Optimistic Query Routing in Clustering-based Approximate Maximum Inner Product Search

Sebastian Bruch
Northeastern University
Boston, MA, USA
s.bruch@northeastern.edu

Aditya Krishnan
Microsoft
New York, NY, USA
adkrishnan@microsoft.com

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

Abstract

Clustering-based nearest neighbor search algorithms partition points into shards to form an index, and search only a subset of shards to process a query. Even though search efficacy is heavily influenced by the algorithm that identifies the shards to probe, it has received little attention in the literature. We study routing in clustering-based maximum inner product search, which includes cosine similarity search. We unpack existing routers and notice the surprising role of optimism. We then take a page from the sequential decision making literature and formalize that insight following the principle of "optimism in the face of uncertainty." In particular, we present a framework that incorporates the moments of the distribution of inner products within each shard to estimate the maximum inner product. We then develop a practical instance of our algorithm that uses only the first two moments to reach the same accuracy as state-of-the-art routers by probing up to 50% fewer points on benchmark datasets without compromising efficiency. Our algorithm is also space-efficient: we design a sketch of the second moment whose size is independent of the number of points and requires $\mathcal{O}(1)$ vectors per shard.

1 Introduction

A fundamental operation in many applications of machine learning is nearest neighbor search [Bruch, 2024]: Given m points denoted $\mathcal{X} \subset \mathbb{R}^d$, it finds the k closest points to a query $q \in \mathbb{R}^d$, where closeness is by vector similarity or distance. We focus on inner product, giving the problem of Maximum Inner Product Search (MIPS), which includes Cosine Similarity search as a special case:

$$\mathcal{S} = \arg\max_{u \in \mathcal{X}}^{(k)} \quad \langle q, u \rangle. \tag{1}$$

As m or d grows, an exact solution is often difficult to obtain within a reasonable budget. The problem is thus relaxed to its *approximate* variant, where we tolerate error to gain speed. The effectiveness of Approximate Nearest Neighbor (ANN) search is characterized by *recall* or *accuracy*, defined as the fraction of true nearest neighbors recalled: $|\tilde{\mathcal{S}} \cap \mathcal{S}|/k$, $\tilde{\mathcal{S}}$ being the set returned by the ANN algorithm.

Clustering-based ANN search: ANN algorithms come in various flavors, from trees [Bentley, 1975, Dasgupta and Sinha, 2015], LSH [Indyk and Motwani, 1998], to graphs [Malkov and Yashunin, 2020, Jayaram Subramanya et al., 2019]. Refer to [Bruch, 2024] for a review of this subject. The method

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

relevant to us is the clustering-based approach, also known as Inverted File, which has received much attention in the literature [Auvolat et al., 2015, Babenko and Lempitsky, 2012, Chierichetti et al., 2007, Bruch et al., 2024, Douze et al., 2024] and is widely adopted in industrial applications.[1]

In this paradigm, $\mathcal{X}$ is partitioned into C *shards* using a clustering function $\zeta : \mathbb{R}^d \to [C]$—typically KMeans with $C = \mathcal{O}(\sqrt{m})$. Accompanying this *index* is a *router* $\tau : \mathbb{R}^d \to [C]^\ell$, which returns ℓ shards likely to contain q's nearest neighbors. If μ_i is the *mean* of the i-th shard, a common router is:

$$\tau(q) = \arg\max^{(\ell)}_{i \in [C]} \quad \langle q, \mu_i \rangle. \tag{2}$$

Processing a query q involves two *independent* subroutines. The first, "routing," returns ℓ shards using $\tau(q)$, and the subsequent step, "scoring," computes inner products between q and the points in the ℓ shards. While the literature has focused on scoring [Jégou et al., 2011, Ge et al., 2014, Wu et al., 2017, Andre et al., 2021, Kalantidis and Avrithis, 2014, Johnson et al., 2021, Norouzi and Fleet, 2013], routing has received little attention. This work turns squarely to the routing step.

We emphasize that, routing is *independent* of the scoring algorithm. For example, we may route using Equation (2) or our novel router to be introduced later. This choice has no algorithmic bearing on scoring whatsoever. In other words, scoring can be accomplished by a linear scan of the chosen shards, computing inner products with Product Quantization (PQ) [Jégou et al., 2011], or searching each shard using a graph ANN algorithm Jayaram Subramanya et al. [2019], Malkov and Yashunin [2020]. All these choices are independent of the type of router utilized in the earlier stage.

The importance of routing: The historical focus on scoring makes sense. Even though routing narrows down the search space, selected shards may contain a large number of points. Scoring must thus be efficient and effective.

We argue that the routing step is important in its own right. First, a more accurate[2] router leads to fewer data points being scored. For example, if shards are balanced in size, access to an oracle router (i.e., one that identifies the shard with the true nearest neighbor) means that the scoring stage must only compute inner products for m/C points to find the top-1 point.

The second reason pertains to scale. As size and dimensionality grows, it is often infeasible to keep the entire index in memory, compression techniques such as PQ notwithstanding. Much of the index must instead rest on secondary storage—cheap but high-latency storage such as disk or blob storage—and accessed when necessary. That line of reasoning has led to the emergence of disk-based graph indexes [Jayaram Subramanya et al., 2019, Singh et al., 2021, Jaiswal et al., 2022].

Translating the same rationale to the clustering-based paradigm implies that shards rest outside of memory, and when a router identifies a subset of shards, scoring must first fetch those shards from storage. This paradigm has gained traction in real-world database systems.[3] This is the context in which we present our research: Storage-backed, clustering-based ANN search systems.

In this framework, a more accurate router lowers the volume of data transferred between storage and memory with implications for storage bandwidth and in-memory cache space utilization, and retrieval throughput. We have included an illustrative example in Appendix A to support this statement.

Existing routers and the role of optimism: We review the relevant literature on routers in Appendix B. Among existing routers, Equation (2), which we call **MEAN**, is the most prominent. It summarizes each shard by its mean point (μ_i's). This is not an unreasonable choice as the mean is the minimizer of the sum of squared deviations from the mean, and is, in fact, natural if ζ is KMeans.

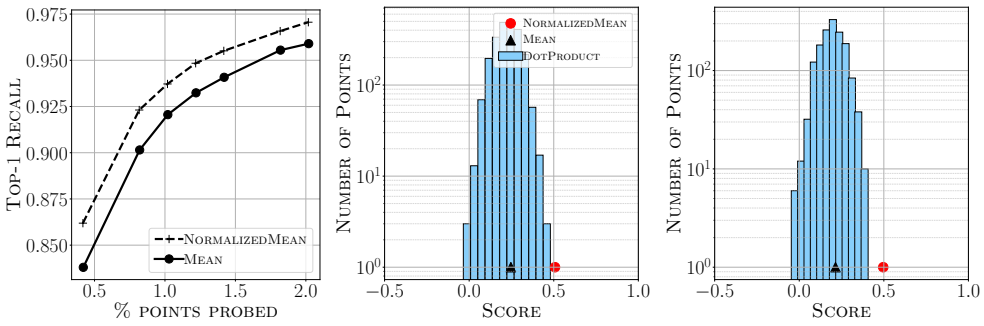
Another common router, which we call **NORMALIZEDMEAN**, belongs to the same family, but its shard representatives are the L_2-normalized means, rather than the unnormalized mean vectors:

$$\tau(q) = \arg\max^{(\ell)}_{i \in [C]} \quad \langle q, \frac{\mu_i}{\|\mu_i\|_2} \rangle. \tag{3}$$

[1] See, for example, `https://turbopuffer.com/blog/turbopuffer`, `https://www.pinecone.io/blog/serverless-architecture/#Pinecone-serverless-architecture`, and `https://research.google/blog/announcing-scann-efficient-vector-similarity-search/`.

[2] We quantify a router's accuracy by the ANN recall in a setup where scoring is by a linear scan over the identified shards, so that the only source of error is the routing procedure.

[3] c.f., documents linked in the footnote above.



(a) (b) (c)

Figure 1: (a) Top-1 recall vs. percentage of points probed on TEXT2IMAGE where points have varying norms; (b) and (c) Distribution of inner products between a shard and a query on GLOVE. Overlaid are scores computed by MEAN and NORMALIZEDMEAN.

This formulation is inherited from *spherical* KMeans [Dhillon and Modha, 2001], which is identical to standard KMeans except that cluster centroids are normalized after every iteration. Because we can assume that $\|q\|_2 = 1$ without loss of generality, Equation (3) routes by the angle between q and μ_i's.

NORMALIZEDMEAN seems appropriate for search over a sphere: When norms are constant, only angles matter in Equation (1), making it reasonable to rank shards by the angle between means and q. Intriguingly, in many circumstances that deviate from that situation, NORMALIZEDMEAN performs more accurately than MEAN, as evidenced in Figure 1(a). Let us unpack this phenomenon.

Consider q and a single shard $\mathcal{P}$ with mean μ. Figure 1(b) visualizes the distribution of inner products of q and all points in $\mathcal{P}$. Overlaid is the inner product of q and μ, as well as their normalized mean. We observe that $\langle q, \mu/\|\mu\|_2 \rangle$ lands in the right tail of the distribution.

That is not surprising when $\|u\|_2 = 1$ for all $u \in \mathcal{P}$. Clearly $\|\mu\|_2 \leq 1$, so that $\langle q, \mu/\|\mu\|_2 \rangle = \langle q, \mu \rangle / \|\mu\|_2 \geq \langle q, \mu \rangle$; NORMALIZEDMEAN amplifies MEAN by a factor of $1/\|\mu\|_2$. What is interesting, however, is that the magnitude of this "boost" correlates with the variance of $\mathcal{P}$: The more concentrated $\mathcal{P}$ is around the direction of μ, the closer μ is to the surface of the sphere, making $\|\mu\|_2$ larger, so that NORMALIZEDMEAN applies a smaller boost to $\langle q, \mu \rangle$. The opposite is true when points are spread out: shards with a higher variance receive a larger lift by NORMALIZEDMEAN.

There are a few caveats. First, it is hard to explain the behavior on sets with varying norms. Second, as we observe in Figure 1(c), NORMALIZEDMEAN's estimate can be overly aggressive. Nonetheless, the insight that a router's score for a shard can be influenced by the shard's variance is worth exploring.

Contributions and outline: From the exercise above, we learned that NORMALIZEDMEAN paints an *optimistic* picture of what the maximum inner product of q and $\mathcal{P}$ could be. MEAN, however, gives a *conservative* estimate. We investigate the ramifications of that insight in this work.

In particular, the research question we study is whether a more principled approach to designing optimistic routers leads to more accurate routing on vector sets with variable norms. This question is not unlike those asked in the online learning literature [Lattimore and Szepesvári, 2020], so it is not surprising that our answer draws from the "principle of optimism in the face of uncertainty."

We apply the Optimism Principle to routing in clustering-based MIPS. Our method, presented in Section 2, uses the concentration of inner products between a query q and points in a shard. In particular, we estimate a score, θ_i, for the i-th shard such that, with some confidence, the maximum inner product of q with points in that shard is at most θ_i. Shards are then ranked by this score. When $\theta_i = \mu_i$, we recover the MEAN router, and when $\theta_i > \mu_i$ routing is optimistic with a controllable degree of optimism. Importantly, our routing function is independent of how shards are formed.

We outline a general framework that can incorporate as much information as is available about the data distribution to estimate θ_i's. We then present a concrete, assumption-free instance of our algorithm that uses the first and second moments of the empirical inner product distribution only.

Furthermore, we make the resulting algorithm space-efficient by designing a sketch [Woodruff, 2014] of the second moment. The end-result is a practical algorithm that is straightforward to implement.

We test our proposal in Section 3 on a variety of ANN benchmark datasets. As our experiments show, our optimistic router achieves the same recall as state-of-the-art routers but with up to a 50% reduction in the total volume of points evaluated per query. We conclude this work in Section 4.

