    - 자연어는 기본적으로 시퀀스이고
    - 특정 단어를 기준으로 한 주변 문맥이 의미 형성에 중요한 역할 수행

      → 자연어 처리에 CNN도 사용되기 시작
    - 합성곱 필터를 넘어서는 문맥은 읽어 내기 어렵다
      - 필터 크기가 3이면 4칸 이상 떨어져 있는 단어 사이의 의미는 잡아내기 어려움

트랜스포머(Transformer)

- **셀프 어텐션 vs 순환 신경망(RNN)**
  - **RNN: 시퀀스 정보 압축에 강점이 있는 구조**
    - 소스 시퀀스를 차례대로 처리
    - 그러나 RNN은 시퀀스 길이가 길어질 수록 정보 압축에 문제 발생
      - 오래 전에 입력된 단어를 잊어버리거나
      - 특정 단어 정보를 과도하게 반영해 전체 정보를 왜곡시키는 문제

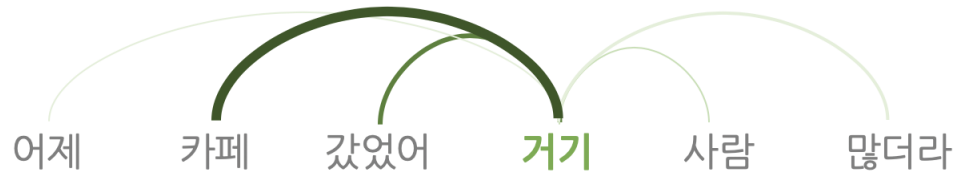
트랜스포머(Transformer)

- **셀프 어텐션 vs 어텐션**

  - **그림을 보면**
    - cafe에 대응하는 소스 언어의 단어: 카페 → 소스 시퀀스의 초반부에 등장
    - cafe라는 단어를 디코딩 할 때, 카페를 반드시 참조
    - 어텐션이 없는 단순 RNN이라면… 워낙 초반에 입력된 단어라 잊었을 가능성이 크고, 이 때문에 번역 품질이 낮아질 수 있다.

트랜스포머(Transformer)

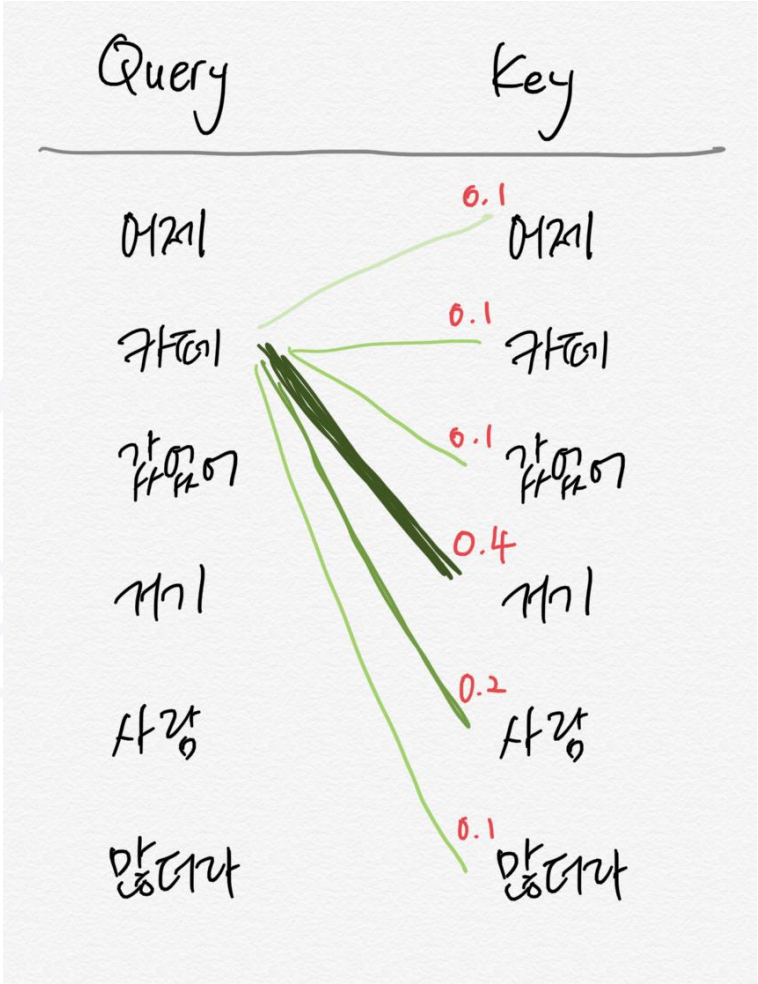
- **셀프 어텐션과 어텐션의 주요 차이**
  - 어텐션은 소스 시퀀스 전체 단어들과 타깃 시퀀스 단어 하나 사이를 연결하는데 사용. 셀프 어텐션은 입력 시퀀스 전체 단어들 사이를 연결



