

Cross-Encoder와 Bi-Encoder (feat. SentenceBERT)

항상해내는사람 김은기 · 2022년 6월 27일

팔로우

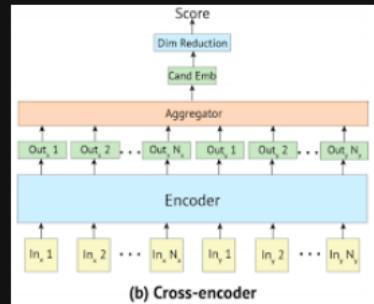
BERT TIL 딥러닝 머신러닝



TIL

▼ 목록 보기

12/16

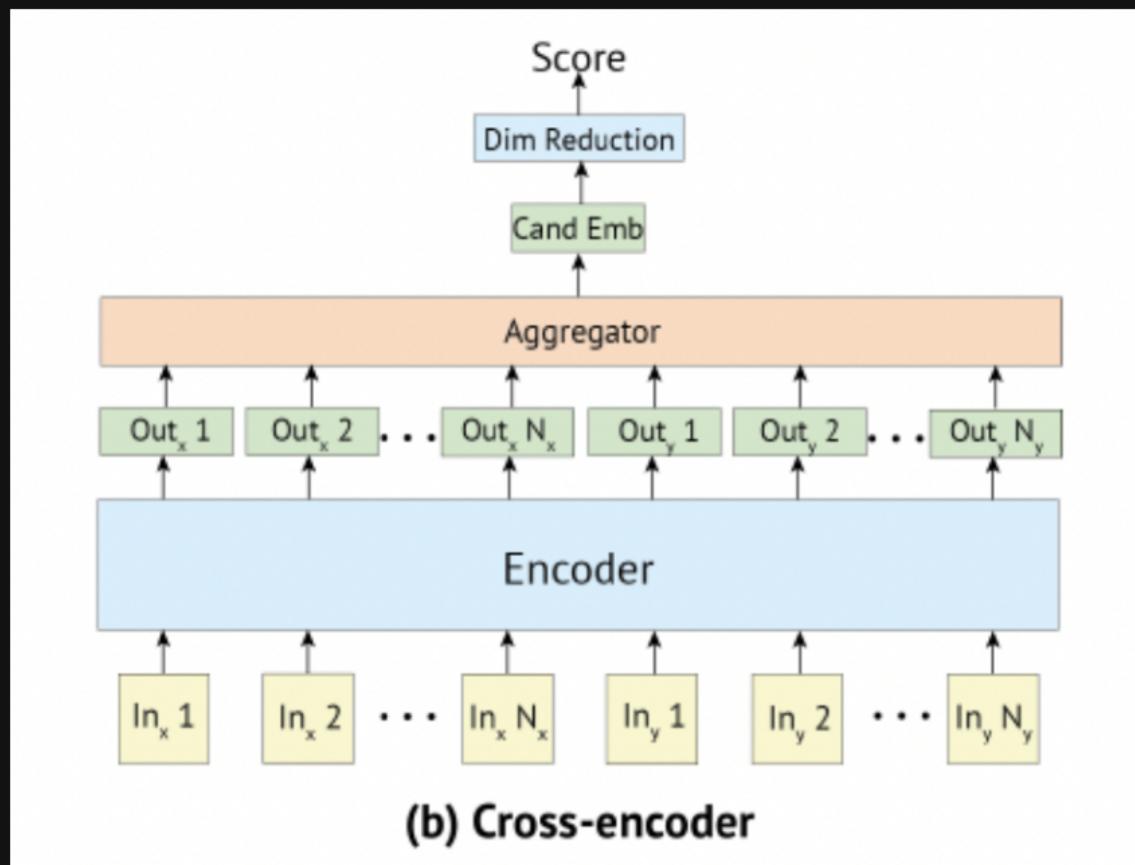


💡 개요

곰파다 프로젝트를 하면서 문장 간 유사도를 계산하는 모델을 구성할 때에 Bi-Encoder 구조 중 하나인 SentenceBERT를 사용해 학습시키고자 했다. 당시에 Cross-Encoder와 Bi-Encoder 방식을 사용할 때 성능 뿐만이 아니라 속도 차이도 매우 커있었다. 하지만 이러한 것이 어떤 원인에서 기인했는지 정확히는 알지 못했고 구조를 가져다 쓰기 바빴던 것 같다. 이에 해당 부분을 정리하고 다음 포스트에서는 1회독을 완료한 SentenceBERT 리뷰를 작성하고자 한다.

