Graph-based Nearest Neighbor Search:
From Practice to Theory

Liudmila Prokhorenkova [*1 2 3]   **Aleksandr Shekhovtsov** [*1 3]

Abstract

Graph-based approaches are empirically shown to be very successful for the nearest neighbor search (NNS). However, there has been very little research on their theoretical guarantees. We fill this gap and rigorously analyze the performance of graph-based NNS algorithms, specifically focusing on the low-dimensional ($d \ll \log n$) regime. In addition to the basic greedy algorithm on nearest neighbor graphs, we also analyze the most successful heuristics commonly used in practice: speeding up via adding shortcut edges and improving accuracy via maintaining a dynamic list of candidates. We believe that our theoretical insights supported by experimental analysis are an important step towards understanding the limits and benefits of graph-based NNS algorithms.

1. Introduction

Many methods in machine learning, pattern recognition, coding theory, and other research areas are based on nearest neighbor search (Bishop, 2006; Chen & Shah, 2018; May & Ozerov, 2015; Shakhnarovich et al., 2006). In particular, the k-nearest neighbor method is included in the list of top 10 algorithms in data mining (Wu et al., 2008). Since modern datasets are mostly massive (both in terms of the number of elements n and the dimension d), reducing the computation complexity of NNS algorithms is of the essence. The nearest neighbor problem is to preprocess a given dataset $\mathcal{D}$ in such a way that for an arbitrary forthcoming query vector q we can quickly (in time $o(n)$) find its nearest neighbors in $\mathcal{D}$.

Many efficient methods exist for NN problem, especially when the dimension d is small (Arya et al., 1998; Bentley,

---
[*]Equal contribution [1]Yandex, Moscow, Russia [2]Moscow Institute of Physics and Technology, Dolgoprudny, Moscow Region, Russia [3]Higher School of Economics, Moscow, Russia. Correspondence to: Liudmila Prokhorenkova <ostroumova-la@yandex-team.ru>.

1975; Meiser, 1993). In particular, the algorithms based on recursive partitions of the space, like k-d threes and random projection trees, are widely used (Bentley, 1975; Dasgupta & Freund, 2008; Dasgupta & Sinha, 2015; Keivani & Sinha, 2018). The most well-known algorithm usable for large d is the Locality Sensitive Hashing (LSH) (Indyk & Motwani, 1998), which is well studied theoretically and widely used in practical applications.

Recently, graph-based approaches were shown to demonstrate superior performance over other types of algorithms in many large-scale applications of NNS (Aumüller et al., 2019). Most graph-based methods are based on constructing a nearest neighbor graph (or its approximation), where nodes correspond to the elements of $\mathcal{D}$, and each node is connected to its nearest neighbors by directed edges (Dong et al., 2011; Hajebi et al., 2011; Wang et al., 2012). Then, for a given query q, one first takes an element in $\mathcal{D}$ (either random or fixed predefined) and makes greedy steps towards q on the graph: at each step, all neighbors of a current node are evaluated, and the one closest to q is chosen. Various additional heuristics were proposed to speed up graph-based search (Baranchuk et al., 2019; Fu et al., 2019; Iwasaki & Miyazaki, 2018; Malkov et al., 2014; Malkov & Yashunin, 2018).

While there is a lot of evidence empirically showing the superiority of graph-based NNS algorithms in practical applications, there is very little theoretical research supporting this. Laarhoven (2018) made the first step in this direction by providing time–space trade-offs for approximate nearest neighbor (ANN) search on sparse datasets uniformly distributed on a d-dimensional Euclidean sphere. The current work significantly extends these results and differs in several important aspects, as discussed in the next section.

Our analysis assumes the uniform distribution on a sphere, and we mostly focus on the dense regime $d \ll \log n$. This setup is motivated by our experiments demonstrating that *uniformization* and *densification* applied to a *general* dataset may improve some graph-based NNS algorithms. The dense regime is additionally motivated by the fact that real-world datasets are known to have low intrinsic dimension (Beygelzimer et al., 2006; Lin & Zhao, 2019).

We first investigate how the greedy search on simple NN graphs work in dense and sparse regimes (Section 4.2). For the dense regime, the time complexity is $\Theta\left(n^{1/d} \cdot M^d\right)$ for some constant M. Here M^d corresponds to the complexity of one step and $n^{1/d}$ to the number of steps.

Then, we analyze the effect of long-range links (Section 4.3): in practice, shortcut edges between distant elements are added to NN graphs to make faster progress in the early stages of the algorithm. This is a core idea of, e.g., navigable small-world graphs (NSW and HNSW) (Malkov et al., 2014; Malkov & Yashunin, 2018). While a naive method of adding long links uniformly at random does not improve the asymptotic query time, we show that properly added edges can reduce the number of steps to $O\left(\log^2 n\right)$. This part is motivated by Kleinberg (2000), who proved a related result for a greedy routing on a two-dimensional lattice. We adapt Kleinberg's result to NNS on a dataset uniformly distributed over a sphere in $\mathbb{R}^d$ (with possibly growing d). Additionally, while Kleinberg's theorem inspired many algorithms (Beaumont et al., 2007; Karbasi et al., 2015; Malkov & Yashunin, 2018), to the best of our knowledge, we are the first to propose an efficient method of constructing properly distributed long edges for a *general* dataset.

We also consider a heuristics known to significantly improve the accuracy: maintaining a dynamic list of several candidates instead of just one optimal point, which is a general technique known as *beam search*. We rigorously analyze the effect of this technique, see Section 4.4.

Finally, in Section 5, we empirically illustrate the obtained results, discuss the most promising techniques, and demonstrate why our assumptions are reasonable.

2. Related Work

Several well-known algorithms proposed for NNS are based on recursive partitions of the space, e.g., k-d trees and random projection trees (Bentley, 1975; Dasgupta & Freund, 2008; Dasgupta & Sinha, 2015; Keivani & Sinha, 2018). The query time of a k-d tree is $O\left(d \cdot 2^{O(d)}\right)$, which leads to an efficient algorithm for the exact NNS in the dense regime $d \ll \log n$ (Chen & Shah, 2018). When $d \gg \log n$, all algorithms suffer from the curse of dimensionality, so the problem is often relaxed to approximate nearest neighbor (ANN) formally defined in Section 3. Among the algorithms proposed for the ANN problem, LSH (Indyk & Motwani, 1998) is the most theoretically studied one. The main idea of LSH is to hash the points such that the probability of collision is much higher for points that are close to each other than for those that are far apart. Then, one can retrieve near neighbors for a query by hashing it and checking the elements in the same buckets. LSH solves c-ANN with query time $\Theta\left(dn^\varphi\right)$ and space complexity $\Theta\left(n^{1+\varphi}\right)$. As-

suming the Euclidean distance, the optimal φ is about $\frac{1}{c^2}$ for data-agnostic algorithms (Andoni & Indyk, 2008; Datar et al., 2004; Motwani et al., 2007; O'Donnell et al., 2014). By using a data-dependent construction, this bound can be improved up to $\frac{1}{2c^2-1}$ (Andoni & Razenshteyn, 2015; Andoni & Razenshteyn, 2016). Becker et al. (2016) analyzed spherical locally sensitive filters (LSF) and showed that LSF can outperform LSH when the dimension d is logarithmic in the number of elements n. LSF can be thought of as lying in the middle between LSH and graph-based methods: LSF uses small spherical caps to define filters, while graph-based methods also allow, roughly speaking, moving to the neighboring caps.

In contrast to LSH, graph-based approaches are not so well understood. It is known that if we want to be able to find the exact nearest neighbor for any possible query via a greedy search, then the graph has to contain the classical Delaunay graph as a subgraph. However, for large d, Delaunay graphs have huge node degrees and cannot be constructed in a reasonable time (Navarro, 2002). In a more recent theoretical paper, Laarhoven (2018) considers datasets uniformly distributed on a d-dimensional sphere with $d \gg \log n$ and provides time–space trade-offs for the ANN search. The current paper, while using a similar setting, differs in several important aspects. First, we mostly focus on the dense regime $d \ll \log n$, show that it significantly differs both in terms of techniques and results, and argue why it is reasonable to work in this setting. Second, in addition to the analysis of plain NN graphs, we are the first to analyze how some additional tricks commonly used in practice affect the accuracy and complexity of the algorithm. These tricks are adding shortcut edges and using beam search. Finally, we support some claims made in the previous work by a rigorous analysis (see Supplementary Materials E). Let us also briefly discuss a recent paper (Fu et al., 2019), which rigorously analyzes the time and space complexity for searching the elements of so-called monotonic graphs. Note that the obtained guarantees are provided only for the case when a query coincides with an element of a dataset, which is a very strong assumption, and it is not clear how the results would generalize to a general setting.

A part of our research is motivated by Kleinberg (2000), who proved that after adding random edges with a particular distribution to a lattice, the number of steps needed to reach a query via a greedy routing scales polylogarithmically, see details in Section 4.3. We generalize this result to a more realistic setting and propose a practically applicable method to generate such edges.

Finally, let us mention a recent empirical paper comparing the performance of HNSW with other graph-based NNS algorithms (Lin & Zhao, 2019). In particular, it shows that HNSW has superior performance over one-layer graphs only

for low-dimensional data. We demonstrate theoretically why this can be the case (assuming uniform datasets): we prove that for plain NN graphs and for $d \gg \sqrt{\log n}$, the number of steps of graph-based NNS is negligible compared with one-step complexity. Moreover, for $d \gg \log n$, the algorithm converges in just two steps. Interestingly, Figure 5 in Lin & Zhao (2019) shows that on uniformly distributed synthetic datasets simple kNN graphs and HNSW show quite similar results, especially when the dimension is not too small. This means that studying simple NN graphs is a reasonable first step in the analysis of graph-based NNS algorithms. In contrast, on real datasets, HNSW is significantly better, which is caused by a so-called diversification of neighbors. However, as we show empirically in Section 5, simple kNN graphs can work comparably to HNSW even on real datasets, if we use the uniformization procedure (Sablayrolles et al., 2018).

3. Setup and Notation

We are given a dataset $\mathcal{D} = \{\mathbf{x}_1, \ldots \mathbf{x}_n\}$, $\mathbf{x}_i \in \mathbb{R}^{d+1}$ and assume that all elements of $\mathcal{D}$ belong to a unit Euclidean sphere, $\mathcal{D} \subset \mathcal{S}^d$. This special case is of particular importance for practical applications since feature vectors are often normalized.[1] For a given query $q \in \mathcal{S}^d$ let $\bar{\mathbf{x}} \in \mathcal{D}$ be its nearest neighbor. The aim of the exact NNS is to return $\bar{\mathbf{x}}$, while in c, R-ANN (approximate *near* neighbor), for given $R > 0$, $c > 1$, we need to find such $\mathbf{x}'$ that $\rho(q, \mathbf{x}') \leq cR$ if $\rho(q, \bar{\mathbf{x}}) \leq R$ (Andoni et al., 2017; Chen & Shah, 2018).[2] By $\rho(\cdot, \cdot)$ we further denote a spherical distance.

Similarly to Laarhoven (2018), we assume that the elements $\mathbf{x}_i \in \mathcal{D}$ are i.i.d. random vectors uniformly distributed on $\mathcal{S}^d$. Random uniform datasets are considered to be the most natural "hard" distribution for the ANN problem (Andoni & Razenshteyn, 2015). Hence, it is an important step towards understanding the limits and benefits of graph-based NNS algorithms.[3] From a practical point of view, real datasets are usually far from being uniformly distributed. However, in some applications uniformity is helpful, and there are techniques allowing to make a dataset more uniform while approximately preserving the distances (Sablayrolles et al., 2018). Remarkably, in our experiments (Section 5) we show that this trick, combined with dimensionality reduction, is beneficial for some graph-based NNS algorithms. We fur-

ther assume that a query vector $q \in \mathcal{S}^d$ is placed uniformly within a distance R from the nearest neighbor $\bar{\mathbf{x}}$ (since c, R-ANN problem is defined conditionally on the event $\rho(q, \bar{\mathbf{x}}) \leq R$). Such nearest neighbor is called *planted*.

In the current paper, we use a standard assumption that the dimensionality $d = d(n)$ grows with n (Chen & Shah, 2018). We distinguish three fundamentally different regimes in NN problems: dense with $d \ll \log(n)$; sparse with $d \gg \log(n)$; moderate with $d = \Theta(\log(n))$.[4] While Laarhoven (2018) focused solely on the sparse regime, we also consider the dense one, which is fundamentally different in terms of geometry, proof techniques, and query time.

As discussed in the introduction, most graph-based approaches are based on constructing a nearest neighbor graph (or its approximation). For uniformly distributed datasets, connecting an element $\mathbf{x}$ to a given number of nearest neighbors is essentially equivalent to connecting it to all such nodes $\mathbf{y}$ that $\rho(\mathbf{x}, \mathbf{y}) \leq \rho^*$ with some appropriate ρ^* (since the number of nodes at a distance at most ρ^* is concentrated around its expectation, see also Supplementary Materials F.3 for an empirical illustration). Therefore, at the preprocessing stage, we choose some ρ^* and construct a graph using this threshold. Later, when we get a query q, we sample a random element $\mathbf{x} \in \mathcal{D}$ such that $\rho(\mathbf{x}, q) < \frac{\pi}{2}$,[5] and perform a graph-based greedy descent: at each step, we measure the distances between the neighbors of a current node and q and move to the closest neighbor, while we make progress. In the current paper, we assume one iteration of this procedure, i.e., we do not restart the process several times.