2 Routing by the Optimism Principle: our proposal

As we observed, NORMALIZEDMEAN is an optimistic estimator, though its behavior is unpredictable. Our goal is to design an optimistic estimator that is statistically principled, thus well-behaved. Throughout this section, all discussions are in the context of a fixed unit-vector q. We denote the i-th shard by $\mathcal{P}_i$, and write S_i for the set of inner products of q and points in $\mathcal{P}_i$: $S_i = \{\langle q, u\rangle : u \in \mathcal{P}_i\}$.

2.1 Formalizing the notion of optimism

We wish to find the *smallest* threshold $\theta_i \geq \langle q, \mu_i\rangle$ for the i-th shard such that the probability that a sample from S_i falls to the left of θ_i is at least $(1+\delta)/2$, for some arbitrary $\delta \in [0, 1]$. Formally, we aim to compute a solution θ_i to the the following optimization problem:

Problem 1 (Optimistic estimator of the maximum inner product in S_i)**.**

$$\inf\{\theta_i : \theta_i \geq \langle q, \mu_i\rangle\} \quad \textit{such that} \quad \Pr_{s\sim S_i}[s \leq \theta_i] \geq \frac{1+\delta}{2}.$$

The optimal θ_i is a probabilistic upper-bound on the maximum attainable value in S_i; that is, with some confidence, we can assert that no value in S_i is greater than θ_i. We then route q by sorting $\mathcal{P}_i$'s by their θ_i in descending order and select the top shards. Our router, OPTIMIST, is defined as follow:

$$\tau(q) = \arg\overset{(\ell)}{\max}_{i\in[C]} \quad \theta_i. \tag{4}$$

Suppose that we have the solution to Problem 1. As $\delta \to 0$, the optimal solution approaches $\theta_i = \max(\langle q, \mu_i\rangle, \langle q, M_i\rangle)$, where M_i is the median of $\mathcal{P}_i$. That is a *conservative* estimate of the maximum inner product. As $\delta \to 1$, θ_i becomes larger, rendering Equation (4) optimistic. At the extreme, the optimal θ_i is the maximum inner product itself—the most optimistic we can get.

Clearly δ controls the degree of optimism. When the distribution is fully known, $\delta = 1$ is an appropriate choice: If we know the distribution of inner products, we can expect to be fully confident about the maximum inner product. But when very little about the distribution is known, then $\delta = 0$ is a sensible choice. In effect, the value of δ reflects our knowledge of the data distribution.

What is left to address is the solution to Problem 1, which is the topic of the remainder of this section. We first describe a general approach that uses as much information as available about the data distribution. We then present a practical approach that is the foundation of the rest of this work.

2.2 General solution

Let $\mathcal{D}$ be an unknown distribution over $\mathbb{R}^d$ from which $\mathcal{P}_i$ is sampled. It is clear that, if we are able to accurately approximate the quantiles of S_i, we can obtain an estimate θ_i satisfying Problem 1.

We motivate our approach by considering a special case where $\mathcal{D}$ is $\mathcal{N}(\mu_i, \Sigma_i)$, a Gaussian with mean $\mu_i \in \mathbb{R}^d$ and covariance $\Sigma_i \in \mathbb{R}^{d\times d}$. In this case, S_i follows a univariate Gaussian distribution with mean $\langle q, \mu_i\rangle$ and variance $q^\top \Sigma_i q$, so that the solution to Problem 1 is simply:

$$\theta_i = \langle q, \mu_i\rangle + \sqrt{q^\top \Sigma_i q} \cdot \Phi^{-1}_{\mathcal{N}(0,1)}\left(\frac{1+\delta}{2}\right), \tag{5}$$

which follows by writing the cumulative distribution function (CDF) of S_i in terms of the CDF of a unit Gaussian, denoted $\Phi_{\mathcal{N}(0,1)}$. Notice that, we first modeled the moments of S_i using the moments of $\mathcal{D}$, then approximated the CDF (equivalently, the quantile function) of the distribution of S_i from its moments—in this special case, the approximation with the first two moments is, in fact, exact.

Our general solution follows that same logic. In the first step, we can obtain the first r moments of the distribution of S_i, denoted $m_j(S_i)$ for $j \in [r]$, from the moments of $\mathcal{D}$, denoted $m_j(\mathcal{D})$. In a subsequent step, we use $m_j(S_i)$'s to approximate the CDF of S_i.

It is easy to see that the j-th moment of S_i can be written as follows:

$$m_j(S_i) = q^{\otimes j} \odot m_j(\mathcal{D}) \approx \frac{1}{n} \cdot q^{\otimes j} \odot \sum_{u_1,\ldots,u_n \sim \mathcal{D}} x^{\otimes j},$$

where $^{\otimes j}$ is the j-fold tensor product, and $\odot$ tensor inner product.

In a second step, we find a distribution $\tilde{S}_i$ such that $m_j(\tilde{S}_i) \approx m_j(S_i)$ for all $j \in [r]$. This can be done using the "method of moments" Pearson [1936] which offers guarantees [Kong and Valiant, 2017, Braverman et al., 2022] in terms of a distributional distance such as the Wasserstein-1 distance.

While using higher-order moments leads to a better approximation, computing $m_j(\cdot)$ for $j > 2$ can be prohibitive considering the dimensionality of real datasets, as the space requirement to store $m_j(\mathcal{D})$ grows as d^j. We leave an exploration of an efficient version of this two-step approach as future work.

2.3 Practical solution via concentration inequalities and sketching

Noting that Problem 1 is captured by the concept of concentration of measure, we resort to results from that literature to find acceptable estimates of θ_i's. In particular, we obtain a solution via an application of the one-sided Chebyshev's inequality, resulting in the following lemma.

Lemma 1. *Denote by μ_i and Σ_i the mean and covariance of the distribution of $\mathcal{P}_i$. An upper-bound on the solution to Problem 1 for $\delta \in (0, 1)$ is:*

$$\theta_i = \langle q, \mu_i \rangle + \sqrt{\frac{1+\delta}{1-\delta}\, q^\top \Sigma_i q}. \tag{6}$$

Proof. The result follows immediately by applying the one-sided Chebyshev's inequality to the distribution, S_i, of inner products between q and points in $\mathcal{P}_i$:

$$\Pr_{s \sim S_i}[s - \langle q, \mu_i \rangle \leq \epsilon] \geq 1 - \frac{q^\top \Sigma_i q}{q^\top \Sigma_i q + \epsilon^2} = \frac{1+\delta}{2} \implies \epsilon^2 = \frac{1+\delta}{1-\delta}\, q^\top \Sigma_i q.$$

Rearranging the terms to match the expression of Problem 1 gives $\theta_i = \langle q, \mu_i \rangle + \epsilon$, as desired. □

We emphasize that Lemma 1 gives a loose upper-bound on the optimal value of θ_i. Because we make no assumptions about the data distribution, the gap between the optimal and estimated θ_i cannot be expressed in closed form. Had we assumed S_i is sub-Gaussian, it would be easy to bound the gap between θ_i obtained from Lemma 1 and the one obtained from sub-Gaussian concentration bounds. Nonetheless, we find that in practice even this sub-optimal solution proves effective.

There is still one challenge with Equation (6): Σ_i can be too large to store as the matrix grows as d^2 for each partition. Contrast that with the cost of MEAN and NORMALIZEDMEAN which only need one d-dimensional vector per partition. To remedy this, we design a compact approximation of Σ_i that can be plugged into Equation (6) to replace Σ_i. We describe that next to complete our algorithm.

2.3.1 Approximating the covariance matrix

Since the procedure we describe is independently applied to each partition, we drop the subscript in Σ_i and focus on a single partition. We seek a matrix $\Sigma^* \in \mathbb{R}^{d \times d}$ that approximates the positive semi-definite (PSD) matrix Σ, by minimizing the following standard approximation error:

$$\text{err}(\Sigma, \Sigma^*) \triangleq \sup_{\|v\|_2 = 1} |v^\top \Sigma v - v^\top \Sigma^* v|. \tag{7}$$

As we seek a *compact* Σ^*, we constrain the solution space to rank-t matrices: $\text{rank}(\Sigma^*) = t$, $t \ll d$.

A standard solution is a result of the Eckhart-Young-Mirsky Theorem: Let $\Sigma = V \Lambda V^\top$ be the eigendecomposition of Σ. Then, $\Sigma^* = [V\sqrt{\Lambda}]_t [V\sqrt{\Lambda}]_t^\top$, where $[\cdot]_t$ selects the first t columns of its argument, optimally approximates Σ under the stated constraints. We denote this solution by Σ_t^{LR}.

Algorithm 1 Indexing and scoring a single partition with OPTIMIST

Input: Partition $\mathcal{P}$ and target rank $t \ll d$.
procedure BUILDROUTERFORPARTITION($\mathcal{P}, t$)
  $\mu \leftarrow \frac{1}{|\mathcal{P}|}\sum_{u\in\mathcal{P}} u$
  $\Sigma \leftarrow \frac{1}{|\mathcal{P}|}\sum_{u\in\mathcal{P}}(u-\mu)(u-\mu)^\top$
  $R_\circ = D^{-1/2}(\Sigma - D)D^{-1/2}$ ▷ D is the diagonal of Σ
  Find eigendecomposition $R_\circ = Q\Lambda Q^\top$
  Sort columns of Q, Λ in non-increasing order
  $\Lambda_t \leftarrow [\Lambda]_t$, $Q_t \leftarrow [Q]_t$
  return $\{\mu, D, \Lambda_t, Q_t\}$
Input: Query $q \in \mathbb{R}^d$ and optimism parameter $\delta > 0$.
procedure SCOREPARTITIONFORQUERY(q, δ; $\{\mu, D, \Lambda_t, Q_t\}$)
  $\tilde{q} \leftarrow q \circ \text{diag}(D^{1/2})$ ▷ Element-wise product
  return $\langle q, \mu\rangle + \sqrt{\frac{1+\delta}{1-\delta}\cdot(\|\tilde{q}\|_2^2 + \tilde{q}^\top Q_t\Lambda_t Q_t^\top \tilde{q})}$

We could stop here and use Σ_t^{LR} in lieu of Σ in Equation (6). However, while that would be the optimal choice in *the general case*, for real-world datasets, we show that we can find a better approximation.

Definition 1 (Masked Sketch of Rank t)**.** *Rewrite Σ as follows, with D its diagonal and $R = \Sigma - D$:*

$$\Sigma = D + R = D^{\frac{1}{2}}(I + \underbrace{D^{-\frac{1}{2}}RD^{-\frac{1}{2}}}_{R_\circ})D^{\frac{1}{2}}.$$

Let $Q\Lambda Q^\top$ be $R_\circ$'s eigendecomposition. We call the following the Masked Sketch of Rank t of Σ:

$$\Sigma_t^{\text{MS}} = D + D^{\frac{1}{2}}[Q]_t[\Lambda]_t[Q]_t^\top D^{\frac{1}{2}}. \tag{8}$$

We next establish that, under certain assumptions, $\text{err}(\Sigma, \Sigma_t^{\text{MS}})$ is lower than $\text{err}(\Sigma, \Sigma_t^{\text{LR}})$.

Lemma 2. *Let $\Sigma \in \mathbb{R}^{d\times d}$ be a PSD matrix with diagonal D for which $\min_{i\in[d]} D_{ii}/\max_{i\in[d]} D_{ii} \geq 1-\epsilon$ for some $\epsilon > 0$. For every $1 \leq t \leq d-1$ such that the $(t+1)$-th eigenvalue of $D^{-1/2}\Sigma D^{-1/2}$ is greater than 1, we have that: $err(\Sigma, \Sigma_t^{\text{MS}}) \leq err(\Sigma, \Sigma_t^{\text{LR}})/(1-\epsilon)$.*

We give the proof in Appendix C and show that for datasets in this work the assumptions hold.

2.3.2 The final algorithm

Using our solution from Lemma 1 and our sketch defined in (8), we describe our full algorithm in Algorithm 1 for building our router and scoring a partition. Notice that, since we only store $\mu, D, [\Lambda]_t$ and $[Q]_t$, the router requires $t+2$ vectors[4] in $\mathbb{R}^d$ per partition. In our experiments, we choose $t \leq 10$ for all datasets except one and show that much of the performance gains from using the whole covariance matrix can be preserved even by choosing a small value of t independent of d.

