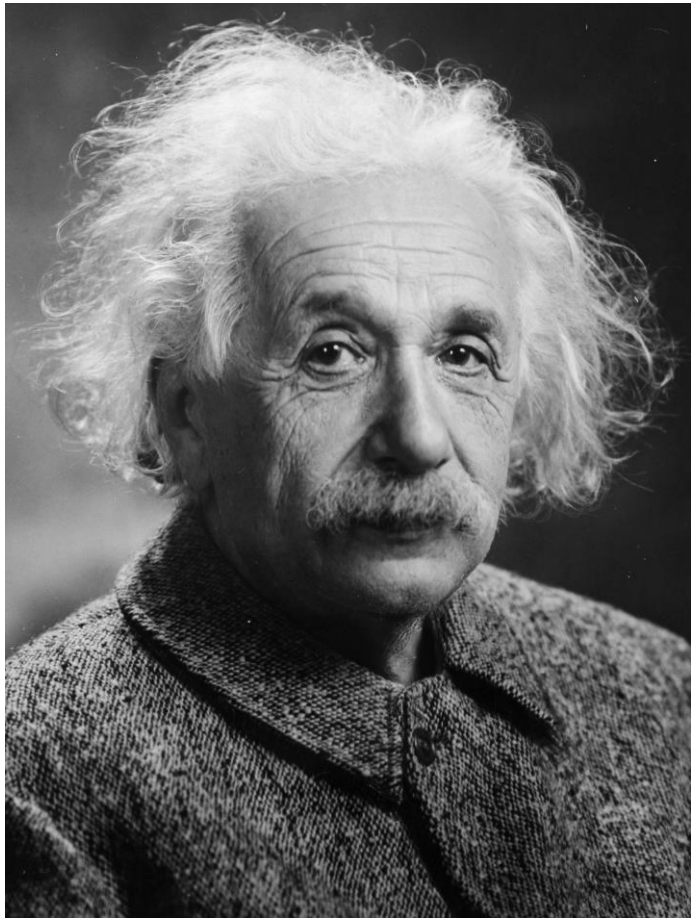
구글 BERT의 정석

BERT의 파생 모델 I: ALBERT, RoBERTa, ELECTRA, SpanBERT

20180376 안제준

ALBERT



크로스 레이어 변수 공유

- 레이어 간에 변수를 공유하는 방법들
  - All-shared : 첫 번째 인코더의 하위 레이어에 있는 모든 변수를 나머지 인코더와 공유
  - Shared feedforward netword : 첫 번째 인코더 레이어의 피드포워드 네트워크의 변수만 다른 인코더 레이어의 피드포워드 네트워크와 공유
  - Shared attention : 첫 번째 인코더 레이어의 멀티 헤드 어텐션의 변수만 다른 인코더 레이어와 공유

펙토라이즈 임베딩 변수화

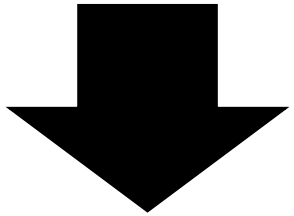
V X H로 직접 투영하는 대신, V X E 와 E X H로 분해하는 방법을 사용

- 먼저 사전의 원-핫 인코딩한 벡터 V를 저차원의 워드피스 임베딩 공간 E로 투영한다(V X E). 이때 워드피스 임베딩의 차원은 V X E = 30000 X 128이 된다.
- 그 다음 워드피스 임베딩 공간 E를 은닉 레이어 H로 투영한다(E X H). 이 때 차원은 EXH = 128X768이 된다.

→즉 V X H 대신 V X E 와 E X H로 분해하는 것이다.

ALBERT 모델 학습

문장1 : she cooked pasta
문장2 : it was delicious



문장1 : it was delicious
문장2 : she cooked pasta

ALBERT와 BERT 비교

Model		Parameters	Layers	Hidden	Embedding	Parameter-sharing
BERT	base	108M	12	768	768	False
	large	334M	24	1024	1024	False
ALBERT	base	12M	12	768	128	True
	large	18M	24	1024	128	True
	xlarge	60M	24	2048	128	True
	xxlarge	235M	12	4096	128	True

모델	파라미터	레이어	히든	임베딩
BERT-base	110M	12	768	768
BERT-large	334M	24	1024	1024
ALBERT-base	12M	12	768	128
ALBERT-large	18M	24	1024	128
ALBERT-xlarge	60M	24	2048	128
ALBERT-xxlarge	235M	12	4096	128

RoBERTa

- MLM 태스크에서 정적 마스킹이 아닌 동적 마스킹 방법을 적용
- NSP 태스크를 제거하고 MLM 태스크만 학습에 사용
- 배치 크기를 증가해 학습
- 토크나이저로 BBPE(byte-level BPE)를 사용

