Prune, Don't Rebuild: Efficiently Tuning α-Reachable Graphs for Nearest Neighbor Search

Zachary Ives*
University of Pennsylvania
zives@engineering.upenn.edu

Jiaming Liang*
University of Pennsylvania
liangjm@engineering.upenn.edu

Ashwin Padaki*
University of Pennsylvania
apadaki@engineering.upenn.edu

Erik Waingarten*
University of Pennsylvania
ewaingar@engineering.upenn.edu

Tian Zhang*
University of Pennsylvania
tianzh@engineering.upenn.edu

Abstract

Over the past decade, graph-based approximate nearest neighbor (ANN) algorithms, such as DiskANN [10] and HNSW [12], have demonstrated state-of-the-art empirical performance. Recent theoretical works [8, 7] introduce the framework of α-*reachability* to obtain worst-case performance guarantees for DiskANN; here, the reachability parameter α gives a trade-off between construction time, query time, and accuracy. In this work, we propose RP-TUNING, an efficient and simple post-hoc algorithm, based on DiskANN's pruning step, and show that

- Theoretically, efficiently adjusting the reachability of an α-reachable graph is possible via pruning: RP-TUNING preserves worst-case reachability guarantees in general metrics and improved guarantees in Euclidean metrics.
- Empirically, RP-TUNING accelerates DiskANN tuning on four datasets by up to 73$\times$ with varied performance trade-offs compared to fully rebuilt graphs.

1 Introduction

Approximate Nearest Neighbor (ANN) search is an essential primitive in modern AI and ML applications, such as retrieval augmented generation [5], linking and matching for knowledge graph construction [16], and curating datasets for model training and fine-tuning. Many large-scale implementations of ANN search adopt graph-based algorithms. DiskANN [10] is widely used, e.g., in Milvus [15] and Microsoft's CosmosDB [14], because of its state-of-the-art empirical performance and scalability. A challenge with algorithms like DiskANN is that the tradeoffs between accuracy, latency, and space are not well understood in practice — requiring users to empirically evaluate the algorithms under different hyperparameter settings, until the right tradeoffs are found.

To address the challenge, we propose RP-TUNING, an efficient and simple post-hoc pruning algorithm derived from the ROBUSTPRUNE subroutine used in the DiskANN index construction. RP-TUNING enables dynamic adjustment of DiskANN's reachability parameter, giving a tunable trade-off between accuracy, latency, and index size. By starting from a high-quality base graph, it avoids the cost of repeated index reconstruction. Leveraging recent theoretical frameworks for α-reachable graphs [8,

*Authors listed in alphabetical order.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

7], we provide a rigorous analysis of RP-TUNING. Namely, we prove that a graph pruned via RP-TUNING preserves specific worst-case reachability guarantees in general metrics while achieving superior worst-case bounds in Euclidean metrics.

Our empirical evaluation across four public datasets demonstrates that RP-TUNING accelerates the DiskANN tuning process by up to 73×. Moreover, DiskANN graphs tuned with RP-TUNING exhibit comparable performance to those rebuilt with the same reachability parameters. In addition, we show that RP-TUNING significantly outperforms a random pruning baseline, and that it can be applied to other graph-based algorithms such as HNSW [12] (See Appendix B).

2 Preliminaries

We consider a dataset P in a metric space (X, D) where X is the universe of points and D is the distance function. For any point $x \in X$ and radius $r > 0$, let $B(x, r) = \{y \in X \mid D(x, y) \leq r\}$ denote the closed ball centered at x with radius r. The doubling constant of X, denoted by C, is the minimum integer such that any ball $B(x, r)$ in X can be covered by at most C balls of radius $r/2$. The *doubling dimension* of the metric space X is defined as $\lambda = \log_2 C$. The *aspect ratio* Δ of P is defined as the ratio between the maximum and minimum distances in P.

A *graph-based ANN index* for P is given by a directed graph $G = (P, E)$; we use (a, b) to denote the one-way edge from a to b. Note that we will abuse P to refer to the dataset and the vertex set of the graph, since in this paper, points in the dataset have a one-to-one correspondence to vertices in the graph. We use $N_{out}(a), N_{in}(a)$ to denote the set of out-neighbors and in-neighbors, respectively, of point $a \in P$ when the graph in the context is clear.

We give a brief review of DiskANN algorithm and its theoretical guarantees below. We refer readers to [10, 8, 7] for full details.

2.1 DiskANN Algorithm Review

There are multiple variants of DiskANN [6, 9, 1]. We review two variants that are related in this paper: the original Vamana heuristics proposed in [10] and the "slow preprocessing" variant formalized in [8].

The search algorithm is the same for both variants (see Algorithm 2). Given the index graph, the search is initialized by seeding a candidate set $\mathcal{L}$ with a designated start vertex s (typically the medoid) and maintaining an empty set of visited vertices $\mathcal{V}$. In each iteration, the algorithm explores the graph by selecting the point p^* from the unexplored candidates in $\mathcal{L}$ that is most proximal to the query q. By immediately adding the out-neighbors $N_{\text{out}}(p^*)$ to the candidate set $\mathcal{L}$ and marking p^* as visited, the algorithm effectively "walks" through the graph's edges toward the query. To ensure the search remains computationally tractable, the candidate set $\mathcal{L}$ is restricted to a maximum size L; if the addition of neighbors causes the set to exceed this bound, $\mathcal{L}$ is pruned to retain only the L points closest to q. This iterative process of greedy selection and neighborhood expansion continues until no unvisited points remain in the search beam, at which point the k best-found candidates are returned.

The index construction of both variants relies on the subroutine ROBUSTPRUNE (see Algorithm 1). ROBUSTPRUNE prunes out-neighbor candidates for the input vertex p and utilizes a parameter $\alpha \geq 1$ to favor "diverse" neighbors that provide long-range shortcuts across the dataset. The algorithm proceeds by iteratively selecting the point p^* in the candidate set $\mathcal{V}$ that is closest to the target point p. Once a candidate is added to $N_{out}(p)$, a pruning condition is applied to all remaining candidates: any point p' that is closer to the newly added out-neighbor p^* than it is to the target point p by a factor of α (i.e., $D(p^*, p') \leq D(p, p')/\alpha$) is removed from the candidate set $\mathcal{V}$. The intuition is that the edge (p, p') is potentially not important if (p, p^*) already helps move to p' since moving from p to p^* decreases the distance to p' by factor α.

The "slow preprocessing" construction executes ROBUSTPRUNE$(p, P\setminus\{p\}, \alpha, n-1)$ for every vertex p in the graph. Notice the candidate set $\mathcal{V} = P \setminus \{p\}$, and the degree limit $R = n - 1$, which results in an $O(n^3)$ time complexity that is impractical at scale. However, it provides a rigorous structural foundation, α-reachability, for analyzing the algorithm's worst-case performance on construction time, query time, and approximation factor.

Definition 2.1 (α**-reachable and** α**-reachability).** Let $\alpha > 1$. We say, in a graph $G = (P, E)$, q is α-reachable from another vertex p if either $(p, q) \in E$, or there exists p' s.t. $(p, p') \in E$ and $D(p', q) \cdot \alpha \leq D(p, q)$. We say a graph G is α-reachable if any vertex in P is α-reachable from any other vertex in P.

Moreover, we define the reachability of a graph G as the maximum α^* such that G is α^*-reachable.

Then, the following result on "slow preprocessing" construction can be proved. Recall λ and Δ denote the dataset's doubling dimension and the aspect ratio.

Lemma 2.2 (Lemma 3.2 and 3.3 in [8])**.** *Applying* RobustPrune$(p, P \setminus \{p\}, \alpha, \infty)$ *for every* $p \in P$ *yields an* α*-reachable graph with maximum degree* $O((4\alpha)^\lambda \log \Delta)$.

