A Brief Comparison of Training-Free Multi-Vector Sequence Compression Methods

Rohan Jha[1] Chunsheng Zuo[1] Reno Kriz[1,2] Benjamin Van Durme[1]

Johns Hopkins University[1] Baltimore, MD, USA

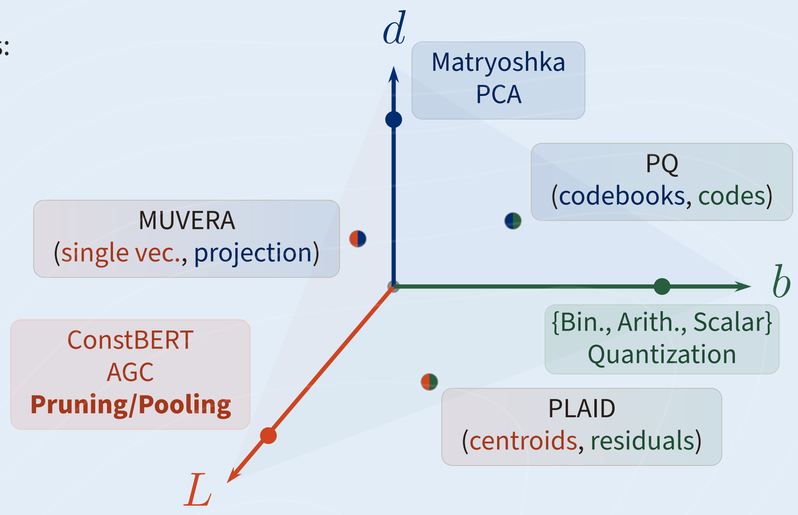
Human Language Technology Center of Excellence[2] Baltimore, MD, USA

Motivation

Multi-vector retrieval models like ColBERT outperform single-vector embeddings, **but their larger index sizes hinder ubiquitous deployment (storage, latency).**

Index size is driven by three axes:

- **Dimensionality (d)**
- **Precision (b)**
- **Sequence Length (L)**



We focus on the third axis: **training-free** methods that reduce the **sequence length (L)** of multi-vector document embeddings.

Compression Methods

How do we reduce the sequence length L of this already-encoded document in our index?

`[D]` `the` `researchers` `found` `that` `large` `language` `models` `can` `generate` `fluent` `but` `factually` `incorrect` `text`

Protect the first token (`[D]` document marker) then apply one of the following:

Token Pruning: Assign each token an **importance score** and retain the top $\tilde{L}$.

Random: Prune tokens at random.

`[D]` `the` ~~`researchers`~~ `found` ~~`that`~~ ~~`large`~~ ~~`language`~~ `models` ~~`can`~~ ~~`generate`~~ ~~`fluent`~~ `but` ~~`factually`~~ ~~`incorrect`~~ `text`

Attention: Score by attention received across all heads/positions in the final transformer layer.

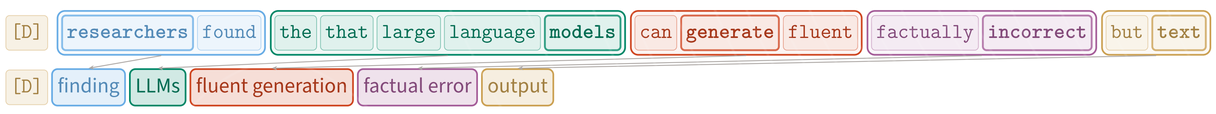
`[D]` ~~`the`~~ `researchers` ~~`found`~~ ~~`that`~~ ~~`large`~~ ~~`language`~~ `models` ~~`can`~~ `generate` ~~`fluent`~~ ~~`but`~~ ~~`factually`~~ `incorrect` `text`

IDF: Score by corpus-level inverse document frequency, preferentially retaining rare terms.

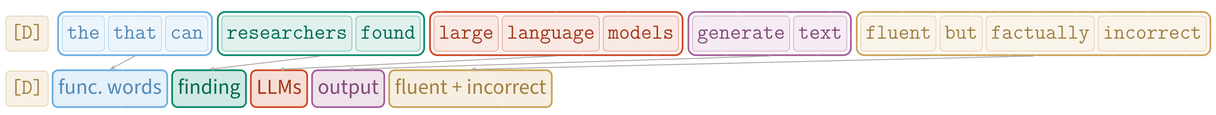
`[D]` ~~`the`~~ `researchers` ~~`found`~~ ~~`that`~~ ~~`large`~~ ~~`language`~~ ~~`models`~~ ~~`can`~~ `generate` `fluent` ~~`but`~~ `factually` `incorrect` ~~`text`~~

Token Pooling: Select $\tilde{L}$ clusters by some **criteria**, then **mean-pool** the embeddings within clusters.