  - 어텐션은 RNN 구조 위에서 동작하지만 셀프 어텐션은 RNN 없이 동작함
  - 타깃 언어의 단어를 1개 생성할 때 어텐션은 1회 수행하지만, 셀프 어텐션은 인코더, 디코더 블록의 개수만큼 반복 수행

트랜스포머(Transformer)

- 셀프 어텐션 계산 예시
  - 셀프 어텐션은 쿼리, 키, 밸류가 서로 영향을 주고 받으면서 문장의 의미를 계산함
  - 쿼리 단어 각각을 대상으로 모든 키 단어와 얼마나 유기적인 관계를 맺는지를 총합이 1인 확률 값으로 나타냄
  - 카페와 가장 관련이 높은 단어는 거기(0.4)



GPT

- GPT(Generative Pre-Training)
  - 2018년, 트랜스포머의 디코더 구조를 참고하여 만든 자연어 처리모델
  - 사전훈련(Pre-Training)을 생성적(Generative) 방식으로 수행하려는 목적을 가지고 있음
  - 트랜스포머는 구글에서 발표, GPT는 OpenAI에서 발표
  - 2019년 GPT-2, 2020년 GPT-3까지 발표됨
  - GPT-2는 언어의 생성에 특히 강하다고 알려졌으며, GPT-3는 지금까지 발표된 가장 거대한 언어모델(1,750억 개에 달하는 매개변수를 보유)

