α-Reachable Graphs for Multi-vector Nearest Neighbor Search

Siddharth Gollapudi [1] **Ravishankar Krishnaswamy** [2] **Ben Landrum** [3] **Nikhil Rao** [4] **Kirankumar Shiragur** [2]
Sandeep Silwal [5] **Harsh Wardhan** [2]

Abstract

In recent years, multi-vector retrieval has emerged as the state-of-the-art in dense retrieval applications by representing queries and documents as sets and employing set-to-set similarity measures. Popularized by the seminal ColBERT work, this paradigm of search offers expressive representations and superior accuracy, albeit at the cost of high storage and computation costs.

To accelerate the adoption of the multi-vector approach in large scale retrieval applications, efficient and easy-to-use algorithms for multi-vector nearest neighbor search are needed. Our work aims to address this as follows:

- We develop a robust theoretical model studying the effects of non-metric similarity functions on the performance of graph-based nearest neighbor data structures. This is particularly relevant for the popular Chamfer distance, on which ColBERT is based.
- Practically, we demonstrate that graph-based data structures can seamlessly support these non-metrics, using the Chamfer similarity as an example. Our algorithm marginally outperforms prior SOTA in the 1Recall@100 setting, while achieving at least 61% more recall for the more relevant 100@100 recall setting.

1. Introduction

The nearest neighbor search problem is fundamental to information retrieval. Given a dataset P and a query q, the goal is to find the point in P closest to q using a similarity function D. However, the "curse of dimensionality" often necessitates relaxations such as c-approximate nearest neighbor search or Recall@k. Typically, P consists of high-dimensional vectors from Machine Learning (ML) models (e.g., text embeddings), and D is a single-vector calculation like Euclidean distance.

While foundational, this single-vector paradigm often fails to capture the nuanced semantics of larger and more complex data objects (Gao et al., 2021; Hofstätter et al., 2022; Lee et al., 2024; Qian et al., 2022; Wang et al., 2021; Yao et al., 2021). Consequently, multi-vector approaches, where each data point $p_i \in P$ is a *collection* of embeddings, have gained traction (Khattab & Zaharia, 2020). Similarity is then measured between these collections, a prominent example being the Chamfer similarity, which aggregates the minimum distances between query embeddings and document embeddings.

This fine-grained multi-vector approach, however, introduces challenges:

- **Computational Cost.** Multi-vector similarities are costlier to compute and require more space to store the representations.
- **Theoretical Guarantees.** Similarities such as Chamfer are not necessarily metrics, possibly invalidating known guarantees of single-vector algorithms.
- **Downstream Tasks.** Usecases for the embeddings, such as retrieval, are tailored to single-vector metrics, such as ℓ_2. Additional work may be needed to adapt them for multi-vector computations.

Thus, recent solutions for nearest neighbor search with multi-vector similarities (all specializing for Chamfer) have been proposed, including:

- **Re-ranking baselines.** These approaches, namely DESSERT (Engels et al., 2024) and PLAID (Santhanam et al., 2022), use an inexpensive, noisy computation to heavily prune the dataset P. The remaining candidates are then re-ranked using the more accurate and costlier multi-vector similarity (such as Chamfer).

[1]UC Berkeley [2]Microsoft Research India [3]Cornell University [4]Microsoft [5]UW-Madison. Correspondence to: Siddharth Gollapudi <sgollapu@berkeley.edu>, Sandeep Silwal <silwal@cs.wisc.edu>.

Proceedings of the 1st Workshop on Vector Databases at International Conference on Machine Learning, 2025.

However, this initial pruning can discard relevant results, compromising accuracy and potentially leading to an $\Omega(n)$ worst-case runtime. Among such baselines, PLAID is often preferred among these due to its empirically higher recall (Dhulipala et al., 2024).

- **MUVERA (Dhulipala et al., 2024).** As the current state-of-the-art, MUVERA offers significant speed improvements (up to 5.7X) over PLAID. It approximates Chamfer similarity with an inner product after applying a suitable transformation, effectively reducing the problem to single-vector search. However, there is additional overhead for computing the transformation. Furthermore, its theoretical accuracy and convergence guarantees are unclear: the transformations for queries and documents differ, potentially leading to an $\Omega(n)$ runtime in worst-case scenarios.

In light of the limitations of prior methods, we are motivated to ask:

Can we obtain a practical nearest neighbor search algorithm capable of handling general multi-vector similarities with provable guarantees?

Our Results: We affirmatively answer this question for a broad class of multi-vector similarity functions, both theoretically and practically. Our approach adapts the DiskANN algorithm (Jayaram Subramanya et al., 2019), a popular single-vector index, by treating it as a *black box*: we substitute its single-vector similarity computations with multi-vector ones. Our key contributions then are:

1. **Theoretical Advances.** We formally introduce *quasimetrics* (Definition 2.2), a concept capturing a wide array of non-metric similarities, including Chamfer. We prove our algorithm converges to a $(1+\epsilon)$-approximate nearest neighbor in sublinear (in n) steps for these quasimetrics (Theorem 3.6). This extends prior DiskANN analysis by (Indyk & Xu, 2023), which was limited to single-vector metrics.

2. **Practical Contributions.** We demonstrate improved empirical results over MUVERA, the previous state-of-the-art. On top of marginal improvements in queries per second (QPS) for 1Recall@100 (defined in Section 4.2), our method achieves significant accuracy gains (at least 61%) on the larger BEIR (Thakur et al., 2021) datasets for the more challenging 100Recall@100 metric. We believe that this metric is particularly informative for large-scale nearest neighbor search systems (Jayaram Subramanya et al., 2019; Fu et al., 2019).

Our approach also offers distinct advantages:

- **Simplicity.** It avoids additional overhead, such as training or computing new embeddings beyond the initial document collections. Existing implementations of DiskANN can be used directly.

- **Generalizability.** Our quasimetric framework (Definition 2.2) extends to a broad class of similarities. This includes Chamfer (where the summed underlying similarity can be any metric, like any ℓ_p norm) and asymmetric similarities that satisfy the triangle inequality, such as hinge-distance (Lai & Hockenmaier, 2017; Lou et al., 2020; Chheda et al., 2021; Roy et al., 2023).

2. Preliminaries

Exact nearest neighbor search in high-dimensional single-vector spaces is challenging due to the "curse of dimensionality (Rubinstein, 2018)." Consequently, researchers typically consider two types of relaxations. Theoretical work usually focuses on the c-approximate nearest neighbor problem, which aims to find a point $p' \in P$ such that its distance to the query q is at most c times the distance to the true nearest neighbor (Indyk & Motwani, 1998). In practice, performance is commonly evaluated using a recall measure, which represents the fraction of the true top-k nearest neighbors retrieved by the search algorithm (Järvelin & Kekäläinen, 2002) (details in Section 4.2).

This work focuses on generalizing nearest neighbor search to the multi-vector setting. In this paradigm, the ambient space $\mathbb{R}^d$ is equipped with a metric $m : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, and each document $p \in P$ is represented with a *set* of l_p vectors, i.e. $p \in \mathbb{R}^{l_p \times d}$. The main multi-vector similarity we consider is the Chamfer similarity, a common similarity measure between two point sets in machine learning, especially in the context of text data (Barrow et al., 1977; Atasu & Mittelholzer, 2019; Bakshi et al., 2024; Kusner et al., 2015; Wan et al., 2019). It is defined as follows:

Definition 2.1 (Chamfer distance)**.** For a query $q \in \mathbb{R}^{l_q \times d}$ the Chamfer distance is defined as

$$C(q,p) = \sum_{i}^{l_q} \arg\min_{j \in [l_p]} (m(q_i, p_j))$$

where q_i or p_j denotes the ith or jth vector in q or p, respectively.