2.3.3 Time complexity analysis

The time complexity of Algorithm 1 is dominated by eigendecomposition which takes $\mathcal{O}(nd^2 + d^3)$ time. However, because we only need the top t eigenvectors and values to build the router, we may use highly-efficient algorithms for computing low-rank approximations of PSD matrices such as the randomized Block Krylov Method (see [Tropp, 2018, Section 6.3] and [Musco and Musco, 2015]).

The time complexity of these algorithms is proportional to $\mathcal{O}(nd\log(d)\times t)$, making them much faster than naïve eigendecomposition. Since t is usually a small constant in our setting, Algorithm 1 can be implemented in near linear time complexity in the size of the data.

3 Experimental evaluation

We put our arguments to the test and evaluate OPTIMIST.

[4]Since $[\Lambda]_t$ can be "absorbed" into $[Q]_t$ with some care taken for the signs of the eigenvalues.

Table 1: Dataset statistics (size m and dimensions d), along with the number of partitions (C) and OPTIMIST's default rank configuration (t) in our main experiments.

DATASET	m	d	C	t
TEXT2IMAGE	10M	200	3,000	4
MUSIC	1M	100	1,024	2
DEEPIMAGE	10M	96	3,000	2
GLOVE	1.2M	200	1,024	4
MSMARCO-MINILM	8.8M	384	3,000	8
NQ-ADA2	2.7M	1,536	1,600	30

Table 2: Relative savings during search to achieve a fixed recall. A saving of $x\%$ means that OPTIMIST searches $x\%$ fewer points than NORMALIZEDMEAN.

RECALL	90%	95%
TEXT2IMAGE	23%	22%
MUSIC	38%	54%
GLOVE	11%	5.5%
MSMARCO-MINILM	22%	7.7%
NQ-ADA2	18%	20%

3.1 Setup

Datasets: Table 1 summarizes in the leftmost block the main properties of the datasets used in our experiments. A complete description can be found in Appendix D.

Clustering: For our main results, we partition the datasets with spherical KMeans [Dhillon and Modha, 2001]. We include in Appendices G and H results from similar experiments but where the clustering algorithm is standard KMeans and Gaussian Mixture Model (GMM). We cluster each dataset into $C = \sqrt{m}$ shards, where m is the number of data points in the dataset.

Evaluation: The independence of routing from scoring simplifies the evaluation protocol: By additivity of latency, efficiency gains from a router translate directly into efficiency gains in the end-to-end search (i.e., routing followed by scoring). We can therefore fix the scoring algorithm and examine routers in isolation to compare their accuracy, latency, and other characteristics.

Once a dataset has been partitioned, we fix the partitioning and evaluate all routers on it to facilitate a fair comparison. We evaluate each router τ as follows. For each test query, we identify the set of shards to probe using τ. We then perform an exact search over the selected shards, obtain the top-k points, and compute recall with respect to the ground-truth top-k set. Because the only source of error is the router's inaccuracy, the measured recall gauges the effectiveness of τ.

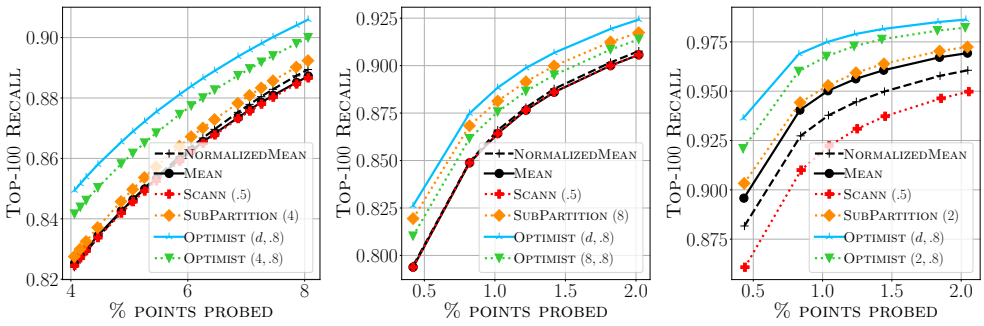
We report recall as a function of the number of *data points* probed, rather than the number of *shards probed*. In this way, a comparison of the efficacy of different routers is unaffected by any imbalance in shard sizes, so that a router cannot trivially outperform another by simply prioritizing larger shards.

Routers: We evaluate the following routers in our experiments:

- MEAN and NORMALIZEDMEAN: Defined in Equations (2) and (3).
- SCANN (T): Similar to MEAN and NORMALIZEDMEAN, but where routing is determined by inner product between a query and the SCANN centroids (c.f., Theorem 4.2 in [Guo et al., 2020]). SCANN has a single hyperparameter T, which we set to 0.5 after tuning.
- SUBPARTITION (t): OPTIMIST stores $t+2$ vectors per partition, where t is the rank in Algorithm 1. Following the KRT method of Gottesbüren et al. [2024], we introduce another baseline where we partition each shard into $t+2$ sub-shards and use the their centroids as the shard's representatives. Routing is based on the maximum inner product of the query with a shard's representatives.
- OPTIMIST (t, δ): t and δ are parameters of Algorithm 1. By default, we set $\delta = 0.8$ and t to values in Table 1, but study their effect in Appendix E. If unspecified, it should be understood that default parameters are used. We write OPTIMIST $(t = d, \cdot)$ to indicate the use of the full covariance matrix.

Code: We have implemented all baseline and proposed routers in the Rust programming language. We have open-sourced[5] our code along with experimental configuration to facilitate reproducibility.

[5]Available at `https://github.com/Artificial-Memory-Lab/optimist-router`



(a) GLOVE (b) MSMARCO-MINILM (c) MUSIC

Figure 2: Top-100 recall vs. volume of probed points. Partitioning is by spherical KMeans. SCANN has parameter T, SUBPARTITION t, and OPTIMIST rank t and degree of optimism δ.

3.2 Main results

Effectiveness: Figure 2 plots recall versus the volume of points examined for select datasets using spherical KMeans. See Appendix F for full results; Appendix G for standard KMeans; and Appendix H for GMM.

Among baselines, NORMALIZEDMEAN generally outperforms MEAN and SCANN, except on MUSIC. OPTIMIST with the full covariance ($t = d$) generally does at least as well as NORMALIZEDMEAN, but often outperforms it significantly. Interestingly, the gains from OPTIMIST widen as k increases. Finally, while OPTIMIST with a rank-t Masked Sketch shows some degradation, it still yields a higher recall than baselines for larger k. SUBPARTITION (t) becomes a strong competitor when k is small.

OPTIMIST shines when data points have varying norms. On MUSIC, at 95% top-100 recall, OPTIMIST needs to probe 54% fewer points than NORMALIZEDMEAN; on average OPTIMIST probes 6,666 points to reach 95% top-100 recall whereas NORMALIZEDMEAN examines 14,463 points.

Table 2 presents savings on all datasets. On DEEPIMAGE, OPTIMIST scans 9% *more* points than NORMALIZEDMEAN to reach 90% recall (55,000 vs 60,000), and 10% more to reach 95% (95,000 vs 105,000). We suspect this is an artifact of the dataset's construction: each point is represented by the top 96 principal components of its original features, leading to unusual partition statistics.

Latency: We now turn to latency which includes routing, fetching the chosen shards from storage, and scoring using PQ with 4-dimensional codebooks. Figure 3 reports latencies for OPTIMIST and NORMALIZEDMEAN. OPTIMIST's gains are more pronounced when shards are stored on blob storage, but even on SSD the gains are substantial. Full results are in Appendix I.

We run experiments on AWS c5.xlarge (4 vCPUs, 8GB memory). Baseline bandwidth of the SSD attached to this machine is 1,150Mbps.[6] Finally, using 4 threads, transferring 4MB of data from blob storage (hosted on Amazon S3) to main memory has P50 latency of 45 milliseconds (ms).[7]

Let us next present in Table 3 a unified view of Figure 3, which reports a breakdown of latency at 95% recall, and Table 2, which shows the relative savings (in terms of the number of data points probed) between OPTIMIST and NORMALIZEDMEAN. It is clear that, with a few exceptions, the overall latency savings are roughly equal to the reduction in the number of points probed. As such, change in the number of points probed is a reasonable proxy to change in overall latency.

Hardware plays an outsize role in latency improvements. For example, a more limited storage bandwidth would boost OPTIMIST's standing. If bandwidth is abundant and number of threads limited, however, OPTIMIST's advantage would become less significant. Other factors that affect latency comparisons include: target recall level; type of compression; and algorithm used for scoring.

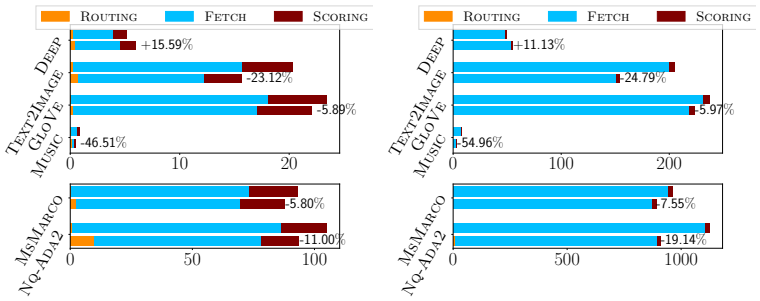
Router size: On most datasets, the router size difference between NORMALIZEDMEAN (1 vector) and OPTIMIST (t vectors) is negligible. On GLOVE, for example, it is 0.8MB versus 4.7MB ($t = 4$)

[6] See `https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-optimized.html`

[7] See `https://github.com/dvassallo/s3-benchmark` for an independent benchmark.

Table 3: Difference between OPTIMIST and NORMALIZEDMEAN in terms of number of points probed and latency. Negative percentages means OPTIMIST leads to gains.

DATASET	# POINTS PROBED	LATENCY (BLOB)	LATENCY (SSD)
TEXT2IMAGE	-22%	-24.8%	-23%
MUSIC	-54%	-55%	-46.5%
DEEPIMAGE	+10%	+11%	+15.6%
GLOVE	-5.5%	-6%	-6%
MSMARCO-MINILM	-7.7%	-7.6%	-5.8%
NQ-ADA2	-20%	-19.1%	-11%

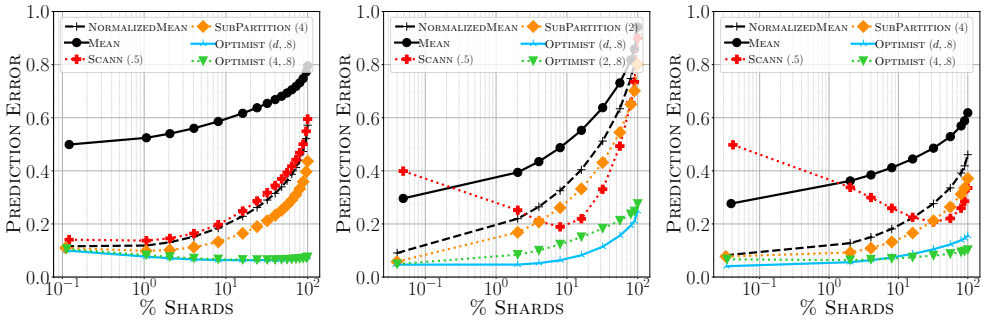


(a) SSD (b) Blob Storage

Figure 3: Mean latency (ms) to reach 95% recall when PQ-compressed shards are on SSD and blob storage. For each dataset, we plot the latency breakdown for NORMALIZEDMEAN (top bar) and OPTIMIST (bottom), and report relative gains (negative value indicates gain by OPTIMIST).

in total. Statistics on other datasets are as follows: MSMARCO-MINILM, 4.4MB vs. 44MB ($t = 8$); MUSIC, 0.4MB vs. 1.6MB ($t = 2$); TEXT2IMAGE, 2.3MB vs. 13.7MB ($t = 4$); DEEPIMAGE, 1.1MB vs. 4.4MB ($t = 2$). The difference is larger on NQ-ADA2: 9.4MB vs 300MB ($t = 30$).