GPT

Transformer.Decoder → GPT1

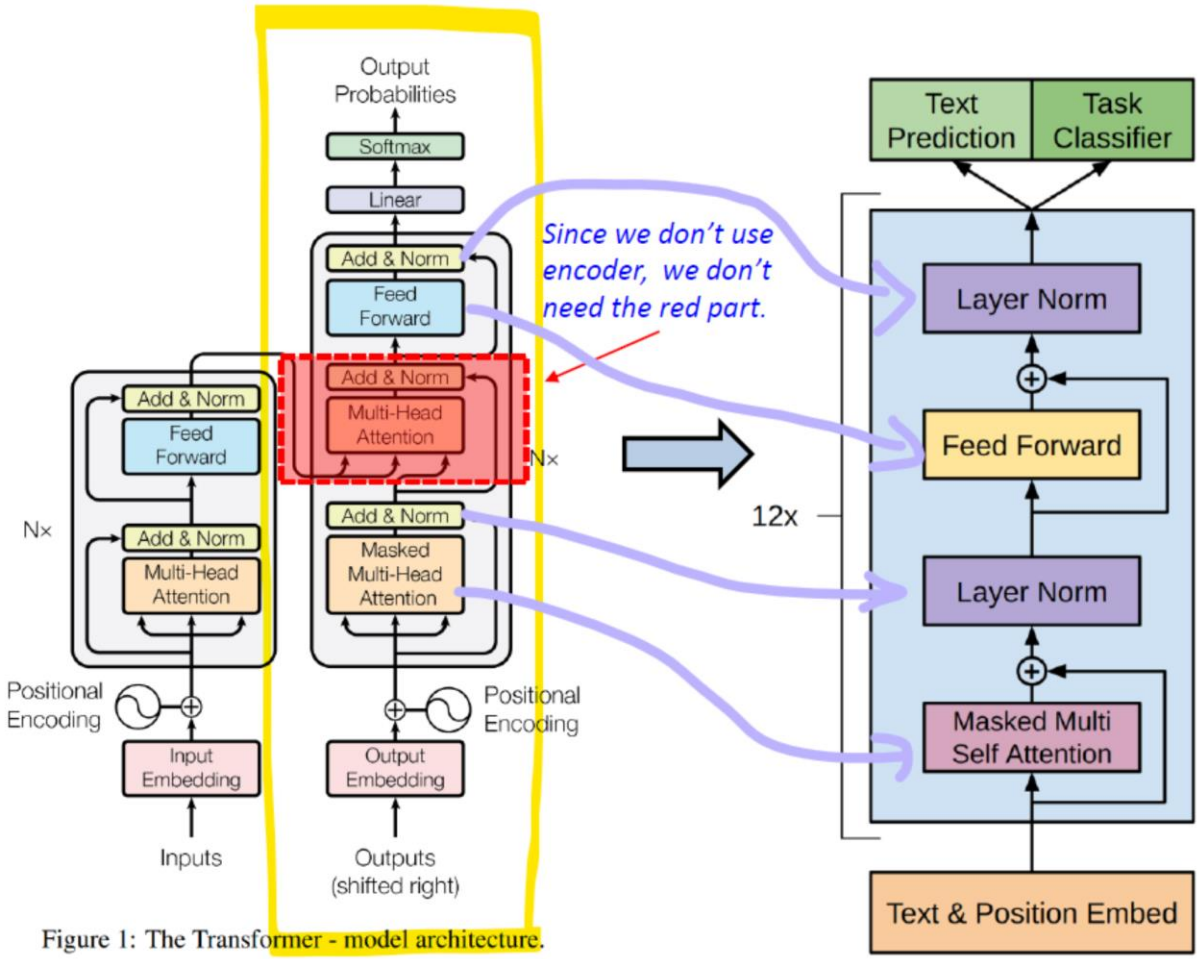


Figure 1: The Transformer - model architecture.

- **GPT-3의 한계점**
  - **낮은 효율성**
    - 1,750억개의 매개변수 → 인간이 평생 보는 정보보다 많은 데이터를 학습해야 함
  - **현실세계의 물리적 상식 부족**
    - "치즈를 냉장고 안에 넣으면 녹을까?" 라는 질문에 "그렇다"라고 대답
    - 세상을 글로만 학습했기 때문. 현실에서 직접 겪어봐야 알 수 있는 매우 당연한 상식을 학습할 기회가 적었음

GPT

- **모든 분야에서 뛰어난 것은 아니다.**
  - 아직까지는 대부분 태스크에서 사람보다 떨어진 성능을 보임
  - 주어진 태스크마다 성능차이가 심함
    - 두가지 이상의 복합연산 능력이 떨어지고
    - 태스크를 수행하기 위해 주어진 데이터가 적을수록 성능이 크게 떨어지는 경향을 보임
- **학습에 사용된 예제를 외운 것인지 실제 추론한 것인지 구분하기 어렵다.**
- **새로운 정보를 수용하기 어렵다. 한마디로 "기억력"이 없다.**
  - 학습된 정보를 토대로 입력 값에 대해 출력 값을 내보낼 수는 있지만, 사람처럼 기억력이라 부를 만한 것이 없다.(그러나 다른 AI도 마찬가지)

GPT

- **방대한 양의 텍스트를 통해 다음 단어를 예측하는 방식으로 학습됨**
  - 주어진 단어에 대해 통계적으로 가장 어울리는 다음 단어를 생성하는 것뿐이며 이해하는 것은 아니라는 비판(이것 역시 다른 AI도 마찬가지)

GPT를 공격하는 사람들이 GPT의 한계점이라고 표현하는 내용중 일부는

GPT 만이 아닌 전체적인 AI 기술에 해당하는 한계임(그냥 깎아내리기 목적?)