It is important to note that 'Chamfer distance' is a bit of a misnomer: this measure is not a true metric because it is asymmetric and does not satisfy the triangle inequality. To accommodate this, we define a broader class of similarity measures:

Definition 2.2 (k-quasimetric)**.** A function D is a k-quasimetric if:

1. $D(p, p) = 0$,
2. For $p \neq q$, $D(p, q) \geq 0$,
3. There exists a $k \geq 1$ such that for any $p_1, p_2, p \in P$, the following inequality holds: $D(p_1, p_2) \leq D(p_1, p) + k \cdot D(p, p_2)$. Note that k is dataset dependent (as shown below).

Indeed, the Chamfer distance is easily shown to be a k-quasimetric.

Lemma 2.3. *Consider a multivector dataset P, where for each $p \in P$, $|p| = l_p$. If m is a metric then the Chamfer distance is a k-quasimetric for $k = \arg\max_{p \in P} l_p$.*

Proof. We just need to verify the third condition. For a point in $a \in p_1$, let $b \in p_2$ and $c \in p$ be its nearest neighbor in p_2 and p respectively. Then $m(a, b) \leq m(a, b')$ for all $b' \in p_2$. In particular, if we choose b' to be the nearest neighbor of c in p_2, we have

$$m(a, b) \leq m(a, b') \leq m(a, c) + m(c, b').$$

Note that the sum of $m(a, c)$ over all a is simply $D(p_1, p)$. Likewise, the sum of $m(c, b')$ over all c is $D(p, p_2)$. Since any $c \in p$ can be mapped to by at most $|p_1|$ many a's $\in p_1$, the claim follows, as only the second sum $m(c, b')$ is counted with multiplicity. □

By generalizing Chamfer to this definition, we are able to extend the theoretical guarantees of single-vector search data structures (reviewed below) to the multi-vector setting at large. We discuss this further in Section 3.

We note that special care must be taken in designating the parameters of D. Since D is not symmetric, $D(q, p)$ is *not* the same as $D(p, q)$. Throughout this paper, we consistently use the first parameter to denote the "query" and the second to denote a "document" from the dataset. This convention aligns with prior work on Chamfer similarity in text retrieval (Dhulipala et al., 2024).

DiskANN Review. Finally, we give a quick overview of DiskANN (Jayaram Subramanya et al., 2019), the main single-vector nearest neighbor search algorithm which we build off of. At a high-level, DiskANN builds a directed graph (called Vamana) on a set of n vectors, satisfying the so called α-reachable property. It guarantees that for every pair x, y of vectors, either the graph has an edge, or there is a vector z such that x can navigate to y via z and the distances are not distorted by more than an α factor. Later Indyk & Xu (2023) showed that an α-reachable graph can be provably used to solve the approximate nearest neighbor problem via a greedy search, with the runtime bounded by the maximum degree of the graph. Furthermore, they were able to bound degree of the graph via the doubling dimension of the underlying dataset, a data-dependent measure of intrinsic dimensionality (Indyk & Naor, 2007; Aumüller & Ceccarello, 2019; Narayanan et al., 2021; Indyk & Xu, 2023). For a set of points P, is defined as the λ_P such that 2^{λ_P} is sufficient to cover any ball in the dataset with balls of half the radius. Full descriptions of the DiskANN algorithm can be found in Appendix A. Note that the results in (Indyk & Xu, 2023) are proven for symmetric distance functions, leaving it unclear whether similar results hold for asymmetric distance functions or more generally for k-quasimetric functions like Chamfer.

Lastly, Δ denotes the *aspect ratio* of the dataset P, defined as $(\max_{p,p' \in P} D(p, p'))/(\min_{p,p' \in P} D(p, p')$.

3. Main Theoretical Results

We now present the main theoretical results of our work, focusing on k-quasimetric functions. Our approach is based on analyzing the graph-based data structure of DiskANN, and we believe a key contribution of our work is proving that DiskANN can be adapted for k-quasimetric functions. Our theoretical findings (with proofs presented later in Section 5) directly support our superior empirical performance compared to all prior approaches for the Chamfer, which is given in Section 4. To this end, we first recall the notion of doubling dimension, with the definition specialized to k-quasimetrics.

Definition 3.1 (Doubling dimension)**.** Consider a multivector dataset P and a k-quasimetric D. The doubling constant is the smallest C such that for all $R > 0$ and any $v \in P$, the ball of radius R centered at v, $B(v, R)$, can be covered by C balls of radius $R/2$ (centered at other points in the dataset inside $B(v, R)$). The doubling dimension is $\log C$.

Intuitively, a dataset in $\mathbb{R}^d$ may not require all d dimensions to represent its full information, and the doubling dimension is a tool used to identify this "intrinsic" dimension of the dataset, similar to the single vector case. Although the definition of doubling dimension is based on balls of radius $R/2$, we can iterate to bound the number of balls of radius R/k needed to cover all points within $B(v, R)$.

Lemma 3.2 (Doubling Dimension Property)**.** *Consider a multivector dataset P whose doubling dimension is x with respect to a k-quasimetric D. For any $p \in P$ and $R > 0$, the ball $B(p, R)$ can be covered with $m \leq (2k)^x$ balls centered around points in $B(p, R)$ of radius at most r/k, where $k > 1$.*

We are now prepared to define α-reachable graphs for k-quasimetric functions. Our definition is a natural extension of the one in (Indyk & Xu, 2023), carefully adapted to account for the asymmetries specific to k-quasimetric functions.

Definition 3.3 (k-quasi α-reachability)**.** Consider a graph $G = (V, E)$ paired with a multivector dataset P, where each $p \in P$ is associated with some node $v \in V$. G is α-reachable for a k-quasimetric D if for any nodes u, v in G, there exists a node t such that the (directed) edge (u, t) exists and $D(t, v) \leq \frac{1}{\alpha} D(u, v)$ for the corresponding points in P.

As discussed earlier, the concept of reachable graphs originates from the DiskANN and its analysis in (Indyk & Xu, 2023), where the authors demonstrate that the local greedy search algorithm on these graphs converges to a good approximate nearest neighbor solution. In our first main result, we present an analogous finding for the k-quasimetric case. The approximation factor for k-quasimetric is more complex than for symmetric distance functions, as it depends on both the quasimetric parameter k and the asymmetry of the distance function. The precise approximation factor and the number of steps required for the local greedy search to reach this approximation are outlined in the following lemma.

Lemma 3.4 (Approximation of greedy search)**.** *Consider a α-reachable graph G with respect to a k-quasimetric function D. For any starting point s, the greedy search procedure takes $O(\text{polylog}(\frac{k\Delta}{\epsilon\nu})\log(n))$ steps to reach a $(\frac{\beta k}{\alpha - k^2} + \frac{\alpha}{\alpha - k^2} + \epsilon)$-approximate nearest neighbor solution with respect to D, where $\beta = \frac{D(a,q)}{D(q,a)}$ and ν is a parameterized by β, k and α. Furthermore, if $\alpha > \Omega((k^2 + \beta k)/\epsilon)$, the greedy search procedure outputs a $(1 + \epsilon)$-approximate nearest neighbor solution.*

Note that for symmetric distance functions, the values of k and β are both equal to 1, which allows us to recover the results from (Indyk & Xu, 2023) for symmetric distance functions. Since the results in (Indyk & Xu, 2023) are tight, we also emphasize that our findings are tight in the worst case.