Cross-Encoder란?

전통적인 BERT 구조를 통해 문장 간 유사도를 구한다고 생각해보자. Classification 혹은 Regression을 하기 위해서는 하나의 Encoder 구조에 결과물을 내놓기 위한 layer를 추가로 쌓게 된다. 그리고 Encoder에 문장 Pair(혹은 Triplet)을 넣어주고 모델에서 나온 CLS 토큰을(혹은 Mean / Max Pooling을 거친 벡터를) 추가된 layer에 넣어주어 결과가 나타나도록 만들어준다.



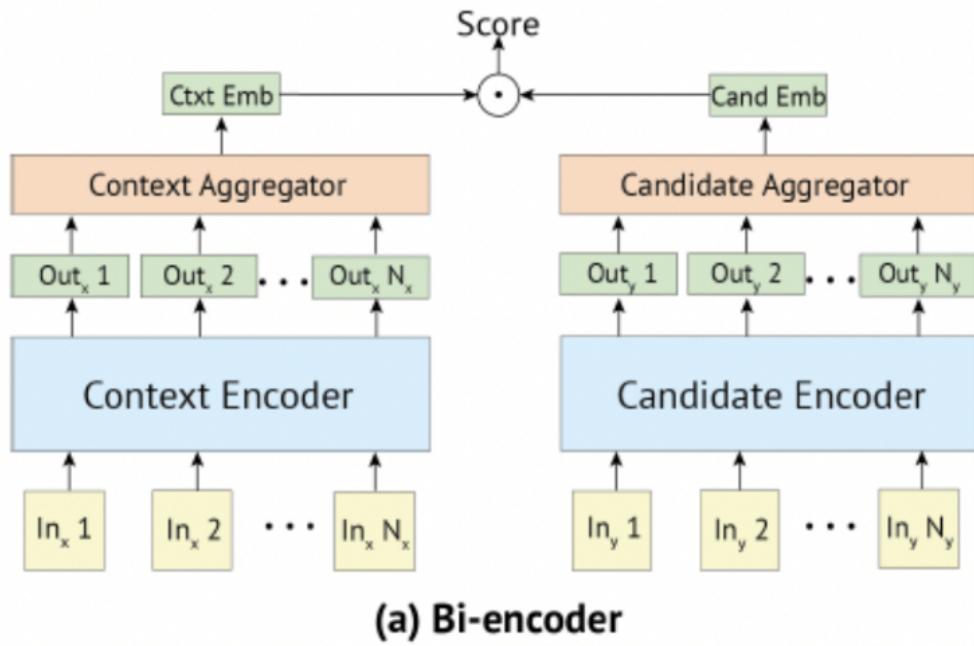
Pre-trained 모델을 가져와서 **fine-tuning**을 진행을 해보았다면 이처럼 두 개의 문장이 한 번에 [SEP]토큰으로 나누어진 구조가 익숙할 것이다. 하지만 여기에는 큰 문제점이 있다.

해당 모델에서 나오는 결과물은 결국 임베딩이 아니라 문장 Pair가 들어갔을 때의 연산 결과이다. 즉, [CLS] 토큰을 구해준 후에 그것을 추가적인 layer의 input으로 넣어주기 때문에 연산에 상당히 오랜 시간이 걸릴 수밖에 없다. 또한 Test 환경에서도 해당 구조를 사용하면 문장 pair가 들어갈 때마다 유사도를 전체 모델에서 구해주어야 하므로 매우 높은 연산이 필요하다. n개의 데이터에 대해서 자신을 제외한 n-1개와 비교를 하며 학습해야 하므로 총 $\frac{n(n-1)}{2}$ 의 연산이 필요하다.

