On the Separation of Logical and Physical Ranking Models for Text Retrieval Applications

Jimmy Lin[1], Xueguang Ma[1], Joel Mackenzie[2] and Antonio Mallia[3]

[1]University of Waterloo, Ontario, Canada
[2]The University of Melbourne, Victoria, Australia
[3]New York University, New York, USA

Abstract

Text retrieval using bags of words is typically formulated as inner products between vector representations of queries and documents, realized in query evaluation algorithms that traverse postings in an inverted index. Viewed in database terms, this captures a tight coupling between the "logical" aspects of ranking (i.e., term weighting) and the "physical" aspects of ranking (query evaluation). We argue that explicitly decoupling these two aspects offers a framework for thinking about the relationship between sparse retrieval techniques and the rapidly growing literature on dense retrieval techniques.

Text retrieval using "bag-of-words" exact match techniques can be distilled into a scoring function between a query q and a document d, $S(q,d) = \sum_{t \in q \cap d} f(t)$, where f is some function of term statistics such as tf, idf, doclength, etc. This formulation covers nearly all major families of retrieval models (probabilistic, vector space, language modeling, divergence from randomness, etc.) and is equivalent to the inner product of two weighted vectors of dimension $|V|$, where V is the vocabulary of the collection. Efficiently generating a top-k ranking of documents from an arbitrarily large collection $\mathcal{C}$ is performed using an inverted index that is traversed by a query evaluation algorithm—both components have been optimized over many decades of research.

To borrow an analogy from database systems, such a design tightly couples the "logical" aspects of ranking—the definition of $S(q,d)$—with the "physical" aspects of ranking—how $S(q,d)$ is computed for all $d \in \mathcal{C}$ to efficiently generate a top-k ranking. In the historical development of information retrieval, this tight coupling made sense because alternatives did not appear to be sufficiently compelling. That is, inverted indexes were the most sensible way to implement a ranking model, especially at scale.

Nevertheless, we are not the first to explore the possibility of logical/physical decoupling in the context of information retrieval. Well over a decade ago, Héman et al. [1] demonstrated that a relational database can be adapted to perform text ranking directly. Specifically, inverted lists can be stored in tables and a ranking model can be expressed as an SQL query, thus leaving the database engine to handle physical query execution, decoupled from the logical specification of the ranking model. We can trace prototypes that attempt to integrate information retrieval and database systems back to the 1990s; see, for example, a special issue of the Bulletin of the Technical Committee on Data Engineering from March 1996, which includes a discussion by Fuhr [2] on the lack of "data independence" in information retrieval systems. Despite some follow-up work in 2014 by Mühleisen et al. [3], the idea of text ranking directly with a relational database never caught on.

However, with growing recent interest in dense retrieval techniques, there are reasons to rethink this tight logical/physical coupling. As an initial exploration, we empirically show that different physical realizations of the same logical ranking model manifest different tradeoffs in terms of quality, time, and space. While this observation is certainly not novel—after all, researchers have been exploring the efficiency of query evaluation algorithms and index compression techniques for decades—we argue that our extension of this discussion to dense retrieval techniques provides a fresh perspective.

Let us begin by highlighting the connections between dense and sparse (i.e., bag-of-words exact match) retrieval. Adopting the terminology of Lin et al. [4], dense retrieval can be captured by the following scoring function: $S(q,d) = \phi(\eta_q(q), \eta_d(d_i))$, where $\eta_\cdot$ are encoders that map queries and documents into representation vectors, typically using transformers. These learned dense representations are then compared using ϕ, which can range in complexity from a simple inner product to a (lightweight) neural network.

