SHORT-PAPER

Pairing Clustered Inverted Indexes with κ-NN Graphs for Fast Approximate Retrieval over Learned Sparse Representations

SEBASTIAN BRUCH

FRANCO MARIA NARDINI, Institute of Information Science and Technologies "Alessandro Faedo", Pisa, PI, Italy

COSIMO RULLI, Institute of Information Science and Technologies "Alessandro Faedo", Pisa, PI, Italy

ROSSANO VENTURINI, University of Pisa, Pisa, PI, Italy

Open Access Support provided by:

University of Pisa

Institute of Information Science and Technologies "Alessandro Faedo"

Published: 21 October 2024

Citation in BibTeX format

CIKM '24: The 33rd ACM International Conference on Information and Knowledge Management
October 21 - 25, 2024
ID, Boise, USA

Conference Sponsors:
SIGIR

CIKM '24: Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (October 2024)
https://doi.org/10.1145/3627673.3679977
ISBN: 9798400704369

Pairing Clustered Inverted Indexes with κ-NN Graphs for Fast Approximate Retrieval over Learned Sparse Representations

Sebastian Bruch
Pinecone
New York, USA
sbruch@acm.org

Franco Maria Nardini
ISTI-CNR
Pisa, Italy
francomaria.nardini@isti.cnr.it

Cosimo Rulli
ISTI-CNR
Pisa, Italy
cosimo.rulli@isti.cnr.it

Rossano Venturini
University of Pisa
Pisa, Italy
rossano.venturini@unipi.it

ABSTRACT

Learned sparse representations form an effective and interpretable class of embeddings for text retrieval. While exact top-k retrieval over such embeddings faces efficiency challenges, a recent algorithm called Seismic has enabled remarkably fast, highly-accurate approximate retrieval. Seismic statically prunes inverted lists, organizes each list into geometrically-cohesive blocks, and augments each block with a summary vector. At query time, each inverted list associated with a query term is traversed one block at a time in an arbitrary order, with the inner product between the query and summaries determining if a block must be evaluated. When a block is deemed promising, its documents are fully evaluated with a forward index. Seismic is one to two orders of magnitude faster than state-of-the-art inverted index-based solutions and significantly outperforms the winning graph-based submissions to the BigANN 2023 Challenge. In this work, we speed up Seismic further by introducing two innovations to its query processing subroutine. First, we traverse blocks in order of importance, rather than arbitrarily. Second, we take the list of documents retrieved by Seismic and "expand" it to include the neighbors of each document using an offline κ-regular nearest neighbor graph; the expanded list is then ranked to produce the final top-k set. Experiments on two public datasets show that our extension, named SeismicWave, can reach almost-exact accuracy levels and is up to 2.2× faster than Seismic.

CCS CONCEPTS

• **Information systems → Retrieval models and ranking**.

KEYWORDS

Learned sparse representations, maximum inner product search, inverted index.

ACM Reference Format:
Sebastian Bruch, Franco Maria Nardini, Cosimo Rulli, and Rossano Venturini. 2024. Pairing Clustered Inverted Indexes with κ-NN Graphs for Fast Approximate Retrieval over Learned Sparse Representations. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3627673.3679977

CIKM '24, October 21–25, 2024, Boise, ID, USA

ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3679977

1 INTRODUCTION

Learned sparse retrieval (LSR) [7, 8, 10, 15, 18] is a family of widely-used techniques that encode an input into *sparse* embeddings—a vector whose dimensions correspond with terms in some dictionary, with nonzero coordinates indicating that the corresponding terms are semantically relevant to the input. Similarity is typically determined by inner product, so that retrieval becomes the problem known as Maximum Inner Product Search (MIPS) [1]: Finding the top-k vectors whose inner product with a query vector is maximal.

Several reasons motivate research on LSR. First, its effectiveness is often on par with *dense retrieval*—which learns dense embeddings [12, 13, 17, 24, 26, 28, 31]. Importantly, studies show that LSR generalizes better to out-of-distribution collections [2, 15]. Second, sparse embeddings inherit many benefits of classical lexical models such as BM25 [27] and are amenable to well-understood algorithms and data structures such as the inverted index. Finally, because each dimension maps to a term, sparse embeddings are *interpretable*.