또한 이는 문장 간의 유사도를 계산해주는 모델을 학습하는 것이기 때문에 문장 임베딩을 Encoder가 학습했다라고 보기는 어렵다. 단일 문장이 Encoder에 들어가서 그것을 의미적으로 적절한 위치에 임베딩시켜주는 것이 목표가 아니라 문장 유사도를 얼마나 잘 내보내는 모델이냐가 목표이기 때문이다.

Bi-Encoder란?

그렇다면 위의 단점을 보완하기 위해서는 어떤 방식을 사용하면 좋을까? 우리의 Encoder구조를 추가적인 layer가 아니라 정말 Encoder 그 역할에 충실한 모델을 만드려면 어떻게 해야할까라는 질문에 대한 대답은 Encoder 위에 추가적인 layer를 쌓지 않으면 된다. Encoder는 문장 임베딩에 충실하도록 만들고 layer가 아니라 추가적인 일부 구조로 결과물(여기서는 유사도)을 구해주는 구조이다. 이를 Bi-Encoder 구조라고 부른다.



A문장, B문장은 하나의 Encoder에 들어가는 것이 아니라 두 Encoder에 각각 들어가게 된다. 문장들은 각각 임베딩이 되며 그것들의 결과가 Score를 연산해주는 구조를 통해 연산이 된다. 위와 같은 구조를 사용하게 되면 미리 학습된 문장들(A, B문장)은 임베딩이 완료되어 저장되게 된다. Test 환경에서 문장들이 주어지면(C, D문장) 모델은 이미 저장된 임베딩을 바탕으로 빠르게 임베딩을 연산해주게 되고 이들에 대한 결과물은 외부적 구조로 연산만 해주면 된다.

이를 통해 매우 빠르고 각각의 역할이 명확한 구조가 형성이 된다. 하지만 Bi-Encoder 구조도 단점이 있다. 하나의 Cross Encoder를 사용하는 것보다 성능이 떨어진다는 것이다. 왜냐하면 Cross Encoder는 그 결과물 자체를 잘 내기 위해 구현된 모델이기 때문이다.

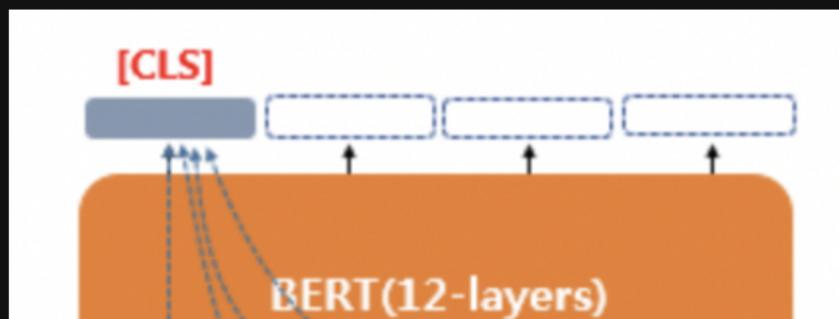
번외) SentenceBERT 란?

