Question Answering with Keyword

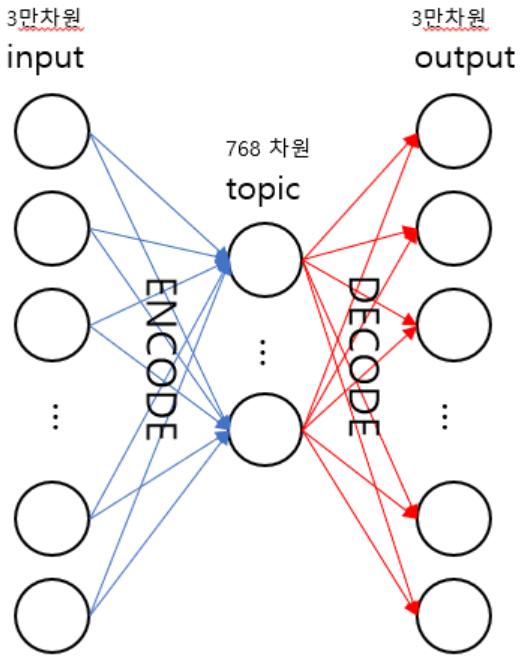
Team : 욕심 많은 민트 초코
Team member : 김정학(2015147557), 박성현(2015147565), 유현석(2015147533)

1. 기존 연구 소개

기존 연구는 BM25 + Topic, Bert + Topic 모델로 이루어져 있다.

① BM25 + Topic

BERT를 사용하지 않고 쿼리(질문)와 도큐먼트(문서)를 매칭해주는 모델은 단어들과 각 단어들이 가지고 있는 토픽을 이용한다. 이 정보들을 BM25 알고리즘을 통해서 문서와 쿼리를 매칭해준다. 주어진 데이터는 쿼리와 문서들로 이루어져 있고 각 단어들이 어떤 토픽을 가지고 있는지는 딥러닝을 통해서 학습을 한다. 이를 위해서 Autoencoder를 사용한다. 쿼리와 문서를 나타내는 방식은 두가지가 있다. 첫번째는 binary 방식이고 두번째는 tf-idf 알고리즘의 idf를 이용하는 것이다. 예를 들어 총 5개의 단어를 가지고 있는 코퍼스속에서 1,1,2,4번 단어로 이루어져있는 문서같은 경우는 바이너리로는 (1,1,0,1,0), idf 방식으로는 다른 문서에서 빈도수에 따라서 바이너리에서 1인 요소에 idf를 곱해준다. idf 방식의 경우에는 다른 문서에서 출현하지 않을수록 값이 높아진다. 주어진 데이터속에서 전체 단어의 수는 총 3만개이다. 따라서 오토인코더는 3만차원의 인풋을 받고 이를 인코더를 통해서 압축화를 하고 압축화되어 있는 텐서를 디코더를 통해서 최대한 원래의 인풋으로 복원을 해야 한다. 이를 도식화하면 아래와 같다.



