Better Retrieval for Generation

Ankit Saha - AI21BTECH11004
Pranav Balasubramanian - AI21BTECH11023

Problem Statement

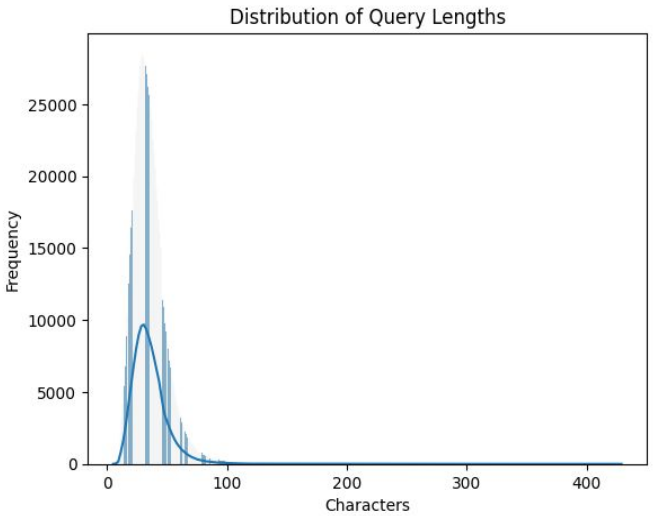
What is Retrieval-Augmented Generation (RAG)?
To address the issue of Large Language Model (LLM) hallucinations and irrelevant outputs, we provide relevant information from a set of retrieved passages (related to the query) to provide the LLM with the necessary context to keep the answer grounded to the query being asked.

This project aims to develop better retrieval strategies that out-perform a set baseline in terms of retrieving and generation quality.

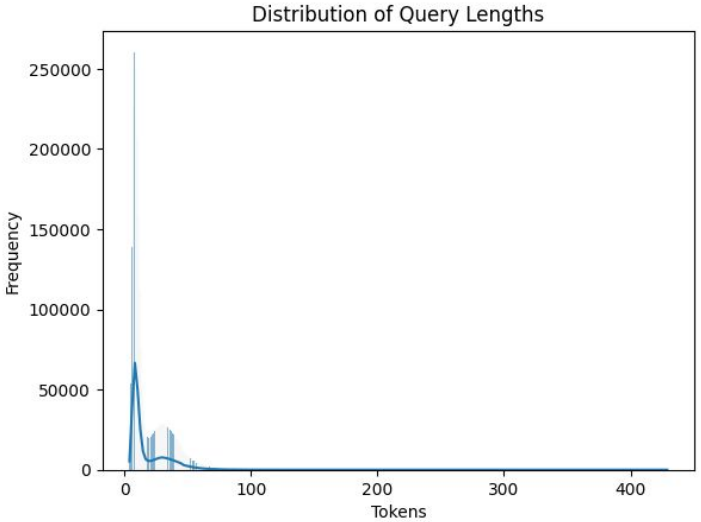
MS-MARCO: Exploratory Data Analysis

Dataset split	Number of queries	Number of passages
Train (7 parts)	808,731	8,087,310
Validation	101,093	1,010,930
Test	101,092	1,010,920