While the above lemma provides a bound on the number of steps, the total number of distance comparisons (and thus the runtime) performed during the search is upper bounded by the number of steps multiplied by the maximum degree of the graph. Therefore, bounding the degree is crucial, and we will address this in the following lemma.

Lemma 3.5 (Degree Bound for k-Quasi α-Reachable Graphs)**.** *For any multivector dataset P and a k-quasimetric D, there exists a α-reachable graph with respect to D whose maximum degree is upper bounded by $O((4\alpha)^x \log(\Delta))$, where x is the doubling dimension of the dataset P with respect to D. Furthermore, there exists an efficient algorithm to compute an α-reachable graph with respect to D with a degree bound of $O((4\alpha)^x \log(\Delta) \log n)$.*

The above result demonstrates the existence of reachable graphs with favorable degree properties and provides an efficient algorithm for constructing these graphs with only a slight degree overhead. By combining these two results, we arrive at our final main theorem, which we summarize below.

Theorem 3.6 (Final Complexity)**.** *Consider a multivector dataset $P \subset \mathbb{R}^d$ whose doubling dimension is x with respect to a k-quasimetric D. There exists a graph based data structure which can be constructed efficiently in time polynomial in $|P|$ and d. For any query q and any $\epsilon > 0$, the data structure outputs a $(\frac{\beta k}{\alpha - k^2} + \frac{\alpha}{\alpha - k^2} + \epsilon)$-approximate nearest neighbor solution with respect to D, where $\beta = \frac{D(a,q)}{D(q,a)}$. The total search time of our algorithm is upper bounded by $\tilde{O}(d(4\alpha)^x \log \Delta)$.[1] Furthermore, by setting $\alpha > \Omega((k^2 + \beta k)/\epsilon)$, our search algorithm outputs a $(1 + \epsilon)$-approximate nearest neighbor solution.*

In practice, we can expect both $\beta = O(1)$ and $k = O(1)$ (see Table 1 and Table 2), as these terms are determined by the gap in size between query and document sets. Thus, this theorem intuitively shows that by setting α to be sufficiently large ($\alpha > \frac{C}{\epsilon}$ for some C), this algorithm can achieve a $(1 + \epsilon)$-approximate nearest neighbor.

Again, we note that analogous results for symmetric distance functions were established by (Indyk & Xu, 2023). Our work is inspired by theirs, and while many of our proofs share a similar spirit, they differ at critical points to address the weaker triangle inequality and the asymmetry inherent in k-quasimetric functions. Furthermore, our results demonstrate the power of DiskANN to handle multivector similarities in a black-box manner and justifies its use in our experiments.

4. Evaluation

In addition to our theoretical framework for graph search on k-quasimetrics, we evaluate DiskANN's empirical performance as a black-box for Chamfer distance, which we refer to as Chamfer-DiskANN.

4.1. Setup

Hardware. All search experiments are performed on an Azure E series VM with 96 vCPUs and 384 GB of RAM.

Datasets. We evaluate all three approaches on the MS-MARCO development split and Quora, NQ, and HotpotQA test split datasets from the BEIR (Thakur et al., 2021) benchmark. We pick these datasets because their sizes represent reasonable workloads for scalable nearest neighbor search. We refer the reader to Appendix B for more details regarding these datasets.

[1] The $\tilde{O}$ notation hides the poly log factors in $\log(\frac{k\Delta}{\epsilon\nu})$ and $\log(n)$.

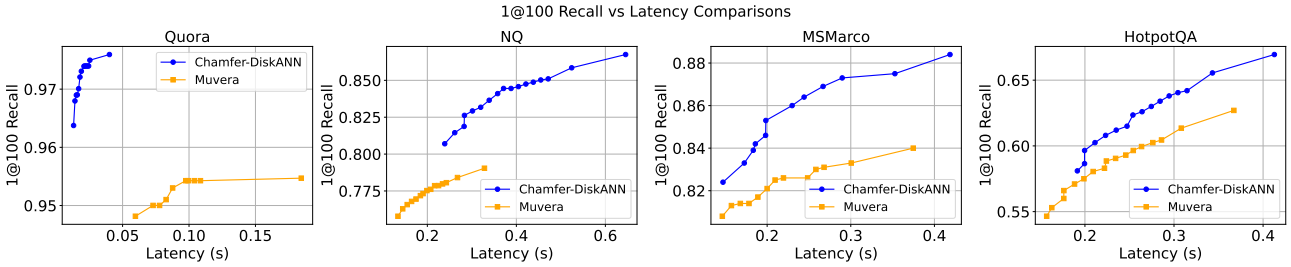

Figure 1. The above plots illustrate the QPS versus 1Recall@100 difference between Chamfer-DiskANN and MUVERA. Chamfer-DiskANN clearly outperforms MUVERA across all datasets.

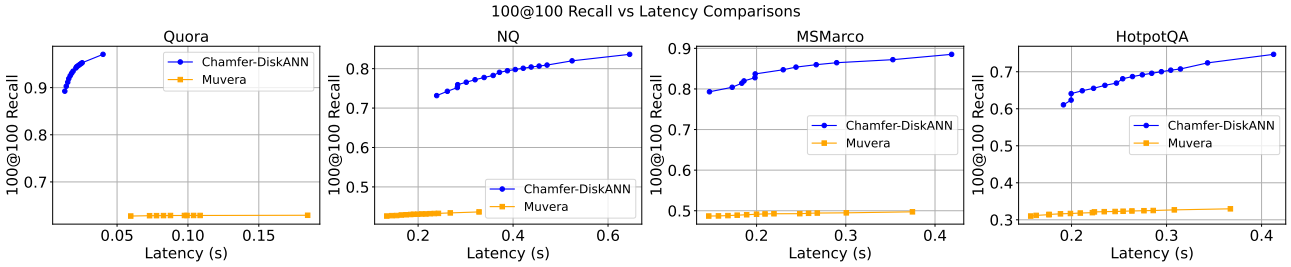

Figure 2. The above plots illustrate the QPS versus 100Recall@100 difference between Chamfer-DiskANN and MUVERA. Chamfer-DiskANN clearly outperforms MUVERA across all datasets.

Embeddings. We use ColBERTv2.0 (Santhanam et al., 2021) to generate embeddings from the MSMARCO and Quora passages. Across all datasets, every embedding has a dimension of 128 and each query passage is fixed to 32 embeddings. For Quora, this implies that a given distance computation between a query and a document uses around 80000 floating point operations. Additionally, for the MUVERA embeddings, we use a custom implementation due to the lack of publicly available software. We use the default recommended parameters from the original publication (Dhulipala et al., 2024), resulting in 10248-dimensional embeddings representing the original multi-vector documents.