모델은 (쿼리, 정답 문서, 거짓 문서)를 하나의 데이터로 사용해서 단어가 어떤 토픽을 가지고 있는지를 학습한다. 따라서 쿼리, 정답문서, 거짓문서를 각각 위 모델을 통과시킨후 그 결과값으로 loss를 구해야한다. Z_q를 쿼리를 인코더만 통과시켰을 때의 값, Z_p 정답문서를 인코더만 통과시켰을 때의 값, Z_n을 거짓문서를 인코더만 통과시켰을 때의 값이라고 하자. Loss의 값은 총 3가지 값의 합으로 구해진다. 첫번째로 Z_q Z_p Z_n을 디코더를 통과시켰을 때 인풋과의 차이점을 사용한다. 이는 인풋이 토픽으로 압축이 잘되기 위함이다. 두번째로는 (Z_q-Z_n) - (Z_q-Z_p)을 사용한다. 이는 쿼리와 정답문서의 토픽 추출이 비슷해지고 쿼리와 거짓문서의 토픽이 멀어지게 하기 위함이다. 마지막으로 Z_q, Z_p, Z_n의 각각 768차원의 값이 1로 수렴하게 하기위한 loss를 사용한다. 이는 학습을 안정화시키기 위함이다. 위 방식으로 모델은 문서와 쿼리의 토픽을 768차원으로 뽑아낼 수 있다. 모델을 학습시킨 후에 단어가 나타내는 토픽을 찾는 방법은 다음과 같다. decode부분의 768차원의 뉴런에서 3만개로 매칭시켜주는 각 가중치를 고려한다. 예를 들어 1번째 뉴런은 3만개의 가중치를 가지고 있는데, 여기서 topk(=10)번째로 값이 큰 가중치까지 정보를 저장한다. 즉 1번째 뉴런에서 가중치가 높은 상위 10개의 단어들이 1,3,5,6,...,11(10개)의 단어들이라면 반대로 1번째 단어는 1번째 토픽을 가진다고 할 수 있다. 이방식으로 각 단어들에 토픽을 연결시켜준다. 이 결과로 어떤 단어는 어려가지 토픽을 가질 수 있고 어떤 단어는 토픽이 아예 매칭이 안될 수도 있다. 이후에 각 문서의 키워드 뒤에 각 단어들이 가지고 있는 토픽을 단순히 더해 데이터를 변형시킨다. 변형된 데이터로 BM25 알고리즘 매칭을 진행하였다.

② Bert + Topic

Bert를 사용한 모델은 먼저 기본 Bert의 동작 방식에 따라 주어진 데이터 Input을 임베딩하는 과정을 거치게 된다.

1) Token Embedding :

각 단어는 word vector table을 모델 입력으로 설정하여 1차원 벡터로 변환된다. 단어를 더 세밀한 단어 조각으로 자르는 것으로 목록에 없는 단어를 해결한다. 예를 들어 playing을 play + ing으로 자르는 방법이다.

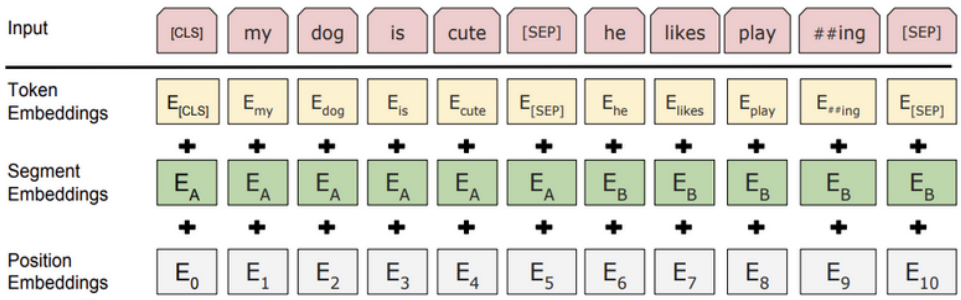
아래의 그림과 같이 "I like dog"를 입력하면 Token Embeddings layer 보내기 전에 두 개의 특수 토큰이 [CLS]의 시작과 [SEP] 끝에 삽입된다. 그 후, 각 단어를 768 차원 벡터로 변환된다.

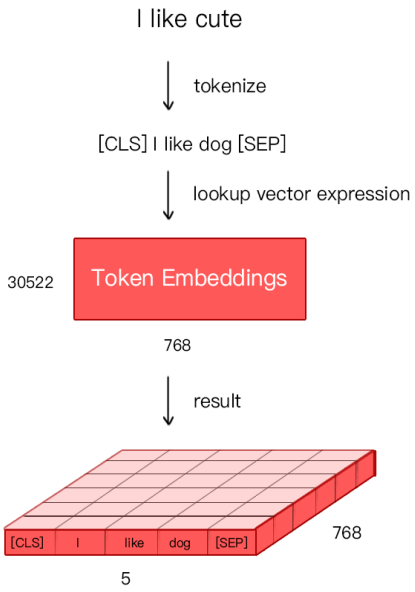
2) Segment Embedding :

Segment Embedding은 토큰화시킨 단어들을 다시 하나의 문장으로 만드는 과정이다.. 이를 통해서 두 텍스트가 의미적으로 유사한지 아닌지 판단할 수 있게 해준다. 한 쌍의 두 문장을 간단히 연결한 다음 첫 번째 문장의 각 토큰에 0을 할당하는 것이고 두 번째는 각 토큰에 1을 할당하는 것이다.