4. Theoretical Analysis

In this section, we overview the obtained theoretical results. We start with a preliminary analysis of the properties of spherical caps and their intersections. These properties give some intuition and are extensively used throughout the proofs. Then, we analyze the performance of the greedy search over plain NN graphs. We mostly focus on the dense regime, but also formulate the corresponding theorem for the sparse one. After that, we analyze how the shortcut edges affect the complexity of the algorithm. We prove that they indeed improve the asymptotic query time, but only in the dense regime. Finally, we analyze the positive effect of the beam search, which is widely used in practice.

---

[1]From a theoretical point of view, there is a reduction of ANN in the entire Euclidean space to ANN on a sphere (Andoni & Razenshteyn, 2015; Valiant, 2015).

[2]There is also a notion of c-ANN, where the goal is to find such $\mathbf{x}'$ that $\rho(q, \mathbf{x}') \leq c\rho(q, \bar{\mathbf{x}})$; c-ANN can be reduced to c, R-ANN with additional $O(\log n)$ factor in query time and $O(\log^2 n)$ in storage cost (Chen & Shah, 2018; Har-Peled et al., 2012).

[3]Also, Andoni & Razenshteyn (2015) show how to reduce ANN on a generic dataset to ANN on a "pseudo-random" dataset for a data-dependent LSH algorithm.

[4]Throughout the paper $\log$ is a binary logarithm (with base 2) and $\ln$ is a natural logarithm (with base e).

[5]We can easily achieve this by trying a constant number of random samples since each sample succeeds with probability $1/2$. This trick speeds up the procedure (without affecting the asymptotics) and simplifies the proofs.

4.1. Auxiliary Results

Let us discuss some technical results on the volumes of spherical caps and their intersections, which we extensively use in the proofs.

Denote by μ the Lebesgue measure over $\mathbb{R}^{d+1}$. By $C_{\mathbf{x}}(\gamma)$ we denote a spherical cap of *height* γ centered at $\mathbf{x} \in \mathcal{S}^d$, i.e., $\{\mathbf{y} \in \mathcal{S}^d : \langle \mathbf{x}, \mathbf{y} \rangle \geq \gamma\}$; $C(\gamma) = \mu\left(C_{\mathbf{x}}(\gamma)\right)$ denotes the volume of a spherical cap of height γ. Throughout the paper, for any variable γ, $0 \leq \gamma \leq 1$, we let $\hat{\gamma} := \sqrt{1 - \gamma^2}$. We say that $\hat{\gamma}$ is the *radius* of a spherical cap.

The volume of a spherical cap defines the expected number of neighbors of a node. We estimate this volume in Supplementary Materials A.1. Despite some additional terms (which can often be neglected), one can think that $C(\gamma) \propto \hat{\gamma}^d$.

The intersections of spherical caps are also important: their volumes are needed to estimate the probably of making a step of graph-based search. Formally, by $W_{\mathbf{x},\mathbf{y}}(\alpha, \beta)$ we denote the intersection of two spherical caps centered at $\mathbf{x} \in \mathcal{S}^d$ and $\mathbf{y} \in \mathcal{S}^d$ with heights α and β, respectively, i.e., $W_{\mathbf{x},\mathbf{y}}(\alpha, \beta) = \{\mathbf{z} \in \mathcal{S}^d : \langle \mathbf{z}, \mathbf{x} \rangle \geq \alpha, \langle \mathbf{z}, \mathbf{y} \rangle \geq \beta\}$. By $W(\alpha, \beta, \theta)$ we denote the volume of such intersection given that the angle between $\mathbf{x}$ and $\mathbf{y}$ is θ. In Supplementary Materials A.2, we estimate $W(\alpha, \beta, \theta)$. We essentially prove that for $\gamma = \frac{\sqrt{\alpha^2 + \beta^2 - 2\alpha\beta\cos\theta}}{\sin\theta}$, $W(\alpha, \beta, \theta)$ (or its complement) $\propto \hat{\gamma}^d$.

Although our results are similar to those formulated by Becker et al. (2016), it is crucial for our analysis that parameters defining spherical caps depend on d and either γ or $\hat{\gamma}$ may tend to zero. In contrast, the results of Becker et al. (2016) hold only for fixed parameters. Also, we extend Lemma 2.2 in their paper by analyzing both $W(\alpha, \beta, \theta)$ and its complement: we need a lower bound on $W(\alpha, \beta, \theta)$ to show that with high probability we can make a step of the algorithm and an upper bound on $C(\alpha) - W(\alpha, \beta, \theta)$ to show that at the final step of the algorithm we can find the nearest neighbor with high probability.

The fact that $C(\gamma) \propto \hat{\gamma}^d$ allows us to understand the principal difference between the dense and sparse regimes. For dense datasets, we assume $d = d(n) = \log(n)/\omega$, where $\omega = \omega(n) \to \infty$ as $n \to \infty$. In this case, it is convenient to operate with radii of spherical caps. An essential property of the dense regime is the fact that the distance from a given point to its nearest neighbor behaves as $2^{-\omega}$, so it decreases with n. Indeed, let $\hat{\alpha}_1$ be the radius of a cap centered at a given point and covering its nearest neighbor, then we have $C(\alpha_1) \sim \frac{1}{n}$, i.e., $\hat{\alpha}_1 \sim n^{-\frac{1}{d}} = 2^{-\omega}$. To construct a nearest neighbor graph, we use spherical caps of radius $M \cdot 2^{-\omega}$ with some constant $M > 1$.

In sparse regime, we assume $d = \omega \log(n)$, $\omega \to \infty$ as $n \to \infty$. In this case, it is convenient to operate with heights of spherical caps. A crucial feature of sparse regime is that the heights under consideration tend to zero as n grows. Informally speaking, all nodes are at spherical distance about $\pi/2$ from each other. Indeed, we have $C(\alpha_1) \sim \frac{1}{n}$, i.e., $\alpha_1^2 \sim 1 - n^{-\frac{2}{d}} = 1 - 2^{-\frac{2}{\omega}} \propto w^{-1}$ and it tends to zero with n. In this case, to construct a graph, we use spherical caps with height equal to $\sqrt{M} \cdot \alpha_1$, where $M < 1$ is some constant.

4.2. Greedy Search on Plain NN Graphs

Dense regime For any constant $M > 1$, let $G(M)$ be a graph obtained by connecting $\mathbf{x}_i$ and $\mathbf{x}_j$ iff $\rho(\mathbf{x}_i, \mathbf{x}_j) \leq \arcsin\left(M \, n^{-1/d}\right)$. The following theorem is proven in Supplementary Materials B.1-B.3.

Theorem 1. *Assume that $d \gg \log\log n$ and we are given some constant $c \geq 1$. Let M be a constant such that $M > \sqrt{\frac{4c^2}{3c^2 - 1}}$, then, with probability $1 - o(1)$, $G(M)$-based NNS solves c, R-ANN for any R (or the exact NN problem if $c = 1$); time complexity is $\Theta\left(d^{1/2} \cdot n^{1/d} \cdot M^d\right) = n^{o(1)}$; space complexity is $\Theta\left(n \cdot d^{-1/2} \cdot M^d \cdot \log n\right) = n^{1+o(1)}$.*

It follows from Theorem 1 that both time and space complexities increase with M (for constant M), so one may want to choose the smallest possible M. When the aim is to find the exact nearest neighbor ($c = 1$), we can take any $M > \sqrt{2}$. When $c > 1$, the lower bound for M decreases with c.

The space complexity straightforwardly depends on M: the radius of a spherical cap defines the expected number of neighbors for a node, which is $\Theta(d^{-1/2} \cdot M^d)$, and additional $\log n$ corresponds to storing integers up to n. The time complexity consists of two terms: the complexity of one step is multiplied by the number of steps. The complexity of one step is the number of neighbors multiplied by d: $\Theta(d^{1/2} \cdot M^d)$. The number of steps is $\Theta\left(n^{1/d}\right)$.

While d is negligible compared with M^d, the relation between $n^{1/d}$ and M^d is non-trivial. Indeed, when $d \gg \sqrt{\log n}$, the term M^d dominates, so in this regime, the smaller M the better asymptotics we get (both for time and space complexities). However, when the dataset is very dense, i.e., $d \ll \sqrt{\log n}$, the number of steps becomes much larger than the complexity of one step. For such datasets, it could be possible that taking $M = M(n) \gg 1$ would improve the query time (as it affects the number of steps). However, in Supplementary Materials B.4, we prove that this is not the case, and the query time from Theorem 1 cannot be improved.

Finally, since all distances considered in the proof tend to zero with n, it is easy to verify that all results stated above for the spherical distance also hold for the Euclidean one.

Sparse regime For any M, $0 < M < 1$, let $G(M)$ be a graph obtained by connecting $\mathbf{x}_i$ and $\mathbf{x}_j$ iff $\rho(\mathbf{x}_i, \mathbf{x}_j) \leq \arccos\left(\sqrt{\frac{2M \ln n}{d}}\right)$. The following theorem holds (see Supplementary Materials B.5 for the proof).

Theorem 2. *For any $c > 1$ let $\alpha_c = \cos\left(\frac{\pi}{2c}\right)$ and let M be any constant such that $M < \frac{\alpha_c^2}{\alpha_c^2+1}$. Then, with probability $1 - o(1)$, $G(M)$-based NNS solves c, R-ANN (for any R and for spherical distance); time complexity of the procedure is $\Theta\left(n^{1-M+o(1)}\right)$; space complexity is $\Theta\left(n^{2-M+o(1)}\right)$.*

Interestingly, as follows from the proof, in the sparse regime, the greedy algorithm converges in at most two steps with probability $1 - o(1)$ (on a uniform dataset). As a result, there is no trade-off between time and space complexity: larger values of M reduce both of them.

One can easily obtain an analog of Theorem 2 for the Euclidean distance. In Theorem 2, α_c is the height of a spherical cap covering a spherical distance $\frac{\pi}{2c}$, which is c times smaller than $\pi/2$. For the Euclidean distance, we have to replace $\frac{\pi}{2}$ by $\sqrt{2}$ and then the height of a spherical cap covering Euclidean distance $\sqrt{2}/c$ is $\alpha_c = 1 - \frac{1}{c^2}$. So, we get the following corollary.

Corollary 1. *For the Euclidean distance, Theorem 2 holds with $\alpha_c = 1 - \frac{1}{c^2}$, i.e., $M < \frac{\left(1-1/c^2\right)^2}{(1-1/c^2)^2+1}$.*

As a result, we can obtain time complexity n^φ and space complexity $n^{1+\varphi}$, where φ can be made about $\frac{1}{(1-1/c^2)^2+1}$. Note that this result corresponds to the case $\rho_q = \rho_s$ in Laarhoven (2018).

4.3. Analysis of Long-range Links

As discussed in the previous section, when $d \gg \sqrt{\log n}$, the number of steps is negligible compared to the one-step complexity. Hence, reducing the number of steps cannot change the main term of the asymptotics.[6] However, if $d \ll \sqrt{\log n}$ (very dense setting), the number of steps becomes the main term of time complexity. In this case, it is reasonable to reduce the number of steps via adding so-called long-range links (or shortcuts) — some edges connecting elements that are far away from each other — which may speed up the search on early stages of the algorithm.

The simplest way to obtain a graph with a small diameter from a given graph is to connect each node to a few neighbors chosen uniformly at random. This idea is proposed by Watts & Strogatz (1998) and gives $O(\log n)$ diameter for the so-called "small-world model". However, having

a logarithmic diameter does *not* guarantee a logarithmic number of steps in graph-based NNS, since these steps, while being greedy in the underlying metric space, may not be optimal on a graph (Kleinberg, 2000). In Supplementary Materials C.1, we formally prove that adding edges uniformly at random cannot improve the asymptotic time complexity. The intuition is simple: choosing the closest neighbor among the long-range ones is equivalent to sampling a certain number of nodes (uniformly at random among the whole set) and then choosing the one closest to q.

This agrees with Kleinberg (2000), who considered a 2-dimensional grid supplied with some random long-range edges. Kleinberg assumed that in addition to the local edges, each node creates one random outgoing long link, and the probability of a link from u to v is proportional to $\rho(u,v)^{-r}$. He proved that for $r = 2$, the greedy graph-based search finds the target element in $O\left(\log^2 n\right)$ steps, while any other r gives at least n^φ with $\varphi > 0$. This result can be easily extended to *constant* $d > 2$, in this case one should take $r = d$ to achieve polylogarithmic number of steps.

Kleinberg (2000) influenced a large number of further studies. Some works generalized the result to other settings (Barrière et al., 2001; Bonnet et al., 2007; Duchon et al., 2006), others used it as a motivation of search algorithms (Beaumont et al., 2007; Karbasi et al., 2015). It is also mentioned as a motivation for a widely used HNSW algorithm (Malkov & Yashunin, 2018). However, Kleinberg's probabilities have not been used directly since 1) it is unclear how to use them for general distributions, when the intrinsic dimension is not known or varies over the dataset, 2) generating properly distributed edges has $\Theta\left(n^2 d\right)$ complexity, which is infeasible for many practical tasks.

We address these issues. First, we translate the result of Kleinberg (2000) to our setting (importantly, we have $d \to \infty$). Second, we show how one can apply the method to general datasets without thinking about the intrinsic dimension and non-uniform distributions. Finally, we discuss how to reduce the computational complexity of graph construction procedure.

Following Kleinberg (2000), we draw long-range edges with the following probabilities:

$$P(\text{edge from } u \text{ to } v) = \frac{\rho(u,v)^{-d}}{\sum_{w \neq u} \rho(u,w)^{-d}}. \quad (1)$$

Theorem 3. *Under the conditions of Theorem 1, sampling one long-range edge for each node according to (1) reduces the number of steps to $O(\log^2 n)$ (with probability $1 - o(1)$).*

This theorem is proven in Supplementary Materials C.2. It is important that in contrast to Kleinberg (2000), we assume $d \to \infty$. Indeed, a straightforward generalization of Kleinberg's result to non-constant d gives an additional 2^d

---

[6]This agrees with the empirical results obtained by Lin & Zhao (2019), where on synthetic datasets the difference between simple kNN graphs and the more advanced HNSW algorithm becomes smaller as d increases and vanishes after $d = 8$.

multiplier, which we were able to remove.