Despite their attractive properties, MIPS over sparse embeddings faced significant efficiency challenges [3, 4, 20, 22]. Recognizing this handicap, the 2023 BigANN challenge at NeurIPS hosted a sparse retrieval track, which evaluated the accuracy-throughput trade-off of submitted solutions on the Splade [9] embeddings of Ms Marco [25]. Results were surprising: the winners were not inverted index-based algorithms, but an adaptation of HNSW [21], a graph-based approximate nearest neighbor (ANN) algorithm.

Motivated by BigANN, Bruch *et al.* proposed Seismic [5], an approximate sparse MIPS algorithm that is highly-accurate yet remarkably fast. In contrast to winning entries of BigANN, Seismic operates on two classic data structures: the inverted and the forward index. The crucial innovation in Seismic is that, each of its inverted lists is organized into geometrically-cohesive blocks, and each block is equipped with a *summary* of the vectors contained in it. Note that, blocks are arranged arbitrarily within each list.

Seismic executes a term-at-a-time strategy to produce the top-k approximate set for a query q. When consuming an inverted list, Seismic first computes the inner product between q and every summary in that list to produce a "potential" score for each block. It then visits blocks in arbitrary order, and compares their

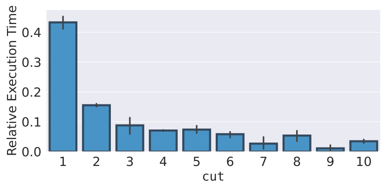

Figure 1: Breakdown of SEISMIC's query processing time per inverted list on SPLADE embeddings of MS MARCO.

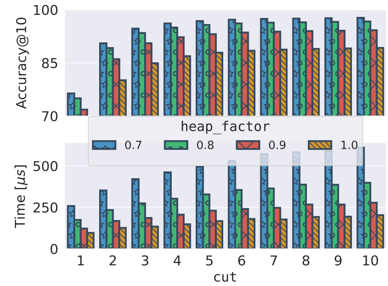

Figure 2: Top-10 accuracy of SEISMIC and search time versus the number of processed inverted lists on SPLADE embeddings of MS MARCO. Accuracy rises quickly and plateaus, while latency increases steadily as more lists are processed.

potential with the smallest score in the top-k heap to determine if a block should be further evaluated. If a block's potential exceeds the threshold, SEISMIC resorts to the forward index to compute the exact inner product between q and every document in that block. Because summaries allow the query processor to skip over a large number of blocks, the algorithm saves substantial computation.

In this paper, we enhance SEISMIC by incorporating two ideas into the query processor to reach higher per-query latency, resulting in a method we name SEISMICWAVE. Our first contribution is to visit blocks in order of their potential, as opposed to the arbitrary order in which they happen to appear in an inverted list.

Second, we leverage the *clustering hypothesis* (CH) [11], which suggests that closely-related documents tend to be relevant to the same queries. We do so by introducing a κ-regular nearest neighbor graph (i.e., a directed graph where each document is connected to its κ nearest neighbors) to "expand" the list of documents returned by SEISMIC. Specifically, we obtain the neighbors of each retrieved document to form an expanded set, and rank the resulting set to extract the top-k subset. This pre-computed structure augments the inverted index—whose lists and blocks capture local similarities—with global information about the similarity between documents.

We must note that, a number of works have paired the inverted index with some realization of CH. MacAvaney *et al.* [19], for example, coupled a standard re-ranking step with a graph-based adaptive exploration, to add to the pool documents that are most similar to the highest-scoring documents. In Lexically-Accelerated Dense Retrieval (LADR), Kulkarni *et al.* [14] use lexical retrieval techniques to seed dense retrieval with a document proximity graph. Our work is related to the above as we also use CH to speed up sparse MIPS.