Indices. For MUVERA, we construct a Maximum Inner Product Search (MIPS) DiskANN index using the C++ implementation from the work of Dobson et al.. An exposition detailing this variant of DiskANN can be found in Appendix A. We use the recommended parameters (Dhulipala et al., 2024) of $R = 200$ (degree bound) and $L = 600$ (beam width). For Chamfer-DiskANN, we change the distance function of the same DiskANN implementation from MIPS to the Chamfer distance, leaving everything else unchanged. As shown in Table 4, the MUVERA indices are uniformly larger than the Chamfer-DiskANN indices, taking up more space and having higher mean degree.

4.2. Main Empirical Evaluation

In this section, we describe two metrics that form the basis of our empirical evaluation. The first metric, 1Recall@100 is a common metric in retrieval and recommendation settings, while 100Recall@100 is a metric we introduce in order to better capture the necessities of large-scale retrieval settings. Our results demonstrate that Chamfer-DiskANN surpasses MUVERA, the current state-of-the-art, on both metrics, even though MUVERA utilizes larger indices. To generate recall/latency curves for all searches, we vary the beamwidth from $L = 100$ to $L = 400$.

1Recall@100 Evaluation. We first evaluate on the *1Recall@100* benchmark commonly found in other works studying multivector retrieval (Dhulipala et al., 2024; Santhanam et al., 2022). In this setting, all approaches are evaluated against a real, human-labeled groundtruth: each query is manually assigned with a small (e.g. 1-4) number of highly relevant documents. For a given query, the goal is to retrieve 100 documents with at least one of the 100 documents being in this ground-truth. This benchmark is best suited for expensive re-ranking routines tailor-made for certain datasets (Ji et al., 2024), because it rewards finding a handful of results (the human-labeled results) from a list of candidates. While this is useful for smaller-scale retrieval use cases, larger-scale retrieval settings require strong accuracy on far more than the top few results: for example, a basic image search on Google may require *hundreds* of valid images for the search.

100Recall@100 Evaluation. To address the downsides of the 1Recall@100 metric for larger-scale ANNS indices, we

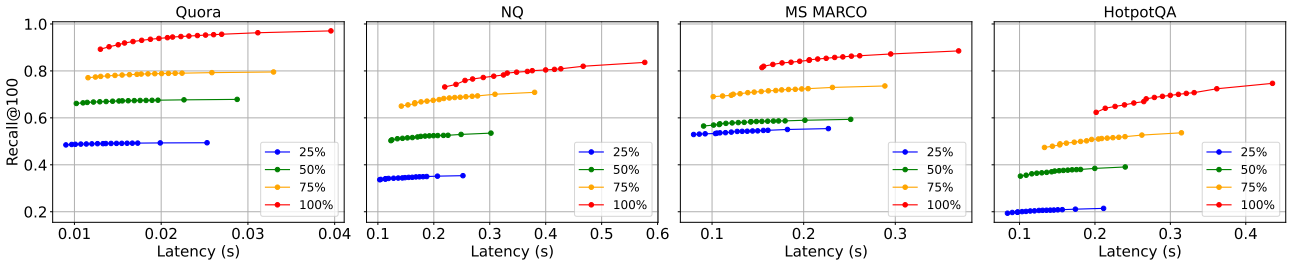

Figure 3. The above plots illustrate the QPS versus 100Recall@100 difference when sampling 25/50/75/100% of the vectors for the queries.

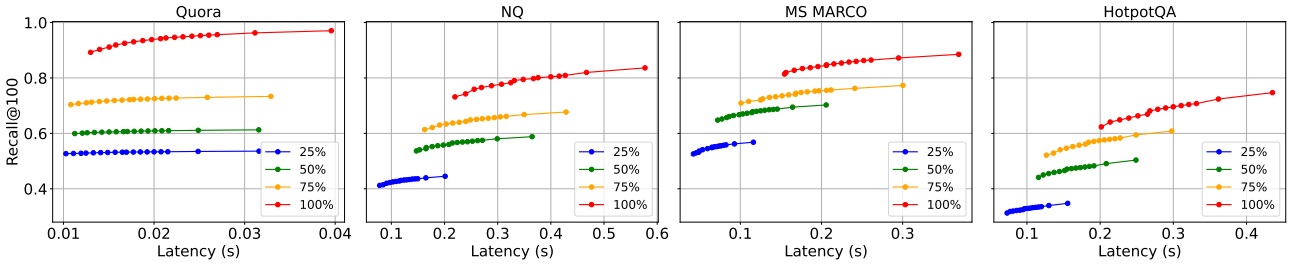

Figure 4. The above plots illustrate the QPS versus 100Recall@100 difference when sampling 25/50/75/100% of the vectors for the documents.

propose a different method of evaluation for this multifilter retrieval: the *100Recall@100* metric. For this benchmark, we first construct a groundtruth of 100 results by computing Chamfer in a bruteforce fashion on a given dataset. Then, we simply evaluate our algorithm on said dataset against the groundtruth. In comparison to the re-ranking friendly 1Recall@100 metric, 100Recall@100 is a better standard for large-scale ANNS indices: it motivates algorithms to quickly and recover a large number of relevant points that are close to the query, rather than optimize for a select few highly-relevant points in the 1Recall@100 setting. We believe that evaluation on this metric is an exciting direction for future work for multivector retrieval in bringing it to large-scale deployments.

Results. Despite the aforementioned issues with the 1Recall@100 benchmark, Figure 1 demonstrates Chamfer-DiskANN's strong performance on this metric: across all datasets, Chamfer-DiskANN performs marginally better ($<1\%$) at worst, suggesting that even Chamfer-DiskANN's worst-case outperforms MUVERA for 1Recall@100. For the 100Recall@100 benchmark, Figure 2 Chamfer-DiskANN demonstrates considerable superiority over MUVERA: Chamfer-DiskANN is at worst **61%** more accurate than MUVERA for all QPS across all datasets. This benchmark highlights one key shortcoming of MUVERA, in that approximating Chamfer with a standard inner product introduces *additional* noise on top of approximating the nearest neighbor search. This approximation introduces an artificial ceiling on the accuracy MUVERA-based solutions can provide. Since the true nearest neighbors are computed with a brute force search (as opposed to human labeling in the case of the 1Recall@100 benchmark), it is possible for Chamfer-DiskANN to achieve perfect recall, while MUVERA cannot due to the inherent approximation.

4.3. Improving Chamfer-DiskANN's Latency: Sampling

The high latencies shown in Figure 1 and Figure 2 illustrate the cost of exactly computing Chamfer. A natural fix to this problem would then be to reduce the input sizes. Indeed, it is known that by sampling a small number of vectors from the input query, one can provably achieve a near linear-time approximation of Chamfer (Bakshi et al., 2024). Inspired by this result, we evaluate the latency/recall tradeoff when sampling both query and document vectors[2]. Figure 3 and Figure 4 demonstrates the negative impact of uniformly sampling a set percentage of the query and document vectors, respectively: in both settings, sampling never provides a superior latency/recall curve.

One possible explanation for this is that the results from Bakshi et al. require the number of vectors per document to be very large: in the datasets evaluated, the number of vectors per document is relatively low (which in turn minimizes the impact of β and k). As such, sampling may become useful for inputs with large numbers of vectors per document,

[2]Note that sampling document vectors does *not* come with the same theoretical guarantees as sampling query vectors.

which would ensure that β and k don't heavily impact the approximation factor of the search.