그래도 중요한 것은...

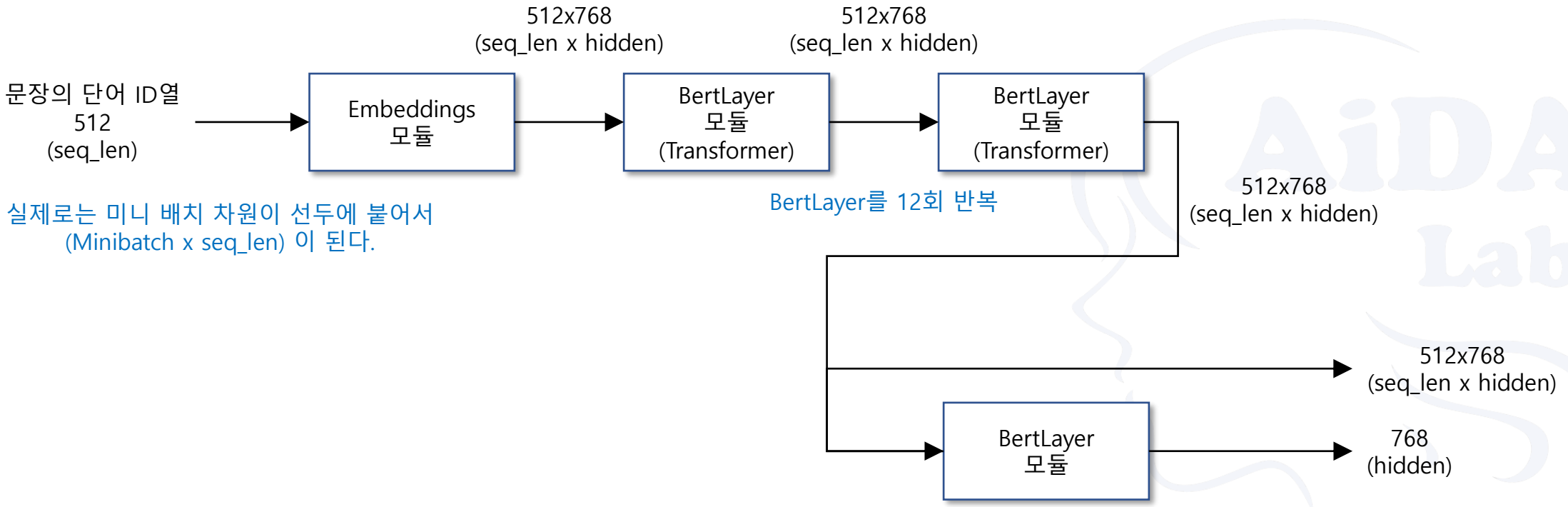
"인간은 다음 단어를 예측하는 방법으로 언어를 학습하지 않았다"는 것

BERT

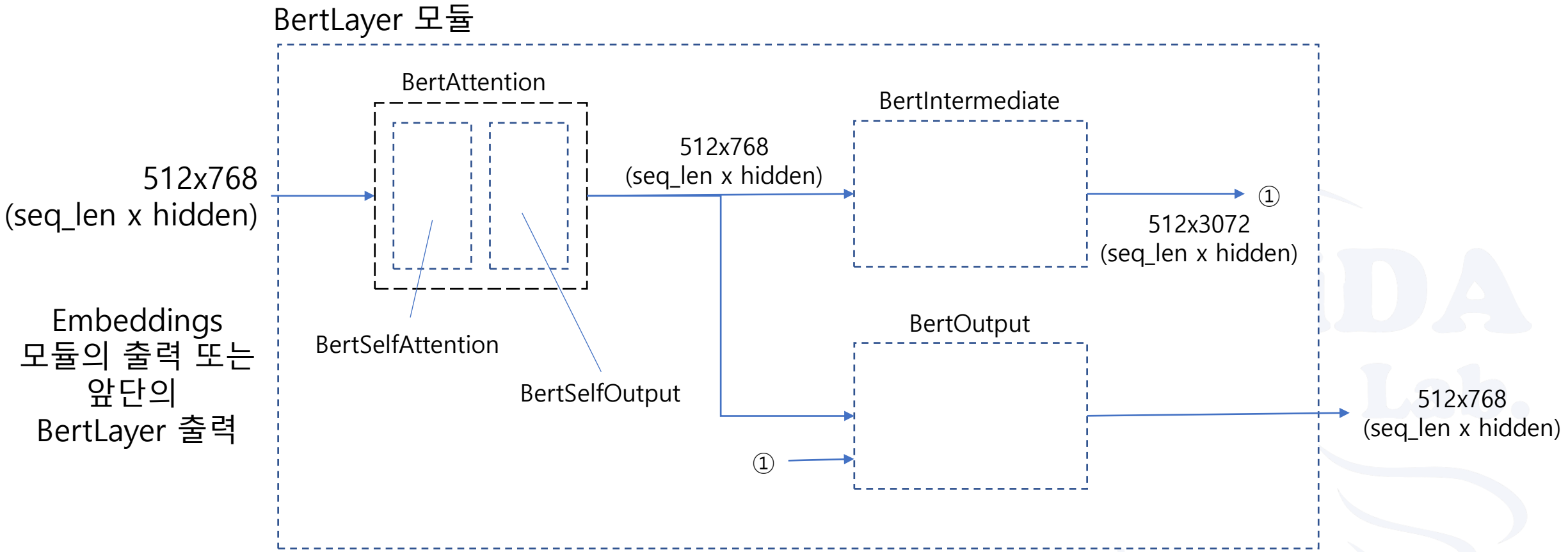
- BERT (Bidirectional Encoder Representation Transformers)
  - 2018년 하반기 Google이 발표한 자연어 처리를 위한 딥 러닝 모델
  - 일부 성능평가에서는 인간보다 높은 정확도를 보임
  - 영상인식 계열에 비하여 발전이 늦은 언어처리 딥 러닝의 한계를 돌파하는 계기가 될 것으로 기대 받음

BERT 모델

- **BERT-Base 모델 기준으로 학습**
- **BERT 모델 구조**



BertLayer 모듈 구성



BERT 모델 구조

- BERT 모델 프로세스
  - 문장을 단어 ID로 하는 ID열(길이는 seq_len=512) → 입력 → Embeddings 모듈 전달

  - Embeddings 모듈은 ID 열을 단어의 특징량 벡터로 변환
    → 단어와 특징량 벡터의 위치 정보를 나타내는 Positional Embedding 추가
    : BERT-Base에서 사용하는 특징량 벡터의 차원 수는 768
      → (모델 구조도에서 hidden으로 표시)

BERT 모델 구조

- Embedding 모듈의 출력 텐서인 (seq_len x hidden) = 512 x 768 → BertLayer로 전달

- BertLayer 모듈: Self-Attention을 이용하여 특징량 변환 수행 → 모듈을 총 12회 반복: 출력 텐서 크기는 입력 텐서와 같은 512 x 768

- 12회 반복된 BertLayer 모듈의 출력 텐서(512 x 768(에서 첫 단어의 특징량 (1 x 768)을 → BertPooler 모듈에 입력

BERT 모델 구조

- 출력 텐서의 첫 단어를 [CLS]로 설정 → 문장의 클래스 분류 등에 사용하기 위한 입력 문장 전체의 특징량을 가지는 부분으로 활용

- 선두 단어의 특징량을 BertPooler 모듈로 변환

- 최종 출력 텐서 (2가지)
  - 12회의 BertLayer 모듈에서 출력된 (seq_len x hidden)=(512x768) 텐서
  - 선두 단어 [CLS]의 특징량(BertPooler 모듈의 출력)인 크기 768의 텐서