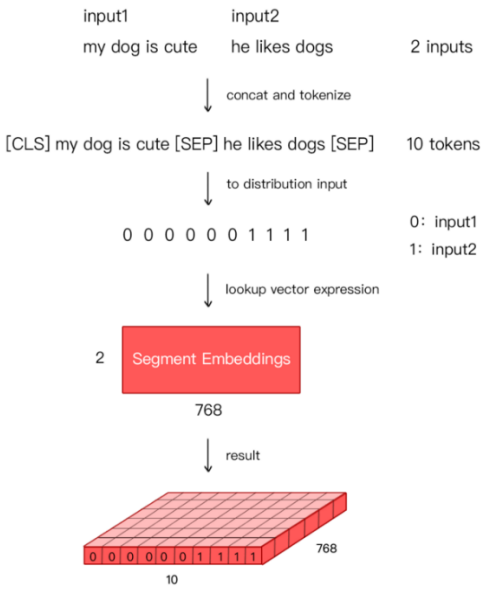
3) Position Embedding :

텍스트의 위치에 따라 의미가 달라지기도 하고 그 자체로도 의미가 달라지기도 한다. 따라서 단어 그 자체가 아닌 의미에 따라서 벡터를 생성하게 도와준다.

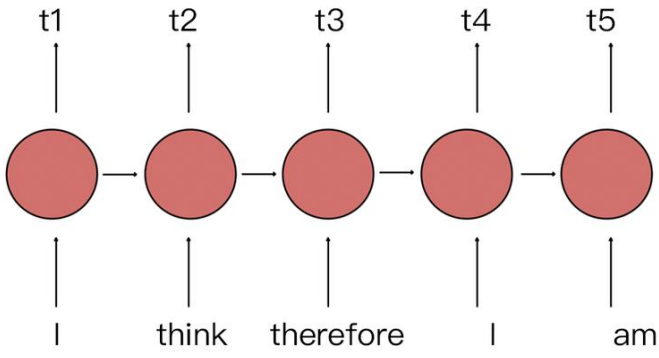




<Token Embedding>



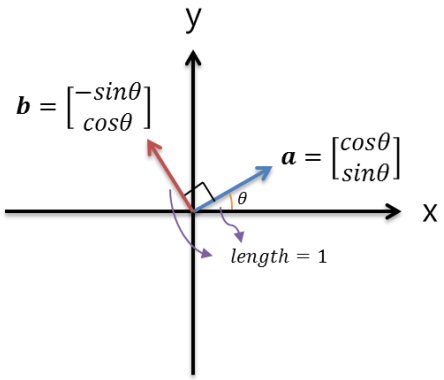
<Segment Embedding>



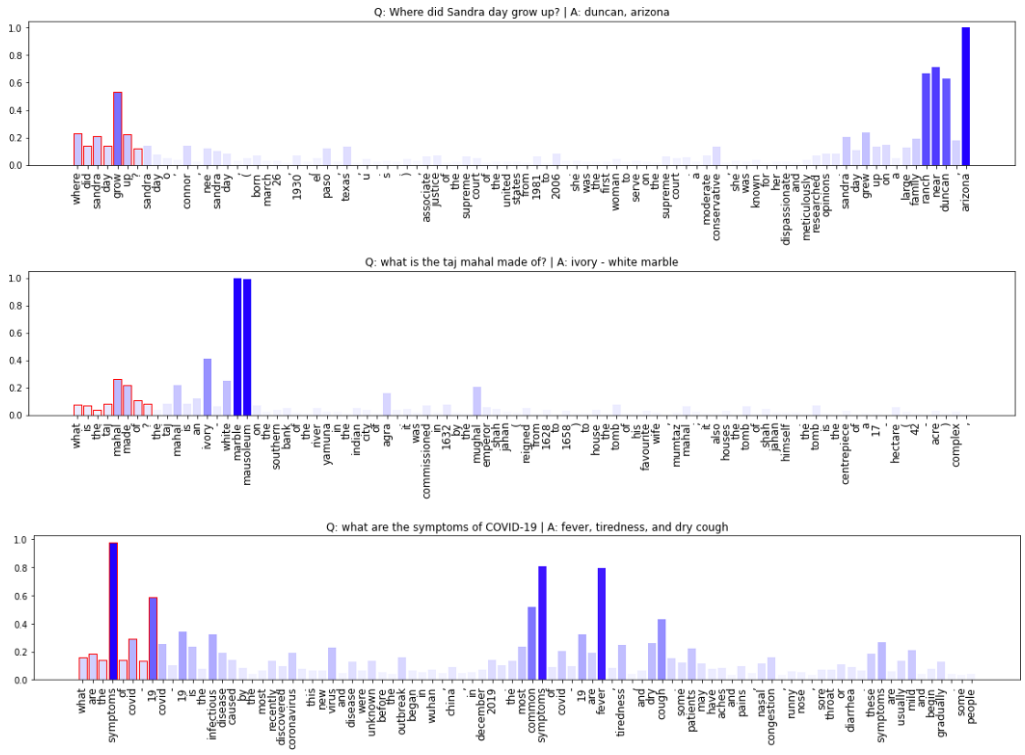
<Position Embedding>

기존 모델은 query 부분, 관련 문서 부분(positive), 관련되지 않는 문서 부분(negative)을 각각 임베딩하여 훈련시킬 데이터를 모두 인코딩을 한 후, pre-training을 진행한다.

위와 같은 과정을 통해서 얻어지는 총 768차원 vector에 위에서 설명한 BM25에 사용한 Topic의 768차원을 합쳐 새로운 vector를 만든다. 그 과정에서 orthogonal loss을 이용하여 Bert만을 사용했을 때 vector와 Topic에서 얻은 새로운 vector에 모두 적합할 수 있도록 학습을 시킨다.