5. Theoretical Analysis

Here we provide proofs for our main results (Lemma 3.4 and Lemma 3.5).

5.1. Proof of Lemma 3.4

Here, we present the proof for Lemma 3.4, which bounds the steps needed for local greedy search to converge to a good approximate solution. First, we state a lemma that provides the approximation guarantee of a locally optimal solution produced by the greedy search. We analyze the locally optimal solution because it is straightforward and serves as a simple sanity check for the expected approximation factor.

Lemma 5.1 (local optimum lemma)**.** *Consider a query q, k-quasimetric function D and a k-quasi α-reachable graph G. For any $\alpha > k^2$, starting from any node s, any local optimum solution v_i for the greedy search algorithm satisfies the following inequality:*

$$d(q, v_i) \leq \left(\frac{\alpha}{\alpha - k^2}\right) d(q, a) + \left(\frac{k}{\alpha - k^2}\right) d(a, q) \ .$$

where a denotes the true nearest neighbor to q, that is, $a = \text{argmin}_{p \in P} D(q, p)$.

Proof. Since D is a k-quasimetric, we have that $D(a, v_i) \leq D(a, q) + kD(q, v_i)$ by the weak triangle inequality axiom. Also, since G is α-reachable by definition, either the edge (v_i, a) exists or there exists some v' such that (v_i, v') exists and $D(a, v') \leq \frac{D(a,v_i)}{\alpha} \leq \frac{D(a,q)+kD(q,v_i)}{\alpha}$.

Since v_i is locally optimal, by definition we have $D(q, v_i) \leq D(q, v')$. Therefore, we have

$$\begin{aligned} D(q, v_i) \leq D(q, v') &\leq D(q, a) + kD(a, v') \\ &\leq D(q, a) + k\left(\frac{D(a, q) + kD(q, v_i)}{\alpha}\right) \end{aligned}$$

Rearranging terms gives the desired result. □

The result above demonstrates that all local optima of the greedy local search algorithm are approximate solutions to the nearest neighbor problem. However, it does not establish a bound on the number of steps required to converge to a good solution. In Lemma 3.4, we extend these proofs to provide a bound on the number of steps needed to reach an arbitrarily good approximate solution as a local optimum. To do this, we first introduce an intermediate lemma, which will be instrumental in proving Lemma 3.4. Below, we state and prove this lemma, and later apply it to prove Lemma 3.4.

Lemma 5.2 (Convergence lemma)**.** *Consider a query q, k-quasimetric function D and a k-quasi α-reachable graph G. Starting from any node v_0, let v_t be the node reached by the greedy search algorithm after t steps. Then for $\alpha > k^2$, v_t satisfies:*

$$d(q, v_t) \leq (\tfrac{k^2}{\alpha})^t D(q, s) + \tfrac{\alpha}{\alpha - k^2} D(q, a) + \tfrac{k}{\alpha - k^2} D(a, q) \tag{1}$$

where a denotes the true nearest neighbor to q, that is, $a = \text{argmin}_{p \in P} D(q, p)$.

Proof. Recall Lemma 5.1's proof, which gives $D(a, v') \leq \frac{D(a,v_t)}{\alpha} \leq \frac{D(a,q)+kD(q,v_t)}{\alpha}$.

We proceed with induction; the base case trivially holds. We now analyze the induction step. Suppose Equation (1) holds for v_t. We wish to show that the inductive claim also holds for v_{t+1}. Since greedy search always picks the next closest node v_{t+1} to the query q, by definition we have that $D(q, v_{t+1}) \leq D(q, v')$ which further implies

$$\begin{aligned} D(q, v_{t+1}) &\leq D(q, v') \leq D(q, a) + kD(a, v') \\ &\leq D(q, a) + k\tfrac{D(a,q)+kD(q,v_t)}{\alpha} \end{aligned}$$

We now simplify the RHS:

$$\begin{aligned} & D(q, a) + k \cdot \tfrac{D(a,q)+kD(q,v_t)}{\alpha} \\ &= \frac{k^2 D(q, v_t)}{\alpha} + \frac{kD(a, q)}{\alpha} + D(q, a) \\ &\leq \frac{k^2}{\alpha} \cdot \frac{k^{2t} D(q, s)}{\alpha^t} + \frac{k^2}{\alpha} \cdot \frac{2\alpha D(q, a)}{\alpha - k^2} \\ &\quad + \frac{kD(a, q)}{\alpha} + D(q, a) \end{aligned}$$

Note that the above inequality comes from applying the inductive hypothesis for t. Rearranging terms then gives us the required bound

$$\frac{k^{2(t+1)} D(q, s)}{\alpha^{t+1}} + \left(\frac{k}{\alpha - k^2}\right) D(a, q) + \left(\frac{\alpha}{\alpha - k^2}\right) D(q, a)$$

concluding the proof. □

The above convergence bound and a simple case analysis forms the proof for Lemma 3.4, which we state below. Due to space constraints we defer its proof to the Appendix D.

Lemma 3.4 (Approximation of greedy search)**.** *Consider a α-reachable graph G with respect to a k-quasimetric function D. For any starting point s, the greedy search procedure takes $O(\text{polylog}(\frac{k\Delta}{\epsilon\nu}) \log(n))$ steps to reach a $(\frac{\beta k}{\alpha - k^2} + \frac{\alpha}{\alpha - k^2} + \epsilon)$-approximate nearest neighbor solution with respect to D, where $\beta = \frac{D(a,q)}{D(q,a)}$ and ν is a parameterized by β, k and α. Furthermore, if $\alpha > \Omega((k^2 + \beta k)/\epsilon)$, the greedy search procedure outputs a $(1 + \epsilon)$-approximate nearest neighbor solution.*

Lemma 5.2 indicates that α-reachability is beneficial for ensuring the convergence of greedy search to a good approximate solution. However, it does not provide guarantees on the complexity of graph-based methods. To bound the complexity, we must establish limits on the degree of the constructed reachable graphs. The total time complexity of greedy search, i.e., the number of distance comparisons made by the algorithm, is proportional to both the graph's degree and the number of steps. Below, we derive a bound on the degree of the reachable graphs.

5.2. Proof of Lemma 3.5

We now present the proof of Lemma 3.5, which establishes a bound on the degree of α-reachable graphs for k-quasimetric functions. Additionally, this result provides an efficient algorithm for constructing these reachable graphs, introducing only a logarithmic overhead to the degree of the graphs. The result and its proof are summarized below.

Lemma 3.5 (Degree Bound for k-Quasi α-Reachable Graphs)**.** *For any multivector dataset P and a k-quasimetric D, there exists a α-reachable graph with respect to D whose maximum degree is upper bounded by $O((4\alpha)^x \log(\Delta))$, where x is the doubling dimension of the dataset P with respect to D. Furthermore, there exists an efficient algorithm to compute an α-reachable graph with respect to D with a degree bound of $O((4\alpha)^x \log(\Delta) \log n)$.*

Proof. We use the notion of a ring from the proof of Lemma 3.3 in (Indyk & Xu, 2023). For $r_1 < r_2$, let $R(p, r_1, r_2)$ be the set of all points in $B(p, r_2)$ but not in $B(p, r_1)$. For each $i \in [\log_2 \Delta]$, we cover $R(p, \frac{D_{max}}{2^i}, \frac{D_{max}}{2^{i-1}})$ with balls of radius $\frac{D_{max}}{\alpha 2^i}$ and by Lemma 3.2 the number of such balls needed is atmost $(4\alpha)^x$. We now provide the construction of our α-reachable graph: for every p, we add out-neighbors of p to be the centers of the balls covering all the points in the rings around p that we defined earlier. In the remainder of the proof, we show that our construction gives an α-reachable graph and then provide a degree bound for our construction. Finally, we end the proof, with its efficient construction.