They also provided a lower bound on worst-case search performance, which is later improved by [7]. In particular, we have

Theorem 2.3 (Theorem 1.1 in [7])**.** *Let* G *be a* α*-reachable graph. For any query* q*, any start point* s*, any constant* $\epsilon > 0$*, and any* $k < L$*,* GreedySearch(s, q, k, L) *in at most* $O\left(L + \log_\alpha \frac{\Delta}{(\alpha-1)\epsilon}\right)$ *steps outputs a set of points* $\{b_1, \ldots, b_L\}$ *such that each* b_j *satisfies:*

$$D(b_j, q) \leq \epsilon + \frac{\alpha}{\alpha - 1} \cdot D(a_j, q) \quad \textit{for } \ell_2 \textit{ metric } D$$
$$D(b_j, q) \leq \epsilon + \frac{\alpha + 1}{\alpha - 1} \cdot D(a_j, q) \quad \textit{for any metric } D$$

where a_j *is the* j*th nearest neighbor to* q*.*

From the above, we can see the parameter α is a key parameter for the worst-case trade-off of the "slow preprocessing" DiskANN (a α-reachable graph), which has

- Storage: $O(n \cdot (4\alpha)^\lambda \log \Delta)$
- Query Time: $O\left(\left(L + \log_\alpha \frac{\Delta}{(\alpha-1)\epsilon}\right) \cdot (4\alpha)^\lambda \log \Delta\right)$
- Approximation: $\left(\epsilon + \frac{\alpha}{\alpha-1}\right)$ for Euclidean metrics; $\left(\epsilon + \frac{\alpha+1}{\alpha-1}\right)$ for general metrics.

As α increases, storage and query time increase while the approximation factor decreases, indicating more accurate output.

The original Vamana index construction, on the other hand, caps the maximum out-degree of the graph for fast construction in practice. Besides RobustPrune, it uses other heuristics, which we will not discuss since we never open the black box in this paper and we refer readers to [10] for details. Despite no clear worst-case guarantee, Vamana can achieve state-of-the-art empirical performance, and the parameter α influences the empirical trade-off in the same directions as it does in the theoretical framework of the slow preprocessing variant.

Algorithm 1 RobustPrune $(p, \mathcal{V}, \alpha, R)$

Input: Graph G, point $p \in P$, candidate set $\mathcal{V}$, desired reachability $\alpha \geq 1$, degree bound R
Output: G is modified by setting at most R new out-neighbors for p
$\mathcal{V} \leftarrow (\mathcal{V} \cup N_{\text{out}}(p)) \setminus \{p\}$
$N_{\text{out}}(p) \leftarrow \emptyset$
while $\mathcal{V} \neq \emptyset$ **do**
  $p^* \leftarrow \arg\min_{p' \in \mathcal{V}} D(p, p')$
  $N_{\text{out}}(p) \leftarrow N_{\text{out}}(p) \cup \{p^*\}$
  if $|N_{\text{out}}(p)| = R$ **then**
    break
  for $p' \in \mathcal{V}$ **do**
    if $\alpha \cdot D(p^*, p') \leq D(p, p')$ **then**
      Remove p' from $\mathcal{V}$

Algorithm 2 GreedySearch(s, q, k, L)

Input: Graph G with start vertex s, query q, result size k, search list size $L \geq k$
Output: Result set $\mathcal{L}$ containing k-approx NNs, and a set $\mathcal{V}$ containing all the visited vertices
Initialize sets $\mathcal{L} \leftarrow \{s\}$ and $\mathcal{V} \leftarrow \emptyset$
while $\mathcal{L} \setminus \mathcal{V} \neq \emptyset$ **do**
  Let $p^* \leftarrow \arg\min_{p \in \mathcal{L} \setminus \mathcal{V}} D(p, q)$
  Update $\mathcal{L} \leftarrow \mathcal{L} \cup N_{\text{out}}(p^*)$ and $\mathcal{V} \leftarrow \mathcal{V} \cup \{p^*\}$
  if $|\mathcal{L}| > L$ **then**
    Update $\mathcal{L}$ to retain closest L points to q
return [closest k points from $\mathcal{L}$; $\mathcal{V}$]

3 RP-TUNING and Worst-Case Guarantees

In this section, we propose RP-TUNING, a simple yet disciplined pruning algorithm based on ROBUSTPRUNE, which efficiently prunes a α-reachable graph G into a sparser graph with a theoretical guarantee of worst-case performance trade-off.

RP-TUNING RP-TUNING takes an initially α_1-reachable graph $G = (P, E)$, and a new reachability target $\alpha_2 < \alpha_1$ as inputs. It simply runs ROBUSTPRUNE$(p, N_{out}(p), \alpha_2, \infty)$ for every $p \in P$ and outputs the modified graph $G' = (P, E')$.

We study the worst-case reachability of the output G' of RP-TUNING(G, α_2) where G is a α_1-reachable graph. It turns out that the worst-case reachability of G' is a function of α_1, α_2 and depends on the metric space and whether the pruning subroutine ROBUSTPRUNE sorts the out-neighbors. To distinguish the unsorted version from ROBUSTPRUNE, we use UNSORTEDROBUSTPRUNE to denote the procedure that is the same as ROBUSTPRUNE but only replaces the step "$p^* \leftarrow \arg\min_{p' \in \mathcal{V}} D(p, p')$" in $Algorithm\ 1$ with "p^* being an arbitrary out-neighbor of p". Our results on the reachability of G' in different settings are summarized in Table 1, and formally stated in Theorem 3.1 and Theorem 3.2.

Table 1: Worst-case reachability after RP-TUNING

	GENERAL	EUCLIDEAN
UNSORTED	$\frac{1}{\frac{1}{\alpha_1}+\frac{1}{\alpha_2}+1}$	$\frac{1}{\frac{1}{\alpha_1}+\frac{1}{\alpha_2}+1}$
SORTED	$\frac{1}{\frac{1}{\alpha_1}+\frac{1}{\alpha_2}}$	$\frac{1}{\frac{1}{\alpha_1}\cdot\sqrt{1-\frac{1}{4\alpha_2^2}}+\frac{1}{\alpha_2}\cdot\sqrt{1-\frac{1}{4\alpha_1^2}}}$

Note $1/(\frac{1}{\alpha_1} \cdot \sqrt{1 - \frac{1}{4\alpha_2^2}} + \frac{1}{\alpha_2} \cdot \sqrt{1 - \frac{1}{4\alpha_1^2}})$ is larger than $\frac{1}{\frac{1}{\alpha_1}+\frac{1}{\alpha_2}}$. For a concrete example, when $\alpha_1 = 3, \alpha_2 = 2$, this expression ≈ 1.226 while $\frac{\alpha_1\alpha_2}{\alpha_1+\alpha_2} = 1.2$.

Theorem 3.1 (Unsorted Reachability)**.** *Let P be a dataset in any metric space (X, D). Fix finite parameters $\alpha_1 \geq \alpha_2 > 1$. Let $G = (P, E)$ be any α_1-reachable graph on P. Let $G' = (P, E')$ be the graph obtained by applying* UNSORTEDROBUSTPRUNE$(p, N_{out}(p), \alpha_2, \infty)$ *for all $p \in P$. Then, the graph G' has reachability $1/\left(1/\alpha_1 + 1/\alpha_2 + 1\right)$ in the worst case.*

Theorem 3.2 (Sorted Reachability)**.** *Let P be a dataset in a metric space (X, D). Fix finite parameters $\alpha_1 \geq \alpha_2 > 1$. Let $G = (P, E)$ be any α_1-reachable graph on P. Let $G' = (P, E')$ be the graph constructed by applying* ROBUSTPRUNE$(p, N_{out}(p), \alpha_2, \infty)$ *for all $p \in P$. Then the following holds in the worst case:*

- *For any metric space, G' has reachability $1/\left(1/\alpha_1 + 1/\alpha_2\right)$.*
- *If (X, D) is a Euclidean space, then G' has reachability $1/\left((1/\alpha_1) \cdot \sqrt{1 - \frac{1}{4\alpha_2^2}} + (1/\alpha_2) \cdot \sqrt{1 - \frac{1}{4\alpha_1^2}}\right)$.*

Notice that the worst-case reachability given by unsorted pruning is less than 1, while the worst-case reachability given by sorted pruning could be larger than 1. This implies sorting is helpful with pruning. Readers may find an analogy in our results to [7], where Euclidean metrics and neighbor sorting guarantee a better worst case reachability. In our problem, neighbor sorting alone already admits a better reachability, and it is further improved in Euclidean metrics. However, Euclidean metrics alone do not admit better reachability.

