COLBERT RETRIEVAL (PRACTICAL)

Martin Fajčík

MUNI PV211 2024

Last edited: 16.04.2024

COLBERT Inference (PLAID)

1. Indexing
    - Storing all vectors for all documents in fp16 has unrealistic memory requirements.
        - For example, 100k documents, 124 tokens on average, fp16 index only for embeddings would take 100,000 x 124 x2 x128 ~ 3,17 GB.

Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT Inference (PLAID)

1. Indexing
    - Storing all vectors for all documents in fp16 has unrealistic memory requirements.
        - For example, 100k documents, 124 tokens on average, fp16 index only for embeddings would take 100,000 x 124 x2 x128 ~ 3,17 GB.
2. Approximate Retrieval
    - Exact retrieval is too slow
        - 32 products on matrix with 100,000 x 124 ~ 12,400,000 embeddings

Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT PLAID

Indexing

1. **Centroids**:
   - Find a set of centroids using **K-means** on random subsample of corpus C.
     - Subsample size ~ square root of # of documents.
     - K ~ rounded down to nearest power of 2 for
       16 x square root of (# of embeddings).

Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT PLAID Indexing – CASE STUDY

Assume ColBERTv2 model with each passage having at most 300 tokens.
Next, assume there is

- 100,000 passages in corpus (<=300 tokens).
- 124 tokens per passage on average.
- model with 128 dimensional vectors.
- 16-bit float type precision used.

Exercise 1

How much memory will centroid vectors cost.

Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT PLAID Indexing – CASE STUDY

Assume ColBERT model with each passage having at most 300 tokens.
Next, assume there is

- 100,000 passages in corpus (<=300 tokens).
- 124 tokens per passage on average.
- model with 128 dimensional vectors.
- 16-bit float type precision used.

Exercise 1

How much memory will centroid vectors cost.

Solution

We have 124 tokens per passage, and 100k passages, this leads to 12 400 000 raw embeddings. K is rounded down to nearest power of 2 for 16 x square root of (# of embeddings). That is K is rounded down to the nearest power of 2 of ~56,000, that is 32 768. Each embedding is save if float 16 type, so the total size in bytes is:

2 bytes x 128 dimensions x 32 768 centroids = 8388608 bytes ~ 8.4 MB.

Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT PLAID

Indexing

2. Compressed Representations:

- Each passage is encoded into matrix P.
- Embeddings in P are compressed.

COLBERT PLAID Indexing – Compression Mechanism

2. **Compressed Representations**:

- Each vector p is stored as
  - index of its nearest centroid C_p, and
  - a residual vector $r_p = p - C_p$
- Each dimension of residual vectors are further quantized into 2 bits.
  - **How to quantize?**
- Theoretical space requirements for 1 representation (b=2, n=128)

$$\log\lceil|C|\rceil + bn$$

Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT PLAID Indexing – Compression Mechanism

2. **Compressed Representations**:

- Each vector p is stored as
  - index of its nearest centroid C_p, and
  - a residual vector $r_p = p - C_p$
- Each dimension of residual vectors are further quantized into 2 bits.
  - Simple quantization = perform **K**-means on each dimension, assign its fp32 intervals into 4 clusters.
- Theoretical space requirements for 1 representation (b=2, n=128) $\log\lceil|C|\rceil + bn$

Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT PLAID Indexing – Compression Mechanism

2. **Compressed Representations**:

- Theoretical space requirements for 1 representation (b=2, n=128)

  $\lceil \log |C| \rceil + bn$

- In practice, authors assume there are up to 2^32 centroids, so
  - **How many bytes will each vector representation consume?**

Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT PLAID Indexing – Compression Mechanism

2. **Compressed Representations**:

- Theoretical space requirements for 1 representation (b=2, n=128) $\lceil \log |C| \rceil + bn$

- In practice, authors assume there are up to 2^32 centroids, so
  - 32 + 2x128 = 288 bits = 36 bytes
- **How much memory it saves from naïve fp16 implementation?**

Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT PLAID Indexing – Compression Mechanism

2. **Compressed Representations**:

- Theoretical space requirements for 1 representation (b=2, n=128) $\lceil \log |C| \rceil + bn$
- In practice, authors assume there are up to 2^32 centroids, so
  - 32 + 2x128 = 288 bits = 36 bytes
- How much memory it saves from naïve fp16 implementation?
  - Naïve = 2 bytes x 128 = 256 bytes (7.1x less)

Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT PLAID Indexing – CASE STUDY

Exercise 2:

Assume ColBERT model with each passage having at most 300 tokens.
Next, assume there is

- 100,000 passages in corpus (<=300 tokens).
- 124 tokens per passage on average.
- model with 128 dimensional vectors.
- 16-bit float type precision used.