3.3 Maximum inner product prediction



(a) GLOVE (b) DEEPIMAGE (c) TEXT2IMAGE

Figure 4: Mean prediction error $\mathcal{E}_\ell(\tau, \cdot)$ of Equation (9) versus ℓ (percent shards).

We claimed that OPTIMIST is statistically principled. It should thus give more accurate estimates of the maximum inner product for all query-partition pairs. We examine that claim in this section.

Fix a dataset with partitions $\mathcal{P}_i$, router τ, and query q. Write τ_i for the score computed by τ for $\mathcal{P}_i$ and q. τ_i's induce an ordering π among $\mathcal{P}_i$'s, so that $\tau_{\pi_i} \geq \tau_{\pi_{i+1}}$. We quantify the inner product

prediction error as follows, for $\ell \in [C]$:

$$\mathcal{E}_\ell(\tau, q) = \frac{1}{\ell} \sum_{i=1}^{\ell} \left| \frac{\tau_{\pi_i}}{\max_{u \in \mathcal{P}_{\pi_i}} \langle q, u \rangle} - 1 \right|. \tag{9}$$

This error is 0 when τ_i's perfectly match the maximum inner product of q and $\mathcal{P}_i$'s. ℓ allows us to factor in the *rank* of partitions: we can measure the error only for the top ℓ shards according to τ. In this way, if we decide that it is not imperative for a router to accurately predict the maximum inner product in low-ranking shards, we can reflect that choice in our calculation.

We measure Equation (9) on all datasets partitioned by spherical KMeans, and all routers considered in this work. Figure 4 reports the results for select datasets—see Appendix J—where for each choice of ℓ, we plot $\mathbb{E}_q[\mathcal{E}_\ell(\tau, q)]$ for all queries. For most routers error grows as $\ell \to C$. OPTIMIST$(t = d, \cdot)$ degrades less severely. Remarkably, MUSIC excepted, when $t \ll d$, the same pattern persists.

4 Concluding remarks

We studied clustering-based maximum inner product search where points are clustered into shards during indexing. Search involves two independent subroutines: a *router* computes a score for each shard with respect to the query and returns the top ℓ shards to probe; then a *scorer* computes (approximate) scores for points in those ℓ shards. We considered a storage-backed system where shards do not rest in memory, but are stored on external storage and must be fetched into memory when a router identifies them. This is an important paradigm for nearest neighbor search at scale.

Within this framework, we focused on routing. We motivated our work by an unusual routing behavior: NORMALIZEDMEAN computes a score that over-estimates the maximum inner product between a query and points in a shard. Interestingly, the extent of over-estimation correlates with the variance of the shard: The more spread-out the points are, the more optimistic the router becomes.

We took the insight that variance should play a role in routing and developed a principled optimistic algorithm, called OPTIMIST, whose score for a shard is a more accurate estimate of the maximum inner product—as confirmed in Figure 4, resulting in smaller ℓ's to achieve a fixed recall. As Figure 3 shows OPTIMIST is particularly attractive when shards rest on some external, high-latency storage.

OPTIMIST has implications beyond latency. Transferring a smaller volume of data to memory means that each query needs a smaller in-memory cache. As a result, cache can be shared among more queries, leading to a higher throughput in systems where memory availability is the main bottleneck.

Our research takes a first step in exploring unsupervised routing for clustering-based MIPS, and identifies a trade-off space that has not been explored before. We have more to do, however. We leave to future work an exploration of a more compact sketch of the covariance; and, an efficient realization of our general solution outlined in Section 2.2.

Acknowledgements. This work was partially supported by the Horizon Europe RIA "EFRA - Extreme Food Risk Analytics" (grant agreement n. 101093026) and the PNRR - M4C2 - Investimento 1.3, Partenariato Esteso PE00000013 - "FAIR - Future Artificial Intelligence Research" - Spoke 1 "Human-centered AI". Horizon Europe and the PNRR programs are funded by the European Commission under the NextGeneration EU program. The views and opinions expressed are solely those of the authors and do not necessarily reflect those of the European Union, nor can the European Union be held responsible for them.

References

Nir Ailon and Bernard Chazelle. The fast johnson–lindenstrauss transform and approximate nearest neighbors. *SIAM Journal on computing*, 39(1):302–322, 2009.

Nir Ailon and Edo Liberty. An almost optimal unrestricted fast johnson-lindenstrauss transform. *ACM Transactions on Algorithms (TALG)*, 9(3):1–12, 2013.

Fabien Andre, Anne-Marie Kermarrec, and Nicolas Le Scouarnec. Quicker adc: Unlocking the hidden potential of product quantization with simd. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(5):1666–1677, 5 2021.

Alex Auvolat, Sarath Chandar, Pascal Vincent, Hugo Larochelle, and Yoshua Bengio. Clustering is efficient for approximate maximum inner product search, 2015.

Artem Babenko and Victor Lempitsky. The inverted multi-index. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3069–3076, 2012.

Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 9 1975.

Vladimir Braverman, Aditya Krishnan, and Christopher Musco. Sublinear time spectral density estimation. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1144–1157, 2022.

Sebastian Bruch. *Foundations of Vector Retrieval*. Springer Nature Switzerland, 2024.

Sebastian Bruch, Franco Maria Nardini, Amir Ingber, and Edo Liberty. Bridging dense and sparse maximum inner product search. *ACM Transactions on Information Systems*, 42(6):1–38, August 2024.

Flavio Chierichetti, Alessandro Panconesi, Prabhakar Raghavan, Mauro Sozio, Alessandro Tiberi, and Eli Upfal. Finding near neighbors through cluster pruning. In *Proceedings of the 26th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 103–112, 2007.

Chih-Yi Chiu, Amorntip Prayoonwong, and Yin-Chih Liao. Learning to index for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(8):1942–1956, 2020.

Sanjoy Dasgupta and Kaushik Sinha. Randomized partition trees for nearest neighbor search. *Algorithmica*, 72(1):237–263, 5 2015.

Inderjit S. Dhillon and Dharmendra S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1):143–175, January 2001.

Yihe Dong, Piotr Indyk, Ilya Razenshteyn, and Tal Wagner. Learning space partitions for nearest neighbor search. In *International Conference on Learning Representations*, 2020.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The faiss library, 2024.

Tiezheng Ge, Kaiming He, Qifa Ke, and Jian Sun. Optimized product quantization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(4):744–755, 2014.

Lars Gottesbüren, Laxman Dhulipala, Rajesh Jayaram, and Jakub Lacki. Unleashing graph partitioning for large-scale nearest neighbor search, 2024.

Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. Accelerating large-scale inference with anisotropic vector quantization. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 604–613, 1998.

Shikhar Jaiswal, Ravishankar Krishnaswamy, Ankit Garg, Harsha Vardhan Simhadri, and Sheshansh Agrawal. Ood-diskann: Efficient and scalable graph anns for out-of-distribution queries, 2022.

Suhas Jayaram Subramanya, Fnu Devvrit, Harsha Vardhan Simhadri, Ravishankar Krishnawamy, and Rohan Kadekodi. Diskann: Fast accurate billion-point nearest neighbor search on a single node. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Herve Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547, 2021.

Yannis Kalantidis and Yannis Avrithis. Locally optimized product quantization for approximate nearest neighbor search. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2329–2336, 2014.

Weihao Kong and Gregory Valiant. Spectrum estimation from samples. *The Annals of Statistics*, 45 (5):2218–2247, 2017.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019.

Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

Yu A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836, 4 2020.

Stanislav Morozov and Artem Babenko. Non-metric similarity graphs for maximum inner product search. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Cameron Musco and Christopher Musco. Randomized block krylov methods for stronger and faster approximate singular value decomposition. *Advances in neural information processing systems*, 28, 2015.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. Ms marco: A human generated machine reading comprehension dataset, November 2016.

Mohammad Norouzi and David J. Fleet. Cartesian k-means. In *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3017–3024, 2013.

Karl Pearson. Method of moments and method of maximum likelihood. *Biometrika*, 28(1/2):34–59, 1936.

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, Doha, Qatar, October 2014.

Harsha Vardhan Simhadri, George Williams, Martin Aumüller, Matthijs Douze, Artem Babenko, Dmitry Baranchuk, Qi Chen, Lucas Hosseini, Ravishankar Krishnaswamny, Gopal Srinivasa, Suhas Jayaram Subramanya, and Jingdong Wang. Results of the neurips'21 challenge on billion-scale approximate nearest neighbor search. In *Proceedings of the NeurIPS 2021 Competitions and Demonstrations Track*, volume 176 of *Proceedings of Machine Learning Research*, pages 177–189, Dec 2022.

Aditi Singh, Suhas Jayaram Subramanya, Ravishankar Krishnaswamy, and Harsha Vardhan Simhadri. Freshdiskann: A fast and accurate graph-based ann index for streaming similarity search, 2021.

Yao Tian, Tingyun Yan, Xi Zhao, Kai Huang, and Xiaofang Zhou. A learned index for exact similarity search in metric spaces. *IEEE Transactions on Knowledge and Data Engineering*, 35(8): 7624–7638, 2023.

Joel A Tropp. Analysis of randomized block krylov methods. *ACM Report*, 2, 2018.

Thomas Vecchiato, Claudio Lucchese, Franco Maria Nardini, and Sebastian Bruch. A learning-to-rank formulation of clustering-based approximate nearest neighbor search. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2024.

David P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1–2):1–157, Oct 2014. ISSN 1551-305X.

Xiang Wu, Ruiqi Guo, Ananda Theertha Suresh, Sanjiv Kumar, Daniel N Holtmann-Rice, David Simcha, and Felix Yu. Multiscale quantization for fast similarity search. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Artem Babenko Yandex and Victor Lempitsky. Efficient indexing of billion-scale datasets of deep descriptors. In *2016 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2055–2063, 2016.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We are explicit in our abstract and introduction about the prior research that motivated this work and describe precisely what our novel contributions are.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We noted in our discussion the limitations of our work. First, we emphasized the sub-optimality of the solution obtained from Lemma 1, but justified its efficacy through empirical experiments. Second, we pointed out that our router, in isolation, is computationally more expensive and requires a larger amount of storage, but explained and demonstrated why this increase in costs can be appropriate depending on the end-to-end setup. Finally, our method performs better when data is stored in a high-latency storage medium, rather than in-memory; we have made a note of that in our discusison.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We state all assumptions and complete proofs for all claims made in this work. For some results, we move the proof to the appendix due to space constraints.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have included all the details of our experiments (including all hyperparameters) as well as a complete description of datasets to facilitate reproducibility. Furthermore, when the work is ready for publication, we plan to release our code and configuration files to open-source.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We have created a double-blind review-compliant Git repository to host all the required code to reproduce every experiment reported in this work (including supplementary figures). In the repository, we have also included links to processed datasets. These links point to data that is stored in a Google Storage Bucket, and made available in such a way that neither reveals the identity of the authors, nor does it require viewers to log into Google services. The repository can be found at `https://github.com/Artificial-Memory-Lab/optimist-router`.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We give all details including hyperparameters. As for train-test splits, the benchmark datasets used in our work come with a test query set that is separate from the data points.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: While we do not explicitly mention the results of statistical significance tests to avoid clutter, we note here that wherever OPTIMIST performs better than the baselines, the differences are statistically significant with a p-value that is often less than 0.001.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Most of our experiments are not compute-heavy and are run on a commercial laptop with standard configuration. For experiments that study the latency of different methods, we state the specifications of the hardware used.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We acknowledge that we have reviewed the NeurIPS code of ethics and that our research conforms, in every respect, with the code.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: Our work does not have any societal impact that we could identify.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our work does not involve the development of models or the release of new data.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We have included this information in our full description of datasets used in our experiments.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: Our research does not result in new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: Our research does not involve crowdsourcing or human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: Our research does not involve human subjects and as such does not require IRB review.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: We do not use LLMs in any shape or form (including for writing, editing, or formatting) in our work.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A An illustrative example

Interestingly, depending on operational factors such as data transfer rates and memory utilization, it would be acceptable for routing to be computationally more expensive as long as it identifies shards more accurately. Let us consider an example to support this statement and motivate our research.