Query Length Analysis

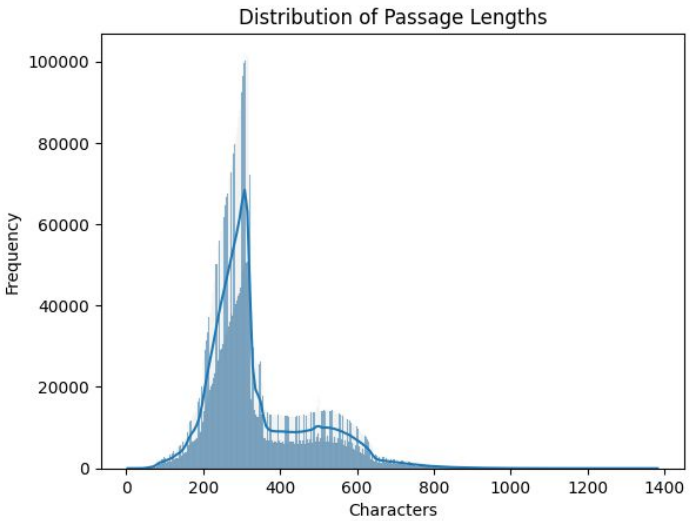


Number of characters

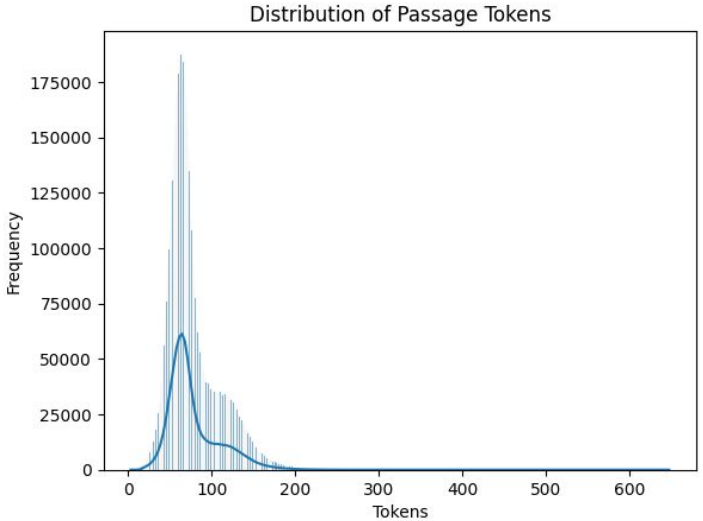


Number of tokens

Passage Length Analysis



Number of characters



Number of tokens

Retrieval Methodology

- The set of all passages is stored in a **FAISS vector database**
- An embedding model is used to first encode each passage
- The encodings are then indexed and stored
- We use the **IndexFlatIP** index, which indexes embeddings based on cosine similarity
- Given a query, encode it using the same embedding model
- Find its nearest neighbours efficiently in the embedding space using the index
- These correspond to the passages with the highest **cosine similarity** with the query, and thus form our retrieved context (most relevant passages to the query)

Embedding Models Tested

- For each query, there are 10 relevant passages with either zero or one passage out of them marked, denoting the passage that was used to generate the answer
- We have tested the following embedding models and measured their top-1 and top-3 accuracies in terms of identifying the marked passage out of the 10 passages
- The marked passage can be thought of as the best passage among the 10 relevant passages
- The following accuracies have been reported for a part of the training dataset because these marked labels are not present in the test dataset

Embedding Models Tested (contd.)

Model name	Top-1 accuracy	Top-3 accuracy
distilbert-base-uncased	12.8248	36.7980
bert-base-uncased	9.3880	28.1643
all-MiniLM-L6-v2	**40.4023**	**75.3562**
bge-large-en-v1.5	44.8449	78.2900
ms-marco-MiniLM-L-12-v2 *(cross-encoder)*	49.7904	83.7385