Estimate the size of saved compressed vectors.

Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT PLAID Indexing – CASE STUDY

Exercise 2:

Assume ColBERT model with each passage having at most 300 tokens.
Next, assume there is
- 100,000 passages in corpus (<=300 tokens).
- 124 tokens per passage on average.
- model with 128 dimensional vectors.
- 16-bit float type precision used.
- 2 bits per dimension quantization.

Estimate the size of saved compressed vectors.

Solution:

We have 124 tokens per passage, and 100k passages, this leads to 12 400 000 raw embeddings. Each is indexed with centroid index (int32) and residual vector (128x 2 bits) = 32 bytes). To find out, how much embeddings belong to each passage, we further need to save passage lengths (int16 will suffice to passages <300), So the result is:

32 bytes x 12 400 000 + 4 bytes x 12 400 000 + 2 bytes x 100 000= 446 600 000 bytes = 446,6 MB.

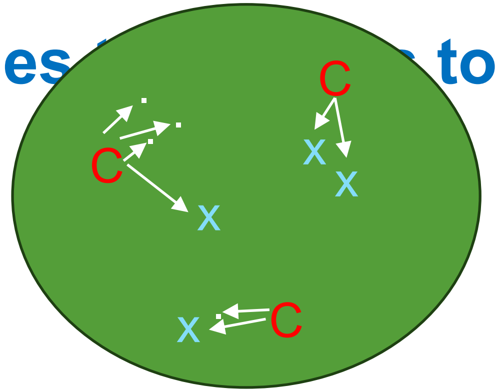
Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT PLAID

Indexing

3. Inverted list of passages

- For each embedding, add passage to an inverted list mapping centroids to their nearest unique passage ID.
  - If 1st embedding in P is nearest to centroid C_1, and 2nd embedding in P is nearest to centroid C_2, both centroids will be mapped to this passage.
- In default implementation, each passage is indexed using **uint32**.
  - **Q: How much passages** **to index?**



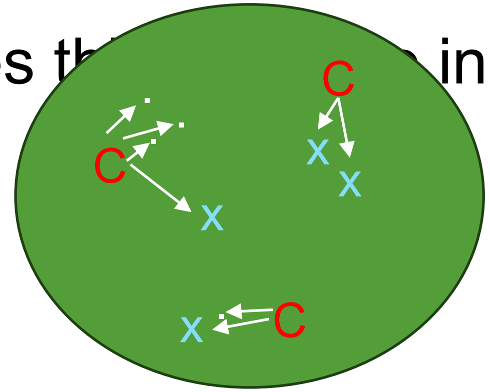
LEGEND
. → passage a
x → passage b
C → centroid location

COLBERT PLAID

Indexing

3. Inverted list of passages

- For each embedding, add passage to an inverted list mapping centroids to their nearest unique passage ID.
    - If 1st embedding in P is nearest to centroid C_1, and 2nd embedding in P is nearest to centroid C_2, both centroids will be mapped to this passage.
- In default implementation, each passage is indexed using **uint32**.
    - Q: How much passages th… index? A: ~4 billion



LEGEND
. → passage a
x → passage b
C → centroid location

COLBERT PLAID Indexing – CASE STUDY

Exercise 3:

Assume ColBERT model with each passage having at most 300 tokens.
Next, assume there is

- 100,000 passages in corpus (<=300 tokens).
- 124 tokens per passage on average.
- Assume each index of each passage is encoded in uint32.
- Assume each cluster is mapped to 198 passages on average.

How much memory will such an inverted index take?