The reason why the reachability of the pruned graph G' is crucial is as follows. First, the following lemma in [8] can help us to bound the maximum degree of the pruned graph G'.

Lemma 3.3 (Lemma 3.3 in [8])**.** *For any $\alpha > 1$, graph $G = (P, E)$, and $p \in P$, applying* RobustPrune$(p, N_{out}(p), \alpha, \infty)$ *to every $p \in P$ guarantees that the maximum degree of the modified graph is $O((4\alpha)^\lambda \log \Delta)$.*

Using the above lemma, we know that the size of G' is $n \cdot O((4\alpha_2)^\lambda \log \Delta)$. The query time of G' now only depends on the number of steps during search, and the approximation of G' can be quantified by its reachability via Theorem 2.3, where the larger the reachability is, the smaller both the number of steps and approximation are. Therefore, the rest of the analysis on query time and approximation is reduced to exploring the reachability of G'. We want it to be as large (or as close to α_2) as possible.

Next, we give the proofs of the above two theorems and include the skipped proof in the appendix.

3.1 Proof of Theorem 3.1

Let $G = (P, E)$ be an α_1-reachable graph and let $G' = (P, E')$ be obtained by applying UnsortedRobustPrune$(p, \emptyset, \alpha_2, \infty)$ at every $p \in P$.

Let α^* denote the reachability of G'. Fix an ordered pair p, z that is tight for α^*-reachability and satisfies $(p, z) \notin E'$ (such a pair must exist since α^* is finite). Then, the directed edge (p, z) was either (a) absent in G, or (b) pruned from G. In case (b), there is some edge $(p, y) \in E$ for which $D(y, z) \leq D(p, z)/\alpha_2$, which gives $\alpha^* \geq \alpha_2$.

In case (a), α_1-reachability of G implies there is some $(p, y) \in E$ such that $D(y, z) \leq D(p, z)/\alpha_1$. This means that applying UnsortedRobustPrune with parameter α_2 either (a1) prunes (p, y) from G', or (a2) there is some edge $(p, x) \in E$ such that $D(x, y) \leq D(p, y)/\alpha_2$. As before, in case (a1) it follows that $\alpha^* \geq \alpha_1$. In case (a2), the edge (p, x) gives a lower bound on the reachability on G'. Namely, α^* is at least the value of the following optimization problem:

$$\begin{aligned} \min_{p,x,y,z \in X} \quad & D(p, z)/D(x, z) \\ \text{subject to} \quad & D(y, z) \leq D(p, z)/\alpha_1 \\ & D(x, y) \leq D(p, y)/\alpha_2 \end{aligned}$$

By invariance under scaling, we may take $D(p, z) = 1$ and flip the objective to get the following crucial lemma.

Lemma 3.4. *Let (X, D) be a metric space. Given parameters $\alpha_1 \geq \alpha_2 \geq 1$, the worst-case reachability of the graph G' obtained by taking an α_1-reachable graph G and applying* UnsortedRobustPrune$(p, \emptyset, \alpha_2, \infty)$ *at every $p \in P$ is $\min\{\alpha_2, 1/\mathrm{OPT}\}$ where OPT is the value of the following optimization problem:*

$$\begin{aligned} \max_{p,x,y,z \in X} \quad & D(x, z) \\ \textit{subject to} \quad & D(y, z) \leq 1/\alpha_1 \\ & D(x, y) \leq D(p, y)/\alpha_2 \\ & D(p, z) = 1 \end{aligned}$$

We solve the optimization problem in a straightforward geometric way. We first just apply triangle inequality and the constraints in the above problem to get an upper bound of $D(x, z)$:

$$\begin{aligned} D(x,z) &\le D(x,y) + D(y,z) \\ &\le D(p,y)/\alpha_2 + 1/\alpha_1 \\ &\le (D(p,z) + D(y,z))/\alpha_2 + 1/\alpha_1 \\ &\le (1 + 1/\alpha_1)/\alpha_2 + 1/\alpha_1 = (\alpha_1 + \alpha_2 + 1)/(\alpha_1\alpha_2), \end{aligned}$$

where the first and third line use triangular inequality and the second and fourth line use the constraints of the optimization problem. We also know $D(x, z)$ in any setting of p, x, y, z that satisfies the problem constraints is a lower bound of the optimal objective since it is a maximization problem. It remains to show a setting where $D(x, z) = (\alpha_1 + \alpha_2 + 1)/(\alpha_1\alpha_2)$.

Lemma 3.5. *In a general metric space or a Euclidean space (X, D), there exists a setting of four points $p, x, y, z \in X$ that satisfies $D(y, z) \le 1/\alpha_1, D(x, y) \le D(p, y)/\alpha_2, D(p, z) = 1$, and $D(x, z) = (\alpha_1 + \alpha_2 + 1)/(\alpha_1\alpha_2)$.*

Proof. In the setting, p, x, y, z are collinear and $D(p, z) = 1, D(z, y) = 1/\alpha_1, D(x, y) = (1 + 1/\alpha_1)/\alpha_2$. The constraints can be easily verified. We then compute $D(x, z) = 1/\alpha_1 + (1 + 1/\alpha_1)/\alpha_2 = (\alpha_1 + \alpha_2 + 1)/(\alpha_1\alpha_2)$ as desired. □

The setting in the proof is shown in Figure 1. The setting exactly makes the inequalities used in

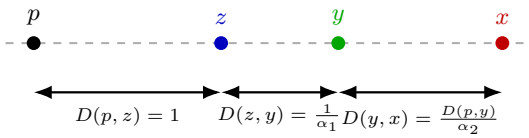

Figure 1: The setting that achieves the optimal objective in the optimization problem in Lemma 3.4.

constructing the upper bound tight. In conclusion, we show the optimization problem solves to $(\alpha_1 + \alpha_2 + 1)/(\alpha_1\alpha_2)$. By Lemma 3.4, the worst-case reachability is $\min(\alpha_1, \alpha_2, (\alpha_1\alpha_2)/(\alpha_1 + \alpha_2 + 1)) = (\alpha_1\alpha_2)/(\alpha_1 + \alpha_2 + 1)$. That completes the proof for Theorem 3.1.

Also notice that, in Figure 1, x is the furthest point from p, while z is the closest one. Later we will see that this is a forbidden pattern when using RobustPrune instead of UnsortedRobustPrune, because x, y, z have to be considered in order of their distances to p in order to achieve an analogous case to (a2).

3.2 Proof of Theorem 3.2

Now we use RobustPrune where neighbors in the graph are considered for pruning in ascending order of their distances to the pivot point. The proof follows the same template as the unsorted case. The key difference is that, because RobustPrune processes candidates in increasing distance from p, any point x that can prune y must be considered no later than y, hence must satisfy $D(p, x) \le D(p, y)$. Moreover, since y serves as the α_1-reachability witness for (p, z), we also have $D(p, y) \le D(p, z)$. This yields the following lemma.

