



2주차 1조

팀원: (강용진), (조선빈), (조현진)

```
lookup.KeyValue  
f.constant(['en  
tf.constant([G  
ce = tf.lookup.StaticV  
init,  
num_oov_buckets=5)  
  
lookup.StaticVocabular  
initializer,  
num_oov_buckets,  
lookup_key_dtype=None  
name=None,
```

베이스 코드의 흐름

베이스 코드의 흐름

데이터 전처리 부분

1. 주어진 test file에서 prompt와 유사도가 큰 문장 상위 3개를 위키에서 가지고옴 그리고 그 문장이 포함된 parquet 파일을 가지고 옴
2. wikipedia_file_data에 위에서 뽑은 문장이 포함된 parquet파일과 넣어줌 파일에서의 문장 위치(id)를 넣어줌
3. wiki_text_data에 1번에서의 상위 문장 3개를 넣어줌
4. process_documents 함수를 통해서 wiki_text_data를 잘 다듬어 줌(processed_wiki_text_data)
5. processed_wiki_text_data를 임베딩한 것 -> wiki_data_embeddings
6. 기존 prompt와 보기 ABCDE를 합쳐주는 col 만듦 -> prompt_answer_stem
7. prompt_answer_stem을 임베딩함 -> question_embeddings
8. contexts에다가 5단어 겹치는 애들(위키에서 뽑은 문장들) 넣어줌(faiss로 효과적으로 찾아줌)
9. 이 중에서 1750 단어를 사용할 것임
10. 이제 trn('prompt', 'A', 'B', 'C', 'D', 'E', 'answer')의 value를 토큰화 해줌
11. 이 토큰화 된 것을 기반으로 predict를 함

시도



시도

1. 9번에서 1750단어만 사용했는데 2000단어 사용
0.807->0.809
2. 유사도 상위 문장 3개 사용했는데 4개로 늘림
0.809->0.813
3. 'NUM_SENTENCES_INCLUDE'를 7개로 늘리기
0.813->0.812
4. 9번에서 1750단어만 사용했는데 2250단어 사용
성능 변화 없음
5. 유사도 상위 문장 3개 사용했는데 5개로 늘림
0.812->0.818

kaggle 제출

Kaggle 제출

나름 100등 증가 시켰다! WOW~

1294	Art Girl		0.807	5	1mo
1182	Hyunjin123		0.818	12	2h