# 9월 13일

어제 LG 취업 상담을 진행했기 때문에 새벽 늦게까지(3~4시)까지 진행하는 바람에 그 여파가 지속되었다.

사실 9시반쯤 일어나 어젯밤에 돌려놓았던 DPR 결과를 확인하기 위해 깼다. 일단 코드 자체 에러는 없었지만

결과데이터를 뜯어보니 index mapping하는데 문제가 있었다.

Faiss 라이브러리를 사용했는데 index변수를 passage 마다 불러와야했는데 처음에 불러오고나서 similarity를 계산하니 해당 passage에 없는 index가 결과로 나와서,, index out of range 에러가 발생했다..

그래서 수정한 코드는 다음과 같다.. → 이것으로 2시간정도 ㅈㄹ함…

```
import faiss
import numpy as np
from transformers import DPRContextEncoder, DPRContextEncoderTokenizer, DPRQuestionEncoder, DPRQuestionEncoderTokenizer
import torch
import numpy as np
import gzip
import pickle
import time
import statistics
import sys
def calculate_dpr_similarity_with_faiss_gpu(model_name_or_path, qa_context, cycle_passage):


    # Load the DPR models and tokenizers
    context_model = DPRContextEncoder.from_pretrained(model_name_or_path)
    context_tokenizer = DPRContextEncoderTokenizer.from_pretrained(model_name_or_path)
    question_model = DPRQuestionEncoder.from_pretrained(model_name_or_path)
    question_tokenizer = DPRQuestionEncoderTokenizer.from_pretrained(model_name_or_path)

    top_documents = []

    for query_list, doc_list_for_query in zip(qa_context, cycle_passage):
        if not doc_list_for_query:
            top_documents.append([])
            continue

        query = query_list if query_list else ""
        doc_texts = [doc[0] for doc in doc_list_for_query if doc]

        # Tokenize and encode the query and documents using the DPR models
        query_inputs = context_tokenizer(query, return_tensors="pt", padding=True, truncation=True)
        doc_inputs = context_tokenizer(doc_texts, return_tensors="pt", padding=True, truncation=True)

        with torch.no_grad():
            query_embeddings = context_model(**query_inputs).pooler_output
            passage_embeddings = context_model(**doc_inputs).pooler_output

          # 이 부분을 document(cycle passage)마다 불렀어야함#########
        # Initialize a GPU device
        res = faiss.StandardGpuResources()

        # Initialize your FAISS index on the GPU
        dimension = 768  # The dimension of DPR embeddings
        index = faiss.index_factory(dimension, "Flat", faiss.METRIC_INNER_PRODUCT)
        index = faiss.index_cpu_to_all_gpus(index)

        # Add data to the GPU-based index
        index.add(passage_embeddings)

        print(index.ntotal)
################################################################################
        # Perform similarity search on the GPU
        # Calculate top_n based on the length of doc_texts
        print(passage_embeddings.shape[0]==len(doc_texts))


        if len(doc_texts) < 5:
            top_n = len(doc_texts)
        else:
            top_n = int(0.2 * len(doc_texts))

        distances, top_doc_indices = index.search(query_embeddings, top_n)


        top_docs = top_doc_indices[0].tolist()  # Convert to a list of top document indices

        top_documents.append(top_docs)
```

```
        print('hi')

    return top_documents



if __name__ =='__main__':

    with open('./data/train_cycle_passage_result.pkl','rb') as f:
        train_cycle_passage = pickle.load(f)
    with open('./data/train_qa_context.pk','rb') as f:
        train_qa_context = pickle.load(f)
    model_name_or_path = "facebook/dpr-ctx_encoder-multiset-base"
    flattened_train_cycle_passage = [doc_list if doc_list else [] for problem in train_cycle_passage for doc_list in problem]
    flattened_trian_qa_context = [query for problem in train_qa_context for query in problem]

    print('Start')
    a = time.time()

    top_doc = calculate_dpr_similarity_with_faiss_gpu(model_name_or_path, qa_context=flattened_trian_qa_context, cycle_passage=flatten

    print('길잉',len(top_doc))
    with open('./data/train_DRP_result1.pkl','wb') as f:
        pickle.dump(top_doc,f)


    print(time.time()-a)
```

지금 시간은 15시 30분으로 아직 진행 중이다.

그렇다면 BM25 결과와 DPR 결과를 가지고 reranking을 진행할 수 있다.

reranking 해서 나오는 결과의 개수를 P_k라고 했을 때 이 $P_k$와 QA_context$_k$를 concat해서 PLM에 넣어서 나오는 결과를 Cycle graph score라 지칭하고,

기존의 GSC의 최종 score는 graph score + context score이다.

여기에 Cycle_passage score을 context에 더해서 new_context score를 구하는 것이다.