정적 마스크 대신 동적 마스크 사용

동적 마스킹 :

먼저 하나의 문장을 10개로 복사

→ 10개의 문장에 대해 무작위로 15%확률 마스크 작업

→ 그럼 10개의 문장은 각기 다른 마스크된 토큰을 가짐

→ 에폭 별로 다른 마스크가 적용된 문장을 입력하게 됨

에폭	**문장**
에폭 1	문장 1
에폭 2	문장 2
.	.
에폭 11	문장 1
에폭 12	문장 2
.	.
에폭 40	문장 10

NSP 태스크 제거

Model	SQuAD 1.1/2.0	MNLI-m	SST-2	RACE
Our reimplementation (with NSP loss):				
SEGMENT-PAIR	90.4/78.7	84.0	92.9	64.2
SENTENCE-PAIR	88.7/76.2	82.9	92.1	63.0
Our reimplementation (without NSP loss):				
FULL-SENTENCES	90.4/79.1	84.7	92.5	64.8
DOC-SENTENCES	90.6/79.7	84.7	92.7	65.6

더 많은 데이터로 학습

- 기존 BERT에서 사용한 데이터셋 이외에 추가로 사용함
  - CC-News
  - Open WebText
  - Stories(크롤 데이터의 일부)

큰 배치 크기로 학습

- BERT → 256개 배치로 100만 단계 동안 사전 학습
- RoBERTa → 8000개 배치로 30만 단계에 동안 사전 학습

- 배치 크기를 키우는 이유 ⇒ 학습 속도를 높일 수 있고 모델 성능 또한 향상시킬 수 있음

BBPE 토크나이저 사용

- BPE와 유사하나 캐릭터 형태가 아닌 바이트 형태의 시퀀스를 사용

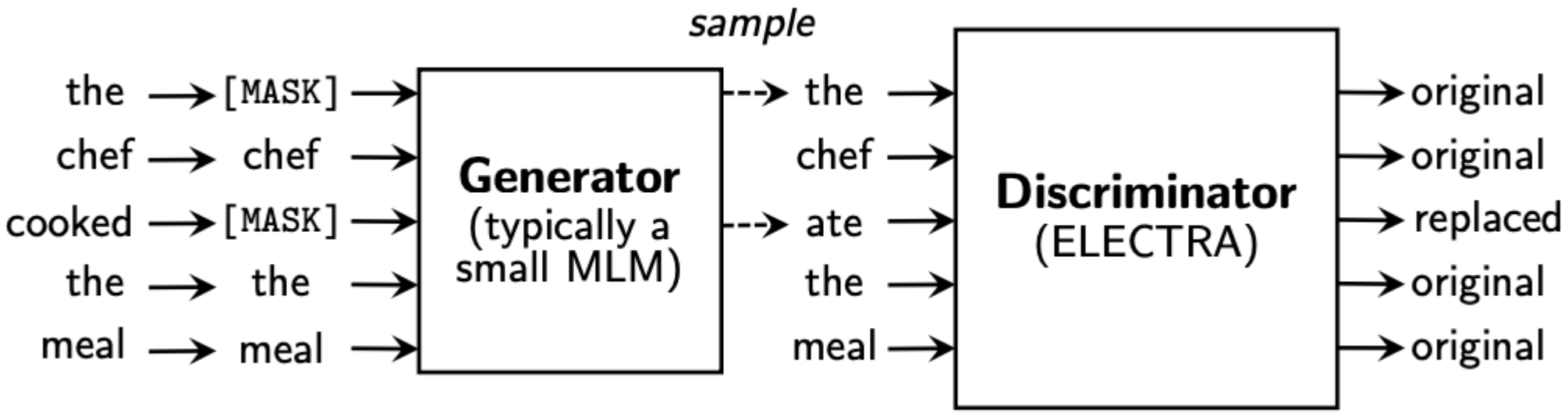
토크나이저 코드

ELECTRA 이해하기

MLM 대신 교체한 토큰 탐지(replaced token detection)태스크를 사용해 학습

교체한 토큰 탐지 태스크 → MLM과 유사하지만 마스킹 대상인 토큰을 다른 토큰으로 변경한 후 이 토큰이 실제 토큰인지 아니면 교체한 토큰인지를 판별하는 형태로 학습을 진행