Note that using long-range edges, we can guarantee $O\left(\log^2 n\right)$ steps, while plain NN graphs give $\Theta\left(n^{1/d}\right)$ (see Theorem 1 and the discussion after that). Hence, reducing the number of steps is reasonable if $\log^2 n < n^{1/d}$, which means that $d < \frac{\log n}{2\log\log n}$.

As follows from the proof, adding more long-range edges may further reduce the number of steps. It is theoretically shown in the following corollary and is empirically evaluated in Supplementary Materials F.3.

Corollary 2. *If for each node we add $\Theta(\log n)$ long-range edges, then the number of steps becomes $O(\log n)$, so the time complexity is $O\left(d^{1/2}\cdot\log n\cdot M^d\right)$ while the asymptotic space complexity does not change compared to Theorem 1. Further increasing the number of long-range edges does not improve the asymptotic complexity.*

Also, we suggest the following trick, which noticeably reduces the number of steps in practice while not affecting the theoretical asymptotics: at each iteration, we check the long-range links first and proceed with the local ones only if we cannot make progress with long edges. We empirically evaluate this trick (called LLF) in Section 5.

It is non-trivial how to apply Theorem 3 in practice due to the dependence of probabilities on d: real datasets usually have a low intrinsic dimension even when embedded to a higher-dimensional space (Beygelzimer et al., 2006; Lin & Zhao, 2019), and the intrinsic dimension may vary over the dataset. Let us show how to make the distribution in (1) dimension-free. For lattices and uniformly distributed datasets, the distance to a node is strongly related to the *rank* of this node in the list of all elements, when they ordered by a distance. Formally, let v be the k-th neighbor of u. Then, we define $\rho_{rank}(u,v) = (k/n)^{1/d}$. For uniform datasets, $\rho(u,v)$ locally behaves as $\rho_{rank}(u,v)$ (up to a constant multiplier) since the number of nodes at a distance ρ from a given one grows as ρ^d. If we replace ρ by ρ_{rank} in (1), we obtain:

$$\text{P(edge to k-th neighbor)} = \frac{1/k}{\sum_{i=1}^n 1/i} \sim \frac{1}{k\ln n}. \quad (2)$$

This distribution is *dimension independent*, i.e., it can be easily used for general datasets.

Finally, we address the problem of the computational complexity of graph construction procedure. For this, we propose to generate long edges as follows: for each source node, we first choose $n^\varphi, 0 < \varphi < 1$, candidates uniformly at random, and then sample a long edge from this set according to the probabilities in (2). Pre-sampling reduces the time complexity to $\Theta\left(n^{1+\varphi}\left(d+\log n^\varphi\right)\right)$, compared with $\Theta(n^2d)$ in Theorem 3. The following lemma shows how it affects the distribution.

Lemma 1. *Let P_k be the probability defined in (2) and P_k^φ be the corresponding probability assuming the pre-sampling of n^φ elements. Then, for all $k > n^{1-\varphi}$, $P_k^\varphi/P_k = \Theta(1/\varphi)$.*

Informally, Lemma 1 says that for most of the elements, the probability does not change significantly. In Section 5, we use pre-sampling with $\varphi = 1/2$ and in Supplementary Materials F.3 we show that this pre-sampling and using (2) instead of (1) does not affect the quality of graph-based NNS.

4.4. Beam Search

Beam search is a heuristic algorithm that explores a graph by expanding the most promising element in a limited set. It is widely used in graph-based NNS algorithms (Fu et al., 2019; Malkov & Yashunin, 2018) as it drastically improves the accuracy. Moreover, it was shown that even a simple kNN graph supplied with beam search can show results close to the state-of-the-art on synthetic datasets (Lin & Zhao, 2019). However, to the best of our knowledge, we are the first to analyze the effect of beam search in graph-based NNS algorithms theoretically.

Theorem 1 states that to solve the exact NN problem with probability $1 - o(1)$, we need to have about M^d neighbors for each node, where $M > \sqrt{2}$. If we reduce M below this bound, then with a significant probability, the algorithm may get stuck in a local optimum before it reaches the nearest neighbor. As follows from the proof, the problem of local optima is critical for the last steps of the algorithm, i.e., large degrees are needed near the query.

Let us show that M can be reduced significantly if the beam search is used instead of the greedy search (see Figure 1 for an illustration). Indeed, assume that we have reached some local neighborhood of the query q, and there are m points closer to q in the dataset. Further, assume that we use beam search with $l \geq m$ best candidates stored. If we can prove that the subgraph on m nodes closest to q is connected, then after m steps of beam search, the candidate set will contain all these nodes, including the nearest neighbor. As a result, with beam search, we may reduce the size of the neighborhood, but instead of getting stuck in a local optimum, we explore the neighborhood until we reach the target. Formally, the following theorem holds.

Theorem 4. *Let $L, R > 1$ be such constants that $L^2\left(1 - \frac{L^2}{4R^2}\right) > 1$ and let $\log\log n \ll d \ll \log n$. Assume that we use beam search with $H = \frac{C\,R^d}{\sqrt{d}}$ (for some sufficiently large C) candidates and we add $\Theta(\log n)$ long-range edges, then $G(L)$-based NNS solves the exact NN problem with probability $1 - o(1)$. The time complexity is $O\left(d \cdot R^d \cdot L^d \cdot \log R\right)$.*
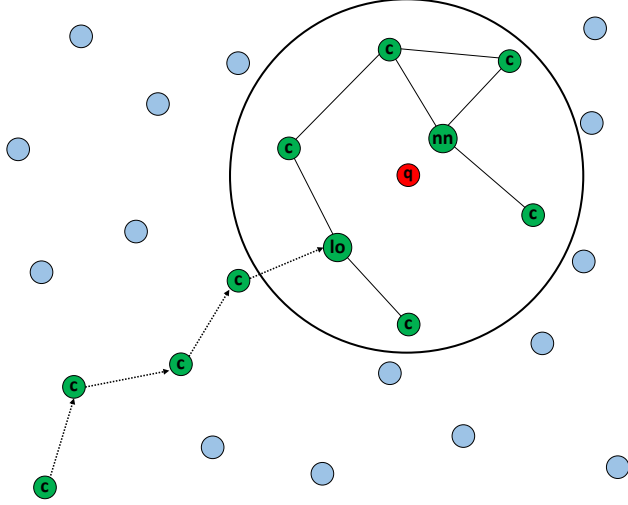
Figure 1. Example of beam search: "q" — query, "nn" — nearest neighbor, "lo" — local optimum for greedy search, "c" — other candidates visited during NNS. Beam search with 7 candidates returns 7 nearest elements to the query since the subgraph induced by them is connected.

As a result, beam search allows to significantly reduce degrees of a graph, which finally leads to time complexity reduction. We empirically illustrate this fact in Table 2 in the next section.

Theorem 4 formulates the time–space trade-off for the beam search: we can reduce L, which determines the space complexity, to any value greater than one, but for small L we have to take large R which increases the query time. The following corollary optimizes the time complexity.

Corollary 3. *In Theorem 4, we can take $L = \sqrt{\frac{3}{2}}$ and any $R > \sqrt{\frac{9}{8}}$. As a result, the main term of the time complexity can be reduced to $\left(\frac{27}{16}\right)^{d/2}$, which is less than $2^{\frac{d}{2}}$ from Theorem 1.*

5. Experiments

In this section, we illustrate the obtained results using synthetic datasets and also analyze whether our findings translate to real data. We also demonstrate how uniformization and densification may improve the quality of some graph-based NNS algorithms, which motivates the assumptions used in our analysis.

5.1. Experimental Setup

Datasets   To illustrate our theoretical results, we generated synthetic datasets uniformly distributed over d-dimensional spheres with $n = 10^6$ and $d \in \{2, 4, 8, 16\}$. We also ex-

Table 1. Real datasets

dataset	dim	# base	# queries	metric
SIFT	128	10^6	10^4	Euclidean
GIST	960	10^6	10^3	Euclidean
GloVe	300	10^6	10^4	Angular
DEEP	96	10^6	10^4	Euclidean

Table 2. KNN graphs on synthetic data

dim	steps	degree	Recall@1	Beam
2	200	20	0.998	1
4	15	60	0.999	1
8	5	300	0.998	1
16	4	2000	0.99	1
16	106	20	0.995	100

perimented with the widely used real-world datasets: SIFT and GIST (Jegou et al., 2010), GloVe (Pennington et al., 2014), and DEEP1B (Babenko & Lempitsky, 2016). These datasets are summarized in Table 1.

Measuring performance   To evaluate the algorithms, we adopt a standard approach and compare their Time-versus-Quality curves. On x-axis, instead of a frequently used Recall@1, we have error $= 1 -$ Recall@1 and use the logarithmic scale for better visualisation, since the most important region is where error is close to zero. On y-axis, for synthetic datasets, we show the number of distance calculations (the smaller the better), since we want to illustrate our theoretical results. For real datasets, we measure the number of queries per second (the larger the better). All algorithms were run on a single core of Intel(R) Core(TM) i7-6800K CPU @ 3.40GHz.

Algorithms   In the experiments, we combine and analyze the techniques discussed in this paper:

- KNN is a simple NN graph, where each node has a fixed number of neighbors;
- KL means that for each node we additionally add edges distributed according to (2), we also use pre-sampling of $\sqrt{n}$ nodes to speed up the construction procedure;
- LLF: at each iteration check *long links first* and consider local neighbors only if required;
- BEAM: use beam search instead of greedy search, always used on real data;
- DIM-RED: dimensionality reduction technique used on real datasets, as discussed in more details in Section 5.3.
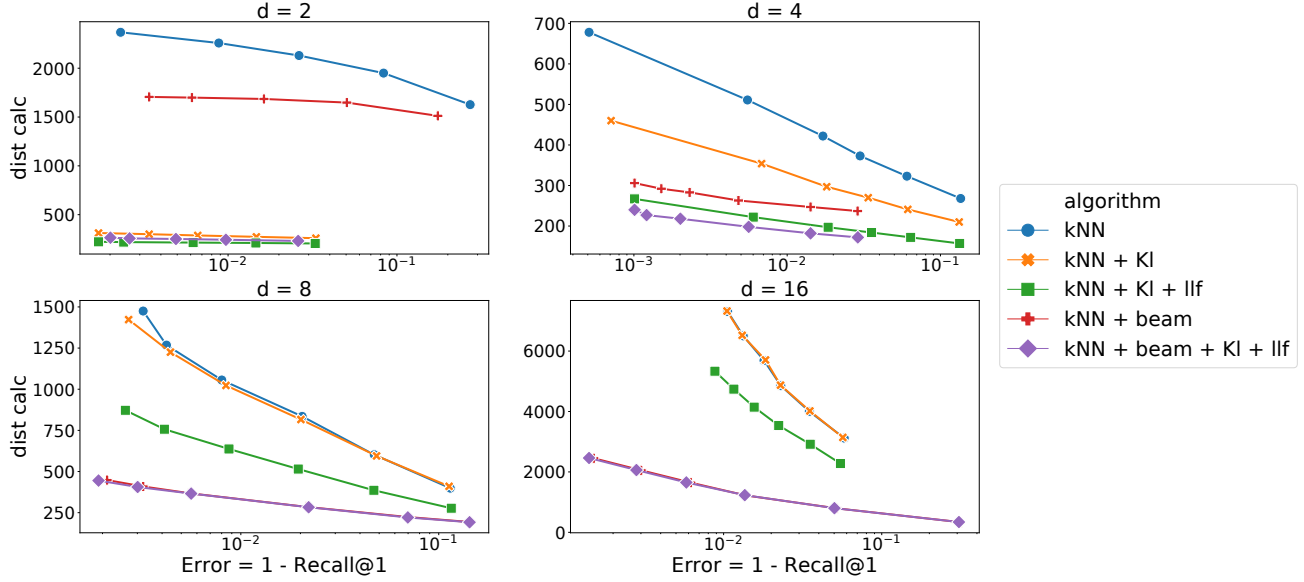
Figure 2. Results on synthetic datasets

To obtain Time-vs-Quality curves, we vary the number of local neighbors for greedy search and the number of candidates for beam search.

For reference, on real datasets, we compare all algorithms with HNSW (Malkov & Yashunin, 2018), as it is shown to provide the state-of-the-art performance on the common benchmarks and its source code is publicly available.

5.2. Synthetic Datasets

In this section, we illustrate our theoretical results from Section 4 on synthetic datasets. The source code of these experiments is publicly available.[7]

Figure 2 shows plain NN graphs (Theorems 1 and 2), the effect of adding long-range links (Theorem 3), the LLF heuristics, and the beam search (Theorem 4).

In small dimensions, the largest improvement is caused by using properly distributed long edges. However, for larger d, adding such edges does not help much, which agrees with our theoretical results: when d is large, the number of steps becomes negligible compared to the complexity of one step (see Section 4.3). Note that LLF always improves the performance, which is expected, since LLF reduces the number of distance computations for local neighbors.

In contrast, the effect of BEAM search is relatively small for $d = 2$, and it becomes substantial as d grows. This agrees with our analysis in Section 4.4: beam search helps to reduce the degrees, which are especially large for sparse

---

[7]https://github.com/Shekhale/gbnns_theory

datasets. Indeed, Theorems 1 and 2 show that graph degrees are expected to explode with d. Table 2 additionally illustrates this: we fix a sufficiently high value of Recall@1 and analyze which values of k would allow KNN to reach such performance (approximately). We see that degrees explode: while 20 is sufficient for $d = 2$, we need 2000 neighbors for $d = 16$. In contrast, the number of steps decreases with d from 200 to 4, which also agrees with our analysis. However, using the beam search with size 100 reduces the number of neighbors back to 20 while still having fewer steps compared to $d = 2$.