We start with the proof of α-reachability. Consider point p and a p' present in the i^{th} ring. From our construction, either there exists a direct edge from p to p' or it must be the case that there exists a ball that covers this point p' and we added an edge to the center of this ball, lets call this point q. As $D(p, p') \geq \frac{D_{max}}{2^i}$ and $D(q, p') \leq \frac{D_{max}}{\alpha 2^i}$, we immediately get that $D(q, p') \leq \frac{1}{\alpha} D(p, p')$ and the α-reachability constraint holds. As the above analysis holds for all p and p', we conclude that the graph constructed is a α-reachable graph.

We now bound the out-degree of every node in the constructed graph. Using Lemma 3.2, which bounds the number of required balls to cover each ring by $m \leq (4\alpha)^x$. As there are at most $\log \Delta$ many rings, we immediately get that the out-degree of every node in our constructed graph is upper bounded by $O((4\alpha)^x \log(\Delta))$.

In the remainder of the proof, we provide an efficient construction of the graph. Now consider an algorithm which iterates over all points $p \in P$. For each $p \in P$, creates rings as earlier and let $R_1, R_2 \ldots, R_{\log_2 \Delta}$ be these rings with respect to point p. For each element in $q \in P$, if $q \in R_i$, we create a set $S_q = B(q, \frac{D_{max}}{\alpha 2^i}) \cap R_i$. These sets S_q give us the required balls and our goal is to use these balls to cover all the points in P. The minimum number of these S_q balls needed to cover all the points in P is upper bounded by $O((4\alpha)^x \log(\Delta))$. However, we don't know the efficient way of picking the minimum number of balls. However, the task at a hand is an instance of the set cover problem, where we wish to cover whole P using sets S_q for all $q \in P$. Set cover is a classical problem in computer science, which is known to be NP-Hard. However, a polynomial time $O(\log |P|)$ approximation algorithm is known for this problem. Invoking this algorithm in our setting, gives us an efficient way to choosing at most $O((4\alpha)^x \log(\Delta) \log |P|)$ balls which cover all the points in P. Adding edges from p to the centers of these balls and repeating this argument for all p concludes our proof. □

6. Conclusion

In this work, we present the first theoretical framework that geometrically models nearest neighbor search for the expressive multi-vector paradigm. We complement our theoretical with compelling empirical results, demonstrating that simply employing DiskANN as a black-box for multi-vector search remarkably outperforms prior state-of-the-art methods in key recall scenarios.

In spite of the significant progress made in this work, open questions and opportunities for improving practical performance remain. For instance, future work could focus on:

- **Hardware Acceleration.** Because of the considerable linear algebra based cost of computing Chamfer, cleverly offloading distance computations to a GPU would allow the CPU to primarily focus on graph traversal.
- **Reducing Input Sizes.** In Section 4.3, we showed that naive reduction of input sizes was a net-negative on search performance. However, developing more nuanced, data-dependent strategies for reducing input vector set sizes could improve latency, although these must be carefully balanced with computational efficiency to ensure scalability.

References

Atasu, K. and Mittelholzer, T. Linear-complexity data-parallel earth mover's distance approximations. In *International Conference on Machine Learning*, pp. 364–373. PMLR, 2019.

Aumüller, M. and Ceccarello, M. The role of local intrinsic dimensionality in benchmarking nearest neighbor search. In *Similarity Search and Applications: 12th International Conference, SISAP 2019, Newark, NJ, USA, October 2–4, 2019, Proceedings 12*, pp. 113–127. Springer, 2019.

Bakshi, A., Indyk, P., Jayaram, R., Silwal, S., and Waingarten, E. Near-linear time algorithm for the chamfer distance. *Advances in Neural Information Processing Systems*, 36, 2024.

Barrow, H. G., Tenenbaum, J. M., Bolles, R. C., and Wolf, H. C. Parametric correspondence and chamfer matching: Two new techniques for image matching. In *Proceedings: Image Understanding Workshop*, pp. 21–27. Science Applications, Inc, 1977.

Chheda, T., Goyal, P., Tran, T., Patel, D., Boratko, M., Dasgupta, S. S., and McCallum, A. Box embeddings: An open-source library for representation learning using geometric structures. *arXiv preprint arXiv:2109.04997*, 2021.

Dhulipala, L., Hadian, M., Jayaram, R., Lee, J., and Mirrokni, V. Muvera: Multi-vector retrieval via fixed dimensional encodings. *arXiv preprint arXiv:2405.19504*, 2024.

Dobson, M., Shen, Z., Blelloch, G. E., Dhulipala, L., Gu, Y., Simhadri, H. V., and Sun, Y. Scaling graph-based ANNS algorithms to billion-size datasets: A comparative analysis. *CoRR*, abs/2305.04359, 2023. doi: 10.48550/arXiv.2305.04359. URL https://doi.org/10.48550/arXiv.2305.04359.

Engels, J., Coleman, B., Lakshman, V., and Shrivastava, A. Dessert: An efficient algorithm for vector set search with vector set queries. *Advances in Neural Information Processing Systems*, 36, 2024.

Fu, C., Xiang, C., Wang, C., and Cai, D. Fast approximate nearest neighbor search with the navigating spreading-out graph. *Proc. VLDB Endow.*, 12(5):461–474, January 2019. ISSN 2150-8097. doi: 10.14778/3303753.3303754. URL https://doi.org/10.14778/3303753.3303754.

Gao, L., Dai, Z., and Callan, J. Coil: Revisit exact lexical match in information retrieval with contextualized inverted list. *arXiv preprint arXiv:2104.07186*, 2021.

Hofstätter, S., Khattab, O., Althammer, S., Sertkan, M., and Hanbury, A. Introducing neural bag of whole-words with colberter: Contextualized late interactions using enhanced reduction. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pp. 737–747, 2022.

Indyk, P. and Motwani, R. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pp. 604–613, 1998.

Indyk, P. and Naor, A. Nearest-neighbor-preserving embeddings. *ACM Transactions on Algorithms (TALG)*, 3(3): 31–es, 2007.

Indyk, P. and Xu, H. Worst-case performance of popular approximate nearest neighbor search implementations: Guarantees and limitations. *Advances in Neural Information Processing Systems*, 36:66239–66256, 2023.

Järvelin, K. and Kekäläinen, J. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.

Jayaram Subramanya, S., Devvrit, F., Simhadri, H. V., Krishnawamy, R., and Kadekodi, R. Diskann: Fast accurate billion-point nearest neighbor search on a single node. *Advances in Neural Information Processing Systems*, 32, 2019.

Ji, Z., Jain, H., Veit, A., Reddi, S. J., Jayasumana, S., Rawat, A. S., Menon, A. K., Yu, F., and Kumar, S. Efficient document ranking with learnable late interactions. *arXiv preprint arXiv:2406.17968*, 2024.

Khattab, O. and Zaharia, M. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pp. 39–48, 2020.