Lemma 3.6. *Let $(X, D(\cdot, \cdot))$ be a metric space, and $p, x, y, z \in X$ be four points. Given parameters satisfying $\alpha_1 \ge \alpha_2 \ge 1$, the worst-case reachability of the graph G' obtained by taking an α_1-reachable graph and applying* RobustPrune$(p, \emptyset, \alpha_2, \infty)$ *at every $p \in P$ is $\min\{\alpha_2, 1/\mathrm{OPT}\}$, where* OPT *is the value of the following optimization problem:*

$$\begin{aligned} \max_{p,x,y,z \in X} \quad & D(x,z) \\ \textit{subject to} \quad & D(y,z) \le 1/\alpha_1 \\ & D(x,y) \le D(p,y)/\alpha_2 \\ & D(x,y) \le D(p,y) \le D(p,z) = 1 \end{aligned}$$

Notice that the only difference in the optimization problem is that there is an ordered condition in the last constraint, which rules out the pattern seen in the previous subsection. However, we can obtain an upper bound on the optimal objective of the problem in the same way using the triangular inequality and problem constraints: $D(x,z) \le D(x,y) + D(y,z) \le D(p,y)/\alpha_2 + 1/\alpha_1 \le 1/\alpha_2 + 1/\alpha_1 = (\alpha_1+\alpha_2)/(\alpha_1\alpha_2)$. Note that the upper bound is smaller than the one in unsorted case. In the general metric space, the upper bound is actually tight.

Lemma 3.7. *In a general metric space (X, D), there exists a setting of four points $p, x, y, z \in X$ that satisfies $D(y,z) \le 1/\alpha_1, D(x,y) \le D(p,y)/\alpha_2, D(x,y) \le D(p,y) \le D(p,z) = 1$, and $D(x,z) = (\alpha_1+\alpha_2)/(\alpha_1\alpha_2)$.*

Proof. The setting is defined by the following distance matrix:

	p	x	y	z
p	0			
x	1	0		
y	1	$1/\alpha_1$	0	
z	1	$\frac{\alpha_1+\alpha_2}{\alpha_1\alpha_2}$	$1/\alpha_2$	0

It is easy to verify that the triangular inequalities and the problem constraints are satisfied. □

Figure 2(a) shows the setting in the proof, where x, y, z are on the (general metric) unit sphere centered at p. Since the lower bound of $D(x,z)$ given by the above setting matches the upper bound, we can conclude that the worst-case reachability is $(\alpha_1\alpha_2)/(\alpha_1+\alpha_2)$ for general metric space by Lemma 3.6.



(a) General Metrics (b) Euclidean

Figure 2: The settings that achieve optimal objectives in the optimization problem in Lemma 3.6 for (a) general metric spaces and (b) Euclidean metrics.

Unfortunately, the optimal setting for general metric space that requires x, y, z are all collinear and on a unit sphere is impossible in Euclidean metrics. That indicates the upper bound derived only using the triangular inequality and the problem constraints is likely to be loose in Euclidean space. We resort to tools for semi-definite programming to solve for Euclidean space as in [7] and get the following lemma.

Lemma 3.8. *Let $\alpha_1 \ge \alpha_2 > 1$. The value of the optimization problem*

$$\begin{aligned} \max_{x,y,z \in \mathbb{R}^d} \quad & \|x-z\|_2 \\ \textit{s.t.} \quad & \|y-z\|_2 \le \|z\|_2/\alpha_1, \\ & \|x-y\|_2 \le \|y\|_2/\alpha_2, \\ & \|x\|_2 \le \|y\|_2 \le \|z\|_2 = 1 \end{aligned}$$

is

$$(1/\alpha_1)\cdot\sqrt{1-1/4\alpha_2^2} \;+\; (1/\alpha_2)\cdot\sqrt{1-1/4\alpha_1^2}.$$

The optimal solution can be obtained by p, x, y, z on the same 2-dimensional plane in Euclidean metrics. In the optimal setting (Figure 2(b)), intuitively speaking, the curvature of the Euclidean unit

sphere pushes x, z towards each other, which results in a smaller maximum $D(x, z)$, thus a larger worst-case reachability for sorted Euclidean metrics.

Lemma 3.6, Lemma 3.7, and Lemma 3.8 complete the proof of Theorem 3.2. We delay the proof of Lemma 3.8 to the appendix.

4 Experimental Evaluation

We have shown theoretical guarantees of the α-reachable graphs pruned by RP-TUNING. In this section, we test RP-TUNING on DiskANN indices to evaluate its real-world performance. We will use 100-Recall@100, a widely adopted metric in practice, to measure query accuracy. 100-Recall@100 is the proportion of the true top 100 nearest neighbors contained in the top 100 results returned by the algorithm.

4.1 Setup

Hardware. We conduct all experiments on a cloud instance equipped with an AMD EPYC 9B14 CPU (15 cores, 30 threads) and 117GB RAM, running Debian GNU/Linux 12.

Datasets. We evaluate our methods on four benchmark datasets: SIFT-1M, GIST-1M, and Deep-1M are obtained from the ANN-Benchmarks repository [2], and MSSPACEV-1M from the BigANN benchmark [13]. **SIFT-1M** and **GIST-1M** [11] are classical computer vision datasets; the former consists of 1 million 128-dimensional SIFT feature vectors with 10,000 query vectors, while the latter contains 1 million 960-dimensional GIST descriptors capturing global image structure, with 1,000 query vectors. **Deep-1M** [3] comprises 1 million 96-dimensional vectors derived from deep neural network embeddings, with 10,000 query vectors. Finally, **MSSPACEV-1M** is a subset of 1 million 100-dimensional vectors from Microsoft's Bing query embeddings [13], originally stored as int8 and converted to float32, with 29,316 query vectors. All datasets use Euclidean distance for similarity.

Tuning Scheme Configuration We implemented the original DiskANN (Vamana)[2] in [10] in Python. We construct the DiskANN indices for the above four datasets with the same parameter setting in [10]: $\alpha_1 = 1.5, R = 70, L$-Build $= 75$. They are referred to as the base or meta indices in the rest of this section. Then we tune the base indices via RP-TUNING and rebuild with $\alpha_2 = 1.4, 1.3, 1.2, 1.1$. During query time, we use Algorithm 2 with the start vertex s being the medoid of the dataset, $k = 100$ (since we will use 100-Recall@100 to measure accuracy). We vary the index search effort by tuning the beam size parameter L of Algorithm 2 as a knob to obtain different trade-offs between recall and query latency.

4.2 Comparison between RP-TUNING and Rebuild for Tuning

Table 2: Runtime comparison at $\alpha_{\text{base}} = 1.5$: Rebuild vs. RP-TUNING (seconds). The rightmost columns show the cumulative time and overall speedup (Total T_{rebuild}/Total T_{prune}) across all four configurations.

	$\alpha = 1.1$			$\alpha = 1.2$			$\alpha = 1.3$			$\alpha = 1.4$			**Total (All 4 α values)**		
Dataset	T_{rebuild}	T_{prune}	Speedup	T_{rebuild}	T_{prune}	Speedup	T_{rebuild}	T_{prune}	Speedup	T_{rebuild}	T_{prune}	Speedup	T_{rebuild}	T_{prune}	Speedup
SIFT	6,728	362	19×	10,383	477	22×	17,993	586	31×	29,822	683	44×	64,926	2,108	31×
DEEP	6,745	359	19×	14,760	479	31×	23,739	594	40×	41,919	704	60×	87,163	2,136	41×
GIST	21,312	480	44×	41,181	688	60×	65,296	838	78×	96,635	1,061	91×	224,424	3,067	73×
MSSPACEV	11,570	412	28×	37,806	559	68×	51,523	685	75×	73,313	1,051	70×	174,212	2,707	64×

We compare the performance of RP-TUNING and rebuild for tuning the base index with four new α values by total processing time cost and recall-QPS trade-off.