2 METHODOLOGY

In this section, we describe our contributions in detail. To make the discussion more concrete, we quickly describe our notation. We denote vectors with lower-case letters (e.g., u and v) and reserve q for the query. We use subscripts to denote specific coordinates (e.g., q_i is the i-th coordinate of q). We write $nz(u) = \{i \mid u_i \neq 0\}$ for the set of non-zero coordinates of a vector.

2.1 Overview of Query Processing in SEISMIC

We briefly review SEISMIC in this section. However, as our review is concise due to space constraints, we refer the interested reader to [5] for a detailed description of the algorithm.

SEISMIC's index comprises of two data structures: a forward index and an inverted index. The forward index is a mapping from document identifiers to raw vectors, and is used to compute exact inner products. The inverted index is made up of one inverted list per coordinate. Each inverted list keeps the id of the "top" λ documents: order in the i-th list is determined by the value of the i-th coordinate. The λ entries in an inverted list are then organized into β geometric blocks by the application of a clustering algorithm to the document vectors in that list. Finally, each block is accompanied by a summary vector that represents its documents.

SEISMIC adopts a term-at-a-time query processing strategy. That means, the retrieval algorithm traverses the top `cut` inverted lists—where `cut` is a hyper-parameter—from $nz(q)$. As it traverses each list, it inserts promising documents into a heap to maintain a current set of top-k documents. Processing an inverted list involves visiting its β blocks sequentially (in arbitrary order), computing the inner product of the block's summary vector with q, comparing the resulting score with the heap's threshold (scaled by `heap_factor`, another hyper-parameter), and evaluating the documents within that block when the block's score exceeds the threshold. If a document's true score—computed using the forward index—exceeds the heap's current threshold, it is inserted into the heap.

2.2 Ordered Block Traversal: First Contribution

The success of SEISMIC rests on the fact that the vast majority of blocks can be skipped without any processing beyond the inner product computation between their summary vectors and q. That is because the heap's threshold can only monotonically increase, so that, as more blocks are visited and evaluated, the likelihood that yet-unseen blocks qualify for further processing decreases. This phenomenon plays out in practice as shown in Figure 1, which visualizes the proportion of SEISMIC's query processing time spent on the first 10 inverted lists: More than 40% of the search time is spent processing the first list, and about 60% on the first two lists.

Our first contribution is to leverage this empirical property. In particular, rather than visiting blocks within the first inverted list in arbitrary order, we sort its blocks by their summary score (i.e., inner product of their summary vector with q) in descending order. The rest of the procedure remains the same. This change ensures that the most promising blocks are evaluated first, and the less promising blocks are more likely to be skipped.

Note that, inner products of q with summaries in an inverted list can be computed efficiently with sparse matrix-vector multiplication. Sorting the blocks is also cheap as β is typically small.

Algorithm 1: Refining Seismic's results with κ-NN graph.

Input: q: query; k: number of results; $\mathcal{N}(\cdot)$: κ-NN graph represented as a function that returns κ neighbors of its argument; Heap: heap with the top-k results from Seismic.
Result: A Heap with the top-k documents.

$S \longleftarrow$ the ids of the vectors in the Heap
for $u \in S$ **do**
  for $v \in \mathcal{N}(u)$ **do**
    $p = \langle q, \text{ForwardIndex}[v] \rangle$
    if Heap.len() $< k$ or $p >$ Heap.min() **then**
      Heap.insert(p, v)
    if Heap.len() $= k + 1$ **then**
      Heap.pop_min()
return Heap

2.3 κ-NN Graph: Second Contribution

We make a second observation: As search for q goes on, Seismic struggles to find good candidate blocks to evaluate. This is evident in Figure 2, which reports latency (in μs) and accuracy for $k = 10$ as a function of the number of evaluated lists (cut) for various values of heap_factor. As is clear from the figure, Seismic's top-k quality rapidly enters the high accuracy region ($> 95\%$), but plateaus and struggles to reach almost-exact accuracy ($> 98\%$). Naturally, exploring new inverted lists increases the execution time.