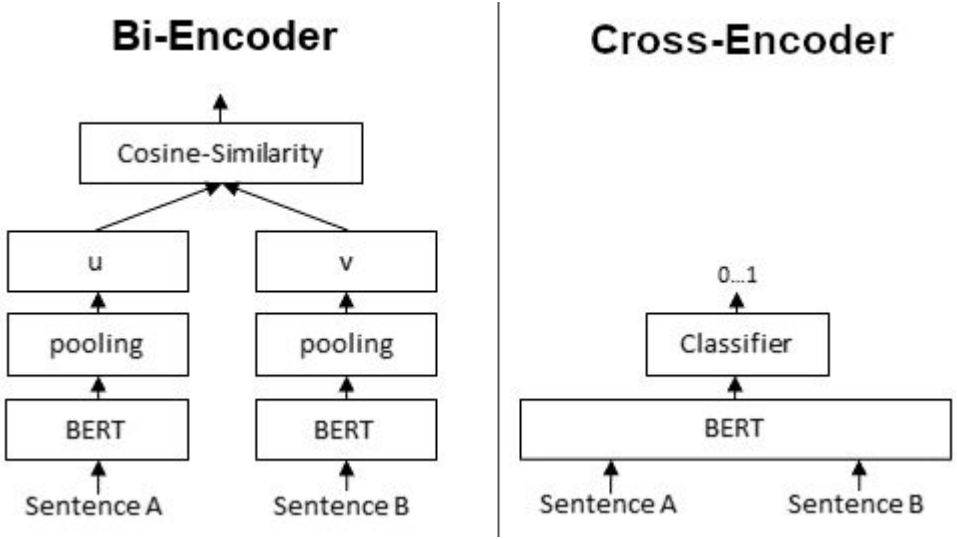
Baseline

- We have chosen the sentence transformer **all-MiniLM-L6-v2** as the baseline because
  - Its embeddings have been optimized for tasks like clustering and semantic similarity unlike BERT and DistilBERT
  - It is fairly lightweight (22.7M parameters) enabling fast fine-tuning and embedding of all passages in the vector database
- For evaluating the baseline, we have indexed all the 1,010,920 test dataset passages in the vector database and retrieved the top-K passages and computed the metrics mentioned in our first presentation for various values of K
- The metrics have been computed considering the 10 relevant passages for each query as the true positives

Baseline Metrics

K	Contextual Precision	Contextual Recall	Contextual Relevancy
3	0.7077	0.1719	0.5706
5	0.6983	0.2546	0.5072
10	0.6591	0.3824	0.3812
100	0.5074	0.6832	0.0681

Improving the Baseline: Reranking



Cross-encoders do not give embeddings!

Cross-Encoders for Reranking

- So far, we have been using encoders to get sentence embeddings
- Alternatively, we can use cross-encoders to directly compare a query and a passage for their similarity
- As seen in one of our previous slides, cross-encoders perform even better than bi-encoders for determining similarity
- However, since cross-encoders do not give embeddings, we cannot use efficient indexing to retrieve the most relevant passages
- Instead, each passage has to be compared with the query one-by-one, which is obviously infeasible when we have a large set of passages

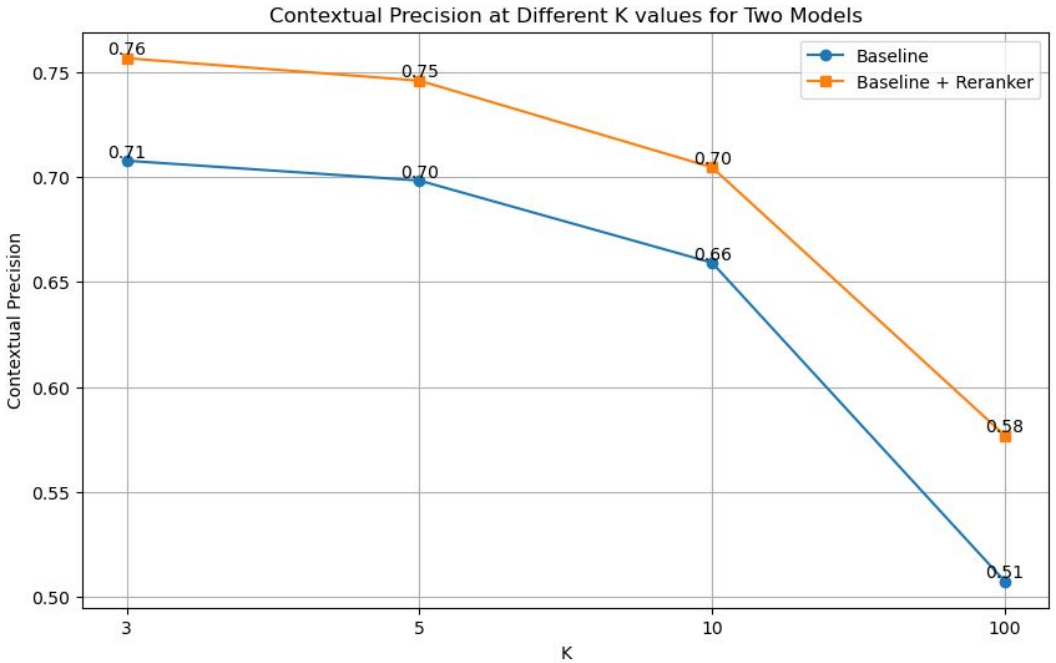
Baseline + Reranker

- We first use the encoder baseline to fetch the 100 closest passages (out of 1,010,920) from the FAISS index
- Then, we use **ms-marco-MiniLM-L-6-v2**, a cross-encoder model, which was trained on the MS-MARCO passage ranking dataset (different from our question-answering dataset)
- The cross-encoder is used to rerank the top 100 passages based on the similarity score with the query, which is then used for retrieving the best (top-K) set of passages
- There is a significant improvement observed in all the metrics