COLBERT PLAID Indexing – CASE STUDY

Exercise 3:

Assume ColBERT model with each passage having at most 300 tokens.
Next, assume there is

- 100,000 passages in corpus (<=300 tokens).
- 124 tokens per passage on average.
- Assume each index of each passage is encoded in uint32.
- Assume each cluster is mapped to 198 passages on average.

How much memory will such an inverted index take?

Solution:

There is 32 768 total clusters (from Exercise 1), so 32 768 x 198 = 6 488 064 mapped passages in total.
These can be saved in a list, if we also separately save a length offset for each cluster.
In this case 32 bits will suffice (using standard dtypes) to encode # of passages mapped to each cluster (remember, there can be the case when almost all passages are mapped to single cluster!).

So in total we have 6 488 064 x 4 bytes + 32768 x 4 bytes = 26 083 328 bytes ~ 26 MB in total.

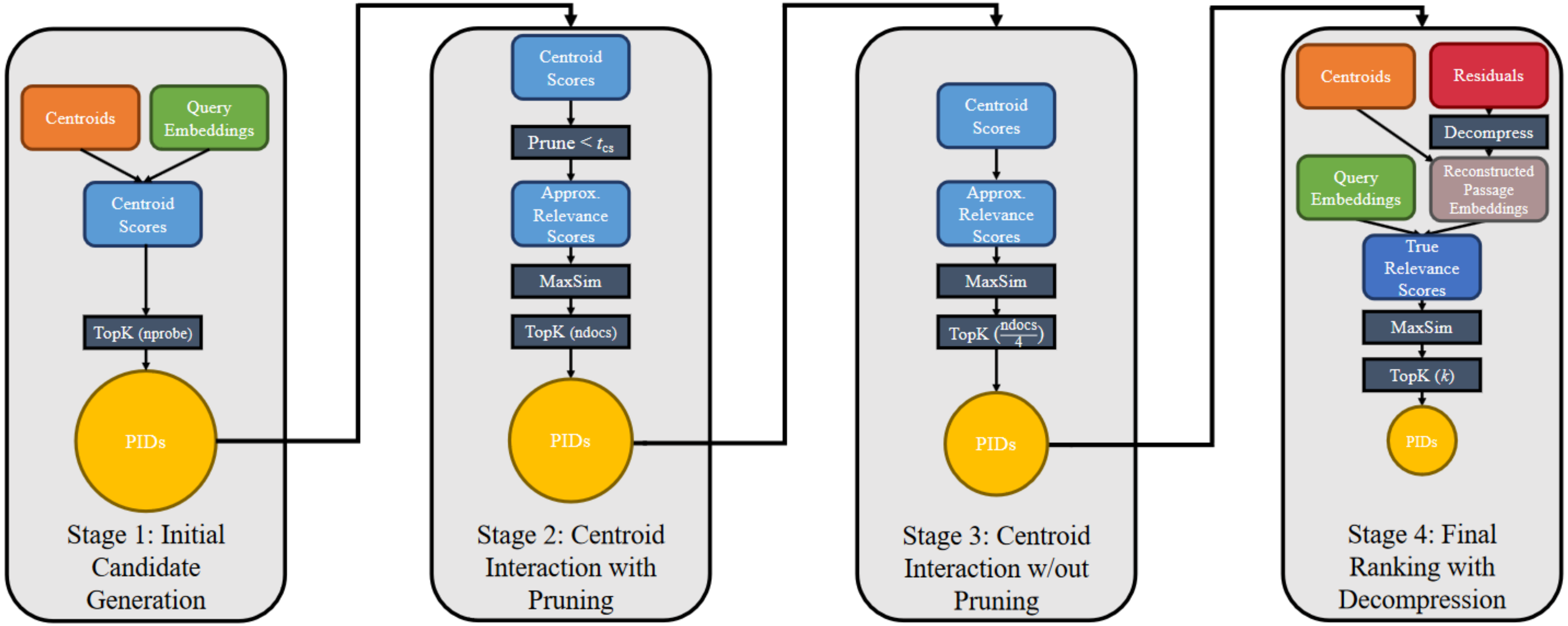
COLBERT PLAID Indexing – CASE STUDY

Index Summary

- Centroids **8.4 MB**
- Embeddings **446,6 MB**
- Inverted Index **26 MB**
- **In total 481 MB**
  - **down from 3,17 GB!**
  - **ONLY ~15% OF THE ORIGINAL SIZE*.**