Our second contribution is to counter the phenomenon above by complementing the Seismic index with auxiliary information that allows it to *refine* the candidate pool. In particular, we construct a κ-NN graph from a collection of documents, wherein each node represents a document and has an outgoing edge to κ documents whose inner product with the source node is maximal. The resulting structure can be stored as a look-up table consisting of pairs of a document id and a list of its κ closest neighbors. The κ-NN graph allows us to identify the nearest neighbors of a document quickly, realizing CH. Formally, let us denote by $\mathcal{N}(u)$ the set of κ closest documents to document $u \in \mathcal{X}$, where $\mathcal{X}$ is the collection:

$$\mathcal{N}(u) = \underset{v \in \mathcal{X}}{\arg\max}^{(\kappa)} \langle u, v \rangle. \tag{1}$$

We use the κ-NN graph as follows. Once Seismic concludes its search for the top-k documents, we take the set of documents in the heap and denote it by $\mathcal{S}$. We then form the *expanded set* $\tilde{\mathcal{S}} = \bigcup_{u \in \mathcal{S}} \left(\{u\} \cup \mathcal{N}(u)\right)$, compute scores for documents in $\tilde{\mathcal{S}}$, and return the top-k subset. This procedure is shown in Algorithm 1.

3 EXPERIMENTS

Datasets. We experiment on two publicly-available datasets: Ms Marco v1 Passage [25] and Natural Questions (NQ) from Beir [29]. Ms Marco is a collection of 8.8M passages in English. In our evaluation, we use the smaller "dev" set of 6,980 queries. NQ is a collection of 2.68M questions in English and a "test" set of 3,452 queries.

Learned Sparse Representations. We evaluate all methods on embeddings generated by two LSR models:

- Splade [8]. Each non-zero entry is the importance weight of a term in the BERT [6] WordPiece [30] vocabulary consisting of 30,000 terms. We use the `cocondenser-ensembledistil` version of Splade that yields MRR@10 of 38.3 on the Ms Marco dev set. The number of non-zero entries in documents (queries) is, on average, 119 (43) for Ms Marco and 153 (51) for NQ.
- Splade-v3 [16]. An improved variant of Splade that incorporates a modified objective and distillation strategy. It yields MRR@10 of 40.3 on the Ms Marco dev set. The number of non-zero entries in documents (queries) is, on average, 168 (24) for Ms Marco.

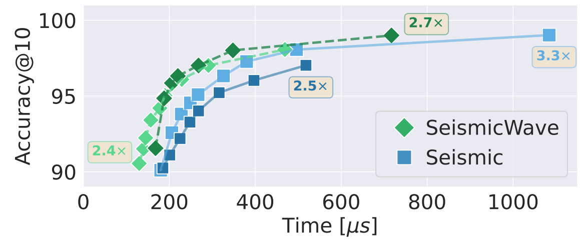

Figure 3: Comparison of Seismic and SeismicWave by memory, latency, and accuracy on Splade embeddings of NQ.

κ-NN Graph. Constructing the (exact) κ-NN graph is expensive due to its quadratic time complexity. As such, we relax the construction to an approximate (but almost-exact) κ-NN graph: each document is connected to its approximate set of top-κ documents. To that end, we use a Seismic index with the following parameters to retrieve the top-κ candidates for every document in the collection: $\lambda = 10{,}000$, $\beta = 2{,}000$, $\alpha = 0.6$, cut $= 15$, heap_factor $= 0.7$. Note that, the κ-NN graph is formed only once for a collection. Storing the κ-NN graph takes $(\lfloor \log_2(n-1) \rfloor + 1) n\kappa$ bits, where n is the size of the collection.

Baselines. The original work by Bruch *et al.* [5] reports a wide gap between Seismic and all other state-of-the-art sparse retrieval algorithms. As such, rather than comparing our extension of Seismic with all baselines, we only contrast SeismicWave with Seismic.