Table 2 demonstrates the tuning efficiency of RP-TUNING. Our method achieves significant speedups compared to the rebuilding baseline across all datasets. We attribute this performance to the fact that RP-TUNING prunes an already-sparse graph, rather than a dense random graph. Notably, the speed-up is the most significant on GIST (achieving up to 73 $\times$ total speedup). That is due to the fact the $\alpha = 1.5$ graph on GIST has average degree 51.3, compared to 65.6 for MSSPACEV.

[2]The index is called Vamana in [10] and "fast preprocessing version" in [8].

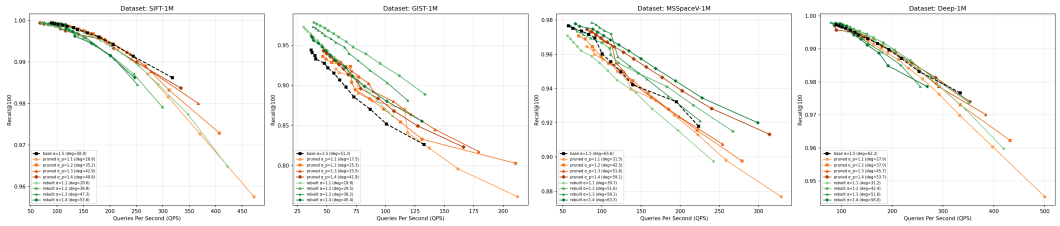

Figure 3: Recall-QPS trade-off frontiers achieved by a base DiskANN graph of $\alpha_1 = 1.5$ then pruned graphs from the base graph via RP-TUNING with $\alpha_2 = \{1.4, 1.3, 1.2, 1.1\}$, and the same base DiskANN graph of $\alpha = 1.5$ along with rebuilt DiskANN graphs of $\alpha = \{1.4, 1.3, 1.2, 1.1\}$.

Figure 3 illustrates the performance of the graphs returned by RP-TUNING compared to the base graphs and the rebuilt graphs. Starting with a base Vamana graph constructed with $\alpha = 1.5$, we apply RP-TUNING using $\alpha \in \{1.4, 1.3, 1.2, 1.1\}$. As shown, graph tuning yields two types of performance gains: (1) a strictly better Pareto frontier for the recall-QPS tradeoff; (2) significantly reduced memory overhead and improved maximum QPS, which is particularly important for resource-constrained and latency-critical settings.

For datasets SIFT and DEEP, pruned and rebuilt graphs only achieve (2), i.e., improved maximum QPS. In the high QPS regime, pruned graphs achieved better performance compared to rebuilt graphs. For example, for any QPS > 350 on SIFT, the recall of pruned graphs exceeds the recall of the rebuilt graphs.

For datasets GIST and MSSPACEV, pruned and rebuilt graphs are able to achieve both (1) and (2), i.e., better recall-QPS tradeoffs as well as improved maximum QPS. For GIST, the rebuilt graph obtains significant recall-QPS performance improvement, while pruned graphs achieve medium improvement on recall-QPS and significantly higher maximum QPS. Note that adjusting the beam size will not further increase QPS for the rebuilt graphs on GIST: the point with maximum QPS on the curve already has beam size 100, which is the largest possible setting for $k = 100$. For MSSPACEV, the best pruned graph ($\alpha_2 = 1.4$) is slightly worse than the best rebuilt graph ($\alpha = 1.4$). However, the best rebuilt graph uses more edges/space (its average degree is 63.3 v.s. 59.1 of the pruned graph). In addition, pruning is tens of times faster than rebuilding.

In Appendix B, we show that RP-TUNING yields improved recall-QPS tradeoffs when applied to HNSW [12], another popular graph-based ANN algorithm.

4.3 Comparison with Random Edge Dropout

The experiments in Subsection 4.2 demonstrate that RP-TUNING achieves performance gains over rebuilt graph indexes. Here, we show that the naive pruning strategy of random edge dropout does not achieve the same improvements.

For each of the graphs pruned from the base graph with $\alpha_1 = 1.5$, we build a randomly pruned graph of the same average degree. For example, for the SIFT dataset with $\alpha_2 = 1.2$, the average degree is 35.2. We create a corresponding dropout graph by randomly deleting $\epsilon = 1 - 35.2/58 \approx 0.39$ fraction of edges per node (since the base graph has average degree 58; see Figure 4). Across all four datasets, all the randomly pruned graphs (except the randomly pruned graph with $\epsilon = 0.211$ on MSSPACEV) have significantly worse Recall-QPS tradeoffs as compared to the pruned graphs. This demonstrates that the performance gains of RP-TUNING are not due to graph sparsity alone, but rather preserving high-quality edges.

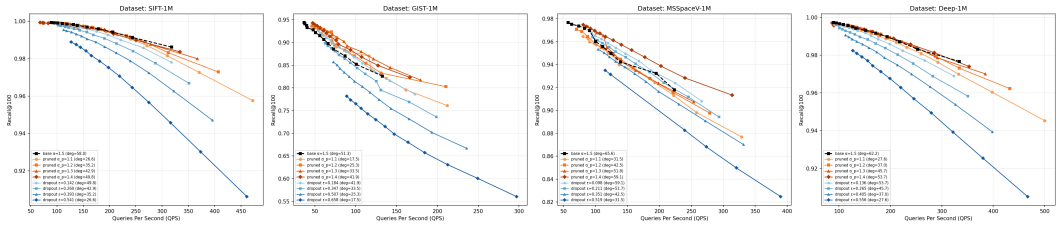

Figure 4: Recall-QPS trade-offs of graphs pruned via RP-TUNING from a base DiskANN graph with $\alpha_1 = 1.5$ and of randomly pruned (dropout) graphs with matching average degree, where ϵ is the fraction of edges deleted per node. Average degrees (deg) are shown in the legends.

5 Conclusions and limitations

In this paper, we proposed a novel index tuning algorithm RP-TUNING for α-reachable graphs, based on a subroutine of DiskANN construction. This strategy provides theoretical guarantees on graph reachability as well as highly efficient performance (with gains of up to 73x). This allows users to improve query latency or optimize the tradeoff with search accuracy and index size.

We identify three primary limitations of RP-TUNING. (1) Theoretically, the worst-case lower bound on reachability (Theorem 3.2) can be vacuous (less than 1) in specific parameter regimes. For example, when $\alpha_1 = 2$ and $\alpha_2 = 1.5$, the guaranteed worst-case reachability in a general metric space is only $6/7$. However, our experiments suggest that real-world datasets do not exhibit this worst-case behavior in practice. (2) Empirically, our evaluation focuses on using RP-TUNING to prune DiskANN and HNSW, leaving open its effectiveness for other graph-based ANN algorithms. (3) RP-TUNING can only decrease graph reachability by pruning; designing efficient methods to increase reachability is an interesting direction for future work.

Acknowledgments and Disclosure of Funding

This work was supported by the National Science Foundation (NSF) Grant No. CCF-2337993; the NSF GRFP under Grant No. DGE-2236662; and funded in part by NSF DBI-2400135 and IIS-2551799.

References

[1] P. Adams, M. Li, S. Zhang, L. Tan, Q. Chen, M. Li, Z. Li, K. Risvik, and H. V. Simhadri. Distributedann: Efficient scaling of a single diskann graph across thousands of computers, 2025. URL `https://arxiv.org/abs/2509.06046`.

[2] M. Aumüller, E. Bernhardsson, and A. Faithfull. ANN-Benchmarks: A benchmarking tool for approximate nearest neighbor algorithms. *Information Systems*, 87:101374, 2020.

[3] A. Babenko and V. Lempitsky. Efficient indexing of billion-scale datasets of deep descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2055–2063, 2016.