Kusner, M., Sun, Y., Kolkin, N., and Weinberger, K. From word embeddings to document distances. In *International conference on machine learning*, pp. 957–966. PMLR, 2015.

Lai, A. and Hockenmaier, J. Learning to predict denotational probabilities for modeling entailment. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pp. 721–730, 2017.

Lee, J., Dai, Z., Duddu, S. M. K., Lei, T., Naim, I., Chang, M.-W., and Zhao, V. Rethinking the role of token retrieval in multi-vector retrieval. *Advances in Neural Information Processing Systems*, 36, 2024.

Lou, Z., You, J., Wen, C., Canedo, A., Leskovec, J., et al. Neural subgraph matching. *arXiv preprint arXiv:2007.03092*, 2020.

Narayanan, S., Silwal, S., Indyk, P., and Zamir, O. Randomized dimensionality reduction for facility location and single-linkage clustering. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 7948–7957. PMLR, 2021. URL http://proceedings.mlr.press/v139/narayanan21b.html.

Qian, Y., Lee, J., Duddu, S. M. K., Dai, Z., Brahma, S., Naim, I., Lei, T., and Zhao, V. Y. Multi-vector retrieval as sparse alignment. *arXiv preprint arXiv:2211.01267*, 2022.

Roy, I., Agarwal, R., Chakrabarti, S., Dasgupta, A., and De, A. Locality sensitive hashing in fourier frequency domain for soft set containment search. *Advances in Neural Information Processing Systems*, 36:56352–56383, 2023.

Rubinstein, A. Hardness of approximate nearest neighbor search. In *Proceedings of the 50th annual ACM SIGACT symposium on theory of computing*, pp. 1260–1268, 2018.

Santhanam, K., Khattab, O., Saad-Falcon, J., Potts, C., and Zaharia, M. Colbertv2: Effective and efficient retrieval via lightweight late interaction. *arXiv preprint arXiv:2112.01488*, 2021.

Santhanam, K., Khattab, O., Potts, C., and Zaharia, M. Plaid: an efficient engine for late interaction retrieval. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pp. 1747–1756, 2022.

Thakur, N., Reimers, N., Rücklé, A., Srivastava, A., and Gurevych, I. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*, 2021.

Wan, Z., Chen, D., Li, Y., Yan, X., Zhang, J., Yu, Y., and Liao, J. Transductive zero-shot learning with visual structure constraint. *Advances in neural information processing systems*, 32, 2019.

Wang, X., Macdonald, C., Tonellotto, N., and Ounis, I. Pseudo-relevance feedback for multiple representation dense retrieval. In *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*, pp. 297–306, 2021.

Yao, L., Huang, R., Hou, L., Lu, G., Niu, M., Xu, H., Liang, X., Li, Z., Jiang, X., and Xu, C. Filip: Fine-grained interactive language-image pre-training. *arXiv preprint arXiv:2111.07783*, 2021.

A. DiskANN Routines

Here we give a description of the DiskANN algorithm and refer the reader to the original paper (Jayaram Subramanya et al., 2019) for more details.

The DiskANN build routine Vamana starts with a random directed graph $G = (V, E)$, where each $v \in V$ corresponds to a point $p \in P$. From an appropriate starting node $s \in V$ and each remaining node $i \in V$, Vamana runs GreedySearch(s, i, k, U) from s to i, saving all visited nodes explored along its path in U. U is then sorted by proximity to i, and truncated at the k-th element. Finally, the truncated U is *pruned* with RobustPrune(i, U, α, R): upon considering a new vertex $p^* \in U$, all remaining vertices $p' \in U$ such that $\alpha \cdot m(p^*, p') \leq m(i, p')$ are removed from U, where m denotes the underlying metric for the space. The resulting set U becomes the out-neighbors of the node i.

The DiskANN build routine admits a very natural search routine for some query q; simply calling GreedySearch(s, q, k, U) will output the top-k approximate nearest neighbors to q. See the algorithm psuedocode below.

Algorithm 1 GreedySearch$(s, x_q, k, \mathcal{L})$

Require: Graph G with initial node s, query vector x_q, boolean variable τ, search list size $\mathcal{L}$.
Ensure: Result set $\mathcal{L}$ containing k approximate nearest neighbors, and a set $\mathcal{V}$ containing all visited nodes.

Initialize sets $\mathcal{L} \leftarrow \{s\}$ and $\mathcal{V} \leftarrow \emptyset$.
while $\mathcal{L} \setminus \mathcal{V} \neq \emptyset$ **do**
  Let $p^* \leftarrow \arg\min_{p \in \mathcal{L} \setminus \mathcal{V}} \|x_p - x_q\|$
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{p^*\}$ and
  $\mathcal{L} \leftarrow \mathcal{L} \cup N_{\text{out}}(p^*)$
  if $|\mathcal{L}| > L$ **then**
    Update $\mathcal{L}$ with the closest L nodes to x_q.
  end if
end while
return [k NNs from $\mathcal{L}$; $\mathcal{V}$]

Algorithm 2 RobustPrune$(p, \mathcal{V}, \alpha, R)$

Require: Graph G, point $p \in P$, candidate set $\mathcal{V}$, distance threshold $\alpha \geq 1$, max outdegree bound R.
Ensure: G is modified by setting at most R 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
  end if
  for $p' \in \mathcal{V}$ **do**
    if $\alpha \cdot d(p^*, p') \leq d(p, p')$ **then**
      Remove p' from $\mathcal{V}$.
    end if
  end for
end while

Algorithm 3 Vamana (P, α, L, R) Indexing Algorithm

Require: Database P with n points where i-th point has coords x_i, parameters α, L, R.
Ensure: Directed graph G over P with out-degree $\leq R$.

Initialize G to an empty graph
Let s denote the medoid of P
Let σ be a random permutation of $[n]$
for $i \in [n]$ **do**
  Let [None; $\mathcal{V}$] $\leftarrow$ GREEDYSEARCH$(s, x_{\sigma(i)}, \text{None}, L)$
  Run ROBUSTPRUNE$(\sigma(i), \mathcal{V}, \alpha, R)$ to update out-neighbors of $\sigma(i)$.
  for $j \in N_{\text{out}}(\sigma(i))$ **do**
    if $|N_{\text{out}}(j) \cup \{\sigma(i)\}| > R$ **then**
      Run ROBUSTPRUNE$(j, N_{\text{out}}(j) \cup \{\sigma(i)\}, \alpha, R)$ to update out-neighbors of j.
    else
      Update $N_{\text{out}}(j) \leftarrow N_{\text{out}}(j) \cup \{\sigma(i)\}$
    end if
  end for
end for

B. Dataset statistics

Statistic	Quora	MSMarco	HotpotQA	NQ
99th pct.	2.79207	3.03490	4.60947	7.02222
99.9th pct.	4.04990	4.48803	6.62863	10.30660
Max	5.93×10^4	1.16×10^2	1.54×10^4	5.32×10^4

Table 1. Selected statistics for the $\beta = \frac{D(a,q)}{D(q,a)}$ parameter across datasets.

Statistic	Quora	MSMarco	HotpotQA	NQ
99th percentile	5.35747	1.66125	3.50287	10.07200
99.9th percentile	13.75402	3.67374	8.20625	26.68467
Max k	88.47741	13.49132	28.80361	96.60723

Table 2. Selected statistics for the k parameter (e.g., from k-quasimetric definition or related analysis) across datasets.