Hyperparameters. When building Seismic and SeismicWave indexes, we first fix a memory budget as a multiple of the size of the forward index. We then sweep the hyper-parameters as follows to find the best configuration that results in an index no larger than the budget: $\lambda \in \{2000, 2500, 3000, 4000, 5000, 6000\}$, $\beta \in \{\lambda/10, \lambda/5\}$, $\alpha \in \{0.4, 0, 5, 0.6\}$, and for SeismicWave, $\kappa \in \{10, 20, 30, 40, 50\}$. We set cut $\in \{1, 2, 3, 4, 5, 6, 7, 8, 10, 12, 14\}$ and heap_factor $\in \{0.7, 0.8, 0.9, 1.0\}$, and report the best configuration.

Metrics. We use three metrics to evaluate all methods:

- Latency (μsec.). The time to retrieve top-k vectors given a query in single-thread mode. Latency does not include embedding time.
- Accuracy. Percentage of true nearest neighbors recalled.
- Index size (MiB). The space the index occupies in memory.

Hardware Details. We implemented the methods in Rust and compile using Rust version 1.77 with the highest level of optimization made available by the compiler. We conduct experiments on a server equipped with one Intel i9-9900K CPU with a clock rate of 3.60 GHz and 64 GiB of RAM. The CPU has 8 physical cores and 8 hyper-threaded ones. We query the index using a single thread.

Embeddings	Budget	Accuracy (%)	90	91	92	93	94	95	96	97	98	99
Splade	1.5×	Seismic	179 (1.5×)	195 (1.7×)	234 (1.9×)	277 (2.0×)	293 (2.0×)	370 (1.9×)	531 (2.2×)	–	–	–
		SeismicWave	116	116	121	141	148	195	237	–	–	–
	2×	Seismic	206 (1.3×)	206 (1.3×)	206 (1.3×)	206 (1.3×)	233 (1.4×)	257 (1.4×)	305 (1.6×)	336 (1.6×)	441 (1.7×)	669 (1.6×)
		SeismicWave	160	160	160	160	160	177	193	218	258	409
Splade-v3	1.5×	Seismic	183 (1.3×)	205 (1.3×)	223 (1.4×)	261 (1.6×)	290 (1.8×)	323 (1.7×)	417 (1.7×)	612 (2.1×)	–	–
		SeismicWave	131	159	159	159	159	186	234	296	470	–
	2×	Seismic	213 (.93×)	261 (1.1×)	261 (1.1×)	261 (1.1×)	261 (1.1×)	319 (1.4×)	319 (1.4×)	425 (1.6×)	519 (1.9×)	806 (1.7×)
		SeismicWave	229	229	229	229	229	229	229	270	270	460

Table 1: Mean latency (μsec/query) at different accuracy cutoffs with speedup (in parenthesis) gained by SeismicWave on Ms Marco. The "Budget" column indicates the memory budget as a multiple of the size of the forward index.

3.1 Results

Ms Marco. Table 1 compares SeismicWave and Seismic on Ms Marco. For each embedding type, we consider two memory budgets expressed as multiples of the size of the collection (4GB for Splade and 5.6GB for Splade-v3). We choose the best index configuration that respects the budget and report the fastest configuration reaching the accuracy cutoffs from 90% to 99%. We also report the speedup in parenthesis gained by SeismicWave over Seismic.

SeismicWave outperforms Seismic in all the evaluation scenarios. On Splade, SeismicWave achieves 2.2× speedup in the 1.5× setting, and is up to 1.7× faster than Seismic in the 2.0× scenario. On Splade-v3, SeismicWave outperforms Seismic in both memory budgets. As Splade-v3 embeddings are less sparse, the memory impact of storing the κ-NN graph is smaller.

Interestingly, the "exact" algorithm Pisa [23] caps at 99% accuracy due to quantization, which enables significant memory savings. At 99% cutoff, SeismicWave takes 409μs; 1.6× faster than Seismic and *two orders of magnitude* faster than Pisa, which takes 95,818μs.

NQ. We also compare the two methods on NQ. The memory budgets used for Ms Marco—1.5× and 2×—are insufficient to achieve satisfying performance with Seismic. We speculate this to be due to the distributional differences between the two datasets: NQ is about 3× smaller but has more non-zero entries per embedding.