[4] M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P.-E. Mazaré, M. Lomeli, L. Hosseini, and H. Jégou. The faiss library. *IEEE Transactions on Big Data*, 2025.

[5] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, Q. Guo, M. Wang, and H. Wang. Retrieval-Augmented Generation for Large Language Models: A Survey. *ArXiv*, abs/2312.10997, 2023. URL `https://api.semanticscholar.org/CorpusID:266359151`.

[6] S. Gollapudi, N. Karia, V. Sivashankar, R. Krishnaswamy, N. Begwani, S. Raz, Y. Lin, Y. Zhang, N. Mahapatro, P. Srinivasan, A. Singh, and H. V. Simhadri. Filtered-diskann: Graph algorithms for approximate nearest neighbor search with filters. In *Proceedings of the ACM Web Conference*

2023, WWW '23, page 3406–3416, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9781450394161. doi: 10.1145/3543507.3583552. URL https://doi.org/10.1145/3543507.3583552.

[7] S. Gollapudi, R. Krishnaswamy, K. Shiragur, and H. Wardhan. Sort before you prune: Improved worst-case guarantees of the diskANN family of graphs. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=JnXbUKtLzz.

[8] P. Indyk and H. Xu. Worst-case performance of popular approximate nearest neighbor search implementations: Guarantees and limitations, 2023. URL https://arxiv.org/abs/2310.19126.

[9] S. Jaiswal, R. Krishnaswamy, A. Garg, H. V. Simhadri, and S. Agrawal. Ood-diskann: Efficient and scalable graph anns for out-of-distribution queries, 2022. URL https://arxiv.org/abs/2211.12850.

[10] S. Jayaram Subramanya, F. Devvrit, H. V. Simhadri, R. Krishnawamy, and R. Kadekodi. Diskann: Fast accurate billion-point nearest neighbor search on a single node. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/09853c7fb1d3f8ee67a61b6bf4a7f8e6-Paper.pdf.

[11] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011.

[12] Y. 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, 2018.

[13] H. V. Simhadri, G. Williams, M. Aumüller, M. Douze, A. Babenko, D. Baranchuk, Q. Chen, L. Hosseini, R. Krishnaswamy, G. Srinivasa, et al. 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, pages 177–189. PMLR, 2022.

[14] N. Upreti, H. V. Simhadri, H. S. Sundar, K. Sundaram, S. Boshra, B. Perumalswamy, S. Atri, M. Chisholm, R. R. Singh, G. Yang, T. Hass, N. Dudhey, S. Pattipaka, M. Hildebrand, M. Manohar, J. Moffitt, H. Xu, N. Datha, S. Gupta, R. Krishnaswamy, P. Gupta, A. Sahu, H. Varada, S. Barthwal, R. Mor, J. Codella, S. Cooper, K. Pilch, S. Moreno, A. Kataria, S. Kulkarni, N. Deshpande, A. Sagare, D. Billa, Z. Fu, and V. Vishal. Cost-Effective, Low Latency Vector Search with Azure Cosmos DB, July 2025. URL http://arxiv.org/abs/2505.05885. arXiv:2505.05885.

[15] J. Wang, X. Yi, R. Guo, H. Jin, P. Xu, S. Li, X. Wang, X. Guo, C. Li, X. Xu, K. Yu, Y. Yuan, Y. Zou, J. Long, Y. Cai, Z. Li, Z. Zhang, Y. Mo, J. Gu, R. Jiang, Y. Wei, and C. Xie. Milvus: A Purpose-Built Vector Data Management System. In *Proceedings of the 2021 International Conference on Management of Data*, pages 2614–2627, Virtual Event China, June 2021. ACM. ISBN 9781450383431. doi: 10.1145/3448016.3457550. URL https://dl.acm.org/doi/10.1145/3448016.3457550.

[16] A. Zeakis, G. Papadakis, D. Skoutas, and M. Koubarakis. Pre-trained embeddings for entity resolution: an experimental analysis. *Proceedings of the VLDB Endowment*, 16(9):2225–2238, 2023.

A Proof of Lemma 3.8

For convenience, we restate the optimization problem here:

$$\begin{aligned} \max_{x,y,z\in\mathbb{R}^d} \quad & \|x-z\|_2 \\ \text{s.t.} \quad & \|y-z\|_2 \le \frac{\|z\|_2}{\alpha_1}, \\ & \|x-y\|_2 \le \frac{\|y\|_2}{\alpha_2}, \\ & \|x\|_2 \le \|y\|_2 \le \|z\|_2 = 1 \end{aligned}$$

Let OPT denote the optimal value of the stated problem, and define

$$\beta(\alpha_1,\alpha_2) := \frac{1}{\alpha_1}\sqrt{1-\frac{1}{4\alpha_2^2}} + \frac{1}{\alpha_2}\sqrt{1-\frac{1}{4\alpha_1^2}}.$$

We prove $\mathrm{OPT} = \beta(\alpha_1,\alpha_2)$.

For convenience of notation, we will let $\|v\|$ denote the ℓ_2-norm of v in this proof.

Upper bound. Let $f(x,y,z) = \|x-z\|^2$. By monotonicity of $\sqrt{\cdot}$, it suffices to upper bound the optimum of f by $\beta(\alpha_1,\alpha_2)^2$. We relax the constraint $\|z\| = 1$ to $\|z\| \le 1$ and rewrite all constraints as quadratic:

$$\begin{aligned} \max_{x,y,z\in\mathbb{R}^d} \quad & f(x,y,z) := \|x-z\|^2 \\ \text{subject to} \quad & g_1(x,y,z) := \|x\|^2 - \|y\|^2 \le 0 \\ & g_2(x,y,z) := \|y\|^2 - \|z\|^2 \le 0 \\ & g_3(x,y,z) := \|y-z\|^2 - \|z\|^2/\alpha_1^2 \le 0 \\ & g_4(x,y,z) := \|x-y\|^2 - \|y\|^2/\alpha_2^2 \le 0 \\ & g_5(x,y,z) := \|z\|^2 - 1 \le 0 \end{aligned}$$

For multipliers $\lambda_1,\dots,\lambda_5 \ge 0$, define the Lagrangian

$$L(x,y,z;\lambda) := f(x,y,z) - \sum_{i=1}^{5}\lambda_i g_i(x,y,z).$$

Then, for any feasible $x,y,z\in\mathbb{R}^d$, one has $L(x,y,z;\,\lambda) \ge f(x,y,z)$, meaning the optimum value of f is at most

$$\sup_{x,y,z\in\mathbb{R}^d} L(x,y,z;\,\lambda).$$

We will now specify our settings of $\lambda_1,\dots,\lambda_5$ as follows. For $i\in\{1,2\}$, set

$$\theta_i := \sin^{-1}\left(\frac{1}{2\alpha_i}\right) \in (0,\pi/6], \qquad h_i := \sin(2\theta_i), \qquad h_{12} := \sin(2(\theta_1+\theta_2)).$$

Note that $1/\alpha_i^2 = 4\sin^2\theta_i = 2(1-\cos 2\theta_i)$, and also $\beta(\alpha_1,\alpha_2) = 2\sin(\theta_1+\theta_2)$. Then, define

$$\lambda_1 = \frac{h_1+h_2-h_{12}}{h_2} \qquad \lambda_2 = \frac{h_1+h_2-h_{12}}{h_1} \qquad \lambda_3 = \frac{h_{12}}{h_1} \qquad \lambda_4 = \frac{h_{12}}{h_2} \qquad \lambda_5 = \beta(\alpha_1,\alpha_2)^2$$

These are all nonnegative since $h_1,h_2,h_{12} > 0$ and $h_{12} = \sin(2\theta_1+2\theta_2) \le \sin(2\theta_1)+\sin(2\theta_2) = h_1+h_2$.