Take Natural Questions [Kwiatkowski et al., 2019] embedded with ADA-002[8] denoted NQ-ADA2, containing 2.7 million 1,536-dimensional points. Applying KMeans and compressing with the typical 4-bit PQ codebooks give shards with about 1,650 points each and 1MB in size. Consider next a commercial machine: AWS c5.xlarge (4 vCPUs, 8GB of memory). Using 4 threads, moving 4MB of data from storage (hosted on Amazon's S3) to memory has a mean latency of 45 milliseconds (ms).[9]

Given this setup, moving 16 NQ-ADA2 shards from blob storage to the machine's memory takes 180ms (P50 latency). On this same machine, routing using Equation (2) takes 0.76ms. As such, so long as a more complex router can achieve the same accuracy as Equation (2) by fetching 16 fewer shards, but by introducing a latency of no greater than 179ms, the overall efficiency of ANN search improves. As we show later, our router on the same dataset with the same configuration takes only about 10.1ms to identify shards, leading to a saving of 170.6ms per query in the given example.

As this example illustrates, storage-backed, clustering-based ANN search offers a trade-off between not just accuracy and speed, but also other efficiency factors such as I/O bandwidth, and memory and cache utilization. This opens the door to more nuanced research. Our work is a step in that direction.

B Related work

As we argued in Section 1, routing accuracy in clustering-based approximate nearest neighbor search is increasingly relevant. Surprisingly, it has received little attention in the literature and most practical implementations of routing do not go beyond the naïve form of Equation (2). In this section, we briefly review the relevant methods that explore this particular topic.

The literature on routing functions can be split into two categories: supervised and unsupervised methods. In the supervised regime, a routing function is learned using a training query distribution. This is best demonstrated by the work of Vecchiato et al. [2024], which formulates the problem of routing as a learning-to-rank task: Given a query the function learns to rank shards, and subsequently uses the learned routing function during search for a test distribution.

That work is similar in spirit to the supervised methods put forth by Chiu et al. [2020], Dong et al. [2020], and Tian et al. [2023]. The difference is that, this set of methods attempt to learn to *partition* the data and form representatives based on the learned functions.

Supervised methods have one caveat: While the methods differ in their approach to the problem, they require a training distribution. Our method, on the other hand, is completely unsupervised and does not have this limitation. Because of this fundamental difference, we do not believe an empirical or theoretical comparison between supervised methods and our method is warranted. In fact, in many instances, the benchmark datasets do not come with a training distribution or have very limited number of training queries.

There are, however, other unsupervised methods in the literature that we have included in our experiments as baselines per Section 3.1 and that we review next in more detail. The first two are the MEAN and NORMALIZEDMEAN routers, which are the *de facto* routing functions in all open-source approximate nearest neighbor search software. These follow the simple form of Equations (2) and (3).

While SCANN is proposed as a quantization method for maximum inner product search [Guo et al., 2020], it implicitly introduces a novel (supervised and unsupervised) routing function—we focus on the unsupervised function. SCANN is based on the idea that, points in a shard should not count equally towards the quantization error. That is unlike Product Quantization Jégou et al. [2011]. Instead, every point is weighted based on how likely it is to maximize inner product with an arbitrary query.

[8] `https://openai.com/index/new-and-improved-embedding-model/`

[9] See `https://github.com/dvassallo/s3-benchmark` for an independent and comprehensive benchmark.

What is relevant to this work is that the SCANN quantization method produces a shard representative that is a single vector, but that is not the centroid or normalized centroid (c.f., Theorem 4.2 in [Guo et al., 2020]). We use this representative to route queries to shards.

Another relevant work is the routing functions proposed by Gottesbüren et al. [2024]. They introduce two routing protocols: KRT and HRT. The latter has strong theoretical guarantees, but both methods perform equally well, with KRT having a slight advantage. As such, our review focuses on KRT.

KRT, which we call SUBPARTITION in Section 3 for clarity, is based on the idea of partitioning each shard into multiple sub-shards and extracting a representative from each sub-shard. This is particularly useful for shards that are far larger than a conventional configuration of clustering-based ANN search would produce. At query time, shards are ranked by a statistic based on the inner products between the query and the shard representatives. In our instantiation of this routing function, we set that statistic to be the mean of the sub-shards.

C Proof of Lemma 2 and justification of its assumptions

Let $\Sigma = D + R$ be the decomposition of the PSD covariance matrix Σ into its diagonal D and residual $R = \Sigma - D$. Let $Q\Lambda Q^\top$ be the orthogonal eigendecomposition of $R_\circ = D^{-1/2}RD^{-1/2}$. Recall that we define Σ_t^{MS}, for some $1 \le t \le d$, as

$$\Sigma_t^{\text{MS}} := D + D^{1/2}[Q]_t[\Lambda]_t[Q]_t^\top D^{1/2}$$

We start by proving Lemma 2, then justify the assumptions of the lemma.

C.1 Proof of Lemma 2

We state a few technical results that will simplify the proof of the lemma.

Fact 1. *For any symmetric matrix $M \in \mathbb{R}^{d\times d}$ with eigendecomposition $USU^\top$, we have that for any $v \in \mathbb{R}^d$,*

$$v^\top M v = \sum_{i=1}^{d} S_i \cdot \langle v, U_i\rangle^2.$$

Fact 2. *Assuming the eigenvalues Λ of $R_\circ$ are sorted in non-increasing order, we have that $I + R_\circ = Q(I+\Lambda)Q^\top$. In words, the eigenvectors of $I + R_\circ$ are the same as $R_\circ$, and the i-th eigenvalue is $\Lambda_i + 1$. As a corollary, since $I + R_\circ$ is PSD, we have that $\Lambda_i \ge -1$ for all $i \in [d]$.*

Proof. This follows easily after noticing that $I = QQ^\top$ because Q is a $d \times d$ matrix with orthonormal columns (and rows). □

Fact 3. *For a diagonal matrix $S \in \mathbb{R}^{d\times d}$ with bounded positive entries, i.e. $0 < l \le S_{ii} \le u$ for all $i \in [d]$, and arbitrary vector $v \in \mathbb{R}^d$, we have that $l\|v\|_2 \le \|Sv\|_2 \le u\|v\|_2$.*

Proof of Lemma 2. First, note that, by the definition of $\text{err}(\cdot,\cdot)$ from Equation (7), we can write:

$$\text{err}(D^{-\frac{1}{2}}\Sigma D^{-\frac{1}{2}}, D^{-\frac{1}{2}}\Sigma_t^{\text{MS}} D^{-\frac{1}{2}}) = \sup_{v\in\mathbb{R}^d} \frac{|v^\top D^{-1/2}(\Sigma - \Sigma\Sigma_t^{\text{MS}})D^{-1/2}v|}{\|v\|_2^2}.$$

Since D has strictly positive entries on the diagonal, we can do a change of variables, setting $u = D^{1/2}v$. Denoting $\max_{i\in[d]} D_{ii}$ by $\|D\|_\infty$, this gives us:

$$\begin{aligned}\text{err}(D^{-\frac{1}{2}}\Sigma D^{-\frac{1}{2}}, D^{-\frac{1}{2}}\Sigma_t^{\text{MS}} D^{-\frac{1}{2}}) &= \sup_{v\in\mathbb{R}^d} \frac{|v^\top(\Sigma - \Sigma_t^{\text{MS}})v|}{\|D^{1/2}v\|_2^2} \\ &\ge \frac{1}{\|D\|_\infty} \cdot \sup_{v\in\mathbb{R}^d} \frac{|v^\top(\Sigma - \Sigma_t^{\text{MS}})v|}{\|v\|_2^2} = \frac{\text{err}(\Sigma, \Sigma_t^{\text{MS}})}{\|D\|_\infty}\end{aligned}$$

where the inequality follows by Fact 3.

Next, consider the following:

$$\begin{aligned}
\text{err}(D^{-\frac{1}{2}}\Sigma D^{-\frac{1}{2}}, D^{-\frac{1}{2}}\Sigma_t^{\text{MS}} D^{-\frac{1}{2}}) &= \sup_{\|v\|_2=1} \left|v^\top D^{-\frac{1}{2}}(\Sigma - \Sigma_t^{\text{MS}})D^{-\frac{1}{2}}v\right| \\
&= \sup_{\|v\|_2=1} \left|v^\top (R_\circ - [Q]_t[\Lambda]_t[Q]_t^\top)v\right| && (10) \\
&= \sup_{\|v\|_2=1} \Big| \sum_{i=t+1}^{d} \Lambda_i \cdot \langle Q_i, v\rangle^2 \Big| && (11) \\
&= \max_{l\in[t+1,d]} |\Lambda_l|. && (12)
\end{aligned}$$

where the second equality follows by the definition of Σ_t^{MS} and the third by Fact 1.

Putting our arguments thus far together, we have established that:

$$\frac{\text{err}(\Sigma, \Sigma_t^{\text{MS}})}{\|D\|_\infty} \leq \max_{l\in[t+1,d]} |\Lambda_l|. \tag{13}$$

Using this result, and noting that, $1 + \Lambda_{t+1} \geq 1$ by assumption, so that $\Lambda_{t+1} \geq 0$; and $\Lambda_l \geq -1$ for all $l \in [d]$ by Fact 2, we can derive the following:

$$\begin{aligned}
\frac{\text{err}(\Sigma, \Sigma_t^{\text{MS}})}{\|D\|_\infty} &\leq \max_{l\in[t+1,d]} |\Lambda_l| \leq 1 + \Lambda_{t+1} \leq \max_{l\in[t+1,d]} |1 + \Lambda_l| && (14) \\
&= \text{err}\Big(\underbrace{I + R_\circ}_{D^{-1/2}\Sigma D^{-1/2}}, [Q]_t[I + \Lambda]_t[Q]_t^\top \Big). && (15)
\end{aligned}$$

All that is left to do is to bound the right-hand side of Equation (15). We do so as follows:

$$\begin{aligned}
&\text{err}(D^{-1/2}\Sigma D^{-1/2}, [Q]_t[I + \Lambda]_t[Q]_t^\top) && (16) \\
&\quad\leq \text{err}(D^{-1/2}\Sigma D^{-1/2}, D^{-1/2}\Sigma_t^{\text{LR}} D^{-1/2}) && (17) \\
&\quad= \sup_{\|v\|_2=1} \left|v^\top D^{-1/2}(\Sigma - \Sigma_t^{\text{LR}})D^{-1/2}v\right| && (18) \\
&\quad= \sup_{\|v\|_2=1} \Big| \sum_{i,j} v_i v_j (\Sigma - \Sigma_t^{\text{LR}})_{ij} / \sqrt{D_{ii}D_{jj}} \Big| && (19) \\
&\quad\leq \sup_{\|v\|_2=1} \Big| \sum_{i,j} v_i v_j (\Sigma - \Sigma_t^{\text{LR}})_{ij} \frac{1}{(1-\epsilon)\|D\|_\infty} \Big| && (20) \\
&\quad\leq \frac{1}{(1-\epsilon)\|D\|_\infty} \sup_{\|v\|_2=1} \left|v^\top (\Sigma - \Sigma_t^{\text{LR}})v\right| && (21) \\
&\quad= \frac{1}{(1-\epsilon)\|D\|_\infty} \text{err}(\Sigma, \Sigma_t^{\text{LR}}), && (22)
\end{aligned}$$

where the first inequality is because the left-hand side is the optimal error by the Eckhart-Young-Mirsky Theorem, and the second inequality is due to the fact that $D_{ii} \geq (1-\epsilon)\|D\|_\infty$.