Figure 3 compares Seismic and SeismicWave by latency and accuracy with two indexes per solution. We annotate each solution with its memory budget as a multiple of the size of the forward index. SeismicWave is up to 1.8× faster than Seismic with a slightly lower memory budget (2.4× vs 2.5×). Additionally, SeismicWave reaches 99% with a budget of 2.7× and a speedup of 1.5×, whereas Seismic requires a budget of 3.3× to reach the same accuracy—a saving of approximately 30% of the inverted index overhead.

3.2 Ablation Study

In Figure 4, we break down the impact of our contributions on the Splade embeddings of Ms Marco with a memory budget of 2× (i.e., 8GB). In this figure, OBT denotes Seismic with Ordered Block Traversal; κ-NN refers to Seismic with the κ-NN graph; and, SeismicWave is the combination of Seismic, OBT, and κ-NN graph.

It is clear that, OBT improves the efficiency of Seismic for lower accuracy cutoffs, but may slightly reduce the maximum achievable accuracy as a result of skipping more blocks in the first list. This degradation is well-compensated by the addition of the κ-NN graph, which reaches almost-exact search. The combination of OBT and the κ-NN graph (SeismicWave) yields the best performance overall, with a further improvement of 10% over the κ-NN graph.

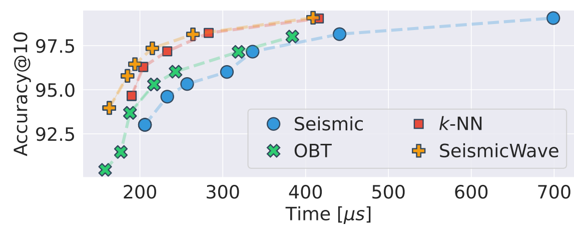

Figure 4: Impact of components of SeismicWave with memory budget of 2× (8GB) on Splade embeddings of Ms Marco.

4 CONCLUSIONS AND FUTURE WORK

We introduced two algorithmic improvements to Seismic [5], a state-of-the-art retrieval algorithm designed for learned sparse representations. In particular, we modified how inverted lists are traversed, and augmented Seismic with global information in the form of a κ-NN graph. As we empirically demonstrated, fo a fixed memory budget, our method, named SeismicWave, outperforms Seismic's latency by a factor of 2 or less. The advantage of SeismicWave is especially evident if a higher accuracy is desired.

We did not use any form of compression in SeismicWave. We leave an empirical examination of the effect of compression and quantization on index size, latency, and accuracy to future work.

5 ACKNOWLEDGEMENTS

This work was partially supported by the Horizon Europe RIA "Extreme Food Risk Analytics" (EFRA), grant agreement n. 101093026, by the PNRR - M4C2 - Investimento 1.3, Partenariato Esteso PE00000013 - "FAIR - Future Artificial Intelligence Research" - Spoke 1 "Human-centered AI" funded by the European Commission under the NextGeneration EU program, by the PNRR ECS00000017 Tuscany Health Ecosystem Spoke 6 "Precision medicine & personalized healthcare" funded by the European Commission under the NextGeneration EU program, by the PNRR IR0000013 "SoBigData.it - Strengthening the Italian RI for Social Mining and Big Data Analytics" funded by the European Commission under the NextGeneration EU program, by the MUR-PRIN 2017 "Algorithms, Data Structures and Combinatorics for Machine Learning", grant agreement n. 2017K7XPAN_003, and by the MUR-PRIN 2022 "Algorithmic Problems and Machine Learning", grant agreement n. 20229BCXNW.