5.3. Real Datasets

While our theoretical results assume uniformly distributed elements, in this section we analyze their viability in real settings.

Let us first discuss a technique (DIM-RED) that exploits uniformization and densification and allows us to improve the quality of KNN significantly. This technique is inspired by our theoretical results and also by the observation that on uniform datasets for $d > 8$ plain NN graphs and HNSW have similar performance (Lin & Zhao, 2019). The basic idea is to map a given dataset to a smaller dimension and at the same time make it more uniform while trying to preserve the neighborhoods. For this purpose, we use the technique proposed by Sablayrolles et al. (2018). At the preprocessing step, we construct a search graph on the new (smaller-dimensional) dataset. This dataset is dense and close to uniform, so our theoretical results are applicable. Then, for a given query q, we obtain a lower-dimensional
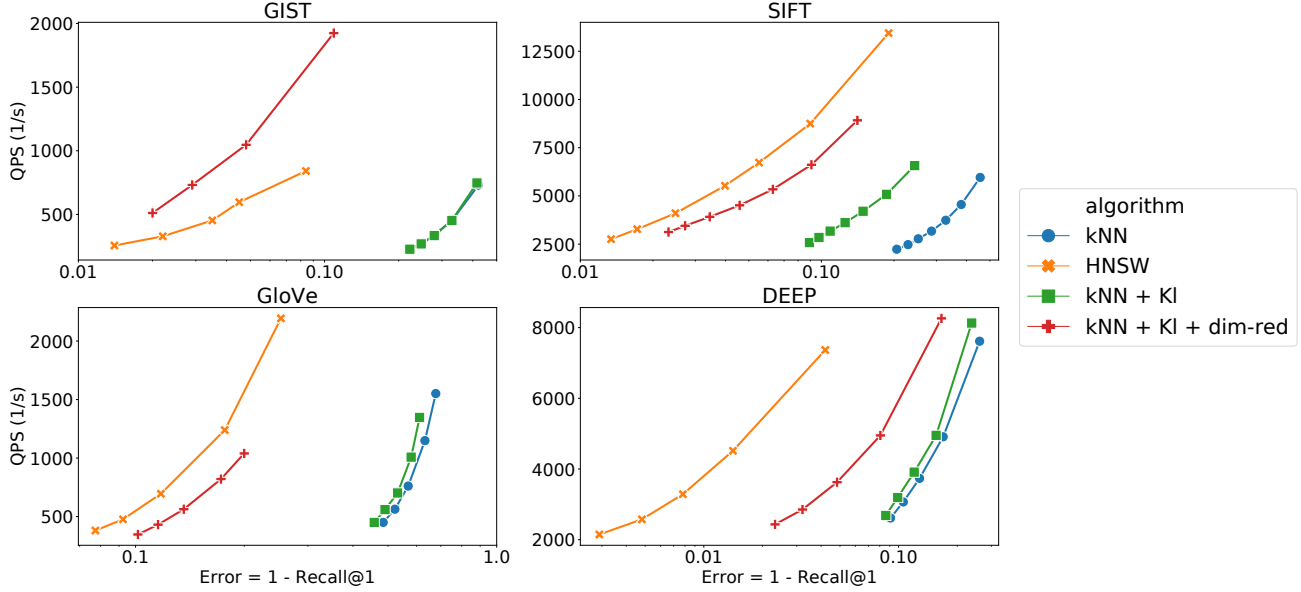
Figure 3. Results on real datasets

q' via the same transformation and then perform the beam search there. Finally, we come back to the original space and find the element closets to the original query from the beam candidates. This technique is called DIM-RED.

We compare all algorithms in Figure 3. Recall that in all cases we use BEAM search since it drastically improves the greedy algorithm. We observe that long edges significantly improve the performance of KNN (in the original space) only for the SIFT dataset. This agrees with our analysis: SIFT has the smallest intrinsic dimension (Lin & Zhao, 2019). However, combining the algorithm with DIM-RED, we get a significant speedup. For GIST, the obtained algorithm is even better than HNSW. This improvement may be caused by the fact that the original dimension is excessive, so we may reduce it without affecting the neighborhoods much. A smaller dimensionality is beneficial since it allows for faster distance computations and may also help to avoid local optima which are critical for sparse datasets.

These results confirm that analysis of uniform distributions and dense datasets is important: while these conditions are not satisfied in real datasets, they are close to being satisfied after the transformation.

The source code for reproducing these experiments is publicly available.[8] Also, in an concurrent research we show that with some additional modifications DIM-RED can be used to obtain a new state-of-the-art graph-based NNS algorithm.

---

[8] https://github.com/Shekhale/gbnns_dim_red

6. Conclusion

In this paper, we theoretically analyze the performance of graph-based NNS, specifically focusing on the dense regime $d \ll \log(n)$. We make an important step from practice to theory: in addition to plain NN graphs, we also analyze the effect of two heuristics widely used in practice: shortcut edges and beam search.

Since graph-based NNS algorithms become extremely popular nowadays, we believe that more theoretical analysis of such methods will follow. A natural direction for future research would be to find broader conditions under which similar guarantees can be obtained. In particular, in the current research, we assume uniform distribution. We supported this assumption empirically, but clearly, the analysis of more general distributions would be useful. While our results can be straightforwardly extended to distributions similar to uniform (e.g., if the density varies in some bounded interval), further generalizations seem to be tricky. Another promising direction is to analyze the effect of diversification (Harwood & Drummond, 2016), which was empirically shown to improve the quality of graph-based NNS (Lin & Zhao, 2019). However, this is reasonable to do in a more general setting, since diversification is especially important for non-uniform distributions.

Acknowledgments

References

Andoni, A. and Indyk, P. Near-optimal hashing algorithms for near neighbor problem in high dimension. *Communications of the ACM*, 51(1):117–122, 2008.

Andoni, A. and Razenshteyn, I. Optimal data-dependent hashing for approximate near neighbors. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pp. 793–801. ACM, 2015.

Andoni, A. and Razensteyn, I. Tight lower bounds for data-dependent locality-sensitive hashing. In *32nd International Symposium on Computational Geometry (SoCG 2016)*, 2016.

Andoni, A., Laarhoven, T., Razenshteyn, I., and Waingarten, E. Optimal hashing-based time-space trade-offs for approximate near neighbors. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 47–66. Society for Industrial and Applied Mathematics, 2017.

Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., and Wu, A. Y. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM (JACM)*, 45(6):891–923, 1998.

Aumüller, M., Bernhardsson, E., and Faithfull, A. ANN-benchmarks: a benchmarking tool for approximate nearest neighbor algorithms. *Information Systems*, 2019.

Babenko, A. and Lempitsky, V. Efficient indexing of billion-scale datasets of deep descriptors. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Baranchuk, D., Persiyanov, D., Sinitsin, A., and Babenko, A. Learning to route in similarity graphs. In *International Conference on Machine Learning*, pp. 475–484, 2019.

Barrière, L., Fraigniaud, P., Kranakis, E., and Krizanc, D. Efficient routing in networks with long range contacts. In *International Symposium on Distributed Computing*, pp. 270–284. Springer, 2001.

Beaumont, O., Kermarrec, A.-M., and Rivière, É. Peer to peer multidimensional overlays: approximating complex structures. In *International Conference On Principles Of Distributed Systems*, pp. 315–328. Springer, 2007.

Becker, A., Ducas, L., Gama, N., and Laarhoven, T. New directions in nearest neighbor searching with applications to lattice sieving. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pp. 10–24, 2016.

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

Beygelzimer, A., Kakade, S., and Langford, J. Cover trees for nearest neighbor. In *Proceedings of the 23rd international conference on Machine learning*, pp. 97–104, 2006.

Bishop, C. M. *Pattern recognition and machine learning*. Springer, 2006.

Bonnet, F., Kermarrec, A.-M., and Raynal, M. Small-world networks: is there a mismatch between theory and practice? 2007.

Chen, G. H. and Shah, D. Explaining the success of nearest neighbor methods in prediction. *Foundations and Trends® in Machine Learning*, 10(5-6):337–588, 2018.

Dasgupta, S. and Freund, Y. Random projection trees and low dimensional manifolds. In *STOC*, volume 8, pp. 537–546. Citeseer, 2008.

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

Datar, M., Immorlica, N., Indyk, P., and Mirrokni, V. S. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry*, pp. 253–262. ACM, 2004.

Dong, W., Moses, C., and Li, K. Efficient k-nearest neighbor graph construction for generic similarity measures. In *Proceedings of the 20th international conference on World wide web*, pp. 577–586. ACM, 2011.

Duchon, P., Hanusse, N., Lebhar, E., and Schabanel, N. Could any graph be turned into a small-world? *Theoretical Computer Science*, 355(1):96–103, 2006.

Fu, C., Xiang, C., Wang, C., and Cai, D. Fast approximate nearest neighbor search with the navigating spreading-out graph. *Proceedings of the VLDB Endowment*, 12(5):461–474, 2019.

Hajebi, K., Abbasi-Yadkori, Y., Shahbazi, H., and Zhang, H. Fast approximate nearest-neighbor search with k-nearest neighbor graph. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

Har-Peled, S., Indyk, P., and Motwani, R. Approximate nearest neighbor: Towards removing the curse of dimensionality. *Theory of computing*, 8(1):321–350, 2012.

Harwood, B. and Drummond, T. Fanng: Fast approximate nearest neighbour graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5713–5722, 2016.

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. ACM, 1998.

Iwasaki, M. and Miyazaki, D. Optimization of indexing based on k-nearest neighbor graph for proximity search in high-dimensional data. *arXiv preprint arXiv:1810.07355*, 2018.

Jegou, H., Douze, M., and Schmid, C. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):117–128, 2010.

Karbasi, A., Ioannidis, S., and Massoulié, L. From small-world networks to comparison-based search. *IEEE Transactions on Information Theory*, 61(6):3056–3074, 2015.

Keivani, O. and Sinha, K. Improved nearest neighbor search using auxiliary information and priority functions. In *International Conference on Machine Learning*, pp. 2578–2586, 2018.

Kleinberg, J. The small-world phenomenon: an algorithmic perspective. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pp. 163–170, 2000.

Laarhoven, T. Graph-based time-space trade-offs for approximate near neighbors. In *34th International Symposium on Computational Geometry (SoCG 2018)*, 2018.

Lin, P.-C. and Zhao, W.-L. Graph based nearest neighbor search: promises and failures. *arXiv. org*, 2019.

Malkov, Y., Ponomarenko, A., Logvinov, A., and Krylov, V. Approximate nearest neighbor algorithm based on navigable small world graphs. *Information Systems*, 45: 61–68, 2014.

Malkov, Y. A. and Yashunin, D. A. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

May, A. and Ozerov, I. On computing nearest neighbors with applications to decoding of binary linear codes. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 203–228. Springer, 2015.

Meiser, S. Point location in arrangements of hyperplanes. *Information and Computation*, 106(2):286–303, 1993.

Motwani, R., Naor, A., and Panigrahy, R. Lower bounds on locality sensitive hashing. *SIAM Journal on Discrete Mathematics*, 21(4):930, 2007.

Navarro, G. Searching in metric spaces by spatial approximation. *The VLDB Journal*, 11(1):28–46, 2002.

O'Donnell, R., Wu, Y., and Zhou, Y. Optimal lower bounds for locality-sensitive hashing (except when q is tiny). *ACM Transactions on Computation Theory (TOCT)*, 6 (1):5, 2014.

Pennington, J., Socher, R., and Manning, C. D. GloVe: global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

Sablayrolles, A., Douze, M., Schmid, C., and Jégou, H. Spreading vectors for similarity search. In *International Conference on Learning Representations*, 2018.

Shakhnarovich, G., Darrell, T., and Indyk, P. *Nearest-neighbor methods in learning and vision: theory and practice (neural information processing)*. The MIT press, 2006.

Valiant, G. Finding correlations in subquadratic time, with applications to learning parities and the closest pair problem. *Journal of the ACM (JACM)*, 62(2):13, 2015.

Wang, J., Wang, J., Zeng, G., Tu, Z., Gan, R., and Li, S. Scalable k-NN graph construction for visual descriptors. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1106–1113, 2012.

Watts, D. J. and Strogatz, S. H. Collective dynamics of small-worldnetworks. *Nature*, 393:440442, 1998.

Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A., Liu, B., Philip, S. Y., et al. Top 10 algorithms in data mining. *Knowledge and information systems*, 14(1):1–37, 2008.

Supplementary Materials

A   Analysis of spherical caps

In this section, we formulate some technical results on the volumes of spherical caps and their intersections, which we extensively use in the proofs. Although they are similar to those formulated in [1], it is crucial for our problem that parameters defining spherical caps depend on d and may tend to zero (both in dense and sparse regimes), while the results in [1] hold only for fixed parameters. Also, in Lemma 2, we extend the corresponding result from [1], as discussed further in this section.

A.1   Volumes of spherical caps

Let us denote by μ the Lebesgue measure over $\mathbb{R}^{d+1}$. By $C_{\mathbf{x}}(\gamma)$ we denote a spherical cap of height γ centered at $\mathbf{x} \in \mathcal{S}^d$, i.e., $\{\mathbf{y} \in \mathcal{S}^d : \langle \mathbf{x}, \mathbf{y} \rangle \geq \gamma\}$; $C(\gamma) = \mu\left(C_{\mathbf{x}}(\gamma)\right)$ denotes the volume of a spherical cap of height γ. Recall that throughout the paper for any variable γ, $0 \leq \gamma \leq 1$, we let $\hat{\gamma} := \sqrt{1 - \gamma^2}$. We prove the following lemma.