To show that $L(x,y,z;\,\lambda) \le \beta(\alpha_1,\alpha_2)^2$, we claim that

$$L(x,y,z;\,\lambda) = \beta(\alpha_1,\alpha_2)^2 - \|ax+by+cz\|^2 \qquad (*)$$

for appropriately chosen scalars $a, b, c \in \mathbb{R}$:

$$a = \sqrt{h_1/h_2} \qquad c = \sqrt{h_2/h_1} \qquad b = -\frac{h_{12}}{\sqrt{h_1 h_2}}.$$

It remains to verify $(*)$ by matching coefficients. Expand

$$\|ax + by + cz\|^2 = a^2\|x\|^2 + b^2\|y\|^2 + c^2\|z\|^2 + 2ab\langle x, y\rangle + 2ac\langle x, z\rangle + 2bc\langle y, z\rangle.$$

- Constant term: L has constant term $\lambda_5 = \beta(\alpha_1, \alpha_2)^2$, matching the RHS.
- Coefficient of $\langle x, y\rangle$: L has coefficient $2\lambda_4$, and the RHS has coefficient $-2ab$. With $a = \sqrt{h_1/h_2}$ and $b = -h_{12}/\sqrt{h_1 h_2}$, we get $-2ab = 2h_{12}/h_2 = 2\lambda_4$.
- Coefficient of $\langle y, z\rangle$: L has coefficient $2\lambda_3$, and the RHS has coefficient $-2bc$. With $c = \sqrt{h_2/h_1}$, we get $-2bc = 2h_{12}/h_1 = 2\lambda_3$.
- Coefficient of $\langle x, z\rangle$: L has coefficient -2, and the RHS has coefficient $-2ac$. Since $ac = \sqrt{h_1/h_2}\sqrt{h_2/h_1} = 1$, these match.
- Coefficient of $\|x\|_2^2$: L has coefficient $1 - \lambda_1 - \lambda_4$. Using $\lambda_1 + \lambda_4 = \frac{h_1+h_2}{h_2}$, we have $1 - \lambda_1 - \lambda_4 = 1 - \frac{h_1+h_2}{h_2} = -\frac{h_1}{h_2} = -a^2$.
- Coefficient of $\|y\|^2$: we must show

  $$\lambda_1 - \lambda_2 - \lambda_3 - \lambda_4 + \lambda_4/\alpha_2^2 = -\frac{h_{12}^2}{h_1 h_2}.$$

  Substituting the definitions of λ_i and using $1/\alpha_2^2 = 2(1 - \cos(2\theta_2))$ gives

  $$\begin{aligned}\lambda_1 - \lambda_2 - \lambda_3 - \lambda_4 + \lambda_4/\alpha_2^2 &= \frac{h_1 + h_2 - h_{12}}{h_2} - \frac{h_1 + h_2 - h_{12}}{h_1} - \frac{h_{12}}{h_1} - \frac{h_{12}}{h_2} + \frac{h_{12}}{h_2\alpha_2^2} \\ &= \frac{h_1 + h_2 - 2h_{12} + h_{12}/\alpha_2^2}{h_2} - \frac{h_2}{h_1} \\ &= \frac{h_1 + h_2 - 2h_{12}\cos(2\theta_2)}{h_2} - \frac{h_2}{h_1}.\end{aligned}$$

  Thus it suffices to show

  $$h_1^2 + h_2^2 - 2h_1 h_{12}\cos(2\theta_2) = h_1 h_2 + h_{12}^2.$$

  Equivalently,

  $$h_2^2 = h_1^2 + h_{12}^2 - 2h_1 h_{12}\cos(2\theta_2),$$

  which follows from the identity $h_{12} = h_1\cos(2\theta_2) + h_2\cos(2\theta_1)$:

  $$\begin{aligned}h_2^2 &= h_2^2\cos^2(2\theta_1) + h_2^2\sin^2(2\theta_1) \\ &= (h_{12} - h_1\cos(2\theta_2))^2 + h_2^2\sin^2(2\theta_1) \\ &= h_{12}^2 + h_1^2\cos^2(2\theta_2) - 2h_1 h_{12}\cos(2\theta_2) + h_2^2\sin^2(2\theta_1) \\ &= h_{12}^2 + h_1^2 - 2h_1 h_{12}\cos(2\theta_2),\end{aligned}$$

  as claimed. Therefore the coefficient of $\|y\|^2$ matches $-b^2 = -h_{12}^2/(h_1 h_2)$.
- Coefficient of $\|z\|^2$: we show that

  $$1 + \lambda_2 - \lambda_3 + \lambda_3/\alpha_1^2 - \lambda_5 = -\frac{h_2}{h_1}.$$

  Substituting $\lambda_2 = \frac{h_1+h_2-h_{12}}{h_1}$ and $\lambda_3 = \frac{h_{12}}{h_1}$ and using $1/\alpha_1^2 = 2(1 - \cos(2\theta_1))$ yields

  $$\begin{aligned}1 + \lambda_2 - \lambda_3 + \lambda_3/\alpha_1^2 - \lambda_5 &= 1 + \frac{h_1 + h_2 - h_{12}}{h_1} - \frac{h_{12}}{h_1} + \frac{h_{12}}{h_1\alpha_1^2} - \lambda_5 \\ &= 2 + \frac{h_2 - 2h_{12}\cos(2\theta_1)}{h_1} - \lambda_5.\end{aligned}$$

It therefore suffices to check that

$$\lambda_5 = \beta(\alpha_1, \alpha_2)^2 = 2 + \frac{2h_2 - 2h_{12}\cos(2\theta_1)}{h_1}.$$

Using $h_{12} = \sin(2(\theta_1 + \theta_2))$ and $h_1 = \sin(2\theta_1)$, $h_2 = \sin(2\theta_2)$, we compute

$$\begin{aligned} 2 + \frac{2h_2 - 2h_{12}\cos(2\theta_1)}{h_1} &= 2 + \frac{2\sin(2\theta_2) - 2\sin(2\theta_1 + 2\theta_2)\cos(2\theta_1)}{\sin(2\theta_1)} \\ &= 2 - 2\cos(2\theta_1 + 2\theta_2) \\ &= (2\sin(\theta_1 + \theta_2))^2 \\ &= \beta(\alpha_1, \alpha_2)^2 = \lambda_5, \end{aligned}$$

where the second equality uses $\sin(2\theta_1 + 2\theta_2) = \sin(2\theta_1)\cos(2\theta_2) + \cos(2\theta_1)\sin(2\theta_2)$. Substituting back gives $1 + \lambda_2 - \lambda_3 + \lambda_3/\alpha_1^2 - \lambda_5 = -h_2/h_1 = -c^2$.

This completes the proof that $\mathrm{OPT} \le \beta(\alpha_1, \alpha_2)$.

Lower bound. We exhibit a solution $x, y, z \in \mathbb{R}^d$ achieving value $\beta(\alpha_1, \alpha_2)$. Let $z = (1, 0)$, let $y = (\cos(2\theta_1), \sin(2\theta_1))$, and let $x = (\cos(2(\theta_1+\theta_2)), \sin(2(\theta_1+\theta_2)))$, where $\theta_i = \sin^{-1}(1/(2\alpha_i))$ as above. Then $\|x\|_2 = \|y\|_2 = \|z\|_2 = 1$, so $\|x\|_2 \le \|y\|_2 \le \|z\|_2 = 1$. Moreover,

$$\|y - z\|_2 = 2\sin(\theta_1) = \frac{1}{\alpha_1}, \qquad \|x - y\|_2 = 2\sin(\theta_2) = \frac{1}{\alpha_2},$$

so all constraints are satisfied with equality. Finally,

$$\|x - z\|_2 = 2\sin(\theta_1 + \theta_2) = \beta(\alpha_1, \alpha_2),$$

so $\mathrm{OPT} \ge \beta(\alpha_1, \alpha_2)$.