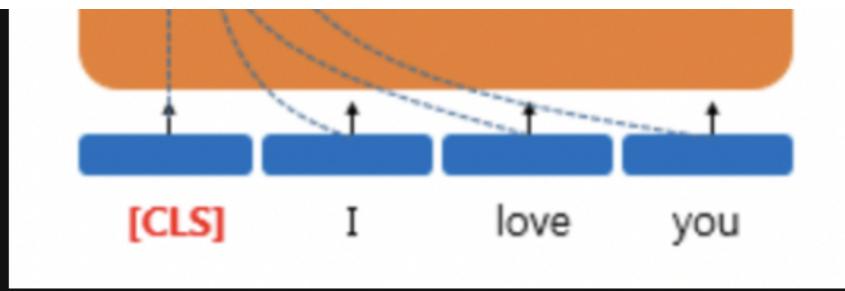
(1) 개요

BERT로부터 문장 벡터를 얻는 방법은 여러 가지가 존재

그중 대표적인 3가지 방법

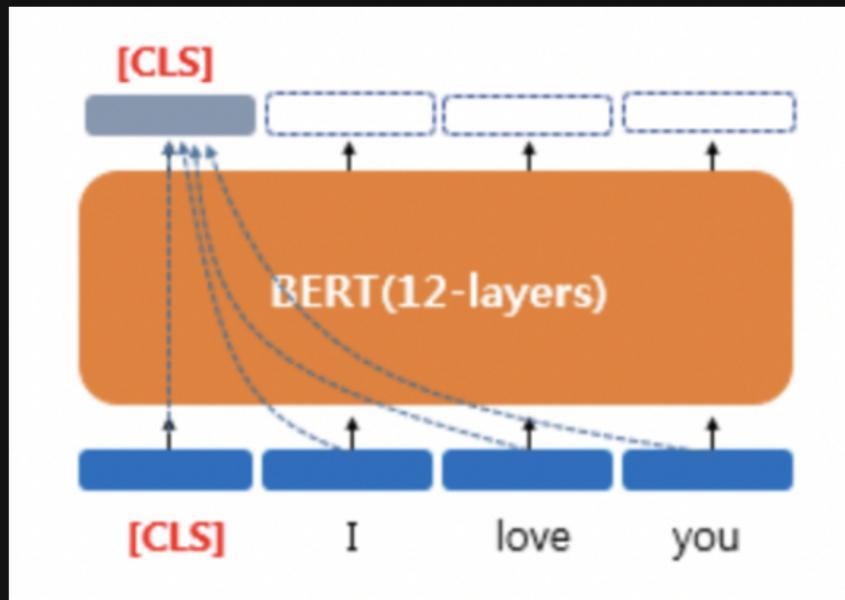
- 1) 첫번째 방법은 [CLS] 토큰의 출력 벡터를 문장 벡터로 간주하는 것





- BERT에서 사용하는 가장 대표적인 방법
- CLS 토큰이 입력된 문장에 대한 총체적인 표현을 압축하고 있다고 간주할 수 있다.

2) 두번째 방법은 BERT의 모든 단어의 출력 벡터에 대해서 평균 풀링을 수행한 벡터를 문장 벡터로 보는 것



이는 CNN 방식에서 사용하는 Pooling과 동일하다

3) 세번째 방법은 2번과 동일하지만 평균 풀링이 아닌 Max Pooling을 진행해주는 것

이때 평균 풀링을 하느냐와 맥스 풀링을 하느냐에 따라서 해당 문장 벡터가 가지는 의미는 다소 다른데, 평균 풀링을 얻은 문장 벡터의 경우에는 모든 단어의 의미를 반영하는 쪽에 가깝다면, 맥스 풀링을 얻은 문장 벡터의 경우에는 중요한 단어의 의미를 반영하는 쪽에 가깝다.

(2) 등장 배경

Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks라는 논문에서 처음 등장했다.

SBERT는 기본적으로 BERT의 문장 임베딩의 성능을 우수하게 개선시킨 모델

기본적인 구조는 동일하게 가져가되 문장 임베딩을 더욱 효과적으로 하고자 했다.