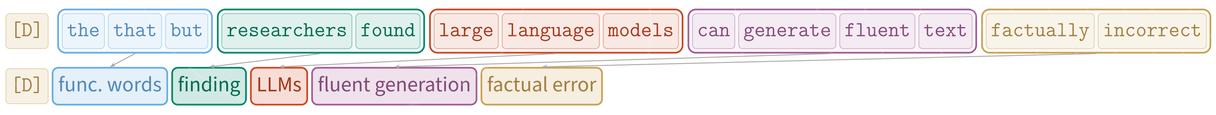
Random / Attention / IDF Pooling: Select k anchors via respective **importance score**, assign tokens to their nearest anchor.



Spherical k-Means: Partition ℓ_2-normalized embeddings via iterative k-means clustering.



Hierarchical (Ward) Pooling: bottom-up agglomerative clustering with cosine distance.



Key Finding

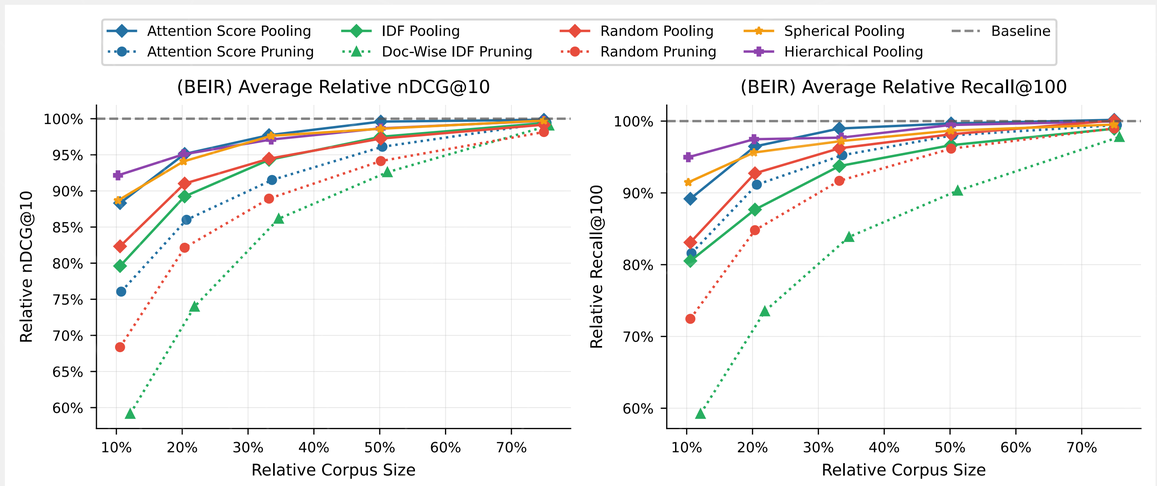
Token pooling is strictly superior to token pruning

(across tested compression ratios, datasets, and metrics)

Caveat: Text-only, concurrent work shows that prune → pool is effective for modalities (e.g. visual documents) with lower info densities / higher distractor rate.