Lemma 1. *Let $\gamma = \gamma(d)$ be such that $0 \leq \gamma \leq 1$. Then*

$$\Theta\left(d^{-1/2}\right) \hat{\gamma}^d \leq C(\gamma) \leq \Theta\left(d^{-1/2}\right) \hat{\gamma}^d \min\left\{d^{1/2}, \frac{1}{\gamma}\right\} .$$

Proof. In order to have similar reasoning with the proof of Lemma 2, we consider any two-dimensional plane containing the vector $\mathbf{x}$ defining the cap $C_{\mathbf{x}}(\gamma)$ and let p denote the orthogonal projection from $\mathcal{S}^d$ to this two-dimensional plane.

The first steps of the proof are similar to those in [1] (but note that we analyze $\mathcal{S}^d$ instead of S^{d-1}, which leads to slightly simpler expressions). Consider any measurable subset U of the two-dimensional unit ball, then the volume of the preimage $p^{-1}(U)$ (relative to the volume of $\mathcal{S}^d$) is:

$$I(U) = \int_{r, \phi \in U} \frac{\mu(S^{d-2})}{\mu(S^d)} \left(\sqrt{1 - r^2}\right)^{d-3} r \, dr \, d\phi .$$

We define a function $g(r) = \int_{\phi : (r, \phi) \in U} d\phi$, then we can rewrite the integral as

$$I(U) = \frac{(d-1)}{4\pi} \int_0^1 \left(1 - r^2\right)^{(d-3)/2} g(r) \, dr^2 .$$

Let $U = p\left(C_{\mathbf{x}}(\gamma)\right)$, then, using $t = \left(1 - r^2\right)/\hat{\gamma}^2$, where $\hat{\gamma} = \sqrt{1 - \gamma^2}$, we get

$$C(\gamma) = \frac{(d-1)}{4\pi} \int_\gamma^1 \left(1 - r^2\right)^{(d-3)/2} g(r) \, dr^2 = \frac{(d-1)\,\hat{\gamma}^{d-1}}{4\pi} \int_0^1 g\left(\sqrt{1 - \hat{\gamma}^2 t}\right) t^{(d-3)/2} \, dt . \tag{1}$$

Note that from Equation (1) we get thvolume of a hemisphere is $C(0) = 1/2$, since $g(r) = \pi$ for all r in this case and $\hat{\gamma} = 1$.

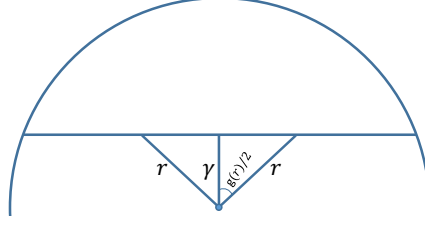Now we consider an arbitrary $\gamma \geq 0$ and note that $g(r) = 2\arccos(\gamma/r)$ (see Figure 1). So, we obtain

Figure 1: $g(r)$

$$C(\gamma) = \frac{(d-1)\,\hat{\gamma}^{d-1}}{2\,\pi} \int_0^1 \arccos\left(\frac{\gamma}{\sqrt{1-\hat{\gamma}^2 t}}\right) t^{\frac{d-3}{2}}\, dt$$

$$= \frac{(d-1)\,\hat{\gamma}^{d-1}}{2\,\pi} \int_0^1 \arcsin\left(\hat{\gamma}\sqrt{\frac{1-t}{1-\hat{\gamma}^2 t}}\right) t^{(d-3)/2}\, dt\,.$$

Now we note that $x \leq \arcsin(x) \leq x \cdot \pi/2$ for $0 \leq x \leq 1$, so

$$C(\gamma) = \Theta\,(d)\,\hat{\gamma}^d \int_0^1 \sqrt{\frac{1-t}{1-\hat{\gamma}^2 t}} \cdot t^{(d-3)/2}\, dt\,.$$

Finally, we estimate

$$\sqrt{1-t} \leq \sqrt{\frac{1-t}{1-\hat{\gamma}^2 t}} \leq \min\left\{1, \sqrt{\frac{1-t}{1-\hat{\gamma}^2}}\right\}. \tag{2}$$

So, the lower bound is

$$C(\gamma) \geq \Theta\,(d)\,\hat{\gamma}^d\,\mathrm{B}\left(\frac{3}{2}, \frac{d-1}{2}\right) = \Theta\,(d)\,\hat{\gamma}^d \left(\frac{d-1}{2}\right)^{-3/2} = \Theta\left(d^{-1/2}\right)\hat{\gamma}^d\,.$$

The upper bounds are

$$C(\gamma) \leq \Theta\,(d)\,\hat{\gamma}^d \int_0^1 t^{(d-3)/2}\, dt = \Theta\,(1)\,\hat{\gamma}^d\,,$$

$$C(\gamma) \leq \Theta\,(d)\,\hat{\gamma}^d \int_0^1 \sqrt{\frac{1-t}{1-\hat{\gamma}^2}} \cdot t^{(d-3)/2}\, dt = \Theta\left(d^{-1/2}\right)\frac{\hat{\gamma}^d}{\gamma}\,.$$

This completes the proof. □

A.2 Volumes of intersections of spherical caps

By $W_{\mathbf{x},\mathbf{y}}(\alpha, \beta)$ we denote the intersection of two spherical caps centered at $\mathbf{x} \in \mathcal{S}^d$ and $\mathbf{y} \in \mathcal{S}^d$ with heights α and β, respectively, i.e., $W_{\mathbf{x},\mathbf{y}}(\alpha, \beta) = \{\mathbf{z} \in \mathcal{S}^d : \langle \mathbf{z}, \mathbf{x}\rangle \geq \alpha, \langle \mathbf{z}, \mathbf{y}\rangle \geq \beta\}$. As for spherical caps, by $W(\alpha, \beta, \theta)$ we denote the volume of such intersection given that the angle between $\mathbf{x}$ and $\mathbf{y}$ is θ.

We analyze the volume of the intersection of two spherical caps $C_{\mathbf{x}}(\alpha)$ and $C_{\mathbf{y}}(\beta)$. In the lemma below we assume $\gamma \leq 1$. However, it is clear that if $\gamma > 1$, then either the caps do not intersect (if $\alpha > \beta \cos\theta$ and $\beta > \alpha \cos\theta$) or the larger cap contains the smaller one.

Lemma 2. *Let* $\gamma = \frac{\sqrt{\alpha^2 + \beta^2 - 2\alpha\beta\cos\theta}}{\sin\theta}$ *and assume that* $\gamma \leq 1$*, then:*

(1) If $\alpha \leq \beta\cos\theta$*, then* $C(\beta)/2 < W(\alpha, \beta, \theta) \leq C(\beta)$ *and*

$$C(\beta) - W(\alpha, \beta, \theta) \leq C_{u,\beta}\,\Theta\left(d^{-1}\right)\hat{\gamma}^d \min\left\{d^{1/2}, \frac{1}{\gamma}\right\};$$

2

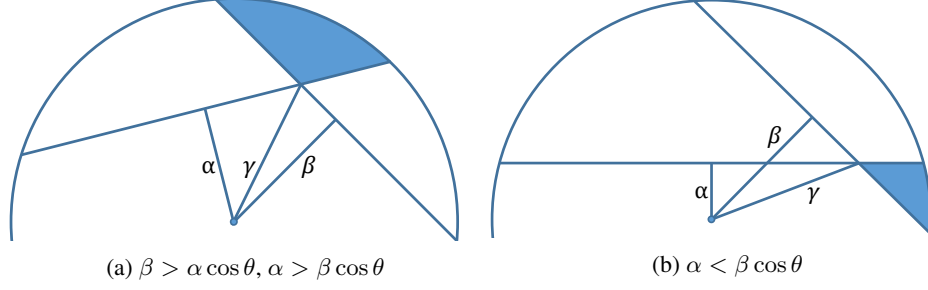(a) $\beta > \alpha \cos \theta, \ \alpha > \beta \cos \theta$ (b) $\alpha < \beta \cos \theta$

Figure 2: Intersection of spherical caps

(2) If $\beta \leq \alpha \cos \theta$, then $C(\alpha)/2 < W(\alpha, \beta, \theta) \leq C(\alpha)$ and

$$C(\alpha) - W(\alpha, \beta, \theta) \leq C_{u,\alpha} \, \Theta \left(d^{-1} \right) \hat{\gamma}^d \min \left\{ d^{1/2}, \frac{1}{\gamma} \right\} ;$$

(3) Otherwise,

$$(C_{l,\alpha} + C_{l,\beta}) \, \Theta \left(d^{-1} \right) \hat{\gamma}^d \leq W(\alpha, \beta, \theta) \leq (C_{u,\alpha} + C_{u,\beta}) \, \Theta \left(d^{-1} \right) \hat{\gamma}^d \min \left\{ d^{1/2}, \frac{1}{\gamma} \right\} ;$$

where

$$C_{l,\alpha} = \frac{\alpha \left(\hat{\alpha} \sin \theta - |\beta - \alpha \cos \theta| \right)}{\gamma \, \hat{\gamma} \, \sin \theta}, \ \ C_{l,\beta} = \frac{\beta \left(\hat{\beta} \sin \theta - |\alpha - \beta \cos \theta| \right)}{\gamma \, \hat{\gamma} \, \sin \theta},$$

$$C_{u,\alpha} = \frac{\hat{\gamma} \, \alpha \, \sin \theta}{\gamma |\beta - \alpha \cos \theta|}, \ \ C_{u,\beta} = \frac{\hat{\gamma} \, \beta \, \sin \theta}{\gamma |\alpha - \beta \cos \theta|}.$$

The cases considered in this lemma are illustrated in Figure 2.

This lemma differs from Lemma 2.2 in [1] by, first, allowing the parameters α, β, θ depend on d and, second, considering the cases (1) and (2), which are essential for the proofs. Namely, we use the lower bound in (3) to show that with high probability we can make a step of the algorithm since the intersection of some spherical caps is large enough (Figure 2a); we use the upper bounds in (1) and (2) to show that at the final step of the algorithm we can find the nearest neighbor with high probability, since the volume of the intersection of some spherical caps is very close to the volume of one of them (Figure 2b), see the details further in the proof.

Proof. Consider the plane formed by the the vectors $\mathbf{x}$ and $\mathbf{y}$ defining the caps and let p denote the orthogonal projection to this plane. Let $U = p(W_{\mathbf{x},\mathbf{y}}(\alpha, \beta))$.

Denote by γ the distance between the origin and the intersection of chords bounding the projections of spherical caps. One can show that

$$\gamma = \sqrt{\frac{\alpha^2 + \beta^2 - 2\alpha\beta \cos \theta}{\sin^2 \theta}} .$$

If $\alpha \leq \beta \cos \theta$, it is easy to see that $W(\alpha, \beta, \theta) > \frac{1}{2} C(\beta)$, since more than a half of $C_{\mathbf{y}}(\beta)$ is covered by the intersection (see Figure 2b). Similarly, if $\beta \leq \alpha \cos \theta$, then $W(\alpha, \beta, \theta) > \frac{1}{2} C(\alpha)$. Now we move to the proof of (3) and will return to (1) and (2) after that.

If $\cos \theta < \frac{\alpha}{\beta}$ and $\cos \theta < \frac{\beta}{\alpha}$, then we are in the situation shown on Figure 2a and the distance between the intersection of spherical caps and the origin is γ. As in the proof of Lemma 1, denote $g(r) = \int_{\phi:(r,\phi)\in U} d\phi$, then the relative volume of $p^{-1}(U)$ is (see Equation (1)):

$$W(\alpha, \beta, \theta) = \frac{(d-1)\, \hat{\gamma}^{d-1}}{4\,\pi} \int_0^1 g \left(\sqrt{1 - \hat{\gamma}^2 t} \right) t^{\frac{d-3}{2}} \, dt .$$

3

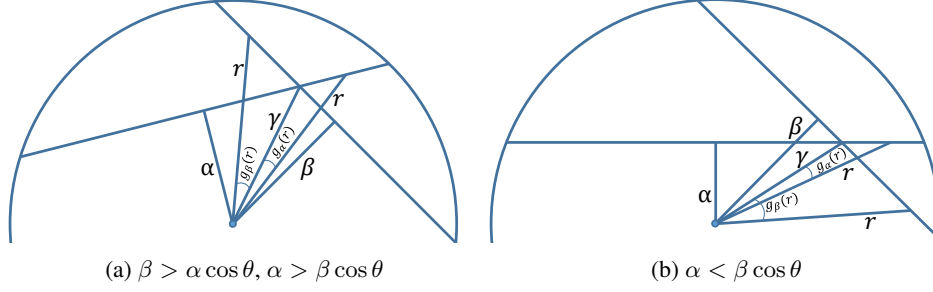(a) $\beta > \alpha \cos\theta$, $\alpha > \beta \cos\theta$ (b) $\alpha < \beta \cos\theta$

Figure 3: $g_\alpha(r)$ and $g_\beta(r)$

The function $g(r)$ can be written as $g_\alpha(r) + g_\beta(r)$, where (see Figure 3a)

$$g_\alpha(r) = \arccos\left(\frac{\alpha}{r}\right) - \arccos\left(\frac{\alpha}{\gamma}\right),$$

$$g_\beta(r) = \arccos\left(\frac{\beta}{r}\right) - \arccos\left(\frac{\beta}{\gamma}\right).$$

Accordingly, we can write $W(\alpha, \beta, \theta) = W_\alpha(\alpha, \beta, \theta) + W_\beta(\alpha, \beta, \theta)$.