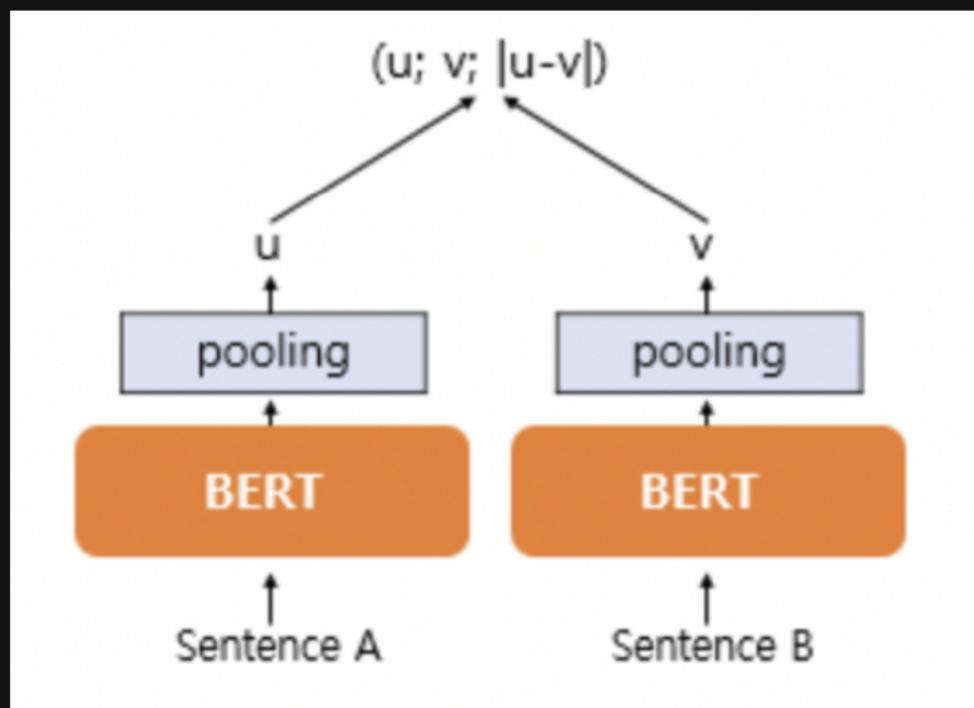
특히나 SentenceBERT(이하 SBERT)는 NLU 혹은 STS 부분에서 좋은 성능을 보이도록 fine-

tuning시키는 것을 목적으로 두는 모델

모델 자체가 이에 대한 Score 위주로 측정이 되어 있다.

(3) 구조

SBERT의 구조는 아래와 같다



해당 형식은 Bi-encoder 형식을 따르는 것을 볼 수 있다.

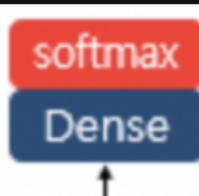
앞서 BERT의 문장 임베딩을 얻기위한 방식이라고 언급했던 평균 풀링 또는 맥스 풀링을 통해서 각각에 대한 문장 임베딩 벡터를 얻는다

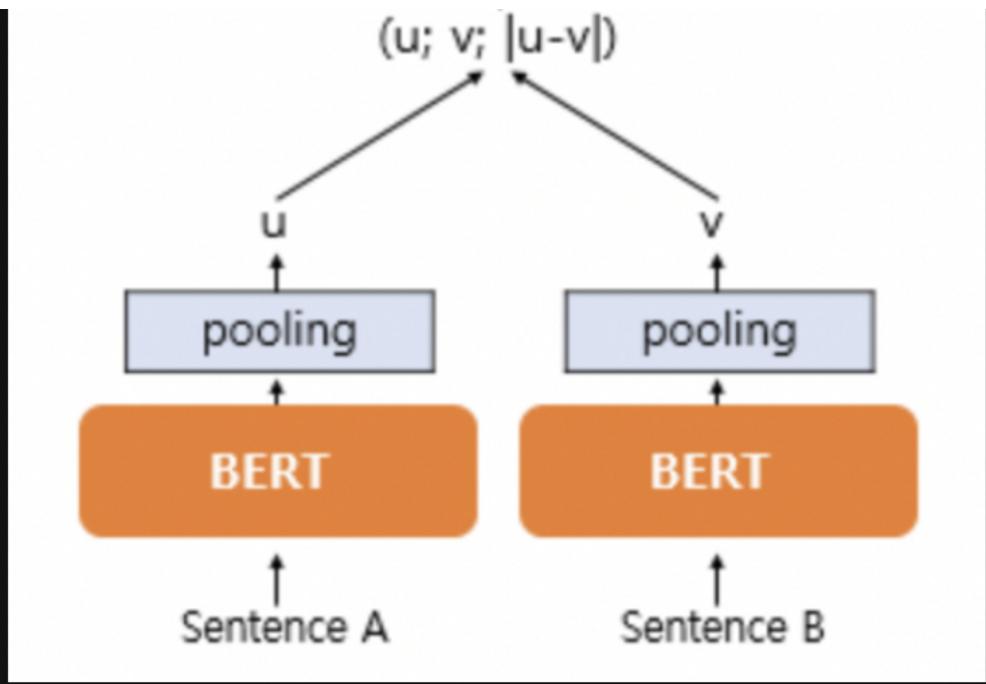
이를 각각 u 와 v 라고 할 때 u 벡터와 v 벡터의 차이 벡터를 구한다. 이 벡터는 수식으로 표현하면 $|u - v|$ 이다.