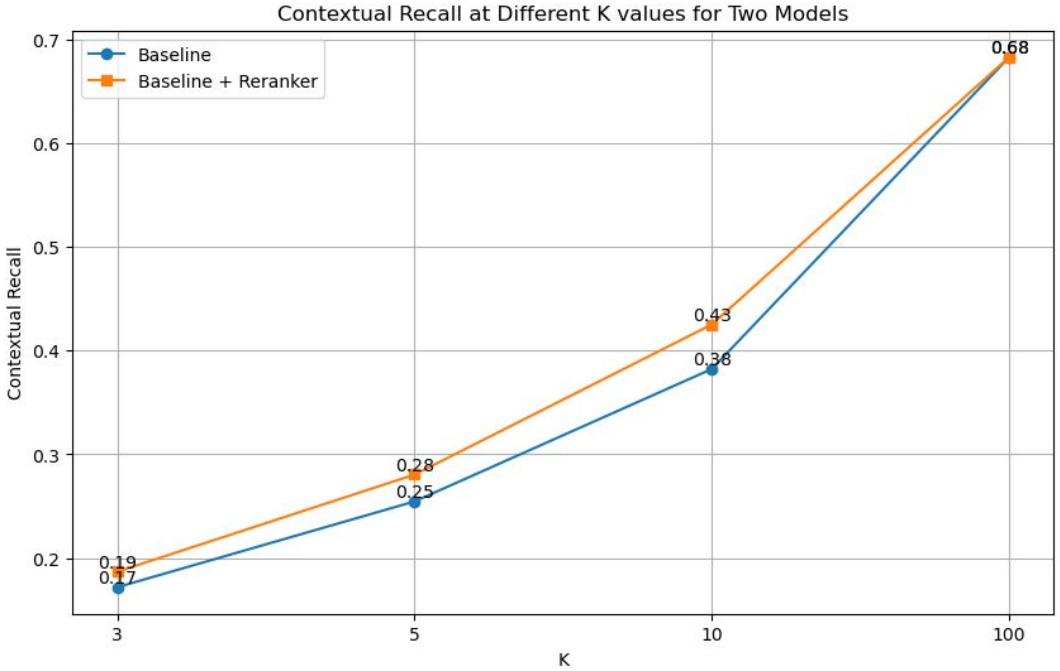
Baseline + Reranker Metrics

K	Contextual Precision	Contextual Recall	Contextual Relevancy
3	0.7564	0.1868	0.6202
5	0.7457	0.2805	0.5584
10	0.7045	0.4253	0.4237
100	0.5768	0.6832	0.0681

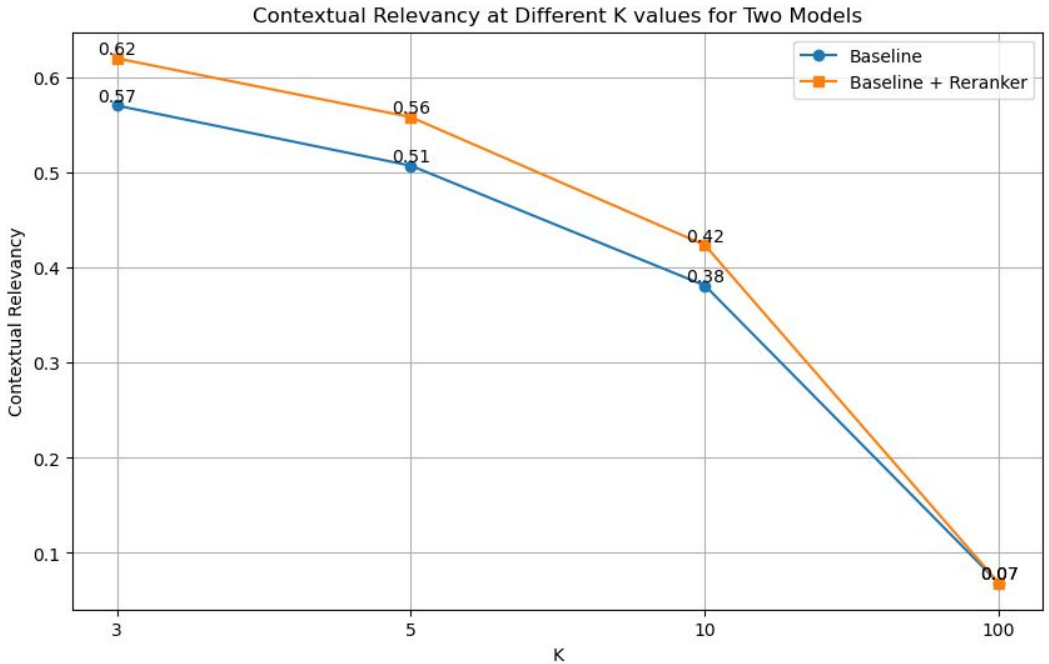
Baseline vs Baseline+Reranker: Contextual Precision