BERT 모델의 사전 작업

- BERT는 네트워크 모델을 두 종류의 언어 작업으로 사전 학습함
  - MLM (Masked Language Model)
  - NSP (Next Sentence Prediction)

BERT 모델의 사전 작업

- MLM
  - CBOW 모델의 확장판 작업
  - CBOW 모델: 문장 중 한 단어를 마스크하여 알 수 없게 하고, 마스크 단어의 앞뒤(약 5단어씩) 정보로 마스크된 단어를 추정하는 모델
  - 입력의 512 단어 중 여러 단어를 마스크하여 마스크된 단어 앞뒤 몇 단어를 지정하지 않고 마스크되지 않은 단어 모두를 사용하여 마스크된 단어를 추정함으로써 해당 단어의 특징량 벡터를 획득하는 작업

BERT 모델의 사전 작업

- **NSP**
  - **BERT 모델은 사전 학습에서 두 개의 텍스트 데이터를 입력함 (512 단어로 두 문장 구성)**
  - **두 문장은 [SEP]로 구분되며 지도 데이터 내에서 두 개의 패턴으로 준비됨**
    - 연속적으로 존재하며 의미 있고 관계가 깊은 문장
    - 전혀 관계가 없고 문맥의 연결이 없는 두 문장
  - **BertPooler 모듈에서 출력된 선두 단어 [CLS]의 특징량으로 입력된 두 개의 문장이 어떤 패턴인지 추론**

BERT 모델의 사전 작업

- **BERT 모델의 두 종류의 사전 학습의 예**
  - **MLM**

    일부 단어가 마스크 된 문장을 입력하고 해당 단어가 Vocabulary의 어떤 낱말인지 맞히기

    (문장) [CLS] I accessed the 마스크 account [SEP] We play soccer at the bank of the 마스크 [SEP]
    답: bank
    답: river