그리고 이 세 가지 벡터를 연결(concatenation)한다. 세미콜론(;)을 연결 기호로 한다면 연결된 벡터의 수식은 다음과 같다.

$$h = (u; v; |u - v|)$$

만약 BERT의 문장 임베딩 벡터의 차원이 n 이라면 세 개의 벡터를 연결한 벡터 h 의 차원은 $3n$ 이 될 것이다.





그리고 이 벡터를 출력층으로 보내 다중 클래스 분류 문제 혹은 **Regression**을 통한 STS 문제를 풀도록 한다.

NLI라면 3가지 class(contradiction, Entailment, Neutral)를 분류하도록 학습시키며 STS의 경우 0~5의 결과값을 내보내도록 학습시킨다.

Classification의 경우라면 분류하고자 하는 클래스의 개수가 k 라면, 가중치 행렬 $3n \times k$ 의 크기를 가지는 행렬 W_y 을 곱한 후에 소프트맥스 함수를 통과시킨다고도 볼 수 있다. 수식은 아래와 같다

$$o = \text{softmax}(W_y h)$$

이러한 구조는 STS(Semantic Textual Similarity)에서도 매우 효과적이다

- STS는 유사도를 0 ~ 5의 Regression 수치로 표현하는 데이터셋이다.

이것도 동일하게 문장 A와 문장 B 각각을 BERT의 입력으로 넣고, 평균 풀링 또는 맥스 풀링을 통해서 각각에 대한 문장 임베딩 벡터를 얻는다.

이를 각각 u 와 v 라고 하였을 때 이 두 벡터의 코사인 유사도를 구한다.

그리고 해당 유사도와 레이블 유사도와의 평균 제곱 오차(Mean Squared Error, MSE)를 최소화하는 방식으로 학습시킨다.

- 마치 **Regression**을 학습하는 것처럼

코사인 유사도의 값의 범위는 -1과 1사이므로 위 데이터와 같이 레이블 스코어의 범위가 0~5점이라면 학습 전 해당 레이블들의 값들을 5로 나누어 값의 범위를 줄인 후 학습할 수 있습니다.

이러한 모델을 학습하는데 있어서 선택에 따라

- 1) 문장 쌍 분류 태스크로만 파인 튜닝 할 수도 있고,
- 2) 무자 싸 히고 태스크로마 파이 트니 하 수도 있으면

1)을 학습한 후에 2)를 학습하는 전략을 세울 수도 있다.

- 해당 방식을 사용하려면 SBERT를 가져와서 fine-tuning시킬 것

참고문헌 :

- <https://wikidocs.net/156176>
- <https://velog.io/@nawnoes/Poly-encoders-architectures-and-pre-training-strategies-for-fast-and-accurate-multi-sentence-scoring>



항상해내는사람 김은기

프리미어와 IDE만 있다면 무엇이든 만들 수 있어

팔로우



이전 포스트

Transfer Learning과 Fine-Tuning

다음 포스트

How to implement Dynamic ...

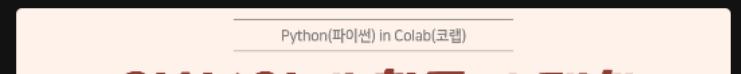


0개의 댓글

댓글을 작성하세요

댓글 작성

관심 있을 만한 포스트



Sentence-BERT: Sentence Embedding using Siamese BERT-Networks

[Paper Review] Sentence-BERT: Sentence Embedding usi...

Intro 문장 간(혹은 문서 간) 유사도 분석에서 좋은 성능을 내고 있는 Sentence-BERT에 대해 알아보려고 한다. 논문 주제는 Sentence-BERT: Sentence Embedding using Siamese BERT-Networks이며, 최근 성능이...