Let us estimate $g_\alpha\left(\sqrt{1 - \hat{\gamma}^2 t}\right)$:

$$g_\alpha\left(\sqrt{1 - \hat{\gamma}^2 t}\right) = \arcsin\left(\sqrt{1 - \frac{\alpha^2}{1 - \hat{\gamma}^2 t}}\right) - \arcsin\left(\sqrt{1 - \frac{\alpha^2}{\gamma^2}}\right)$$

$$= \arcsin\left(\sqrt{\left(1 - \frac{\alpha^2}{1 - \hat{\gamma}^2 t}\right)\frac{\alpha^2}{\gamma^2}}\sqrt{\left(1 - \frac{\alpha^2}{\gamma^2}\right)\frac{\alpha^2}{1 - \hat{\gamma}^2 t}}\right)$$

$$= \Theta\left(\frac{\alpha\sqrt{\gamma^2 - \alpha^2}}{\gamma\sqrt{1 - \hat{\gamma}^2 t}}\right)\left(\sqrt{1 + \frac{\hat{\gamma}^2(1 - t)}{\gamma^2 - \alpha^2}} - 1\right).$$

Note that

$$\left(\sqrt{1 + \frac{\hat{\gamma}^2}{\gamma^2 - \alpha^2}} - 1\right)(1 - t) \le \sqrt{1 + \frac{\hat{\gamma}^2(1 - t)}{\gamma^2 - \alpha^2}} - 1 \le \frac{\hat{\gamma}^2}{2(\gamma^2 - \alpha^2)}(1 - t).$$

Now, we can write the lower bound for $W_\alpha(\alpha, \beta, \theta)$. Let

$$C_{l,\alpha} = \left(\sqrt{1 + \frac{\hat{\gamma}^2}{\gamma^2 - \alpha^2}} - 1\right)\frac{\alpha\sqrt{\gamma^2 - \alpha^2}}{\gamma\,\hat{\gamma}} = \frac{\alpha\left(\hat{\alpha}\sin\theta - |\beta - \alpha\cos\theta|\right)}{\gamma\,\hat{\gamma}\,\sin\theta},$$

$C_{l,\beta}$ can be obtained by swapping α and β.

Then the lower bound is

$$W(\alpha, \beta, \theta) \ge \Theta(d)\,\hat{\gamma}^d\,(C_{l,\alpha} + C_{l,\beta})\int_0^1 \frac{1 - t}{\sqrt{1 - \hat{\gamma}^2 t}}t^{(d-3)/2}\,dt$$

$$\ge \Theta(d)\,\hat{\gamma}^d\,(C_{l,\alpha} + C_{l,\beta})\int_0^1 (1 - t)\,t^{(d-3)/2}\,dt = \Theta(d^{-1})\,\hat{\gamma}^d\,(C_{l,\alpha} + C_{l,\beta}).$$

Now we define $C_{u,\alpha}$ (and, similarly, $C_{u,\beta}$) as

$$C_{u,\alpha} = \frac{\hat{\gamma}}{(\gamma^2 - \alpha^2)}\cdot\frac{\alpha\sqrt{\gamma^2 - \alpha^2}}{\gamma} = \frac{\hat{\gamma}\,\alpha\,\sin\theta}{\gamma\,|\beta - \alpha\cos\theta|}.$$

4

Then

$$W(\alpha, \beta, \theta) \leq \Theta(d) \, \hat{\gamma}^d \, (C_{u,\alpha} + C_{u,\beta}) \cdot \int_0^1 \frac{1-t}{\sqrt{1-\hat{\gamma}^2 t}} t^{(d-3)/2} \, dt \, .$$

We use the upper bound

$$\frac{1-t}{\sqrt{1-\hat{\gamma}^2 t}} \leq \min \left\{ \sqrt{1-t}, \frac{1-t}{\gamma} \right\}$$

and obtain

$$W(\alpha, \beta, \theta) \leq \Theta(d^{-1}) \, \hat{\gamma}^d \, (C_{u,\alpha} + C_{u,\beta}) \min \left\{ d^{\frac{1}{2}}, \frac{1}{\gamma} \right\} \, ,$$

which completes the proof of (3).

Now, let us finish the proof for (1) and (2). If $\alpha \leq \beta \cos \theta$, then we are in a situation shown on Figure 2b. In this case, the bounds on $W(\alpha, \beta, \theta)$ are obvious. To estimate $C(\beta) - W(\alpha, \beta, \theta)$, we can directly follow the above proof for (3) and the only difference would be that $g(r) = g_\beta(r) - g_\alpha(r)$ instead of $g(r) = g_\alpha(r) + g_\beta(r)$. Note that we need only the upper bound and we simply say that $g(r) \leq g_\beta(r)$. The proof for (2) is similar with $g(r) = g_\alpha(r) - g_\beta(r)$. $\qquad\square$

B Greedy search on plain NN graphs

B.1 Proof overview

Let α_M denote the height of a spherical cap defining $G(M)$. By $f = f(n) = (n-1)C(\alpha_M)$ we denote the expected number of neighbors of a given node in $G(M)$. Then, it is clear that the complexity of one step of graph-based search is $\Theta(f \cdot d)$ (with high probability), so for making k steps we need $\Theta(k \cdot f \cdot d)$ computations (see Section B.2.1). The number of edges in the graph is $\Theta(f \cdot n)$, so the space complexity is $\Theta(f \cdot n \cdot \log n)$ (see Section B.2.2).

To prove that the algorithm succeeds, we have to show that it does not get stuck in a local optimum until we are sufficiently close to q. If we take some point $\mathbf{x}$ with $\langle \mathbf{x}, q \rangle = \alpha_s$, then the probability of making a step towards q is determined by $W(\alpha_M, \alpha_s, \arccos \alpha_s)$. In all further proofs we obtain lower bounds for this value of the form $\frac{1}{n} g(n)$ with $1 \ll g(n) \ll n$. From this, we easily get that the probability of making a step is at least $1 - (1 - g(n)/n)^{n-1} = 1 - e^{-g(n)(1+o(1))}$.

A fact that will be useful in the proofs is that the value $W(\alpha_M, \alpha_s, \arccos \alpha_s)$ is a monotone function of α_s (see Section B.2.3). I.e., if we have a lower bound for some α_s, then for all smaller values we have this bound automatically.

By estimating the value $W(\alpha_M, \alpha_s, \arccos \alpha_s)$, we obtain (in further sections) that with probability $1 - o(1)$ we reach some point at distance at most $\arccos \alpha_s$ from q. Then, to achieve success, we may either jump directly to $\bar{\mathbf{x}}$ at the next step or to already have $\arccos \alpha_s \leq cR$ if we are solving c, R-ANN.

To limit the number of steps, we additionally show that with a sufficiently large probability at each step we become "ε closer" to q. In the dense regime, it means that the sine of the angle between the current position and q becomes smaller by at least some fixed value.

Finally, let us emphasize that several consecutive steps of the algorithm cannot be analyzed independently. Indeed, if at some step we moved from $\mathbf{x}$ to $\mathbf{y}$, then there were no points in $C_{\mathbf{x}}(\alpha_M)$ closer to q than $\mathbf{y}$ by the definition of the algorithm. Consequently, the intersection of $C_{\mathbf{x}}(\alpha_M)$, $C_{\mathbf{y}}(\alpha_M)$ and $C_q(\langle q, \mathbf{y} \rangle)$ contains no elements of the dataset. The closer $\mathbf{y}$ to $\mathbf{x}$ the larger this intersection. However, the fact that at each step we become at least "ε closer" to q allows us to bound the volume of this intersection and to prove that it can be neglected.

B.2 Auxiliary results

B.2.1 Time complexity

Let v be an arbitrary node of G and let $N(v)$ denote the number of its neighbors in G. Recall that $f = (n-1)C(\alpha_M)$.

5

Lemma 3. *With probability at lest $1 - \frac{4}{f}$ we have $\frac{1}{2}f \le N(v) \le \frac{3}{2}f$.*

Proof. The number of neighbors $N(v)$ of a node v follows Binomial distribution $\mathrm{Bin}(n-1, C(\alpha_M))$, so $\mathbb{E}N(v) = f$. From Chebyshev's inequality we get

$$\mathrm{P}\left(|N(v) - f| > \frac{f}{2}\right) \le \frac{4\,\mathrm{Var}(N(v))}{f^2} \le \frac{4}{f},$$

which completes the proof. $\qquad\square$

To obtain the final time complexity of graph-based NN search, we have to sum up the complexities of all steps of the algorithm. We obtain the following result.

Lemma 4. *If we made k steps of the graph-based NNS, then with probability $1 - O\left(\frac{1}{kf}\right)$ the obtained time complexity is $\Theta\left(kfd\right)$.*

Proof. Although the nodes encountered in one iteration are not independent, the fact that we do not need to measure the distance from any point to q more than once allows us to upper bound the complexity by the random variable distributed according to $\mathrm{Bin}(k(n-1), C(\alpha_M))$. Then, we can follow the proof of Lemma 3 and note that one distance computation takes $\Theta(d)$.

To see that the lower bound is also $\Theta\left(kfd\right)$, we note that more than a constant number of steps are needed only for the dense regime. For this regime, we may follow the reasoning of Lemma 10 to show that the volume of the intersection of two consecutive balls is negligible compared to the volume of each of them. $\qquad\square$

B.2.2 Space complexity

Lemma 5. *With probability $1 - O\left(\frac{1}{fn}\right)$ we have $\frac{1}{4}fn \le E(G) \le \frac{3}{4}fn$.*

Proof. The proof is straightforward.[1] For each pair of nodes, the probability that there is an edge between them equals $C\left(\alpha_M\right)$. Therefore, the expected number of edges is

$$\mathbb{E}(E(G)) = \binom{n}{2}C\left(\alpha_M\right).$$

It remains to prove that $E(G)$ is tightly concentrated near its expectation. For this, we apply Chebyshev's inequality, so we have to estimate the variance $\mathrm{Var}(E(G))$. One can easily see that if we are given two pairs of nodes e_1 and e_2, then, if they are not the same (while one coincident node is allowed), then $\mathrm{P}(e_1, e_2 \in E(G)) = C(\alpha_M)^2$. Therefore,

$$\mathrm{Var}(E(G)) = \sum_{e_1, e_2 \in \binom{\mathcal{D}}{2}} \mathrm{P}(e_1, e_2 \in E(G)) - (\mathbb{E}E(G))^2$$

$$= \sum_{\substack{e_1, e_2 \in \binom{\mathcal{D}}{2} \\ e_1 \ne e_2}} \mathrm{P}(e_1, e_2 \in E(G)) + \mathbb{E}E(G) - (\mathbb{E}E(G))^2 = \binom{n}{2}C\left(\alpha_M\right)\left(1 - C\left(\alpha_M\right)\right).$$

Applying Chebyshev's inequality, we get

$$\mathrm{P}\left(|E(G) - \mathbb{E}(E(G))| > \frac{E(G)}{2}\right) \le \frac{4\,\mathrm{Var}(E(G))}{E(G)^2} = \frac{4\left(1 - C(\alpha)\right)}{E(G)}. \qquad (3)$$

From this, the lemma follows. $\qquad\square$

It remains to note that if we store a graph as the adjacency lists, then the space complexity is $\Theta\left(E(G) \cdot \log n\right)$.

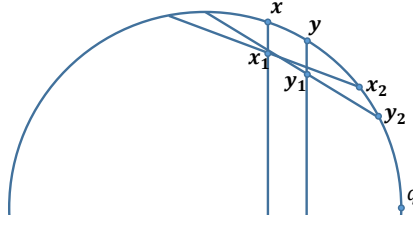[1]Similar proof appeared in, e.g., [4].

Figure 4: Monotonicity of $W(\alpha_M, \alpha_s, \arccos \alpha_s)$

B.2.3 Monotonicity of $W(\alpha_M, \alpha_s, \arccos \alpha_s)$

Lemma 6. $W(\alpha_M, \alpha_s, \arccos \alpha_s)$ *is a non-increasing function of* α_s.

Proof. We refer to Figure 4, where two spherical caps of height α_M are centered at $\mathbf{x}$ and $\mathbf{y}$, respectively, and note that we have to compare "curved triangles" $\triangle \mathbf{x} \mathbf{x}_1 \mathbf{x}_2$ and $\triangle \mathbf{y} \mathbf{y}_1 \mathbf{y}_2$. Obviously, $\rho(\mathbf{x}, \mathbf{x}_2) = \rho(\mathbf{y}, \mathbf{y}_2)$, $\angle \mathbf{x} \mathbf{x}_2 \mathbf{x}_1 = \angle \mathbf{y} \mathbf{y}_2 \mathbf{y}_1$, but $\angle \mathbf{x} \mathbf{x}_1 \mathbf{x}_2 < \angle \mathbf{y} \mathbf{y}_1 \mathbf{y}_2$. From this and the spherical symmetry of $\mu(p^{-1}(\cdot))$ (p was defined in the proof of Lemma 2) the result follows. $\qquad\square$

B.3 Proof of Theorem 1 (greedy search in dense regime)

Recall that for dense datasets ($d = \log n / \omega$), it is convenient to operate with radii of spherical caps (if α is a height of a spherical cap, then we say that $\hat{\alpha}$ is its radius). Let $\hat{\alpha}_1$ be the radius of a cap centered at a given point and covering its nearest neighbor, then we have $C(\alpha_1) \sim \frac{1}{n}$, i.e., $\hat{\alpha}_1 \sim n^{-\frac{1}{d}} = 2^{-\omega}$. We further let $\delta := 2^{-\omega}$.

We construct a graph $G(M)$ using spherical caps with radius $\hat{\alpha}_M = M \delta$. Then, from Lemma 1, we get $f = \Theta\left(n \, d^{-1/2} M^d \delta^d\right) = \Theta\left(d^{-1/2} M^d\right)$. So, the number of edges in $G(M)$ is $\Theta\left(d^{-1/2} \cdot M^d \cdot n\right)$ and the space complexity is $\Theta\left(d^{-1/2} \cdot M^d \cdot n \cdot \log n\right)$ (see Section B.2.2).