최종 결과물은 아래의 형태와 같게 각각의 단어들에 대한 값으로 산출되어지게 된다. 즉 관련도가 높은 단어일수록 주위 단어들에 비해서 높은 값을 갖게 된다. 문장을 뽑아낼 때에는 관련도가 높은 시작점부터 다시 낮아지는 끝나는 점까지 설정해주면 된다.



2. 기존 연구 문제점

① BM25 + Topic

기존의 BM25 모델로 토픽을 학습했을 때, 토픽이 치우쳐지는 현상을 발견하였다. 예를 들면, 기존의 모델은 '3'이 426개, 'two'가 311, 'is'가 227개의 토픽을 가지고 있다. 즉 많이 사용되는 단어들이 토픽으로 뽑히는 경향이 있었다. 상대적으로 희귀한 단어들은 아예 토픽이 없는 경우가 대다수였다. 따라서 토픽을 이용해서 데이터를 변형시킬 때, 토픽이 주는 정보가 거의 없을 수가 있다. 기존 BM25모델은 토픽과 키워드를 적절히 사용해서 매칭을 시키는 것이 목표였지만, 사실은 키워드만을 가지고 매칭을 시키고 있을 수가 있었다. 또 다른 문제로는 토픽을 단순히 문서 뒤에 더하기만 한다는 것이다. 각 단어가 나타내는 특성이 다를 수 있는데, 이를 단순히 더하는 것은 특성을 다 활용하지 않는 것이라고 결론을 내렸다.

② Bert + Topic

기존 Bert 모델은 확실히 BM25 모델보다는 월등한 성능을 보여주었다.