We conclude that $\mathrm{OPT} = \beta(\alpha_1, \alpha_2)$.

B Additional Experiments

B.1 Experiments with HNSW

While RP-TUNING is motivated by the DiskANN algorithm, we demonstrate its versatility by applying it to the widely-used HNSW algorithm [12] with the FAISS HNSW implementation [4]. Adapting RP-TUNING to HNSW presents two issues: (1) it is not a single graph, but several graphs in a layered structure, (2) it does not use a reachability parameter α. To address (1), we exclusively prune the base layer graph. To address (2), we empirically determine a base α value using the following procedure:

- We sample 5000 random pairs of vertices (s, t) and compute the s-t reachability, i.e., the maximum value of $D(s,t)/D(u,t)$ over all outneighbors u of s.
- We take the 60th, 70th, 80th, and 90th percentiles of these values to determine the four settings used for RP-TUNING.

Across all four datasets (SIFT, DEEP, GIST, MSSPACEV), the pruned HNSW indices match or outperform the baseline in their recall-QPS tradeoffs (see Figure 5). We also show the best QPS with at least 0.98 recall over the four pruned HNSW indices and the original index (see Table 3). Note the total prune time is the sum of time over the four settings using a python implementation. Those demonstrates that RP-TUNING preserves high-quality edges even for graph-based algorithms outside of the DiskANN family.

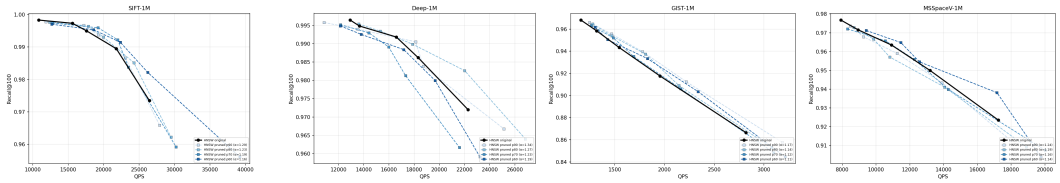

Figure 5: Recall-QPS tradeoffs of pruned HNSW graphs.

Dataset	Orig. QPS	Best Pruned QPS	Improvement	Best Level	Total Prune Time
SIFT-1M	21,786	26,216	+20.3%	p60 ($\alpha = 1.156$)	1,209s
GIST-1M	764	846	+10.8%	p60 ($\alpha = 1.105$)	1,500s
MSSpaceV-1M	5,447	6,596	+21.1%	p80 ($\alpha = 1.190$)	2,132s
Deep-1M	18,324	22,014	+20.1%	p80 ($\alpha = 1.269$)	1,485s

Table 3: HNSW RobustPrune: best QPS at recall ≥ 0.98, QPS improvement over original, and total pruning cost for four variants (p60, p70, p80, p90).

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: The abstract and introduction clearly state the two main contributions: (1) theoretical worst-case reachability guarantees for the α-reachable graphs pruned by RP-TUNING (Section 3), and (2) the empirical evaluation for efficient DiskANN index tuning (Section 4), as well as HNSW graph pruning (Appendix).

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: The paper discusses that the theoretical guarantees apply to the slow-preprocessing variant of DiskANN, and that the worst-case reachability of a pruned graph may be even below 1 in the worst case, as noted in Section 5. The paper also explicitly acknowledges that the application of RP-TUNING to graph-based ANN algorithms other than DiskANN and HNSW is to be explored. Moreover, we do not study any methods that increase graph reachability.

   Guidelines:

   - The answer [N/A] 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: All assumptions (metric space, $\alpha_1 \geq \alpha_2 > 1$, α-reachability of the input graph) are explicitly stated in the theorem statements. Full proofs of Theorem 3.1 and Theorem 3.2 are given in Sections 3.1 and 3.2; the proof of Lemma 3.8 (the SDP bound for Euclidean space) is deferred to Appendix A.

   Guidelines:

   - The answer [N/A] 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: Section 4.1 specifies the hardware (AMD EPYC 9B14, 117GB RAM), all four datasets (SIFT, GIST, Deep, MSSPACEV, which are publicly available from github repositories: `https://github.com/erikbern/ann-benchmarks` and `https://github.com/microsoft/SPTAG/tree/main/datasets/SPACEV1B`), implementation details (Python, Vamana with $\alpha_1 = 1.5, R = 70, L$-Build $= 75$), and evaluation protocol (100-Recall@100 with varying beam size L). The settings of dropout (Section 4.3) and HNSW pruning (Appendix B) experiments are specified as well.

Guidelines:

- The answer [N/A] 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: [No]

   Justification: We do not release code. All four datasets (SIFT, GIST, DEEP, MSSPACEV) are publicly available from github repositories: `https://github.com/erikbern/ann-benchmarks` and `https://github.com/microsoft/SPTAG/tree/main/datasets/SPACEV1B`. For datasets that contain more than one million points, we only load the first one million points. We brute-force the ground truth for $k = 100$ nearest neighbors.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4.1 specifies all relevant details: hardware, datasets with dimensions and query counts, index construction parameters ($\alpha_1 = 1.5, R, L$-Build), tuning targets ($\alpha_2 \in \{1.4, 1.3, 1.2, 1.1\}$), and the query-time evaluation setup (medoid as start vertex, $k = 100$, varying beam size L).

   Guidelines:

   - The answer [N/A] 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: [No]

   Justification: The paper does not report error bars. RP-TUNING is deterministic given the base graph, and each base index is constructed once and fixed for all downstream tuning, so the pruned graphs do not vary across runs. Other sources of variability (random graph initialization in Vamana construction for the base and rebuilt indices, random edge deletion in the dropout baseline, random pair sampling for the HNSW α estimate, and timing noise in QPS and runtime measurements) are not quantified.

   Guidelines:

   - The answer [N/A] 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: Section 4.1 specifies the compute environment: AMD EPYC 9B14 CPU (15 cores, 30 threads) with 117GB RAM on Debian GNU/Linux 12. Table 2 reports absolute runtimes in seconds for both rebuild and RP-TUNING across all datasets and configurations.

   Guidelines:

   - The answer [N/A] 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: This paper presents algorithmic research on approximate nearest neighbor index tuning using public benchmark datasets. It does not involve human subjects, sensitive data, or applications with foreseeable ethical risks.

   Guidelines:

   - The answer [N/A] 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: [N/A]

    Justification: This paper presents fundamental algorithmic research on approximate nearest neighbor index tuning. The work improves computational efficiency of ANN indices and has no foreseeable direct negative societal impact; it enables more efficient vector search systems but does not introduce new capabilities that could be misused.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release models, pre-trained weights, or scraped datasets. The experimental datasets are standard public benchmarks with no high risk for misuse.

    Guidelines:

    - The answer [N/A] 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: All datasets and code used are cited with their original papers and are used solely for benchmarking, consistent with their terms. SIFT-1M and GIST-1M [11] are from the INRIA TEXMEX corpus (`http://corpus-texmex.irisa.fr/`); the BigANN benchmark lists the SIFT data under CC0. Deep-1M is a subset of Yandex Deep1B [3] (CC BY 4.0). MSSPACEV-1M is a subset of Microsoft SPACEV1B [13] (O-UDA license, `https://github.com/microsoft/SPTAG/tree/main/datasets/SPACEV1B`). The HNSW experiments use FAISS [4] (MIT license). DiskANN (Vamana) and RP-TUNING are our own Python implementations.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The paper does not introduce new datasets, models, or software assets.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The paper does not involve human subjects research and therefore requires no IRB approval.

    Guidelines:

    - The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not used as any important, original, or non-standard component of the core methodology in this research.

    Guidelines:

    - The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.