* - original size didn't counted in passage lengths, but that is negligible in terms of size

COLBERT Retrieval



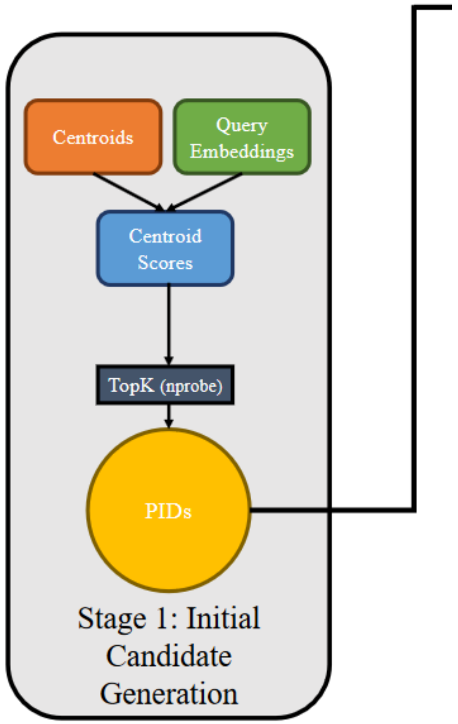
Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT Retrieval

- Stage 1
  - Let C be the matrix of all centroid embeddings.
  - Let Q be the query embeddings, obtained after encoding query.
  - Compute similarities between each

    $$S = CQ^{\top}$$

  - the passages “close” to the top-nprobe centroids per query token are the **initial candidate set**



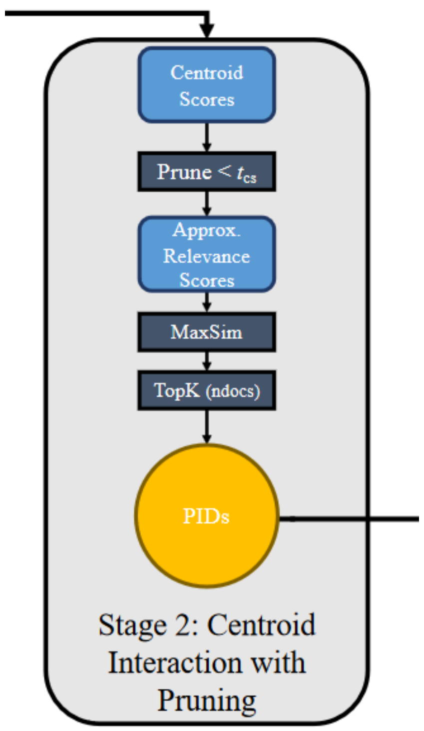
Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT Retrieval

- Stage 2: Representational Composition
  - Suppose **I** is the list of the centroid indices mapped to each of the tokens in the candidate set
  - Compute approximate similarity, where **each token is represented by its nearest centroid embedding**

$$\tilde{D} = \begin{bmatrix} S[I_1] \\ S[I_2] \\ \vdots \\ S[I_{|P|}] \end{bmatrix} \qquad S = CQ^\top$$