Let us now analyze the distance $\arccos \alpha_s$ up to which we can make steps towards the query q (with sufficiently large probability). This is stated in Lemma 9, but before that let us prove some auxiliary results.

First, let us analyze the behavior of the main term $\hat{\gamma}^d$ in $W(x, y, \arccos z)$ when $\hat{x}, \hat{y}, \hat{z} = o(1)$, which is the case for the considered situations in the dense regime.

Lemma 7. *If* $\hat{x}, \hat{y}, \hat{z} = o(1)$, *then* $\hat{\gamma}$ *defined in Lemma 2 for* $W(x, y, \arccos z)$ *is*

$$\hat{\gamma} \sim \frac{\sqrt{2\left(\hat{x}^2 \hat{y}^2 + \hat{y}^2 \hat{z}^2 + \hat{x}^2 \hat{z}^2\right) - \left(\hat{x}^4 + \hat{y}^4 + \hat{z}^4\right)}}{2\hat{z}}.$$

Proof. By the definition, $\gamma = \sqrt{\frac{x^2 + y^2 - 2xyz}{1 - z^2}}$. Then

$$\hat{\gamma}^2 = 1 - \gamma^2 = \frac{\hat{x}^2 + \hat{y}^2 + \hat{z}^2 - 2 + 2\sqrt{\left(1 - \hat{x}^2\right)\left(1 - \hat{y}^2\right)\left(1 - \hat{z}^2\right)}}{\hat{z}^2}$$

$$\sim \frac{2\left(\hat{x}^2 \hat{y}^2 + \hat{y}^2 \hat{z}^2 + \hat{x}^2 \hat{z}^2\right) - \left(\hat{x}^4 + \hat{y}^4 + \hat{z}^4\right)}{4\hat{z}^2}.$$

$\qquad\square$

Now, we analyze $W(\alpha_M, \alpha_s, \arccos \alpha_s)$ and we need only the lower bound. Recall that we use the notation $\delta = 2^{-\omega}$.

Lemma 8. *Assume that* $\hat{\alpha}_s = s \delta$ *and* $\hat{\alpha}_M = M \delta$.

- *If* $M \geq \sqrt{2}s$, *then* $W(\alpha_M, \alpha_s, \arccos \alpha_s) \geq \frac{1}{n} \cdot s^{d + o(d)}$.

7

- *If $M < \sqrt{2}s$, then $W(\alpha_M, \alpha_s, \arccos \alpha_s) \geq \frac{1}{n} \cdot \left(M^2 - \frac{M^4}{4s^2}\right)^{d/2+o(d)}$.*

Proof. First, assume that $M \geq \sqrt{2}s$. In this case we have $\alpha_M < \alpha_s^2$, so we are under the conditions (1)-(2) of Lemma 2 (see Figure 2b) and, using Lemma 1, we get $W(\alpha_M, \alpha_s, \arccos \alpha_s) > \frac{1}{2}C(\alpha_s) = \Theta\left(d^{-1/2}s^d\delta^d\right) = \frac{1}{n}s^{d+o(d)}$.

If $M < \sqrt{2}s$, then, asymptotically, we have $\alpha_M > \alpha_s^2$, so the case (3) of Lemma 2 can be applied. Let us use Lemma 7 to estimate $\hat{\gamma}$:

$$\hat{\gamma}^2 = \delta^2 \left(M^2 - \frac{M^4}{4s^2}\right)(1 + o(1)) \ .$$

And now from Lemma 2 we get

$$W(\alpha_M, \alpha_s, \arccos \alpha_s) \geq C_l \Theta\left(d^{-1}\right)\hat{\gamma}^d \ ,$$

where C_l corresponds to the sum of $C_{l,\alpha}$ and $C_{l,\beta}$ in Lemma 2. So, it remains to estimate C_l:

$$C_l = \frac{\alpha_M \left(\hat{\alpha}_M \hat{\alpha}_s + \alpha_M \alpha_s - \alpha_s\right) + \alpha_s \left(\hat{\alpha}_s^2 + \alpha_s^2 - \alpha_M\right)}{\gamma \hat{\gamma} \hat{\alpha}_s}$$

$$= \Theta\left(\frac{Ms\delta^2 + \sqrt{(1 - M^2\delta^2)(1 - s^2\delta^2)} - \sqrt{1 - s^2\delta^2} + s^2\delta^2 + 1 - s^2\delta^2 - \sqrt{1 - M^2\delta^2}}{\delta^2}\right)$$

$$= \Theta(1) \cdot \frac{M(s - M/2)\delta^2 + \frac{1}{2}M^2\delta^2}{\delta^2} = \Theta(1) \ .$$

Therefore,

$$W(\alpha_M, \alpha_s, \arccos \alpha_s) \geq \frac{1}{n} \cdot \left(M^2 - \frac{M^4}{4s^2}\right)^{d/2+o(d)} \ .$$

$\square$

From Lemma 8, we can find the conditions for M and s to guarantee (with sufficiently large probability) making steps until we are in the cap of radius $s\delta$ centered at q. The following lemma gives such conditions and also guarantees that at each step we can reach a cap of a radius at least $\varepsilon\delta$ smaller for some constant $\varepsilon > 0$.

Lemma 9. *Assume that $s > 1$. If $M > \sqrt{2}s$ or $M^2 - \frac{M^4}{4s^2} > 1$, then there exists such constant $\varepsilon > 0$ that $W(\alpha_M, \alpha_s, \arcsin(\hat{\alpha}_s + \varepsilon\delta)) \geq \frac{1}{n}S^{d(1+o(1))}$ for some constant $S > 1$.*

Proof. First, let us take $\varepsilon = 0$. Then, the result directly follows from Lemma 8. We note that a value M satisfying $M^2 - \frac{M^4}{4s^2} > 1$ exists only if $s > 1$.

Now, let us demonstrate that we can take some $\varepsilon > 0$. The two cases discussed in Lemma 8 now correspond to $M^2 \geq s^2 + (s + \varepsilon)^2$ and $M^2 < s^2 + (s + \varepsilon)^2$, respectively. If $M > \sqrt{2}s$, then we can choose a sufficiently small ε that $M^2 \geq s^2 + (s + \varepsilon)^2$. Then, the result follows from Lemma 8 and the fact that $s > 1$. Otherwise, we have $M^2 < s^2 + (s + \varepsilon)^2$ and instead of the condition $M^2 - \frac{M^4}{4s^2} > 1$ we get (using Lemma 7)

$$\frac{2\left(M^2s^2 + s^2(s + \varepsilon)^2 + M^2(s + \varepsilon)^2\right) - \left(M^4 + s^4 + (s + \varepsilon)^4\right)}{4(s + \varepsilon)^2} > 1.$$

As this holds for $\varepsilon = 0$, we can choose a small enough $\varepsilon > 0$ that the condition is still satisfied. $\square$

This lemma implies that if we are given M and s satisfying the above conditions, then we can make a step towards q since the expected number of nodes in the intersection of spherical caps is much larger than 1. Formally, we can estimate from below the values $g(n)$ for all steps of the algorithm by $g_{min}(n) = S^{d(1+o(1))}$. So, according to Section B.1, we can make each step with probability $1 - O\left(e^{-S^{d(1+o(1))}}\right)$. Moreover, each step reduces the radius of a spherical cap centered at q and
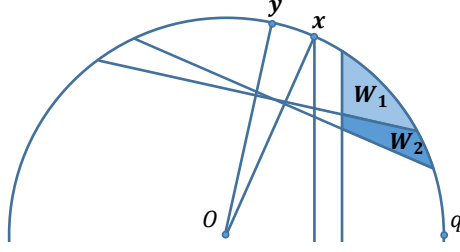
Figure 5: Dependence of consecutive steps

containing the current position by at least $\varepsilon\delta$. As a result, the number of steps (until we reach some distance $\arccos\alpha_s$) is $O\left(\delta^{-1}\right) = O\left(2^{\omega}\right)$.

To estimate the overall success probability, we have to take into account that the consecutive steps of the algorithm are dependent. In Section B.1, it is explained how the previous steps of the algorithm may affect the current one: the fact that at some step we moved from $\mathbf{y}$ to $\mathbf{x}$ implies that there were no elements closer to q than $\mathbf{x}$ in a spherical cap centered at $\mathbf{y}$. However, we can show that this dependence can be neglected.

Lemma 10. *The dependence of the consecutive steps can be neglected and does not affect the analysis.*

Proof. The main idea is illustrated on Figure 5. Assume that we are currently at a point $\mathbf{x}$ with $\rho(\mathbf{x}, q) = \arcsin(\hat{\alpha}_s + \varepsilon\delta)$. Then, as in the proof of Lemma 9, we are interested in the volume $W(\alpha_M, \alpha_s, \arcsin(\hat{\alpha}_s + \varepsilon\delta))$, which corresponds to $W_1 + W_2$ on Figure 5. Assume that at the previous step we were at some point $\mathbf{y}$. Given that all steps are "longer than ε", the largest effect of the previous step is reached when $\mathbf{y}$ is as close to $\mathbf{x}$ as possible and $\mathbf{x}$ lies on the geodesic between $\mathbf{y}$ and q. Therefore, the largest possible volume is $W(\alpha_M, \alpha_s, \arcsin(\hat{\alpha}_s + 2\varepsilon\delta))$, which corresponds to W_1 on Figure 5. It remains to show that W_1 is negligible compared with $W_1 + W_2$.

If $M < \sqrt{2}s$, then the main term of $W(\alpha_M, \alpha_s, \arcsin(\hat{\alpha}_s + \varepsilon\delta))$ is $\hat{\gamma}^d$ with

$$\hat{\gamma}^2 = \frac{2\left(M^2 s^2 + s^2(s+\varepsilon)^2 + M^2(s+\varepsilon)^2\right) - \left(M^4 + s^4 + (s+\varepsilon)^4\right)}{4(s+\varepsilon)^2}$$

$$= \frac{M^2 + s^2}{2} - \frac{\left(M^2 - s^2\right)^2}{4(s+\varepsilon)^2} - \frac{(s+\varepsilon)^2}{4}.$$

The main term of $W(\alpha_M, \alpha_s, \arcsin(\hat{\alpha}_s + 2\varepsilon\delta))$ is $\hat{\gamma}_1^d$ with

$$\hat{\gamma}_1^2 = \frac{M^2 + s^2}{2} - \frac{\left(M^2 - s^2\right)^2}{4(s+2\varepsilon)^2} - \frac{(s+2\varepsilon)^2}{4}.$$

It is easy to see that $M < \sqrt{2}s$ implies that $\hat{\gamma}_1^2 < \hat{\gamma}^2$. As a result, $W(\alpha_M, \alpha_s, \arcsin(\hat{\alpha}_s + 2\varepsilon\delta)) = o\left(W(\alpha_M, \alpha_s, \arcsin(\hat{\alpha}_s + \varepsilon\delta))\right)$ (similarly to the other proofs, it is easy to show that the effect of the other terms in Lemma 2 is negligible compared to $(\hat{\gamma}_1/\hat{\gamma})^d$). Moreover, since at each step we reduce the radius of a spherical cap centered at q by at least $\varepsilon\delta$, any cap encountered in one iteration intersects with only a constant number of other caps, so their overall effect is negligible, which completes the proof.

Finally, let us note that as soon as we have $M > \sqrt{2}s$ (with $s > 1$), with probability $1 - o(1)$ we find the nearest neighbor in one step. $\qquad\square$

Having this result on "almost independence" of the consecutive steps, we can say that the overall success probability is $1 - O\left(2^{\omega}e^{-S^{d+o(1)}}\right)$. Assuming $d \gg \log\log n$, we get $O\left(2^{\omega}e^{-S^{d(1+o(1))}}\right) =$

9

$O\left(2^{\frac{\log n}{d}}e^{-\log n}\right) = o(1)$. This concludes the proof for the success probability $1 - o(1)$ up to choosing suitable values for s and M.

Let us discuss the time complexity. With probability $1 - o(1)$ the number of steps is $\Theta\left(\delta^{-1}\right)$: the upper bound was already discussed; the lower bound follows from the fact that $M\delta$ is the radius of a spherical cap, so we cannot make steps longer than $\arcsin(M\delta)$, and with probability $1 - o(1)$ we start from a constant distance from q. The complexity of each step is $\Theta\left(f \cdot d\right) = \Theta\left(d^{1/2} \cdot M^d\right)$, so overall we get $\Theta\left(d^{1/2} \cdot 2^\omega \cdot M^d\right)$.

It remains to find suitable values for s and M. Before we continue, let us analyze the conditions under which we find exactly the nearest neighbor at the next step of the algorithm. Assume that the radius of a cap centered at q and covering the currently considered element is $\hat{\alpha}_s$ and $\hat{\alpha}_s = s\,\delta$, $\hat{\alpha}_M = M\delta$. Further assume that the radius of a spherical cap covering points at distance at most R from q is $\hat{\alpha}_r = r\delta = \sin R$ for some $r < 1$. The following lemma gives the conditions for M and s such that at the next step of the algorithm we find the nearest neighbor $\bar{x}$ with probability $1 - o(1)$ given that $\bar{x}$ is uniformly distributed within a distance R from q.

Lemma 11. *If for constant M, s, r we have $M^2 > s^2 + r^2$, then*

$$C(\alpha_r) - W(\alpha_M, \alpha_r, \arccos \alpha_s) \le C(\alpha_r)\beta^d$$

with some $\beta < 1$.

Proof. First, recall the lower bound for $C(\alpha_r)$: $C(\alpha_r) \ge \Theta\left(d^{-1/2}\right)\delta^d r^d$.