	기존 BM25 모델	기존 Bert 모델
Score (R@1000)	94.20	99.97

[R@1000(recall@1000, 재현률) : 관련순으로 1000개의 문서를 ranking을 매겼을 시 실제 관련 문서 중 1000등 안에 들어간 비율]

그러나 1000등 안에 들어가는 것을 가지고 평가 지표를 삼았기 때문에 99.97 수치이지만 성능 향상의 여지는 충분히 보였다.

	R@1000	R@500	R@100
Score	99.97	99.70	94.39

실제로 평가 ranking 범위를 500, 100까지 줄이니 94.39까지 떨어졌다.

그리고 1000개의 문서 안에서 답의 rank를 세부적으로 보았을 때 BM25의 답 순위가 우위를 보이는 경우가 확인됐다. Bert가 문맥 상의 자연어 처리 성능에 대해 뛰어나지만 키워드 매칭에선 약간 부족한 성능을 보인다고는 이미 알려져 있고 실제로 위 Bert모델의 경우를 분석한 결과 키워드적으로 BM25에 비해 부족한 부분이 있어 일부 성능에서 낮은 순위를 보인다고 예측할 수 있었다.

3. 연구 방법

개발 환경 : 구글 colab, jupyter notebook

① BM25 + Topic

기존 BM25모델의 토픽 치우침을 개선하기 위해서 topk의 수를 높이는 방식을 도입했다. 하지만 단순히 topk를 높이게 되면 발생하는 문제가 크게 두가지가 있다. 첫번째로는 topk가 높아지면 토픽을 더할 때 데이터의 크기가 너무 커지게 된다. 키워드보다 토픽의 길이가 더 길어져 토픽으로만 문서매칭을 할 수가 있다. 또 다른 문제로는 희귀한 단어들이 토픽을 가질 수 있게도 되지만 기존에 토픽을 많이 가지고 있던 단어들은 심하면 768개의 토픽을 모두 가질 수 있다. 따라서 토픽의 개념이 무의미해진다. 이를 해결하기 위해서 topk의 크기를 늘이지만 각 단어가 가지고 있는 토픽수를 임의로 줄이는 방식을 사용했다. 예를 들어 'is'단어가 700개의 토픽과 매칭이 된다면 700개를 다 사용하지 않고, 700/(log(700)개만 사용한다. 또한 만약 'water'가 5개의 토픽과 매칭이 된다면 5개를 다 사용한다. 즉 min(x,x/(logx))개 만큼 사용해서 토픽이 많으면 줄이고 적으면 그대로 사용한다.

토픽을 그대로 더하는 것을 해결하기 위해서 다음과 같은 가정을 하였다. 토픽을 적게 포함하고있는 단어는 그 토픽에 가중치를 많이 주어 토픽을 뚜렷하게 해야하고, 토픽을 많이 포함하고 있는 단어는 토픽보다 키워드에 가중치를 주어서 같은 토픽을 많이 가지고 있는 문서보다 키워드에 더 가중치를 주어야 매칭이 더 잘 될 것이라는 가정이다. 따라서 각 단어가 포함하고 있는 토픽의 수에 따라서 토픽수가 적으면 토픽을 두번 더하고 토픽수가 많으면 키워드를 두번 더하는 방식을 도입하였다.

② Bert + Topic

Test 과정은 633개의 쿼리와 그 쿼리의 관련 문서 번호 답들을 가지고 BM25와 Bert 모델에서 1000순위의 관련 문서를 뽑은 답들과 비교해 성능을 평가한다. 단순히 평가 지표만을 확인하지 않고 실제로 어떤 쿼리에서 성공을 했고 실패를 했는지 따로 각각 분리를 했고 관련 문서 답들의 순위 평균을 내 실제로 어떤 쿼리에서 순위적으로 더 높은 성능을 냈는지 쉽게 확인할 수 있게 데이터를 정제했다. 그 후 데이터를 가지고 키워드에 보다 좋은 기능을 보일 것으로 예상되는 BM25와 Bert를 비교하기 위해 각각의 토큰에 대한 idf 값으로 분류 및 분석을 진행했다.