Focusing on the case where ϕ is defined as an inner product (which encompasses many dense retrieval techniques [5, 6, 7, 8, 9, 10]), the top-k ranking problem is usually cast as nearest neighbor search. In many cases, a brute-force scan over the representation vectors suffices for latency-insensitive batch querying, e.g., with "flat" indexes in Facebook's Faiss library [11], but in other cases,

DESIRES 2021 – 2nd International Conference on Design of Experimental Search & Information REtrieval Systems, September 15–18, 2021, Padua, Italy
jimmylin@uwaterloo.ca (J. Lin)

an approximate nearest neighbor (ANN) technique such as HNSW [12] is necessary for low latency top-k ranking, e.g., using nmslib. Here, we note that the logical/physical dichotomy applies: the same *logical* ranking model (i.e., the definitions of $\eta_\cdot$ and ϕ) can be realized *physically* in different ways, e.g., brute-force scans for batch querying or HNSW for low-latency retrieval.[1]

Further developing this connection, we note that the formulation of dense and sparse retrieval is mathematically equivalent. Specifically, $S(q, d)$ in both cases is defined as the inner product between document and query vectors—the only difference lies in the characteristics of those vectors, e.g., their dimensionality, how they are computed, etc. This means that, in principle, we could "mix and match" logical and physical ranking models—and indeed, this has already been done before. For example, Teofili and Lin [13] evaluated a number of (not very efficient) techniques for performing top-k ranking on dense vectors using Lucene; Tu et al. [14] explored using HNSW for BM25 ranking.

Here, we provide a case study that further investigates this idea for sparse retrieval. We began with DeepImpact [15] as the logical ranking model, on the MS MARCO passage corpus. As points of comparison, we also considered bag-of-words BM25, on both the original passages from the corpus and the results of applying document expansion with doc2query–T5 [16].

We experimented with different physical ranking models: Anserini [17], PISA [18], and the HNSW [12] implementation in nmslib. Note that HNSW can perform search on real-valued DeepImpact representation vectors directly, but Anserini and PISA require quantization first (in both cases, into 8 bits). For HNSW, we use $M = 40$, $\text{bf}_{\text{construct}} = 2500$ and $\text{ef}_{\text{search}} = 1000$, similar to Tu et al. [14]. PISA runs using MaxScore processing after document reordering [19]. Experiments were conducted in memory on a Linux machine with two 3.50 GHz Intel Xeon Gold 6144 CPUs and 512 GiB of RAM. Results on the development queries are shown in Table 1, where we report output quality (MRR@10), query latency (ms), and index size (MB). In all cases, we set retrieval depth to $k = 1000$.

We see that the *same* logical ranking model manifests a diverse range of quality/time/space tradeoffs in different physical implementations. Comparing Anserini and PISA, they achieve the same level of effectiveness (small differences due to tie-breaking effects), but PISA is quite a bit faster (although its indexes are slightly larger). HNSW quality is worse because of the approximate nature of nearest neighbor search, but its queries are much faster than Lucene and almost as fast as PISA (but require much larger indexes).

[1] Although note that database engines are in general responsible for the faithful execution of a logical computation, which is not the case here with HNSW due to its approximations.

Method	Quality	Time	Space
	MRR@10	Latency (ms)	Index Size (MB)
Anserini (Lucene)			
(1a) Bag of words	0.187	40.1	661
(1b) doc2query–T5	0.277	62.8	1036
(1c) DeepImpact (quantized)	0.325	244.1	1417
PISA			
(2a) Bag of words	0.187	8.3	739
(2b) doc2query–T5	0.276	11.9	1150
(2c) DeepImpact (quantized)	0.326	19.4	1564
nmslib HNSW			
(3a) DeepImpact	0.299	21.9	6686
(3b) DeepImpact (quantized)	0.298	22.5	6686

Table 1
Experimental results on the development queries of the MS MARCO passage ranking test collection.

What do we make of these results? In truth, the comparison between Lucene and PISA is not particularly surprising, as researchers have performed experiments along these lines for many decades—comparing alternative implementations of the same *class* of solutions, in this case, document-at-a-time query evaluation on inverted indexes. However, HNSW represents a fundamentally different physical realization of the logical ranking model based on hierarchical navigable small-world graphs, an approach that is very different from inverted indexes. While it is true that HNSW currently does not provide a compelling solution—it is dominated by PISA in terms of both effectiveness and efficiency, HNSW is a relative newcomer. In contrast, PISA benefits from techniques that have been optimized and refined over decades. It is entirely possible that as HNSW receives more attention, the performance gap will close.