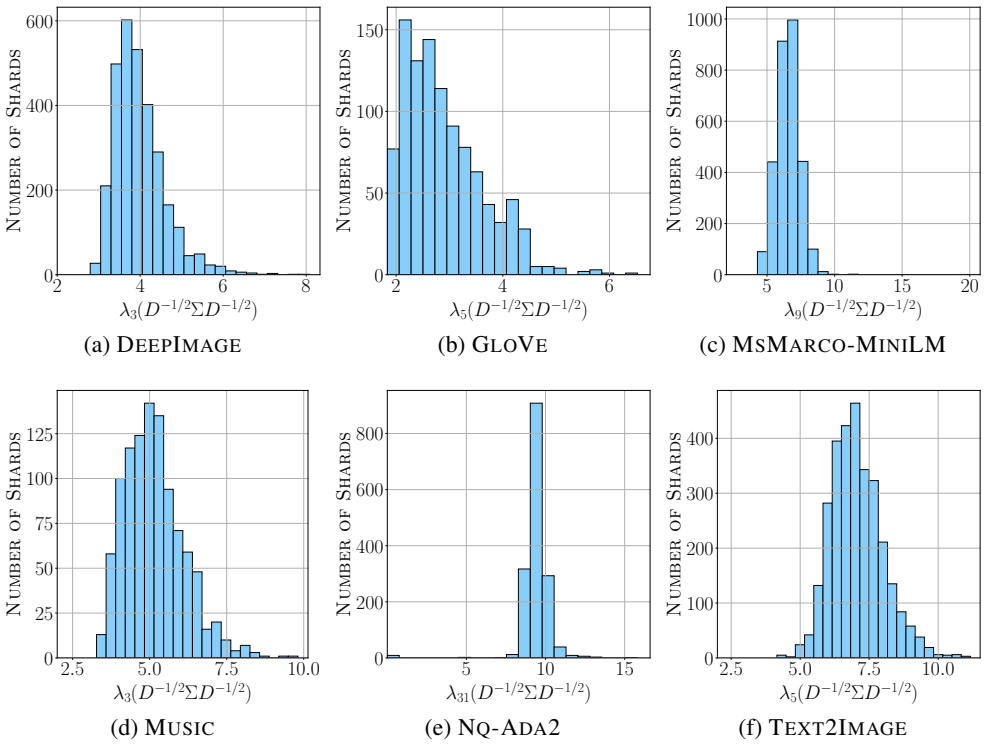
Putting everything together, we have shown that:

$$\frac{\text{err}(\Sigma, \Sigma_t^{\text{MS}})}{\|D\|_\infty} \leq \frac{1}{(1-\epsilon)\|D\|_\infty} \text{err}(\Sigma, \Sigma_t^{\text{LR}}),$$

as desired. □

C.2 Assumptions of Lemma 2

Recall that we make two assumptions about the covariance matrix Σ and the eigendecomposition of its symmetrization, $D^{-1/2}\Sigma D^{-1/2}$:



(a) DEEPIMAGE (b) GLOVE (c) MSMARCO-MINILM

(d) MUSIC (e) NQ-ADA2 (f) TEXT2IMAGE

Figure 5: Histogram of the $(t+1)$-th eigenvalue. For each dataset, we pick the partitioning and t from Table 1. Plots show that almost all shards for all datasets have $(t+1)$-th eigenvalue bounded away from 1, except for a few shards for NQ-ADA2.

1. The $(t+1)$-th eigenvalue of $D^{-1/2}\Sigma D^{-1/2}$ is greater than or equal to 1. In particular, letting $D^{-1/2}\Sigma D^{-1/2} = Q(I+\Lambda)Q^\top$ be the orthogonal eigendecomposition of the symmetrization, we assume $1 + \Lambda_{t+1} \geq 1$.
2. The diagonal of Σ has the property that $\min_{i\in[d]} \Sigma_{ii} \geq (1-\epsilon)\max_{i\in[d]} \Sigma_{ii}$ for some $\epsilon \in (0, 1]$.

First assumption. Notice that $D^{-1/2}\Sigma D^{-1/2} = I + D^{-1/2}RD^{-1/2}$ is symmetric PSD, so we have that $\text{tr}(D^{-1/2}\Sigma D^{-1/2}) = \sum_{i=1}^{d} 1 + \Lambda_i = d$. Hence by definition there must exist some t for which $1 + \Lambda_{t+1} \geq 1$. While in the worse case we cannot hope for the existence of an eigenvalue larger than this, in practice, including for the datasets we consider in this work, it can be shown that in fact the eigenvalues of $D^{-1/2}\Sigma D^{-1/2}$ are larger than 1 across shards and datasets–see Figure 5.

Second assumption. While in the worst case, the diagonal of Σ can have arbitrarily large entries compared to its smallest entries, in practice this is rarely the case. While we do not explore how to remove this assumption, there are several mechanisms to do so in practice such as applying random rotations or pseudo-random rotations Ailon and Chazelle [2009], Woodruff [2014], Ailon and Liberty [2013] to the data points in each shard before using Algorithm 1. It is well known (e.g. Lemma 1 in Ailon and Chazelle [2009]) that after applying such transforms, the coordinates of the vectors are "roughly equal," thereby ensuring that the diagonal of the covariance has entries of comparable magnitude. We leave the exploration of removing this assumption to future work.

D Datasets

The following is a complete description of the datasets used in this work:

- **TEXT2IMAGE**: A cross-modal dataset, where data and query points may have different distributions in a shared space [Simhadri et al., 2022]. We use a subset consisting of 10 million 200-dimensional data points along with a subset of 10,000 test queries. The dataset is available under the terms of the Creative Commons Attribution 4.0 International license.
- **MUSIC**: 1 million 100-dimensional points [Morozov and Babenko, 2018] with 1,000 queries. To the best of our knowledge, the dataset comes with no information on the license under which it is made available (per https://github.com/stanis-morozov/ip-nsw?tab=readme-ov-file).
- **DEEPIMAGE**: Subset of 10 million 96-dimensional points from the billion deep image features dataset [Yandex and Lempitsky, 2016] with 10,000 queries. The dataset is available under the terms of Apache license 2.0.
- **GLOVE**: 1.2 million, 200-dimensional word embeddings trained on tweets [Pennington et al., 2014] with 10,000 test queries. The dataset is available under the terms of the Public Domain Dedication and license v1.0.
- **MSMARCO-MINILM**: MS MARCO Passage Retrieval v1 [Nguyen et al., 2016] is a question-answering dataset consisting of 8.8 million short passages in English. We use the "dev" set of queries for retrieval, made up of 6,980 questions. We embed individual passages and queries using the ALL-MINILM-L6-V2 model[10] to form a 384-dimensional vector collection. The dataset is available under the terms of the Creative Commons Attribution 4.0 International license. The model used to embed the dataset is available under the terms of Apache license 2.0.
- **NQ-ADA2**: 2.7 million, 1,536-dimensional embeddings of the Natural Questions dataset [Kwiatkowski et al., 2019] with the ADA-002 model.[11] The dataset is available under the terms of Apache license 2.0.

We note that the last four datasets are intended for cosine similarity search. As such we normalize these collections prior to indexing, reducing the task to MIPS of Equation (1).

E Effect of hyper-parameters on OPTIMIST

Recall that OPTIMIST takes two parameters: t, the rank of the covariance sketch, and δ, the degree of optimism. We examine the effect of these parameters on the performance of OPTIMIST.

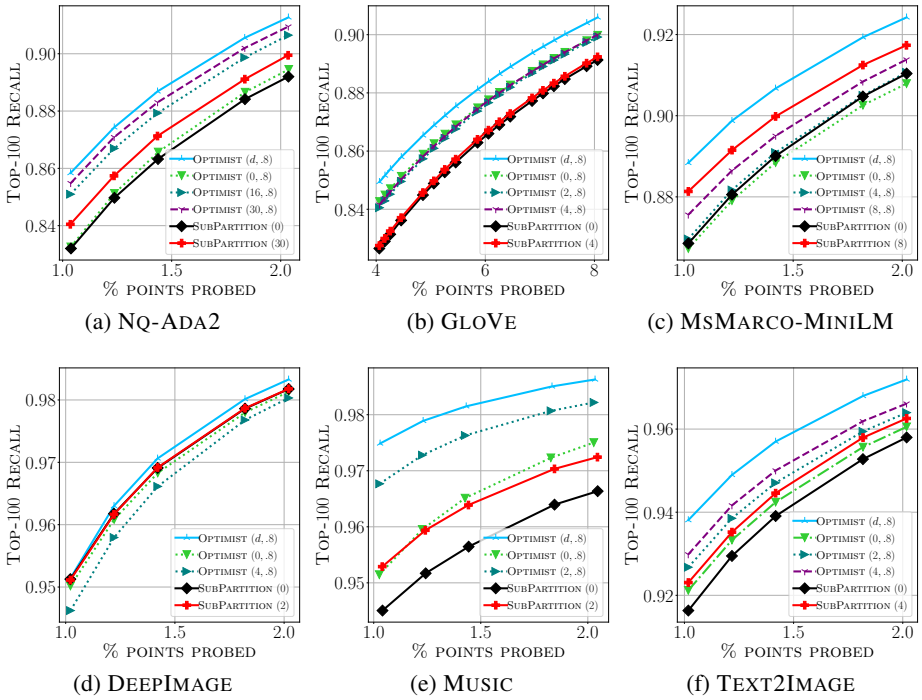
Figure 6 visualizes the role played by t. It comes as no surprise that larger values of t lead to a better approximation of the covariance matrix. What we found interesting, however, is the remarkable effectiveness of a sketch that simply retains the diagonal of the covariance, denoted by OPTIMIST$(0, \cdot)$, in the settings of k we experimented with (i.e., $k \in \{1, 10, 100\}$).

In the same figure, we have also included two configurations of SUBPARTITION: one with 2 sub-partitions, SUBPARTITION(0), and another with $t + 2$ sub-partitions, SUBPARTITION(t), for the largest t. These help put the performance of OPTIMIST with various ranks in perspective. In particular, we give the SUBPARTITION baseline the same amount of information and contrast its recall with OPTIMIST.

We turn to Figure 7 to understand the impact of δ. It is clear that encouraging OPTIMIST to be too optimistic can lead to sub-optimal performance. That is because of our reliance on the Chebyshev's inequality, which can prove too loose, leading to an overestimation of the maximum value. Interestingly, $\delta \in (0.6, 0.8)$ appears to yield better recall across datasets.

[10] Checkpoint at `https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2`.

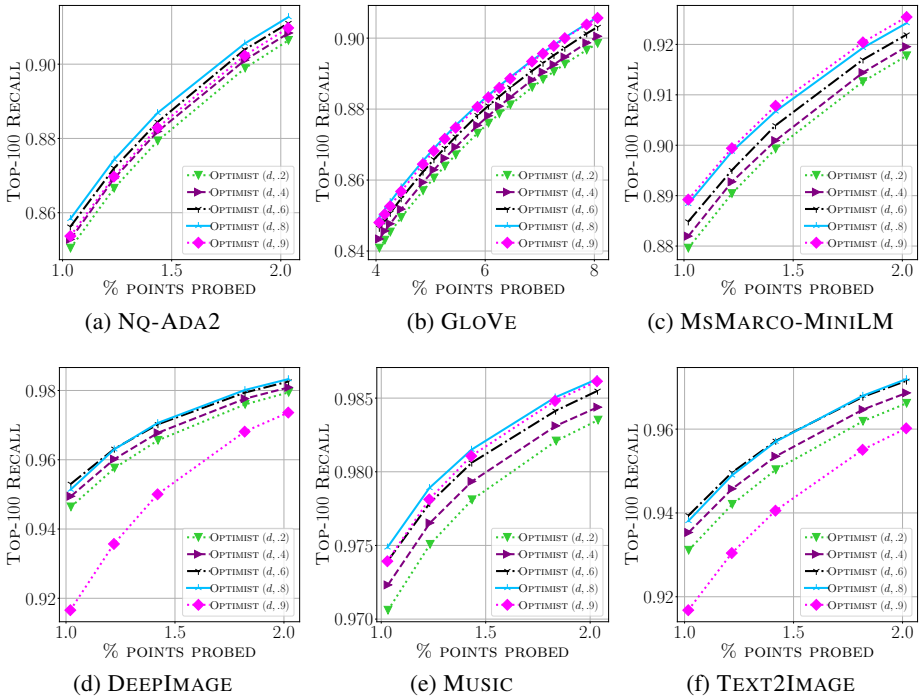
[11] `https://openai.com/index/new-and-improved-embedding-model/`



(a) NQ-ADA2 (b) GLOVE (c) MSMARCO-MINILM

(d) DEEPIMAGE (e) MUSIC (f) TEXT2IMAGE

Figure 6: Top-100 recall vs. volume of probed data as we change the rank parameter (t). OPTIMIST$(0, \cdot)$ is a sketch that is the diagonals only. Partitioning is with Spherical KMeans.