교체한 토큰 판별 태스크 이해하기



ELECTRA의 생성자와 판별자 이해하기

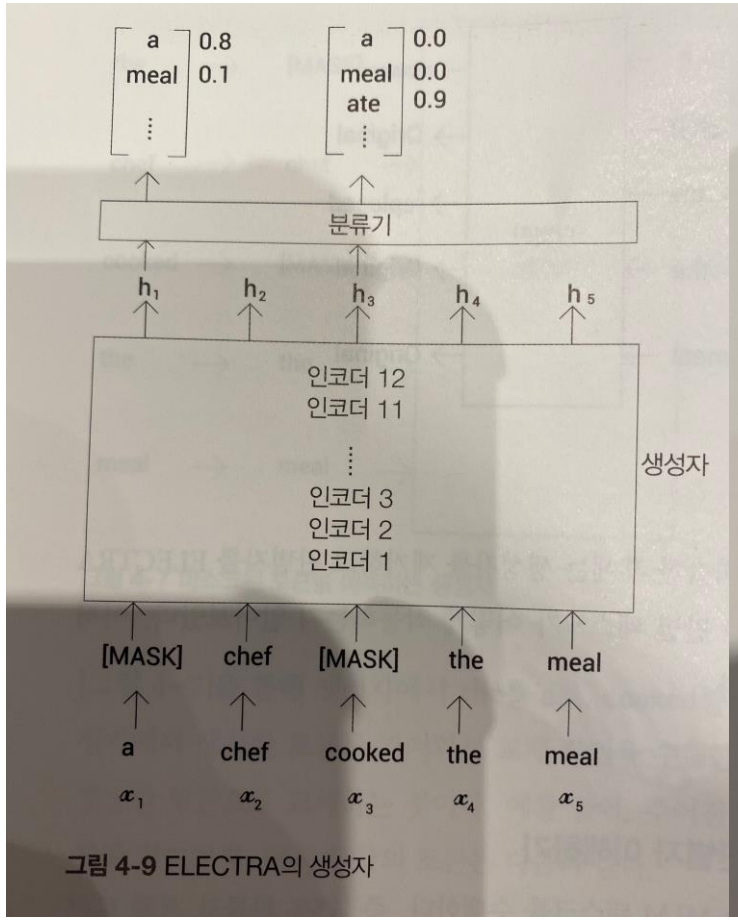


그림 4-9 ELECTRA의 생성자

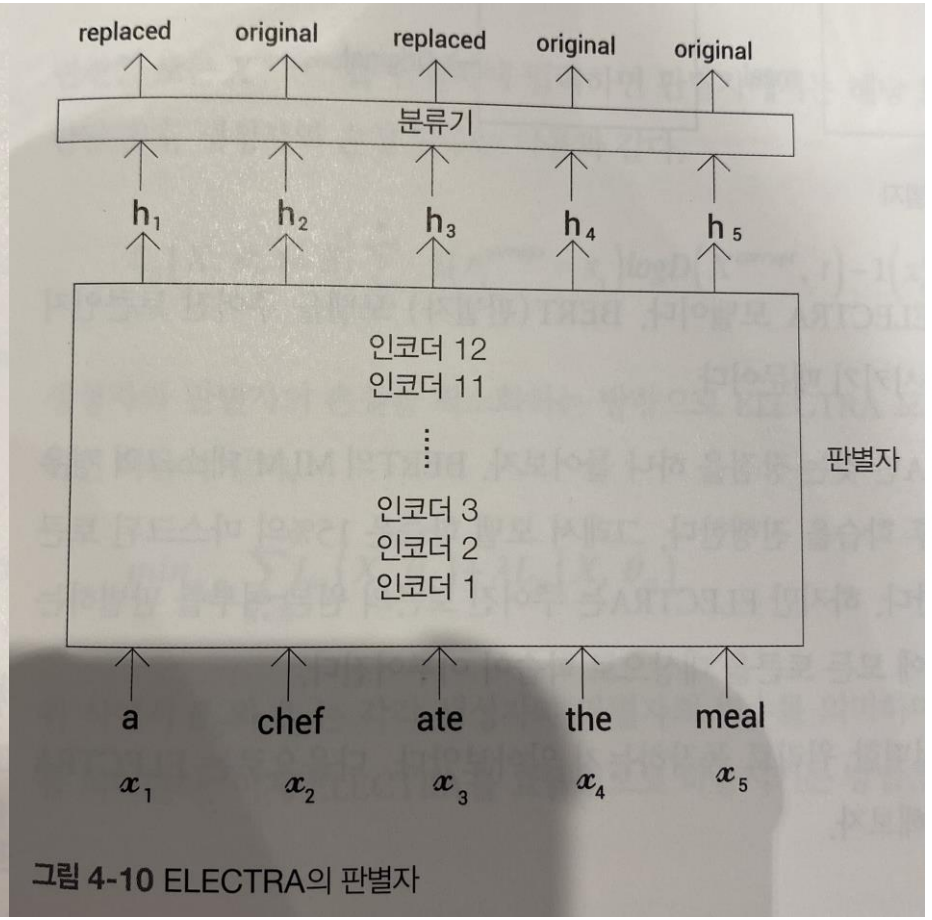


그림 4-10 ELECTRA의 판별자

$$D(X,t) = sigmoid(w^T h_D(X)_t)$$

SpanBERT로 span 예측

SpanBERT는 2개의 목표를 설정한다.