Baseline vs Baseline+Reranker: Contextual Recall



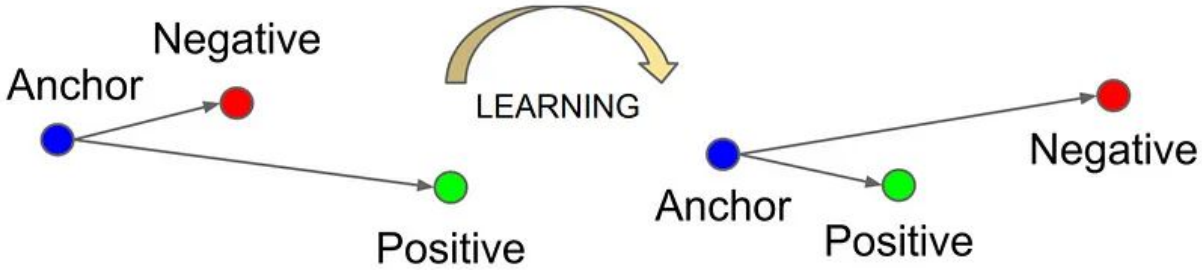
Baseline vs Baseline+Reranker: Contextual Relevancy



Improving the Baseline: Fine-Tuning

- We have then fine-tuned the baseline retriever **all-MiniLM-L6-v2** on our MS-MARCO dataset
- We have used the **TripletMarginWithDistanceLoss** as the loss function with the distance as the cosine distance
- Essentially, the loss takes a triplet as input: (anchor, positive, negative)
- The model will try to minimize the distance between the anchor (query) and the positive (relevant passage) and maximize the distance between the anchor (query) and the negative (irrelevant passage)

Triplet Margin With Distance Loss



$$\frac{1}{n}\sum_{i=1}^{n}\max\{d(a_i,p_i)-d(a_i,n_i)+\text{margin},0\}$$

Fine-tuning the Retriever

- We have used all the 10 relevant passages for each query as the positives to form the triplets
- For negatives, we have randomly sampled passages corresponding to other queries
- The margin has been set as 1.0
- Adam optimizer with a learning rate of 1e-3 has been used
- 128,000 triplets were used for fine-tuning as of now

Fine-tuned + Reranker Metrics

K	Contextual Precision	Contextual Recall	Contextual Relevancy
3	0.0006	0.00006	0.0002
5	0.0006	0.00006	0.0001
10	0.0006	0.00006	0.00006
100	0.0006	0.00008	0.000008

- We can see that all the metrics are very bad
- We will look into this and try to fix it

Work Distribution

Ankit

- Literature survey
- Evaluated embedding models
- Implemented evaluation metrics
- Fine-tuned the baseline embedding model
- Evaluated the improved and baseline retrievers

Pranav

- Literature survey
- Exploratory data analysis
- Set up FAISS vector database
- Evaluated embedding models
- Implemented reranking

Upcoming Plans

- Fixing the extremely poor performance after fine-tuning
- Implement a mixture of expert retrievers as suggested by us in the first presentation
- Implement the likelihood ratio test for retrieved passage relevance by us in the first presentation
- Compare our improved retrievers with our baseline retriever and existing retrievers in literature

References

- Sentence Transformers
- Cross-Encoders
- Intution of Triplet Loss
- TripletMarginWithDistanceLoss
- Loss Functions