$$S_{\tilde{D}} = \sum_{i=1}^{|Q|} \max_{j=1}^{|\tilde{D}|} \tilde{D}_{i,j}$$



Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

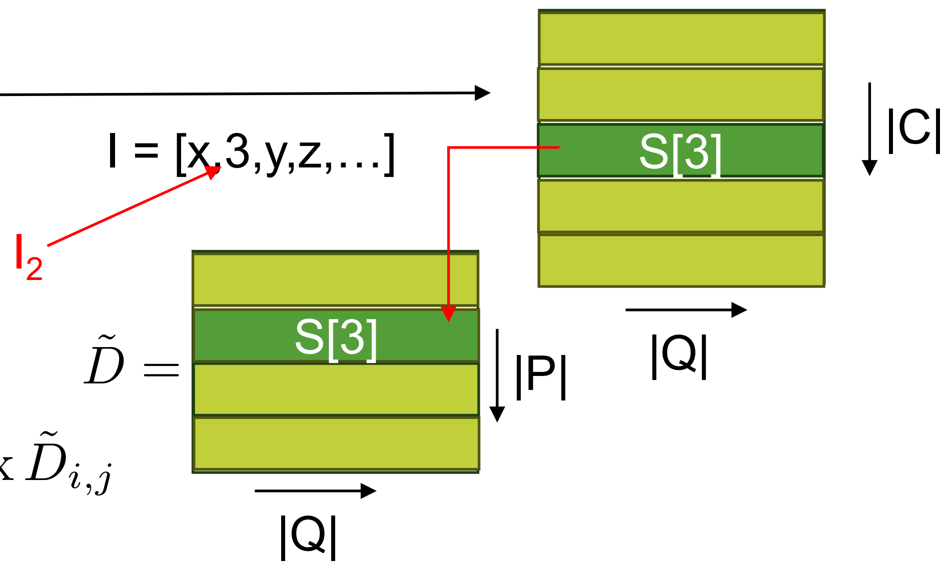
COLBERT Retrieval

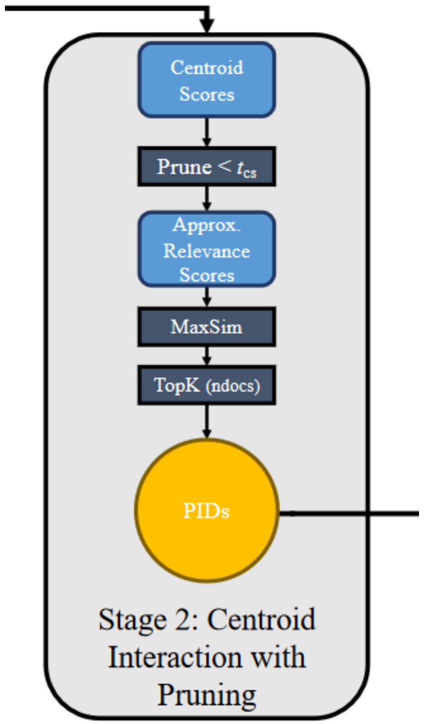
- Stage 2: <u>Representational Composition</u>
  - Suppose **I** is the list of the centroid indices mapped to each of the tokens in the candidate set

$$S = CQ^\top$$

$$\tilde{D} = \begin{bmatrix} S[I_1] \\ S[I_2] \\ \vdots \\ S[I_{|P|}] \end{bmatrix}$$