Statistic	MSMARCO	Hotpot	NQ	Quora
# Queries	6,980	7,405	3,452	10,000
# Corpus	8.84M	5.23M	2.68M	523K
Avg. # Emb. per Doc.	78.8	68.65	100.3	18.28

Table 3. Dataset information.

In this section, we provide a series of key statistics that characterize the datasets we evaluate on. In particular, we point the reader towards Table 1 and Table 2, which show that both β and k are usually very small. The theory then suggests that we should get a good (i.e. $(1+\epsilon)$) approximate nearest neighbor for searches on these datasets, which as Figure 1 and Figure 2 shows, is in fact the case.

C. Doubling dimension property proof (Lemma 3.2)

Here we prove the property of doubling dimension that is very crucial for all our main results.

Lemma 3.2 (Doubling Dimension Property)**.** *Consider a multivector dataset P whose doubling dimension is x with respect to a k-quasimetric D. For any $p \in P$ and $R > 0$, the ball $B(p, R)$ can be covered with $m \leq (2k)^x$ balls centered around points in $B(p, R)$ of radius at most r/k, where $k > 1$.*

Proof. Define $\text{subcenters}(p, r)$ to be a set of centers of minimum number of balls of radius $\frac{r}{2}$ needed to cover all of $B(p, r) \cap P$. Let m be the smallest integer such that the following inequality holds $2^m \geq k$; note that $k \geq 2^{m-1}$. Now consider the following construction of 2^{mx} balls of radii r/k to cover all points in $B(p, r) \cap P$. Let S_0 be a singleton set containing p and for any $i \geq 1$ let S_i be defined as follows,

$$S_i = \bigcup_{p' \in S_{i-1}} \text{subcenters}(p', r/2^{i-1}) \, .$$

We can claim that balls of radii $r/2^i$ centered around points in S_i cover all points in $B(p, r)$ and $|S_i| \leq 2^{ix}$. We prove this claim using induction. Note that the claim trivially holds for $i = 0$ by the definition of $B(p, r)$ and definition of doubling dimension respectively. By induction over i, we assume that the claim holds upto $i-1$ and show the claim for i. For i the induction claim holds as for any point p' in S_{i-1}, all points covered by $B(p', r/2^{i-1})$ are also covered by balls of radii $r/2^i$ around $\text{subcenters}(p, r)$. Therefore, balls of radii $r/2^m \leq r/k$ centered around points in S_m covers all points in $B(p, r)$. Furthermore, it is immediate that $|S_i| \leq \sum_{p' \in S_{i-1}} |\text{subcenters}(p', r/2^{i-1}) \leq |S_{i-1}|2^x \leq 2^{(i-1)x}2^x \leq 2^{ix}$. We conclude the proof and the induction claim holds for all $i \geq 0$. The lemma statement holds by applying the induction claim for S_m. Note that the cardinality of S_m is upper bounded by 2^{mx}. As $k \geq 2^{m-1}$, we get that, $2^{mx} \leq (2k)^x$ and we conclude the proof. □

Dataset	Method	Mean Deg.	Idx Size (GB)
HotpotQA	Chamfer	51.0865	1.02
	MUVERA	93.6500	1.85
NQ	Chamfer	61.6716	0.63
	MUVERA	83.7390	0.85
MS MARCO	Chamfer	67.8025	2.27
	MUVERA	99.4127	3.31
Quora	Chamfer	24.4257	0.05
	MUVERA	48.8036	0.10

Table 4. Comparison of Chamfer and MuVERA Indices

D. Proof of Lemma 3.4

Lemma 3.4 (Approximation of greedy search)**.** *Consider a α-reachable graph G with respect to a k-quasimetric function D. For any starting point s, the greedy search procedure takes $O(\text{polylog}(\frac{k\Delta}{\epsilon\nu})\log(n))$ steps to reach a $(\frac{\beta k}{\alpha - k^2} + \frac{\alpha}{\alpha - k^2} + \epsilon)$-approximate nearest neighbor solution with respect to D, where $\beta = \frac{D(a,q)}{D(q,a)}$ and ν is a parameterized by β, k and α. Furthermore, if $\alpha > \Omega((k^2 + \beta k)/\epsilon)$, the greedy search procedure outputs a $(1+\epsilon)$-approximate nearest neighbor solution.*

Proof. We proceed by cases:

1. Suppose $D(q, s) > 2kD_{max}$. Then we have that $D(q, a) > D(q, s) - kD(a, s) > D(q, s) - kD_{max} > \frac{D(q,s)}{2}$. Then, using equation (1), we can expand the

approximation ratio $c_i = \frac{D(q,v_i)}{D(q,a)}$ as follows:

$$\begin{aligned}\frac{D(q,v_i)}{D(q,a)} &\leq \frac{z^{-t}D(q,s) + \frac{kD(a,q)+\alpha D(q,a)}{\alpha - k^2}}{D(q,a)} \\ &= \frac{z^{-t}D(q,s)}{D(q,a)} + \frac{k\beta+\alpha}{\alpha-k^2} \\ &\leq 2z^{-t} + \frac{k\beta+\alpha}{\alpha-k^2} = 2z^{-t} + M\end{aligned}$$

Therefore, for $\epsilon > 0$, we obtain a $(M+\epsilon)$-approximate nearest neighbor in $\log_z \frac{2}{\epsilon}$ steps.

2. Now, suppose $D(q,s) \leq 2kD_{max}$, and $D(q,a) \geq vD_{min}$. By equation (1), we can get an $(M+\epsilon)$-approximate nearest neighbor for $z^{-t}D(q,s) < \epsilon D(q,a)$. Using the given bounds on $D(q,s)$ and $D(q,a)$ results in needing $\log_z\left(\frac{2k\Delta}{\epsilon v}\right)$ steps.

3. Finally, suppose $D(q,s) \leq 2kD_{max}$, and $D(q,a) < vD_{min}$. Say v_t (i.e. the result at step t), such that $D(q,v_t) > D(q,a)$, is not the nearest neighbor. We now wish to show a lower bound for $D(q,v_t)$. See that by definition,

$$\begin{aligned}D(a,v) &\leq D(a,q) + kD(q,v) \\ &= \beta D(q,a) + kD(q,v) \\ &\leq \beta D(q,v) + kD(q,v)\end{aligned}$$

Thus $\frac{D_{min}}{\beta+k} \leq \frac{D(a,v)}{\beta+k} \leq D(q,v)$. With Equation (1), if v_t is not the nearest neighbor, it satisfies

$$\begin{aligned}\frac{D_{min}}{\beta+k} \leq D(q,v_t) &\leq z^{-t}D(q,s) + \frac{kD(a,q)+\alpha D(q,a)}{\alpha-k^2} \\ &\leq z^{-t}2kD_{max} + \frac{vD_{min}(k\beta+\alpha)}{\alpha-k^2}\end{aligned}$$

Canceling then rearranging terms gives

$$\frac{1}{\beta+k} - \left(\frac{k}{\alpha-k^2}\right)\beta v - \left(\frac{\alpha}{\alpha-k^2}\right)v \leq z^{-t}2k\Delta$$

Therefore, for $t < \log_z \frac{2k\Delta}{v}$, the algorithm reaches the exact nearest neighbor.

□