2021년 10월 10일 · 0개의 댓글

by jaehyeong.an_

3



트랜스포머(Transformer)와 어텐션 매커니즘(Attention Mec...

1. 배경(Background) 2017년에 Attention is All You Need라는 논문이 나온 이후로 어텐션 구조는 최초에 제안되었던 NLP 분야는 물론 Computer Vision 분야와 Time Series 분야까지 다양한 분야에서 적용...

2022년 12월 14일 · 4개의 댓글

by 환공지능

22

[딥러닝] 언어모델, RNN, GRU, LSTM, Attention, Transfor...

언어모델에 대한 기초적인 정리

2021년 8월 22일 · 1개의 댓글

일상/연애 한국어 대화 BERT로 이진 분류 모델 만들기

이론편

veloo.io/@seolini43

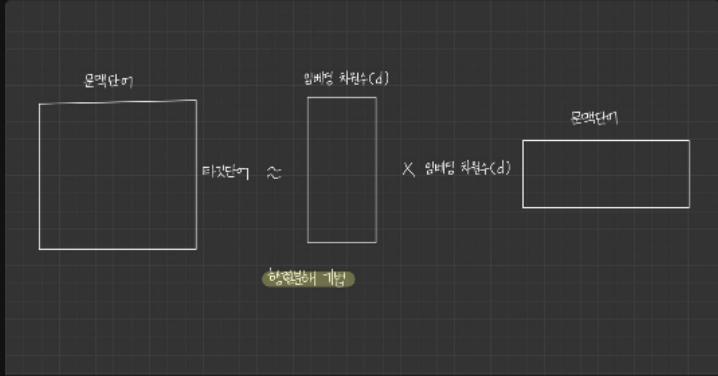
[파이썬] 일상/연애 주제의 한국어 대화 'BERT'로 이진 분류 ...

BERT를 이용한 프로젝트 - 이론편입니다!

2021년 6월 22일 · 0개의 댓글

by Seolini

6



임베딩(Embedding)이란?

임베딩이란 자연어처리에서 사람이 쓰는 자연어를 기계가 이해할 수 있도록 숫자형태인 vector로 바꾸는 과정 혹은 일련의 전체 과정을 의미합니다. 단어나 문장 각각을 벡터로 변환해 벡터 공간(Vector space)으로 ...

2022년 8월 7일 · 0개의 댓글

by AI Scientist를 목표로!

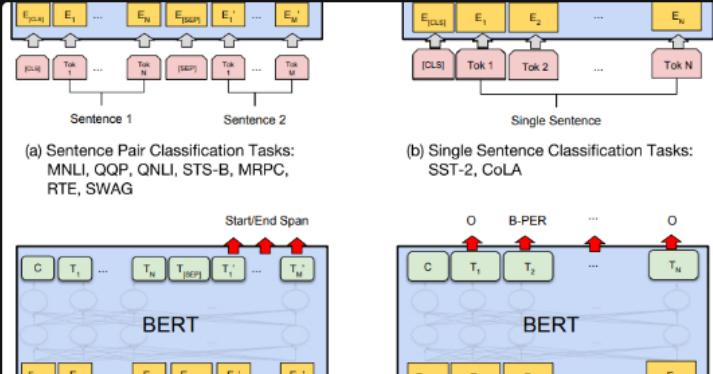
4

Fine-tuning SBERT by sentence-transformers

[Basic NLP] sentence-transformers 라이브러리를 활용한 ...

Intro 이전 포스트에서 소개한 SentenceBERT를 어떻게 학습하는지 논문 및 sentence-transformers 공식 깃헙을 기준으로 몇 가지 방법을 알아보고 어떤 방법이 가장 좋은 성능을 내었느지 소개하고자 한다. 1. SBERT...