DPR과 BM25, 그리고 reranking에서 top_N개의 passage(doc)를 가져오는 방법을 정했다.

1. DPR & BM25

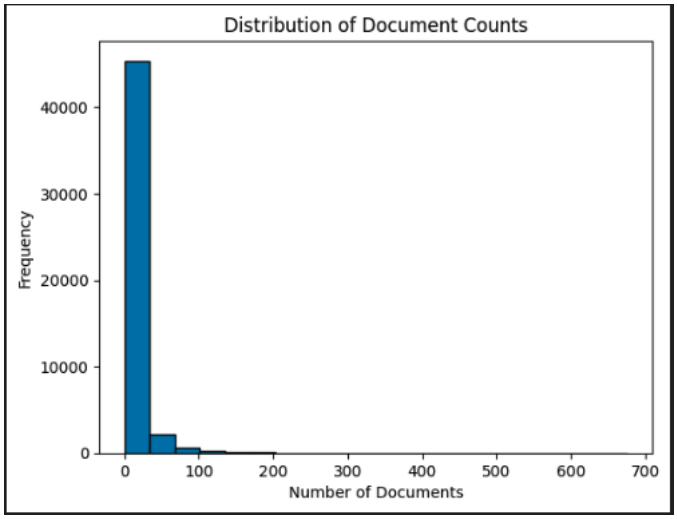
- 개수가 5개보다 적을 경우 그대로 뽑고 이외에는 전체 개수의 20%

2. Reranking

- 5개 미만 : 그대로 뽑기
- 5개 이상 10개 미만 : 5개
- 10개 이상 20개 미만 :10개
- 20개 이상 40개 미만 : 10개
- 40개 이상 60개 미만 : 20개
- 60개 이상 100개 미만 : 30개
- 100개 이상 : 40개

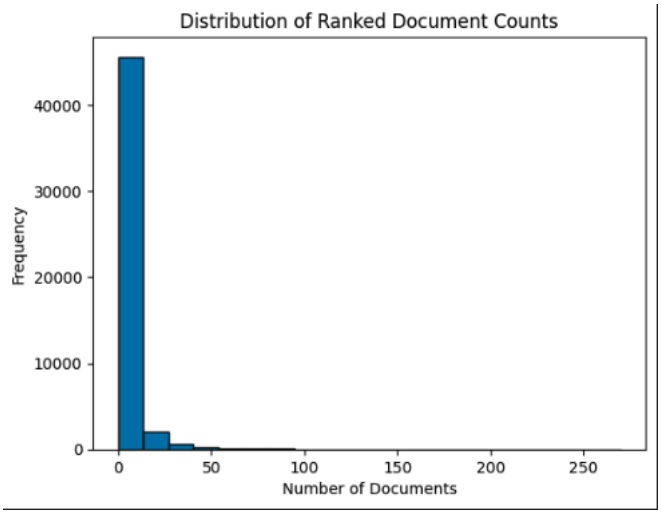
위와 같이 정한 이유는 분포를 통해 살펴 보자

### train data의 Document 분포




- cycle_passage가 제일 많은 개수 676
- cycle_passage가 제일 적은 개수 0
- total cycle_passage 개수 508462
- 평균 cycle_passage 개수 10.43962632173288
- cycle_passage 개수 중간 값 4

### ranked(DPR + BM25) train data의 Document 분포

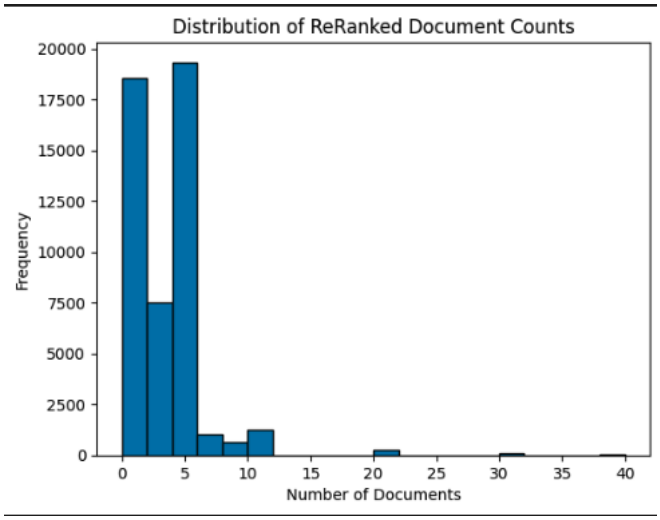



- cycle_passage가 제일 많은 개수 270
- cycle_passage가 제일 적은 개수 0
- total cycle_passage 개수 252064
- 평균 cycle_passage 개수 5.175320808951853
- cycle_passage 개수 중간 값 2