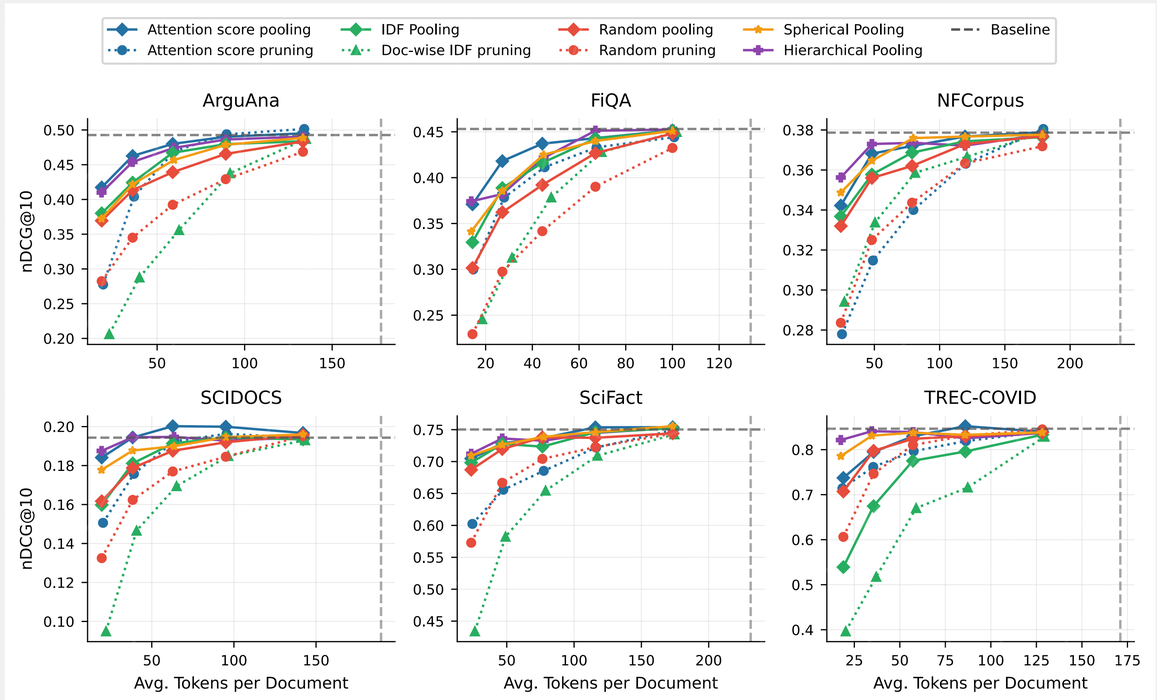
- Pooling preserves semantic density by **aggregating** information rather than discarding it.
- At severe compression ($r \leq 0.20$), pruning performance collapses while pooling degrades more gracefully.
- Pooling achieves **>2x greater index compression** than pruning at equivalent retrieval quality.
- Hierarchical Pooling at 5x compression (∼38 tokens/doc) retains **95.7%** nDCG@10 and **98.1%** Recall@100 on BEIR subset.

Results: BEIR (Avg. Relative nDCG@10, Recall@100)



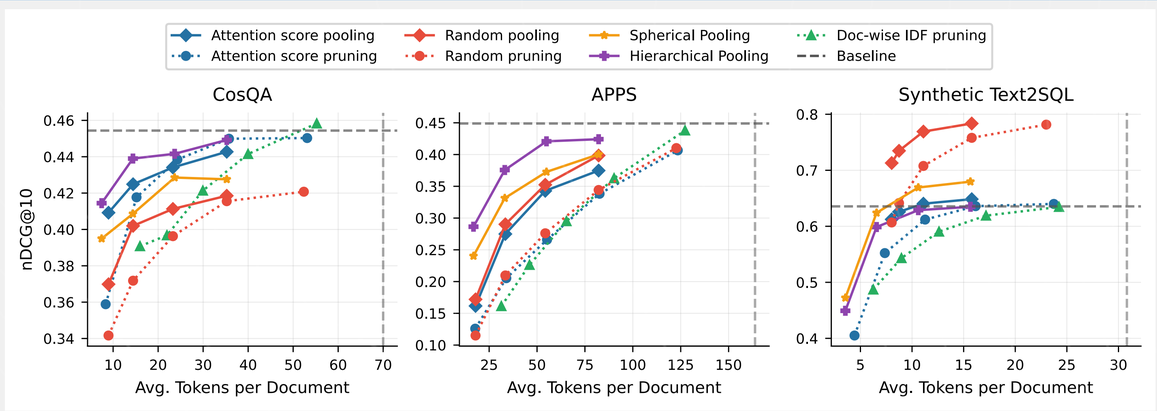
Takeaway: Pooling methods consistently outperform pruning across all compression ratios. Even random pooling beats the best pruning strategy at aggressive compression levels.

Results: BEIR (Per-Dataset nDCG@10)



Takeaway: Hierarchical Pooling is the best overall method, but tightly matched on some domains by other pooling methods.

Results: CoIR (Text-to-Code Retrieval)



Takeaway: Hierarchical Pooling shows a larger advantage over simpler pooling methods on code retrieval tasks, with greater variance due to fewer tokens on average.

Compression Cost vs. Quality Trade-Off

While offline indexing cost is usually ignored, this assumption breaks down when **LM agents (re)construct search indices on-the-fly**.

- **Attention Pooling** (cheapest): $\mathcal{O}(L\tilde{L}d)$
- **Spherical k-Means** (I iters): $\mathcal{O}(IL\tilde{L}d)$
- **Hierarchical Pooling**: $\mathcal{O}(L^2d + L^2 \log L)$

Practical guidance:

For *offline* indexing: use Hierarchical Pooling.

For *agentic / ad-hoc* indexing: Attention Pooling offers an attractive cost–quality trade-off.

LIR'26: The 1st Late Interaction Workshop @ ECIR 2026
April 02, 2026, Delft, The Netherlands