2022년 3월 1일 · 7개의 댓글



BERT

<https://keep-steady.tistory.com/19> BERT설명내용-

[BERT 참 고링크 1\) Illustrated Bert: <http://jalammar...>](https://github.com/chullhwan-song/Reading-Paper/issues/202)

2020년 8월 7일 · 0개의 댓글



Attention (additive)	$n^2 \cdot d$	$n^2 \cdot d$
Recurrent	$n \cdot d^2$	$n \cdot d$
Convolutional	$n \cdot d^2$	$n \cdot d$
Multi-Head Attention with linear transformations. For each of the h heads, $d_q = d_k = d_v = d/h$	$n^2 \cdot d + n \cdot d^2$	$n^2 \cdot h + n \cdot d$
Recurrent	$n \cdot d^2$	$n \cdot d$
Convolutional	$n \cdot d^2$	$n \cdot d$

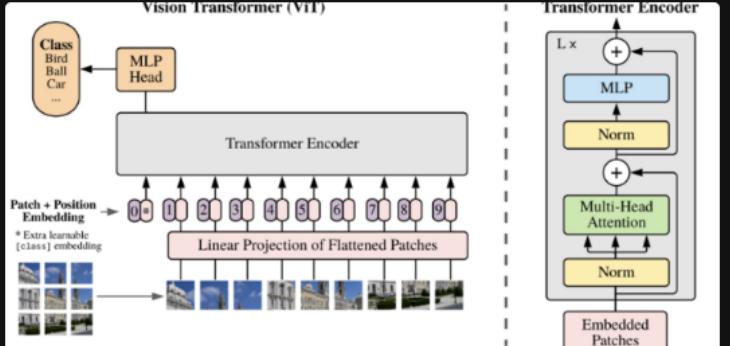
Linear Transformer

투박스 컨퍼런스 주제로 무려 pretrain 모델 만들기를 선정했다... 아직도 이게 맞나 싶긴 하지만 gpu 충분하고 다들 충분히 모델이나 관련 지식이 충분하니까 학부생으로서 할 수 있는 가장 좋은 프로젝트가 될 수 있지...

2021년 11월 5일 · 0개의 댓글



BERT를 활용한 한국어 문서 추출요약 보



Vision Transformer(ViT) 논문 리뷰

ViT(비전 트랜스포머) 논문 읽기

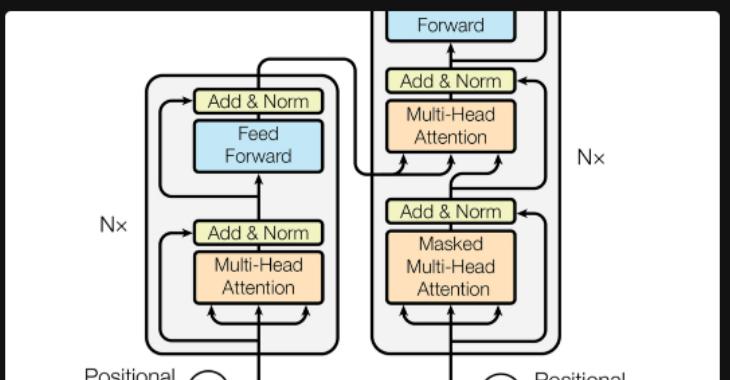
2023년 4월 9일 - 2개의 댓글



9. Multi-modal Learning

사람이 여러 개의 감각기관을 활용하여 문제를 해결하듯이 딥러닝에도 이를 적용해보자. 한 type의 데이터가 아닌 다른 특성을 갖는 데이터 type들을 같이 사용하는 활용하는 학습법 e.g.) Text, audio data 각각의 데이터들은 서로 다른 타입으로 자료구...

2022년 5월 13일 · 0개의 댓글



[논문 스터디] Week 4-5] Attention is All You Need

딥러닝 기반의 여러 요약 모델을 공부하고 있던 중, 한국어 데이터로 학습한 추출요약 모델이 있으면 좋겠다 싶어서 만들어 보았습니다. 보노보노는 뭔가 허전해서 넣었습니다. 감사합니다.

2021년 4월 10일 · 21개의 댓글

 by ragoon

♥ 17

2021년 3월 25일 · 1개의 댓글

 by 김재희

♥ 2