### Reranked(SentenceBERT) train data의 Document 분포

cycle_passage가 제일 많은 개수 40
cycle_passage가 제일 적은 개수 0

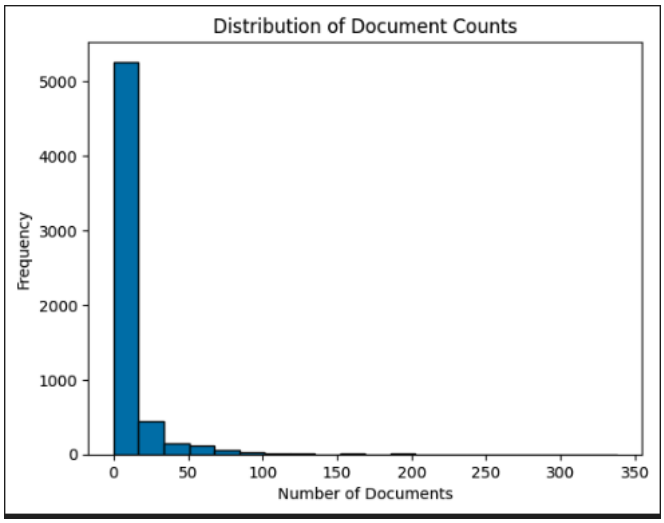
total cycle_passage 개수 133492
평균 cycle_passage 개수 2.7408274304486193
cycle_passage 개수 중간 값 2




---

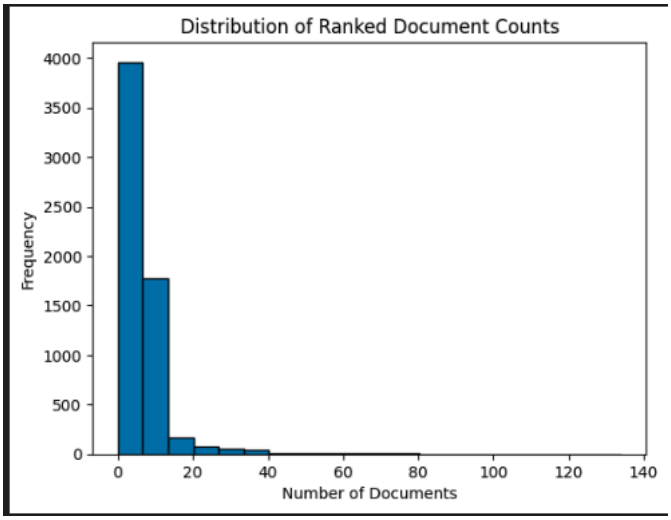
### Dev data의 Document 분포

- cycle_passage가 제일 많은 개수 338
- cycle_passage가 제일 적은 개수 0
- total cycle_passage 개수 59756
- 평균 cycle_passage 개수 9.788042588042588
- cycle_passage 개수 중간 값 4



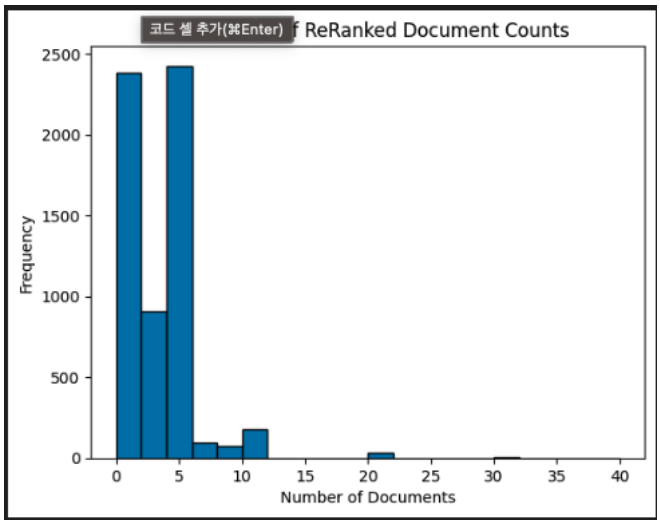

### ranked(DPR + BM25) dev data의 Document 분포

- cycle_passage가 제일 많은 개수 134
- cycle_passage가 제일 적은 개수 0
- total cycle_passage 개수 30638
- 평균 cycle_passage 개수 5.018509418509418
- cycle_passage 개수 중간 값 2




### Reranked(SentenceBERT) train data의 Document 분포

- cycle_passage가 제일 많은 개수 40
- cycle_passage가 제일 적은 개수 0
- total cycle_passage 개수 16241
- 평균 cycle_passage 개수 2.66027846027846
- cycle_passage 개수 중간 값 2




1. 내일 이거 확인하고, reranking까지 진행한다음 (statistics.ipynb)
2. upgrade cycle count 제작하기 (LAB1.ipynb)

- train data는 제작완료