$$S_{\tilde{D}} = \sum_{i=1}^{|Q|} \max_{j=1}^{|\tilde{D}|} \tilde{D}_{i,j}$$



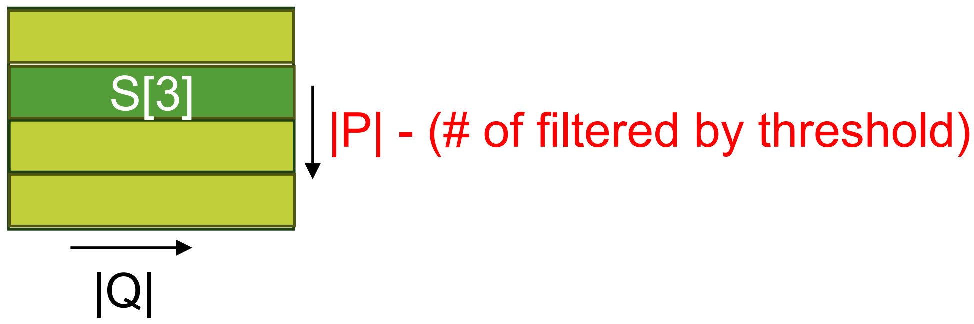


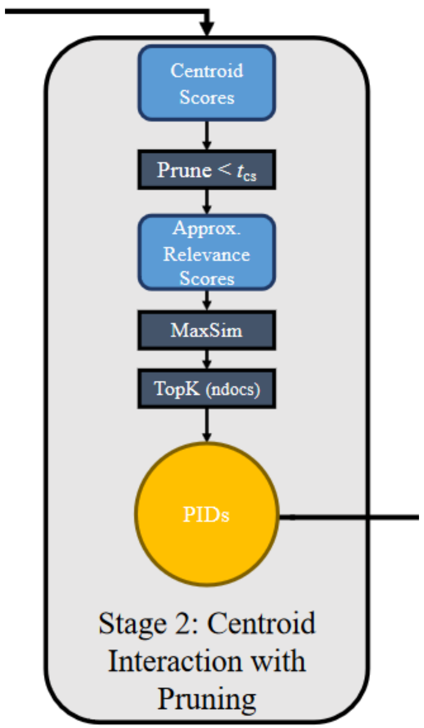
Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT Retrieval

- Stage 2: <u>Pruning</u>
  - To make matrix $\tilde{D}$ even smaller, only tokens, whose corresponding centroid *i* has max similarity to query higher then threshold are considered
    - In implementation, these rows can be actually pre-flagged in S

$$\max_{j=1}^{|Q|} S_{i,j} \geq t_{cs} \qquad \tilde{D} =$$



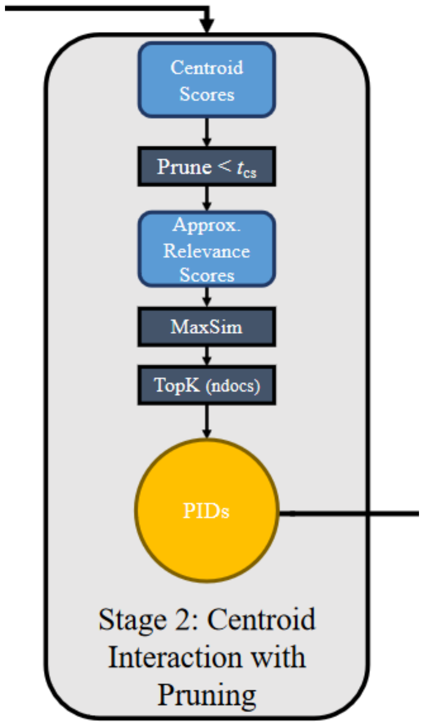


Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT Retrieval

- Stage 2
  - Top-ndocs scoring documents are selected into passage IDs (PIDs) candidate set for stage 3

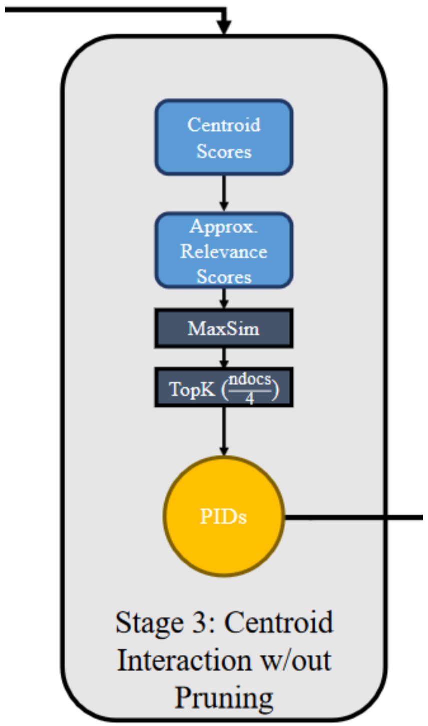
$$S_{\tilde{D}} = \sum_{i=1}^{|Q|} \max_{j=1}^{|\tilde{D}|} \tilde{D}_{i,j}$$



Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT Retrieval

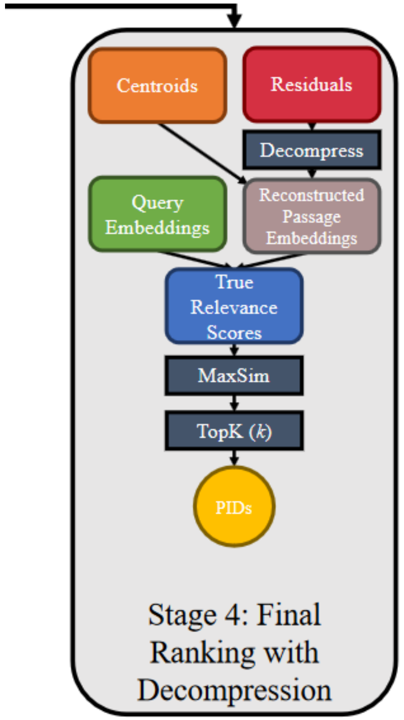
- Stage 3
  - Same as stage 2, but without filtering, considering all rows of S.
  - Top-(ndocs/4) passages are selected for final processing (**Stage 4**).



Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

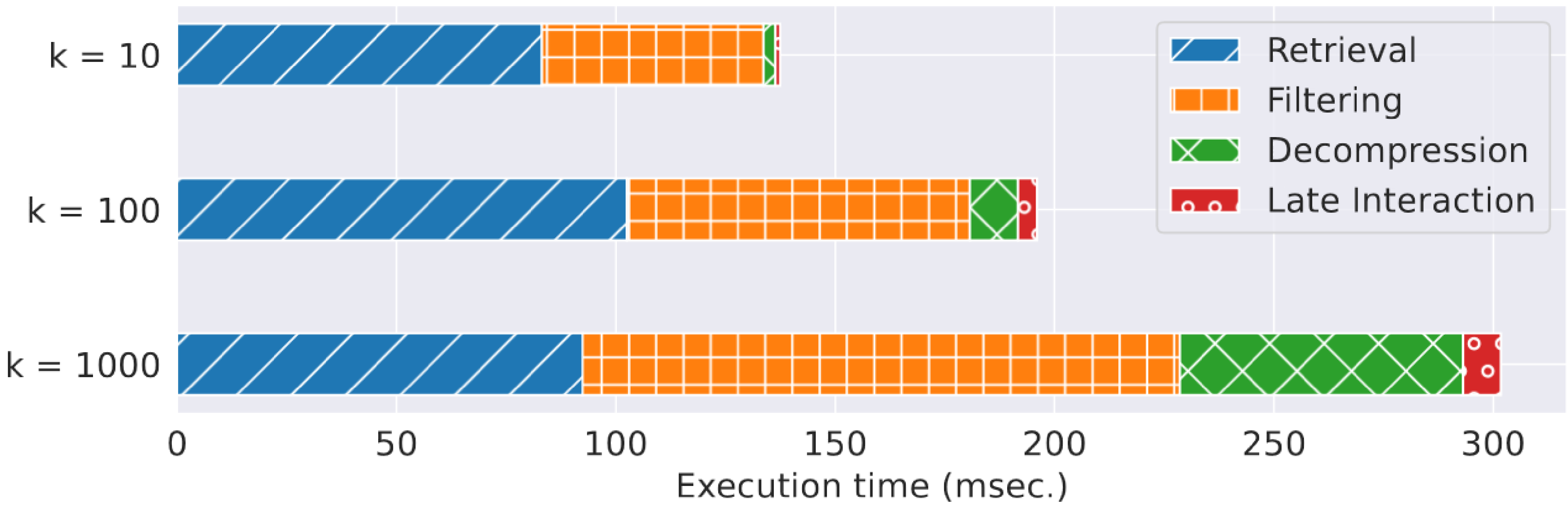
COLBERT Retrieval

- Stage 4
  - Final set of candidates is decompressed, and exact max-sim operation is computed for these in FP32.
  - All steps are done to minimize
    - candidate index lookup (looking for cluster/residual of particular passage)
    - decompressed of candidates (both steps are very slow).
    - **Board: How to do dequantization?**
  - Final **top-k** results are returned, along with similarities.



Santhanam, Keshav, et al. "PLAID: an efficient engine for late interaction retrieval." *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2022.

COLBERT Retrieval Speed on MS-MARCO



Nardini, Franco Maria, Cosimo Rulli, and Rossano Venturini. "Efficient Multi-vector Dense Retrieval with Bit Vectors." *European Conference on Information Retrieval*. Cham: Springer Nature Switzerland, 2024.