4. 결과

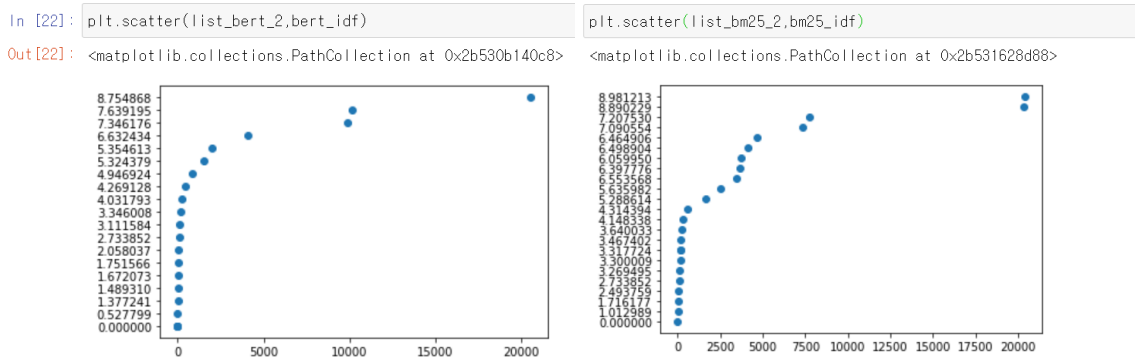
① BM25 + Topic

기존 BM25에 새로운 알고리즘을 도입했더니 다음과 같은 결과가 확인되었다.

	기존 모델	제안하는 모델
점수(R@1000)	94.2	94.8

기존의 모델보다 향상된 성능을 보이는 것을 확인할 수 있다.

② Bert + Topic



Bert보다 BM25에서 더 정확도가 높았던 질문들은 보다 높은 idf 값을 갖는 token들을 많이 포함하고 있었다. 즉 키워드와 idf가 높은 단어들이 bm25에서 bert보다 정확도를 높여준다는 사실을 확인할 수 있었다.

5. 향후 계획

현재는 토픽수가 많으면 줄이고 적으면 그대로 사용하는 함수에 임의로 log함수를 사용했지만, 앞으로 적합한 함수를 도입해야할 것이다. 또한 토픽에 가중치를 줘야하는지 키워드에 가중치를 줘야하는지에 대해서 임의로 분위수 개념을 사용하였다. 마찬가지로 이를 위해서 더 적합한 방식 혹은 함수를 조사 및 도입 할 것이다.

몇 개 예시를 갖고 idf값을 기준으로 분석을 했을 때 bert와 bm25에서 각각 결과가 더 좋았던 질문들에서 차이가 보였다. 이를 전체 데이터로 확대 시킴으로써 일반화하는 과정을 진행할 예정이다. 그 와중에서 idf 값처럼 bm25와 bert를 구분 지을 수 있는 몇 가지 특징들을 더 분석하여 찾을 예정이다. 그 후 그 해당 기준에 따라 bm25에서의 순위와 bert에서의 순위에 가중치를 주어서 최종적인 순위를 만들어 나갈 예정이다.

주차	목표
7	베이스 모델에 키워드 매칭 적용 방법 리서치
8	모델 키워드 적용 구현, 기존 모델과 비교 및 중간 보고서 준비
9	중간발표
10	피드백으로 모델 개선 및 최종 모델 구현 방안 수립
11	알고리즘 최적화 진행
12	다른 모델과 비교 및 앙상블 방법 적용
13	최종 모델 구현 및 결정
14	최종 보고서 준비 및 발표
15	졸업 전시회 준비

6. 팀원 구성 및 역할

김정학(조장) : bm25 성능 향상 알고리즘 구상 및 코드 구현

박성현(조원) : bm25, bert 모델 결과 데이터 정제 및 분석

유현석(조원) : 정제 데이터 세부 분석 및 결과 시각화

7. 참고 자료(그림 출처)

https://developpaper.com/interpretation-of-the-paper-the-theory-of-bert/

https://sacko.tistory.com/4

https://towardsdatascience.com/explainbertqa-687abe3b2fcc