REFERENCES

[1] Sebastian Bruch. 2024. *Foundations of Vector Retrieval*. Springer Nature Switzerland.
[2] Sebastian Bruch, Siyu Gai, and Amir Ingber. 2023. An Analysis of Fusion Functions for Hybrid Retrieval. *ACM Transactions on Information Systems* 42, 1, Article 20 (August 2023), 35 pages.
[3] Sebastian Bruch, Franco Maria Nardini, Amir Ingber, and Edo Liberty. 2023. An Approximate Algorithm for Maximum Inner Product Search over Streaming Sparse Vectors. *ACM Transactions on Information Systems* 42, 2, Article 42 (November 2023), 43 pages.
[4] Sebastian Bruch, Franco Maria Nardini, Amir Ingber, and Edo Liberty. 2024. Bridging Dense and Sparse Maximum Inner Product Search. *ACM Transactions on Information Systems* (2024). *(to appear)*.
[5] Sebastian Bruch, Franco Maria Nardini, Cosimo Rulli, and Rossano Venturini. 2024. Efficient Inverted Indexes for Approximate Retrieval over Learned Sparse Representations. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Washington, DC, USA).
[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 4171–4186.
[7] Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2021. SPLADE v2: Sparse Lexical and Expansion Model for Information Retrieval. arXiv:2109.10086 [cs.IR]
[8] Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2022. From Distillation to Hard Negative Sampling: Making Sparse Neural IR Models More Effective. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Madrid, Spain). 2353–2359.
[9] Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2023. Towards Effective and Efficient Sparse Neural Information Retrieval. *ACM Transactions on Information Systems* (December 2023).
[10] Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021. SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada). 2288–2292.
[11] N. Jardine and C.J. van Rijsbergen. 1971. The use of hierarchic clustering in information retrieval. *Information Storage and Retrieval* 7, 5 (1971), 217–240.
[12] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. 6769–6781.
[13] Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China). 39–48.
[14] Hrishikesh Kulkarni, Sean MacAvaney, Nazli Goharian, and Ophir Frieder. 2023. Lexically-Accelerated Dense Retrieval. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Taipei, Taiwan). 152–162.
[15] Carlos Lassance and Stéphane Clinchant. 2022. An Efficiency Study for SPLADE Models. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Madrid, Spain). 2220–2226.
[16] Carlos Lassance, Hervé Déjean, Thibault Formal, and Stéphane Clinchant. 2024. SPLADE-v3: New baselines for SPLADE. arXiv:2403.06789 [cs.IR]
[17] Jimmy Lin, Rodrigo Frassetto Nogueira, and Andrew Yates. 2021. *Pretrained Transformers for Text Ranking: BERT and Beyond*. Morgan & Claypool Publishers.
[18] Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, Nazli Goharian, and Ophir Frieder. 2020. Expansion via Prediction of Importance with Contextualization. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China). 1573–1576.
[19] Sean MacAvaney, Nicola Tonellotto, and Craig Macdonald. 2022. Adaptive Re-Ranking with a Corpus Graph. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (Atlanta, GA, USA). 1491–1500.
[20] Joel Mackenzie, Andrew Trotman, and Jimmy Lin. 2021. Wacky Weights in Learned Sparse Representations and the Revenge of Score-at-a-Time Query Evaluation. arXiv:2110.11540 [cs.IR]
[21] Yu A. Malkov and D. A. Yashunin. 2020. Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42, 4 (4 2020), 824–836.
[22] Antonio Mallia, Joel Mackenzie, Torsten Suel, and Nicola Tonellotto. 2022. Faster Learned Sparse Retrieval with Guided Traversal. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Madrid, Spain). 1901–1905.
[23] Antonio Mallia, Michal Siedlaczek, Joel Mackenzie, and Torsten Suel. 2019. PISA: Performant Indexes and Search for Academia. In *Proceedings of the Open-Source IR Replicability Challenge co-located with 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, OSIRRC@SIGIR 2019, Paris, France*. 50–56.
[24] Franco Maria Nardini, Cosimo Rulli, and Rossano Venturini. 2024. Efficient Multi-vector Dense Retrieval with Bit Vectors. In *Advances in Information Retrieval*. 3–17.
[25] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. (November 2016).
[26] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.
[27] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3.. In *TREC (NIST Special Publication, Vol. 500-225)*, Donna K. Harman (Ed.). National Institute of Standards and Technology (NIST), 109–126.
[28] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. ColBERTv2: Effective and Efficient Retrieval via Lightweight Late Interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 3715–3734.
[29] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models. In *35th Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.
[30] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation.
[31] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. In *International Conference on Learning Representations*.