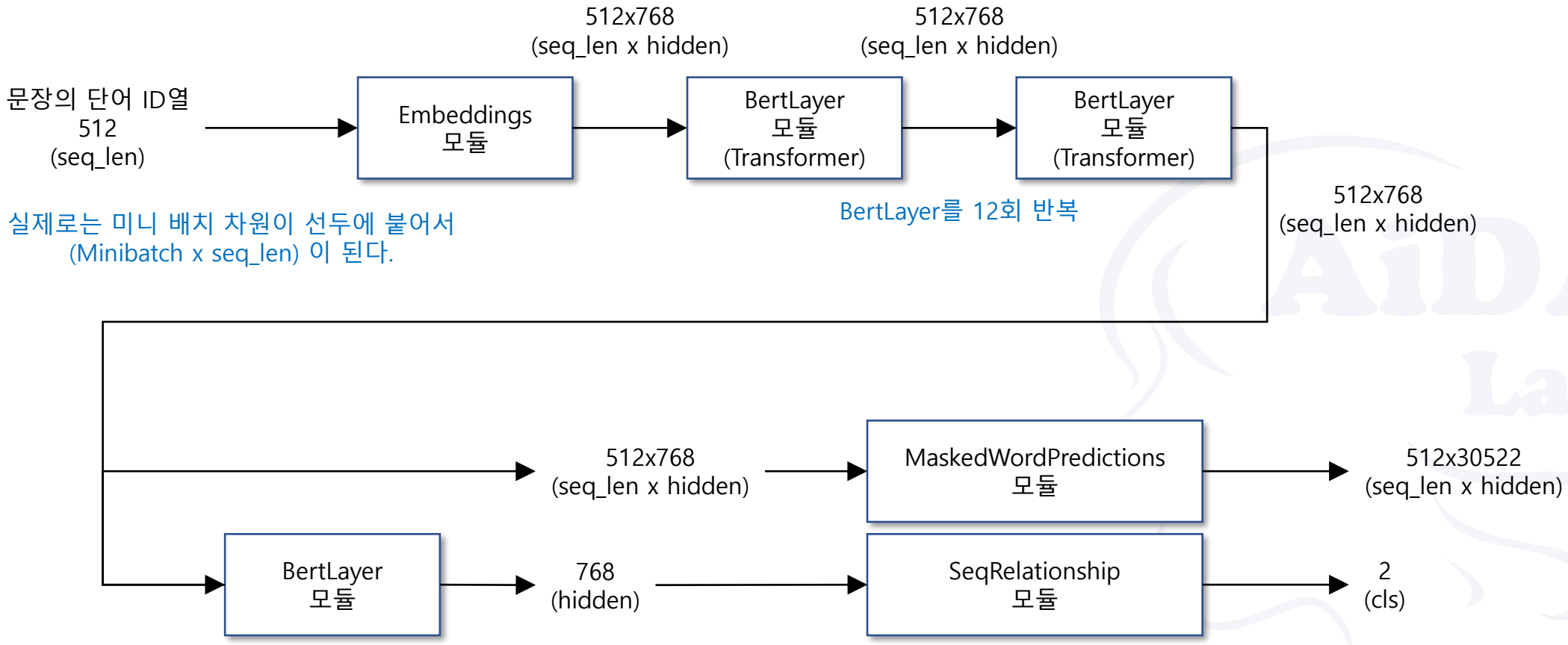
  - **NSP**

    두 문장을 입력하고 두 문장의 의미가 연결되어 있는지 맞히기

    (문장) [CLS] I accessed the bank account [SEP] We play soccer at the bank of the river. [SEP]
    답은 관계성이 없음

BERT 모델의 사전 작업

- 사전 학습을 실시하는 BERT 모델 구조



BERT 모델의 사전 작업

- 두 언어 작업을 해결하기 위하여 모듈 연결
  - 기본 모델에 MaskedWordPredictions 모듈과 SeqRelationship 모듈을 붙여 두 종류의 사전 작업인 MLM과 NSP를 잘 수행할 수 있도록 기본 모델 학습

BERT 모델의 사전 작업

- MaskedWordPredictions 모듈
  - BertLayer 출력(seq_len x hidden)=(512x768)을 입력하고 (seq_len x vocab_size) = (512x30,522) 출력
  - vocab_size (30,522)는 BERT의 vocabulary 전체의 단어 수(영어의 경우)
  - 입력된 512 단어가 전체 vocabulary 단어의 어느 것인지 (512x30522)에 대하여 소프트맥스 함수를 계산하여 도출
  - 실제로 추정하는 것은 입력 단어 512개 전체가 아닌 마스크된 알 수 없는 단어 뿐

BERT 모델의 사전 작업

- **SeqRelationship 모듈**
  - **BertPooler 모듈에서 출력된 선두 단어 [CLS]의 특징량 벡터를 전결합층에 입력하여 클래스의 수가 2인 분류를 실행**
  - **전 결합층의 출력 크기 2 → 아래의 2 패턴 중 어느 쪽인지 판정하기 위함**
    - 연속적으로 존재하며 의미 있고 관계가 깊은 문장
    - 전혀 관계가 없고 문맥의 연결이 없는 두 문장

BERT의 특징

- BERT의 세가지 특징
  - 문맥에 의존한 단어 벡터 표현을 만들 수 있게 되었다.
  - 자연어 처리 작업에서 파인 튜닝이 가능해 졌다.
  - Attention에 의한 설명과 시각화가 간편해 졌다.