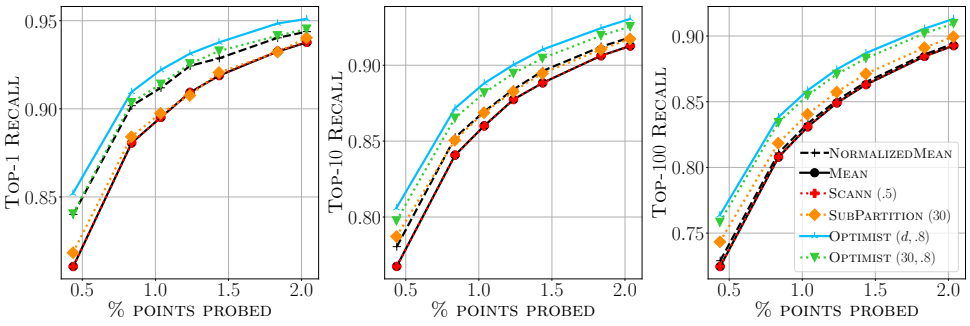


(a) NQ-ADA2 (b) GLOVE (c) MSMARCO-MINILM

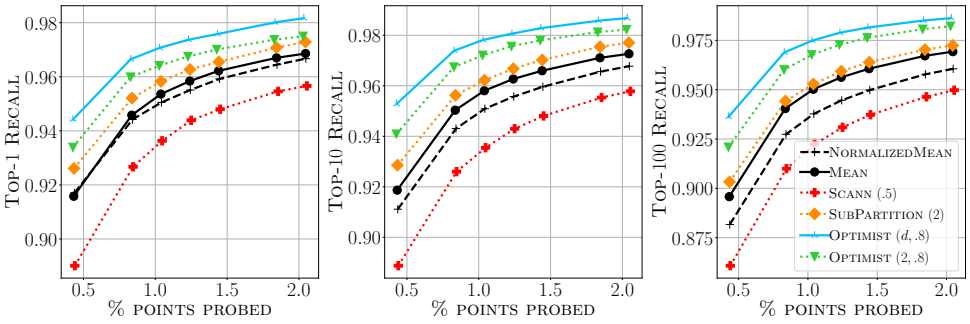
(d) DEEPIMAGE (e) MUSIC (f) TEXT2IMAGE

Figure 7: Top-100 recall vs. volume of probed data, comparing a range of values of δ. As $\delta \to 1$, OPTIMIST becomes more optimistic. Partitioning is with Spherical KMeans.

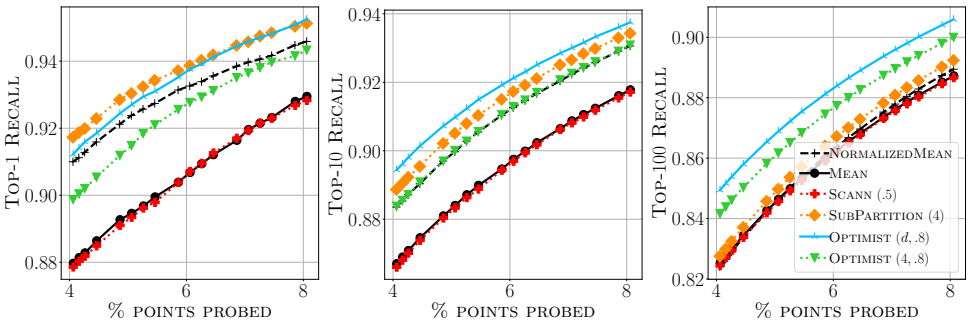
F Experiments with spherical KMeans



(a) NQ-ADA2

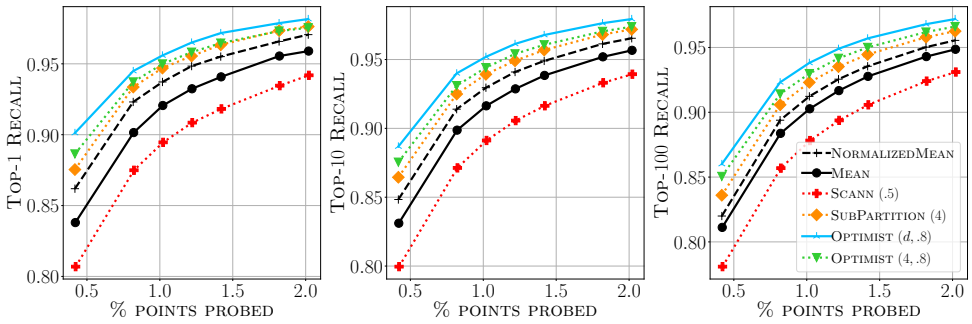


(b) MUSIC

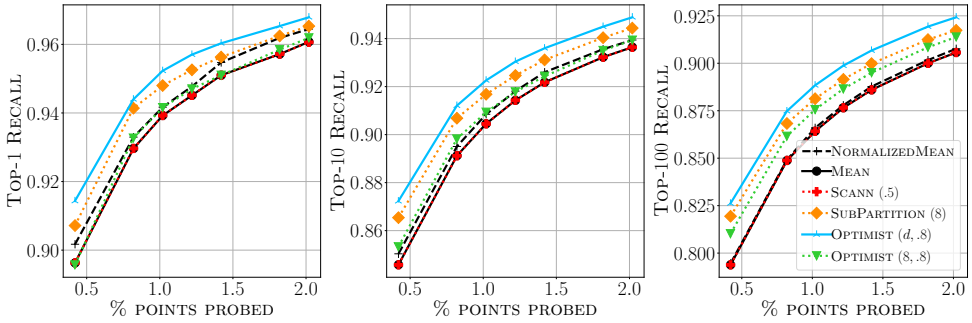


(c) GLOVE

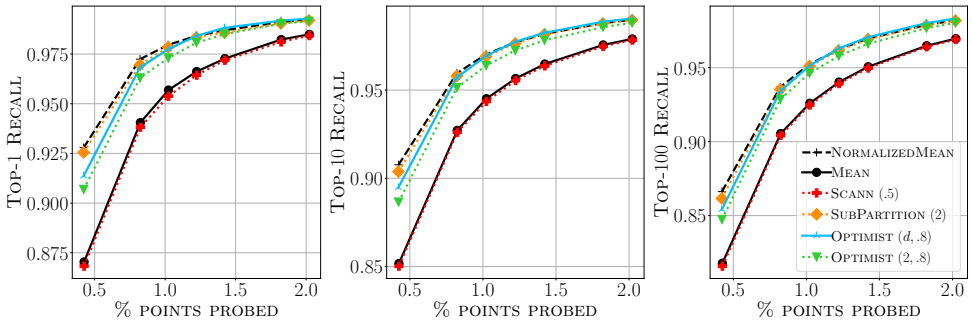
Figure 8: Top-k recall vs. volume of probed data. Partitioning is with Spherical KMeans. SCANN has parameter T, SUBPARTITION t (leading to $t+2$ sub-partitions per shard), and OPTIMIST rank t and degree of optimism δ.



(d) Text2Image



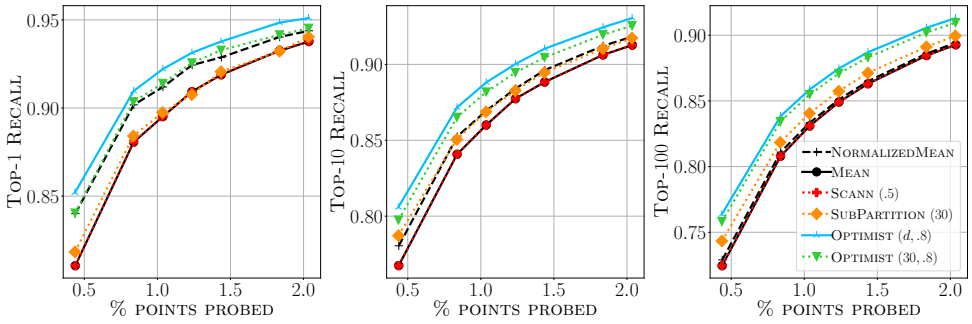
(e) MsMarco-MiniLM



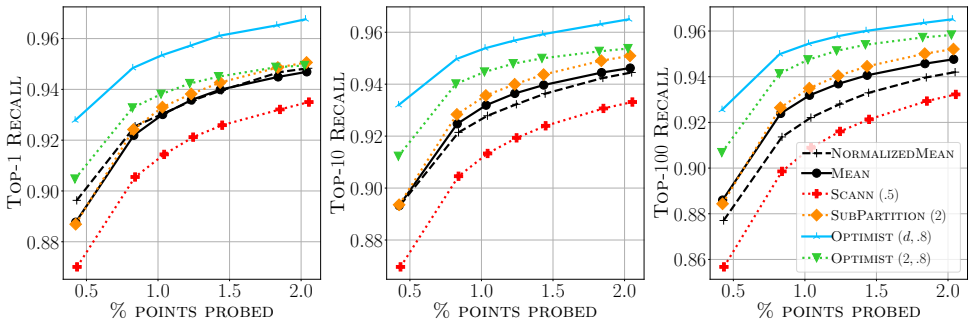
(f) DeepImage

Figure 8: Top-k recall vs. volume of probed data. Partitioning is with Spherical KMeans. ScaNN has parameter T, SubPartition t (leading to $t+2$ sub-partitions per shard), and Optimist rank t and degree of optimism δ.