Furthermore, dense and sparse representations are not discrete categories, but rather lie on a continuum. Currently, the size (in terms of the number of dimension) of sparse representations equals the vocabulary size of the corpus, and dense representations typically have hundreds of dimensions. What if we "densify" sparse representations and "sparisfy" dense representations—for example, to yield vectors that are ten thousand dimensions? How then will the tradeoffs manifest? We believe that separating the logical and physical aspects of ranking will enable future innovations to progress independently and provide a helpful framework for exploring quality, time, and space tradeoffs in future studies of dense and sparse retrieval, as well as hybrids and points in between.

Acknowledgements

We'd like to thank Arjen de Vries for helpful comments on an earlier draft of this piece.

References

[1] S. Héman, M. Zukowski, A. P. de Vries, P. A. Boncz, Efficient and flexible information retrieval using MonetDB/X100, in: Proceedings of the Third Biennial Conference on Innovative Data Systems Research (CIDR 2007), Asilomar, California, 2007, pp. 96–101.

[2] N. Fuhr, Models for integrated information retrieval and database systems, Bulletin of the Technical Committee on Data Engineering 19 (1996) 3–13.

[3] H. Mühleisen, T. Samar, J. Lin, A. de Vries, Old dogs are great at new tricks: column stores for IR prototyping, in: Proceedings of the 37th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2014), Gold Coast, Australia, 2014, pp. 863–866.

[4] J. Lin, R. Nogueira, A. Yates, Pretrained transformers for text ranking: BERT and beyond, arXiv:2010.06467 (2020).

[5] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, W.-t. Yih, Dense passage retrieval for open-domain question answering, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020, pp. 6769–6781.

[6] L. Xiong, C. Xiong, Y. Li, K.-F. Tang, J. Liu, P. N. Bennett, J. Ahmed, A. Overwijk, Approximate nearest neighbor negative contrastive learning for dense text retrieval, in: Proceedings of the 9th International Conference on Learning Representations (ICLR 2021), 2021.

[7] S. Hofstätter, S. Althammer, M. Schröder, M. Sertkan, A. Hanbury, Improving efficient neural ranking models with cross-architecture knowledge distillation, arXiv:2010.02666 (2020).

[8] L. Gao, Z. Dai, T. Chen, Z. Fan, B. V. Durme, J. Callan, Complementing lexical retrieval with semantic residual embedding, in: Proceedings of the 43rd European Conference on Information Retrieval (ECIR 2021), Part I, 2021, pp. 146–160.

[9] S. Hofstätter, S.-C. Lin, J.-H. Yang, J. Lin, A. Hanbury, Efficiently teaching an effective dense retriever with balanced topic aware sampling, in: Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021), 2021, pp. 113–122.

[10] S.-C. Lin, J.-H. Yang, J. Lin, In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval, in: Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021), 2021, pp. 163–173.

[11] J. Johnson, M. Douze, H. Jégou, Billion-scale similarity search with GPUs, arXiv:1702.08734 (2017).

[12] Y. A. Malkov, D. A. Yashunin, Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs, Transactions on Pattern Analysis and Machine Intelligence 42 (2020) 824–836.

[13] T. Teofili, J. Lin, Lucene for approximate nearest-neighbors search on arbitrary dense vectors, arXiv:1910.10208 (2019).

[14] Z. Tu, W. Yang, Z. Fu, Y. Xie, L. Tan, K. Xiong, M. Li, J. Lin, Approximate nearest neighbor search and lightweight dense vector reranking in multi-stage retrieval architectures, in: Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval, 2020, pp. 97–100.

[15] A. Mallia, O. Khattab, T. Suel, N. Tonellotto, Learning passage impacts for inverted indexes, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 1723–1727.

[16] R. Nogueira, J. Lin, From doc2query to docTTTTTquery, 2019.

[17] P. Yang, H. Fang, J. Lin, Anserini: Reproducible ranking baselines using Lucene, Journal of Data and Information Quality 10 (2018) Article 16.

[18] A. Mallia, M. Siedlaczek, J. Mackenzie, T. Suel, PISA: performant indexes and search for academia, in: Proceedings of the Open-Source IR Replicability Challenge (OSIRRC 2019): CEUR Workshop Proceedings Vol-2409, Paris, France, 2019, pp. 50–56.

[19] J. Mackenzie, M. Petri, A. Moffat, Faster index reordering with bipartite graph partitioning, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 1910–1914.