BERT의 특징

- **문맥에 의존한 단어 벡터 표현을 만들 수 있게 되었다.**
  - **어떤 언어라도 단어의 의미가 단 하나인 경우는 적음**
    - (예) bank: 은행, 강변 이라는 의미가 있음
  - **다양한 의미를 가진 각 단어들은 문맥에 따라 단어의 의미가 바뀜**
  - **BERT는 문맥에 맞는 단어의 벡터 표현이 가능함**

BERT의 특징

- **BERT는 12단 Transformer를 사용**
  - Embedding 모듈에서 단어ID를 단어 벡터로 변환할 때는 은행의 bank와 강변의 bank는 동일한 길이(768)의 단어 벡터
  - 12단의 Transformer를 거치는 동안 단어 bank의 위치에 있는 특징량 벡터는 변화함
  - 변화 결과, 12단 째의 출력인 단어, bank의 위치에 있는 특징량 벡터는 최종적으로 은행 bank와 강변 bank가 서로 다른 벡터가 됨
  - 여기서 말하는 특징량 벡터란, 사전 학습의 MLM이 풀어놓은 특징량 벡터
  - 문장 중의 단어 bank와 그 주변 단어와의 관계성을 바탕으로 하여 Transformer의 Self-Attention 처리로 작성됨
  - 동일한 단어도 주변 단어와의 관계성에 따라 문맥에 맞는 단어 벡터가 생성됨

BERT의 특징

- **자연어 처리 작업에서 파인 튜닝이 가능해 졌다.**
  - **BERT를 기반으로 다양한 자연어 처리를 수행하려면**
    - 두 언어 작업에서 사전 학습한 가중치 파라미터를 BERT 모델의 가중치로 설정
    - BERT 모델 구조 그림에서 나타낸 (seq_len x hidden)=(512x768) 텐서와 (hidden)=(768)의 두 텐서를 출력
    - 두 텐서를 실행하고 싶은 자연어 처리 작업에 맞춘 어댑터 모듈에 투입
    - 작업에 따른 출력 획득
    - (예) 긍정적/부정적 감정 분석의 경우, 어댑터 모듈로 하나의 전결합층을 추가하는 것 만으로 문자의 판정이 가능해짐

BERT의 특징

- 모델을 학습할 때, 기반이 되는 BERT와 어댑터 모듈의 전결합층 양쪽 모두를 파인 튜닝으로 학습
- BERT의 출력에 어댑터 모듈을 연결하여 다양한 자연어 처리 작업 수행 가능
- Object Detection을 위한 SSD 모델, 자세 추정을 위한 OpenPose 모델에서 사용된 기반 네트워크인 VGG와 같은 역할을 BERT가 수행
- 적은 문서 데이터로도 성능 좋은 모델의 작성이 가능함
- 자연어 처리 작업도 화상 작업처럼 전이학습 및 파인 튜닝을 적용할 수 있게 된 것이 BERT가 주목받은 요인의 하나임

BERT의 특징

- **BERT는 어떻게 화상 작업의 기본 모델인 VGG와 같은 전이학습 및 파인 튜닝의 기반 역할을 수행할 수 있을까?**
  - VGG 모델과 같이 화상 처리에서 화상 분류가 가능한 네트워크는 물체 감지나 시맨틱 분할에도 유효함
  - BERT도 사전작업 MLM을 풀 수 있는 **단어를 문맥에 맞는 특징량 벡터로 변환할 수 있는 능력**이 단어의 의미를 정확하게 파악할 수 있게 함
  - 사전작업 NSP로 **문장이 의미 있게 연결되었는지 여부를 판정할 수 있는 능력**이 문장의 의미를 이해할 수 있게 함
  - 단어와 문장의 의미를 이해할 수 있도록 사전학습을 하고 있으므로 자연어 처리 작업인 감정 분석 등에도 응용이 가능해짐

BERT의 특징

- **선구적인 범용 언어 모델의 사례를 만듦**
  - 단어와 문장의 의미를 제대로 파악해야 하는 사전 작업의 수행
  - 사전 작업으로 학습한 가중치를 기반으로, 어댑터를 자연어 처리 작업에 맞게 교체하여 파인 튜닝을 수행

→ 이러한 처리의 흐름이 자연어 처리에서의 하나의 표준이 될 것으로 기대됨

BERT의 특징

- Attention에 의한 설명과 시각화가 간편해 졌다.
  - Attention: 결과에 영향을 준 단어의 위치정보
  - Attention을 시각화 함으로써 인간이 추론 결과를 설명하기가 쉬워짐