- MLM

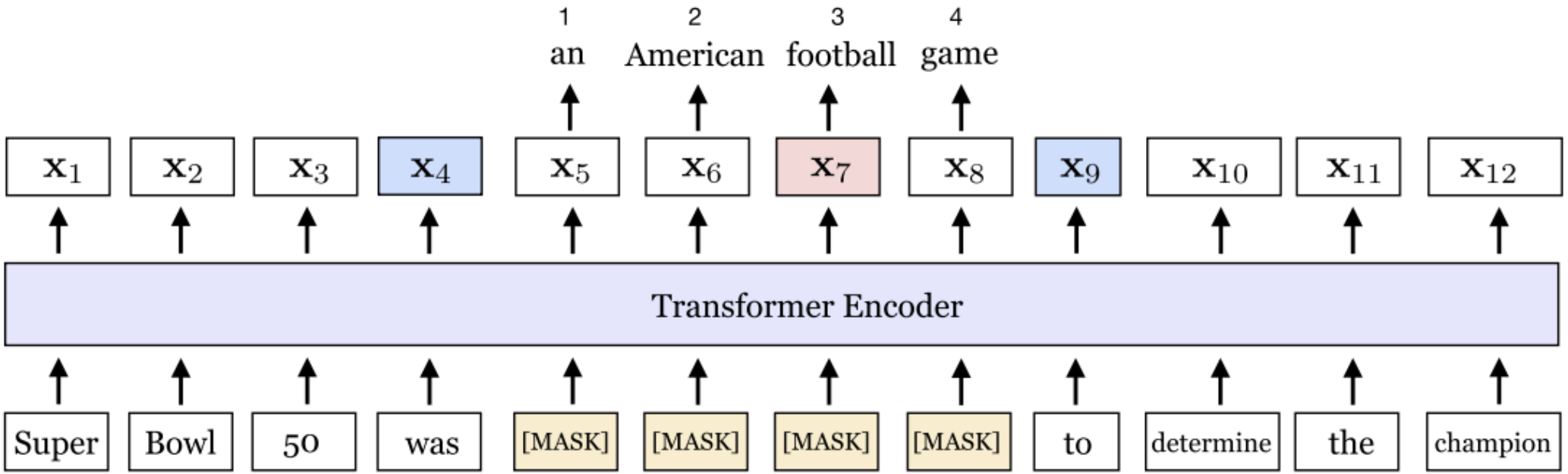
  마스크된 토큰을 예측하기 위해 해당 토큰의 표현만 사용

- SBO

  마스크된 토큰을 예측하기 위해 스팬 경계 토큰의 표현과 마스크된 토큰의 위치 임베딩 정보를 사용

SpanBERT의 아키텍처 이해하기

$$\mathcal{L}(\text{football}) = \mathcal{L}_{\text{MLM}}(\text{football}) + \mathcal{L}_{\text{SBO}}(\text{football})$$
$$= -\log P(\text{football} \mid \mathbf{x}_7) - \log P(\text{football} \mid \mathbf{x}_4, \mathbf{x}_9, \mathbf{p}_3)$$



SpanBERT 탐색

$z_i = f$(R_(s-1),R_(e+1),R_(i-s+1))

이 때 f(.)은 기본적으로 2개의 피드포워드 네트워크와 GeLU활성화 함수로 구성된다.

$h_0 =$ [R_s-1;R_e+1;P_i-s+1]

$h_1 = LayerNorm(GeLU(W_1 h_0))$

$z_i = LayerNorm(GeLU(W_2 h_1))$

마치며

- ALBERT는 BERT의 가벼운 형태이며, 크로스 레이어 변수 공유와 펙토라이즈 임베딩 변수화라는 두 가지 변수 감소 방법을 사용한다는 사실을 확인했다. 또한 ALBERT에서 사용하는 SOP태스크도 다뤘다. SOP는 모델의 목표가 주어진 문장 쌍이 뒤집어졌는지 여부를 분류하는 이진 분류 태스크이다.
- RoBERTa는 학습 시 MLM 태스크만 사용하며, 동적 마스킹 방법으로 큰 배치 크기로 학습한다. 토크나이저로 BBPE 토크나이저를 사용하며 사전 크기는 5만이다.
- ELECTRTA는 MLM태스크를 사전학습에서 사용하는 대신 교체한 토큰 판별이라는 새로운 태스크를 사전 학습에 사용했다. 교체한 토큰 판별 태스크는 [MASK]로 토큰을 마스킹하는 대신에, 토큰을 다른 토큰으로 교체하고 주어진 토큰이 실제 토큰인지 교체된 것인지는 예측하도록 모델을 학습시켰다.
- SpanBERT는 MLM과 SBO태스크를 사용해 학습을 진행한다.