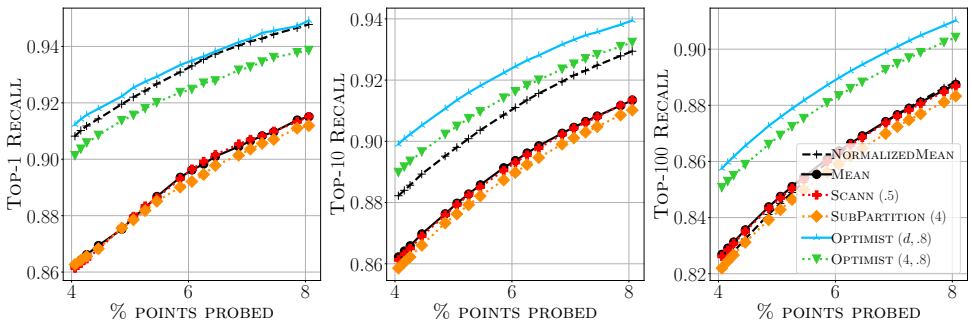
G Experiments with standard KMeans



(a) NQ-ADA2

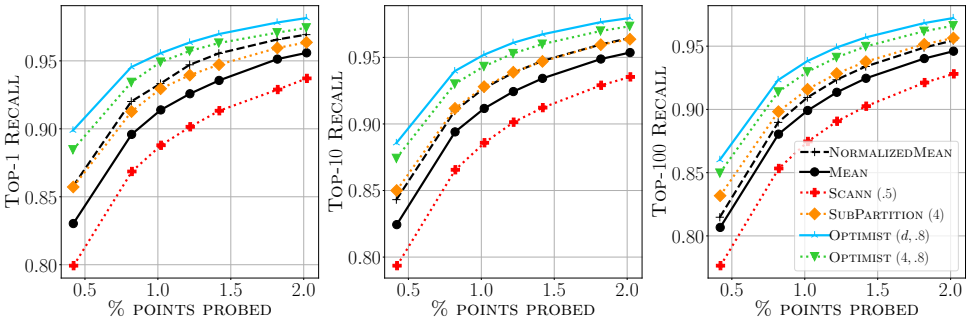


(b) MUSIC

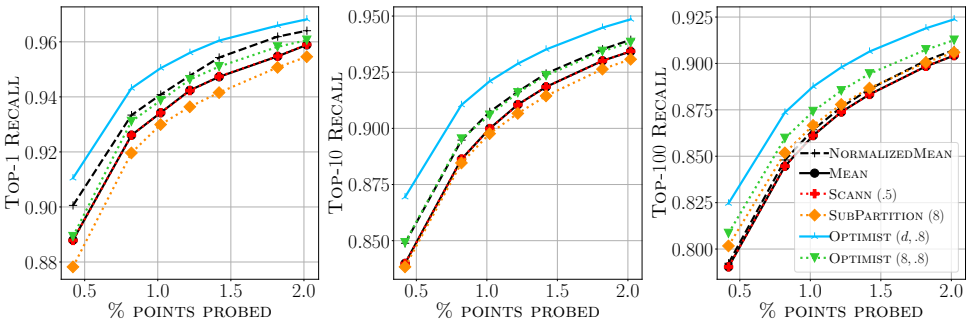


(c) GLOVE

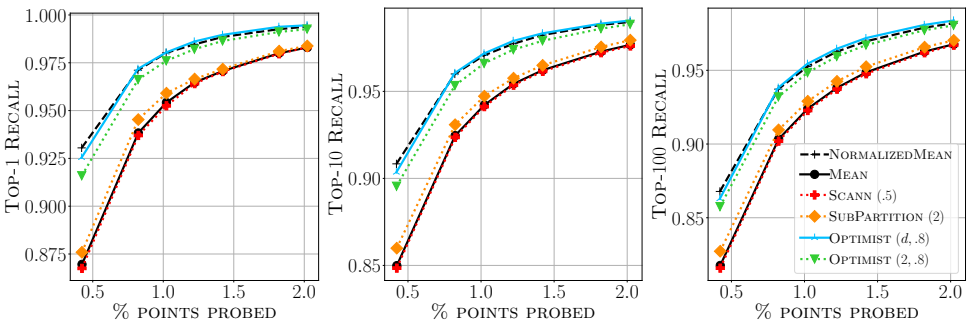
Figure 9: Top-k recall vs. volume of probed data. Partitioning is with Standard KMeans. SCANN has parameter T, SUBPARTITION t (leading to $t+2$ sub-partitions per shard), and OPTIMIST rank t and degree of optimism δ.



(d) Text2Image



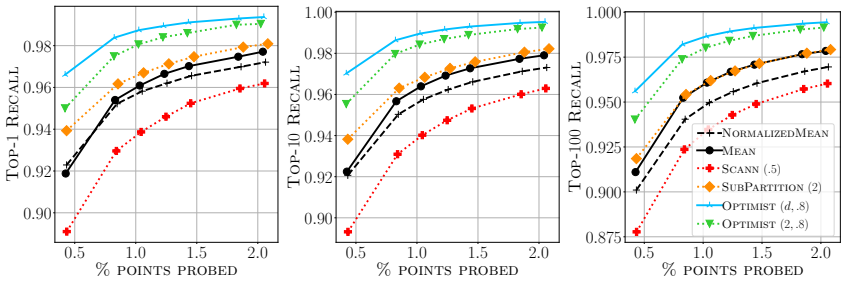
(e) MsMarco-MiniLM



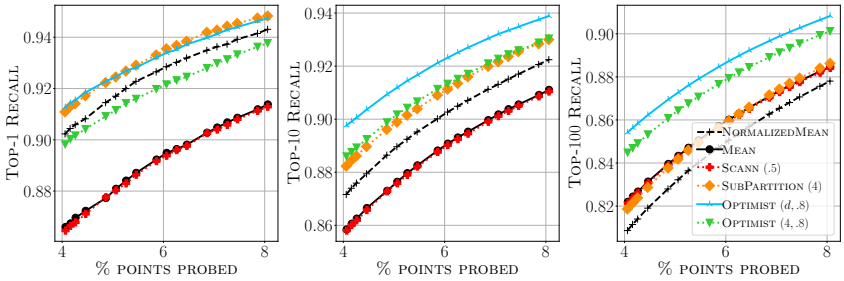
(f) DeepImage

Figure 9: Top-k recall vs. volume of probed data. Partitioning is with Standard KMeans. SCANN has parameter T, SUBPARTITION t (leading to $t+2$ sub-partitions per shard), and OPTIMIST rank t and degree of optimism δ.

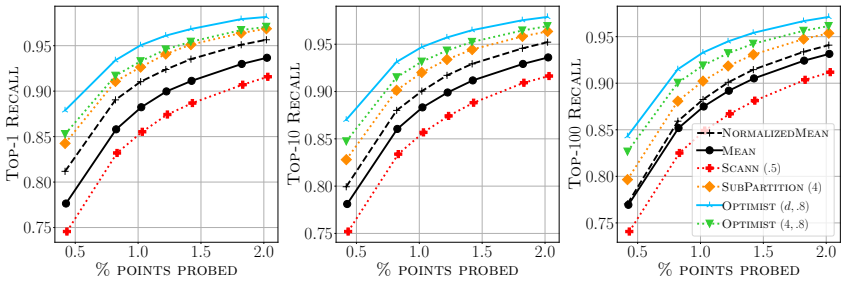
H Experiments with GMM



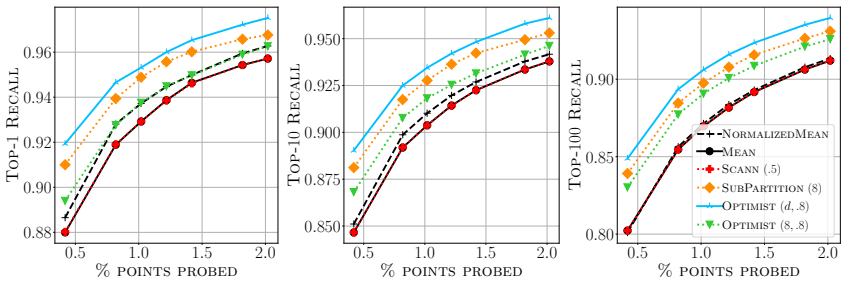
(a) MUSIC



(b) GLOVE

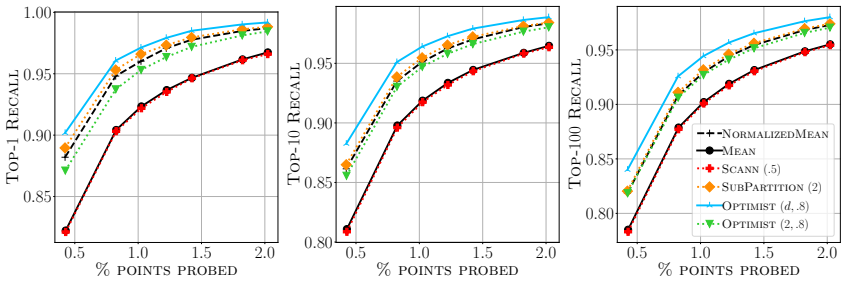


(c) TEXT2IMAGE



(d) MSMARCO-MINILM

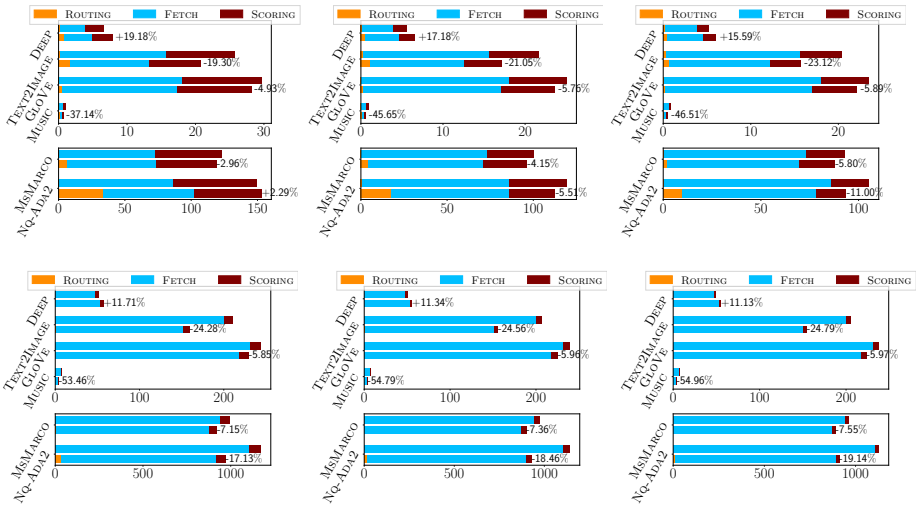
Figure 10: Top-k recall vs. volume of probed data. Partitioning is with Gaussian Mixture Model. SCANN has parameter T, SUBPARTITION t (leading to $t+2$ sub-partitions per shard), and OPTIMIST rank t and degree of optimism δ. We note that, due to the dimensionality of NQ-ADA2, we were unable to complete GMM clustering on this particular dataset.



(e) DEEPIMAGE

Figure 10: Top-k recall vs. volume of probed data. Partitioning is with Gaussian Mixture Model. SCANN has parameter T, SUBPARTITION t (leading to $t+2$ sub-partitions per shard), and OPTIMIST rank t and degree of optimism δ. We note that, due to the dimensionality of NQ-ADA2, we were unable to complete GMM clustering on this particular dataset.

I Latency comparison



(a) 1 thread (b) 2 threads (c) 4 threads

Figure 11: Mean latency (ms) to reach 95% recall, using different number of threads (columns), when PQ-compressed shards are on SSD (top row) and blob storage (bottom row). For each dataset, we plot the latency breakdown for NORMALIZEDMEAN (top bar) and OPTIMIST (bottom bar), and report relative gains (negative value indicates gain by OPTIMIST).

J Maximum inner product prediction

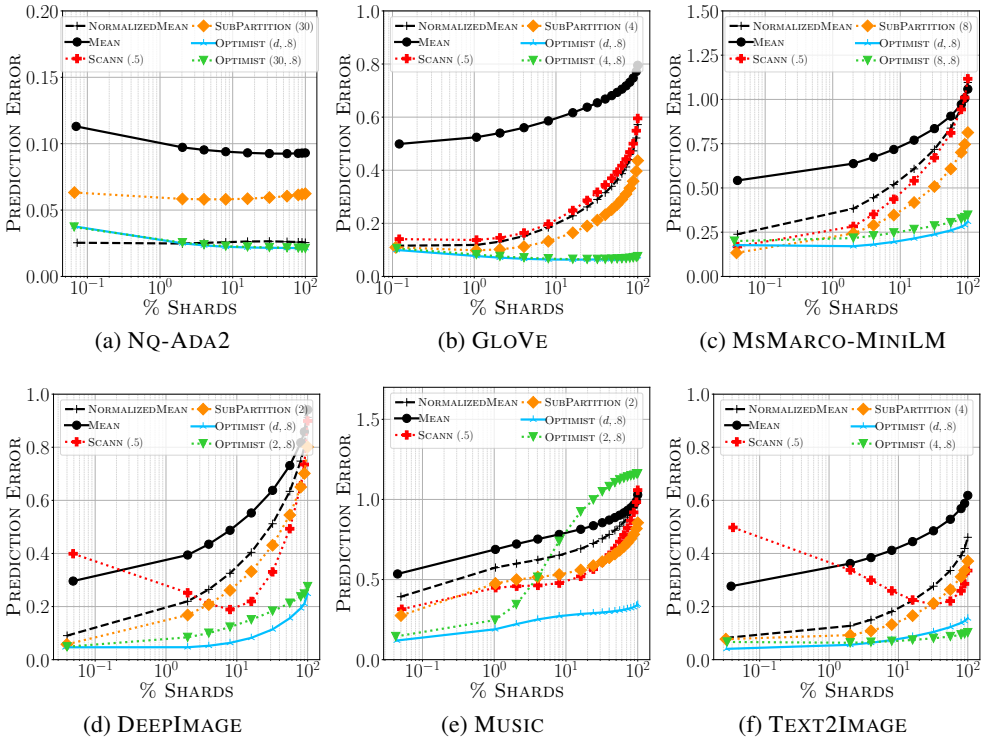


Figure 12: Mean prediction error $\mathcal{E}_\ell(\tau, \cdot)$, defined in Equation (9), versus ℓ (expressed as percent of total number of shards), for various routers and datasets.