Since we have $M^2 > s^2 + r^2$, then asymptotically we have $\alpha_M < \alpha_s\alpha_r$, so the cases (1)-(2) of Lemma 2 should be applied (see Figure 2b). Let us estimate $\hat{\gamma}^2$ (Lemma 7):

$$\hat{\gamma}^2 \sim \delta^2 \cdot \frac{2\left(M^2 s^2 + M^2 r^2 + s^2 r^2\right) - \left(M^4 + r^4 + s^4\right)}{4\,s^2}$$

$$= \delta^2 \cdot \frac{-\left(M^2 - s^2 - r^2\right)^2 + 4\,s^2 r^2}{4\,s^2} = \delta^2 r^2\left(1 - \Theta(1)\right).$$

Therefore $\hat{\gamma}^d \le \delta^d r^d \beta^d$ with some $\beta < 1$.

It only remains to estimate the other terms in the upper bound from Lemma 2:

$$\frac{\hat{\gamma}\,\alpha_r\,\hat{\alpha}_s}{\gamma|\alpha_M - \alpha_r\,\alpha_s|}\Theta\left(d^{-1}\right)\min\left\{d^{1/2}, \frac{1}{\gamma}\right\} = O\left(d^{-1}\right),$$

from which the lemma follows. $\qquad\square$

Now we are ready to finalize the proof. We solve c, R-ANN if either we have $\arcsin(s\delta) \le cR$ or we are sufficiently close to q to find the exact nearest neighbor $\bar{x}$ in the next step of the algorithm. Let us analyze the first possibility. Let $\sin R = r\delta$, then we need $s < cr$. According to Lemma 9, it is sufficient to have $r\,c > 1$ and either $M \ge \sqrt{2}\,rc$ or $M^2 - \frac{M^4}{4r^2 c^2} > 1$. Alternatively, according to Lemma 11, to find the exact nearest neighbor with probability $1 - o(1)$, it is sufficient to reach such s that $M^2 > s^2 + r^2$. For this, according to Lemma 9, it is sufficient to have $s > 1$, $M^2 > s^2 + r^2$, and either $M \ge \sqrt{2}s$ or $M^2 - \frac{M^4}{4s^2} > 1$.

One can show that if the following conditions on M and r are satisfied, then we can choose an appropriate s for the two cases discussed above:

    (a) $r\,c > 1$ and $M^2 > 2\,r^2\,c^2\left(1 - \sqrt{1 - \frac{1}{r^2 c^2}}\right)$;

    (b) $M^2 > \frac{2}{3}\left(r^2 + 1 + \sqrt{r^4 - r^2 + 1}\right)$.

To succeed, we need either (a) or (b) to be satisfied. The bound in (a) decreases with r ($r > 1/c$) and for $r = \frac{1}{c}$ it equals $\sqrt{2}$. The bound in (b) increases with r and for $r = 1$ it equals $\sqrt{2}$. To find a general bound holding for all r, we take the "hardest" $r \in (\frac{1}{c}, 1)$, where the bounds in (a) and (b) are equal to each other. This value is $r = \sqrt{\frac{4c^2}{(c^2+1)(3c^2-1)}}$ and it gives the bound $M > \sqrt{\frac{4c^2}{3c^2-1}}$ stated in the theorem.

B.4 Larger neighborhood in dense regime

As discussed in the main text, it could potentially be possible that taking $M = M(n) \gg 1$ improves the query time. The following theorem shows that this is not the case.

Theorem 1. *Let $M = M(n) \gg 1$. Then, with probability $1 - o(1)$, graph-based NNS finds the exact nearest neighbor in one iteration; time complexity is $\Omega\left(d^{1/2} \cdot 2^\omega \cdot M^{d-1}\right)$; space complexity is $\Theta\left(n \cdot d^{-1/2} \cdot M^d \cdot \log n\right)$.*

As a result, when $M \to \infty$, both time and space complexities become larger compared with constant M (see Theorem 1 from the main text).

Proof. When M grows with n, it follows from the previous reasoning that the algorithm succeeds with probability $1 - o(1)$. The analysis of the space complexity is the same as for constant M, so we get $\Theta\left(d^{-1/2} \cdot M^d \cdot n \cdot \log n\right)$. When analyzing the time complexity, we note that the one-step complexity is $\Theta\left(d^{-1/2} \cdot M^d\right)$. It is easy to see that we cannot make steps longer than $O\left(M \cdot 2^{-\omega}\right)$. This leads to the time complexity $\Omega\left(d^{1/2} \cdot M^{d-1} \cdot 2^\omega\right)$. $\qquad\square$

B.5 Proof of Theorem 2 (greedy search in sparse regime)

For sparse datasets, instead of radii, we operate with heights of spherical caps. In this case, we have $C(\alpha_1) \sim \frac{1}{n}$, i.e., $\alpha_1^2 \sim 1 - n^{-\frac{2}{d}} = 1 - 2^{-\frac{2}{\omega}} \sim \frac{2\ln 2}{\omega}$. We further denote $\frac{2\ln 2}{\omega}$ by δ.

We construct $G(M)$ using spherical caps with height α_M, $\alpha_M^2 = M\delta$, where M is some constant. Then, from Lemma 1, we get that the expected number of neighbors of a node is $f = \Theta\left(n\, d^{O(1)}(1 - M\delta)^{d/2}\right) = n^{1-M+o(1)}$. From this and Section B.2.2 the stated space complexity follows. The one-step time complexity $n^{1-M+o(1)}$ follows from Section B.2.1.

Our aim is to solve c, R-ANN with some $c > 1$, $R > 0$. If $R \geq \pi/2c$, then we can easily find the required near neighbor within a distance $\pi/2$, since we start $G(M)$-based NNS from such point. Let us consider any $R < \frac{\pi}{2c}$. It is clear that in this case we have to find the nearest neighbor itself, since $R\,c$ is smaller than the distance to the (non-planted) nearest neighbor with probability $1 + o(1)$. Note that $\alpha_c = \cos\frac{\pi}{2c} < \alpha_R := \cos R$.

Lemma 12. *Assume that $\alpha_M^2 = M\delta$, $\alpha_s^2 = s\delta$, $\alpha_\varepsilon^2 = \varepsilon\delta$, $M, s > 0$ are constants, and $\varepsilon \geq 0$ is bounded by a constant. If $s + M < 1$, then*

$$W(\alpha_M, \alpha_s, \arccos\alpha_\varepsilon) \geq \frac{1}{n} \cdot n^{\Omega(1)}\,.$$

Proof. Asymptotically, we have $\alpha_M > \alpha_s\alpha_\varepsilon$ and $\alpha_s > \alpha_M\alpha_\varepsilon$, so we are under the condition (3) in Lemma 2. First, consider the main term of $W(\alpha_M, \alpha_s, \arccos\alpha_\varepsilon)$:

$$\hat{\gamma}^d = \left(1 - \frac{M\delta + s\delta - 2\sqrt{Ms\varepsilon\delta}\,\delta}{1 - \varepsilon\delta}\right)^{d/2} = e^{-\frac{d}{2}\delta\left(M + s + O(\sqrt{\delta})\right)} = n^{-\left(M + s + O(\sqrt{\delta})\right)} = \frac{1}{n} \cdot n^{\Omega(1)}.$$

It remains to multiply this by $\Theta\left(d^{-1}\right)$ and $C_l = C_{l,\alpha} + C_{l,\beta}$ (see Lemma 2). It is easy to see that $C_l = \Omega(1)$ in this case, so both terms can be included to $n^{\Omega(1)}$, which concludes the proof. $\qquad\square$

It follows from Lemma 12 that if $M + s < 1$, then we can reach a spherical cap with height $\alpha_s = \sqrt{s\delta}$ centered at q in just one step (starting from a distance at most $\pi/2$). And we get $g(n) = n^{\Omega(1)}$.

Recall that $M < \frac{\alpha_c^2}{\alpha_c^2+1}$ and let us take $s = \frac{1}{\alpha_c^2+1}$, then we have $M + s < 1$. The following lemma discusses the conditions for M and s such that at the next step of the algorithm we find $\bar{x}$ with probability $1 - o(1)$.

Lemma 13. *If for constant M and s we have $M < s\alpha_R^2$, then*

$$C(\alpha_R) - W(\alpha_M, \alpha_R, \arccos\alpha_s) = C(\alpha_R)n^{-\Omega(1)}\,.$$

Proof. First, recall the lower bound for $C(\alpha_R)$: $C(\alpha_R) \geq \Theta\left(d^{-1/2}\right)\left(1-\alpha_R^2\right)^{d/2}$.

Note that since we have $M < s\alpha_R^2$, then the cases (1)-(2) of Lemma 2 should be applied (see Figure 2b). Let us estimate γ:

$$\gamma^2 = \frac{M\delta + \alpha_R^2 - 2\sqrt{Ms}\alpha_R\delta}{1-s\delta} = \alpha_R^2 + \delta\left(M - 2\sqrt{Ms}\alpha_R + s\alpha_R^2\right) + O\left(\delta^2\right)$$

$$= \alpha_R^2 + \delta\left(\sqrt{M} - \sqrt{s}\alpha_R\right)^2 + O\left(\delta^2\right) = \alpha_R^2 + \Theta(\delta).$$

Therefore,

$$\hat{\gamma}^d = \left(1-\alpha_R^2\right)^{d/2}\left(1-\Theta(\delta)\right)^{d/2} = \left(1-\alpha_R^2\right)^{d/2} n^{-\Omega(1)}.$$

It only remains to estimate the other terms in the upper bound from Lemma 2:

$$\frac{\hat{\gamma}\,\alpha_R\,\hat{\alpha}_s}{\gamma|\alpha_M - \alpha_R\,\alpha_s|}\Theta\left(d^{-1}\right)\min\left\{d^{1/2}, \frac{1}{\gamma}\right\} = O\left(\delta^{-1}d^{-1}\right),$$

from which the lemma follows. $\qquad\square$

Note that in our case we have $M < \frac{\alpha_c^2}{\alpha_c^2+1} < \frac{\alpha_R^2}{\alpha_c^2+1} = s\alpha_R^2$. From this Theorem 2 follows.

C Long-range links

C.1 Random edges

The simplest way to obtain a graph with a small diameter from a given graph is to connect each node to a few random neighbors. This idea is proposed in [8] and gives $O\left(\log n\right)$ diameter for the so-called "small-world model". It was later confirmed that adding a little randomness to a connected graph makes the diameter small [2]. However, we emphasize that having a logarithmic diameter does not guarantee a logarithmic number of steps in graph-based NNS, since these steps, while being greedy in the underlying metric space, may not be optimal on a graph.

To demonstrate the effect of long edges, assume that there is a graph G', where each node is connected to several random neighbors by directed edges. For simplicity or reasonings, assume that we first perform NNS on G' and then continue on the standard NN graph G. It is easy to see that during NNS on G', the neighbors considered at each step are just randomly sampled nodes, we choose the one closest to q and continue the process, and all such steps are independent. Therefore, the overall procedure is basically equivalent to a random sampling of a certain number of nodes and then choosing the one closest to q (from which we then start the standard NNS on G).

Theorem 2. *Under the conditions of Theorem 1 in the main text, performing a random sampling of some number of nodes and choosing the one closest to q as a starting point for graph-based NNS does not allow to get time complexity better than $\Omega\left(d^{1/2} \cdot e^{\omega(1+o(1))} \cdot M^d\right)$.*

Proof. Assume that we sample $e^{l\omega}$ nodes with an arbitrary $l = l(n)$. Then, with probability $1 - o(1)$, the closest one among them lies at a distance $\Theta\left(e^{-\frac{l\omega}{d}}\right)$. As a result, the overall time complexity becomes $\Theta\left(e^{-\frac{l\omega}{d}} \cdot d^{1/2} \cdot e^{\omega} \cdot M^d + d \cdot e^{l\omega}\right)$. If $l = \Omega(d)$, then the term $d \cdot e^{l\omega} = d \cdot e^{\Omega(d)\omega}$ dominates $d^{1/2} \cdot e^{\omega} \cdot M^d$, otherwise we get $\Theta\left(d^{1/2} \cdot e^{\omega(1+o(1))} \cdot M^d\right)$, which proves Theorem 2. $\quad\square$

C.2 Proof of Theorem 3 (effect of proper long edges)

Recall that we assume the following probability distribution:

$$\text{P(edge from } u \text{ to } v) = \frac{\rho(u,v)^{-d}}{\sum_{w\neq u}\rho(u,w)^{-d}}. \tag{4}$$

First, we estimate the denominator. In the lemma below we consider only the elements w with $\rho(u,w) > n^{-\frac{1}{d}}$. However, it easily follows from the proof that adding only edges with $\rho(u,v) > n^{-\frac{1}{d}}$ does not affect the reasoning.

From Theorem 1 in the main text, we know that without long edges we need $O(n^{\frac{1}{d}})$ steps, which is less than $\log^2 n$ for $d > \frac{\log n}{2 \log \log n}$. So, in this case Theorem 3 follows from Theorem 2. Hence, in the lemma below we can assume that $d < \frac{\log n}{2 \log \log n}$.

Lemma 14. *If $d < \frac{\log n}{2 \log \log n}$, then*

$$\mathbb{E}\left(\rho(u, w)^{-d}\right) = \Theta\left(\frac{\log n}{\sqrt{d}}\right).$$

Proof. Note that $\mathbb{E}\left(\rho(u, w)^{-d}\right) = \mathbb{E}\rho^{-d}(\mathbf{1}, w)$, where $\mathbf{1} = (1, 0, \ldots, 0)$. So, similarly to Lemma 1,

$$\mathbb{E}\left(\rho(\mathbf{1}, w)^{-d}\right) = \frac{\mu(S^{d-1})}{\mu(S^d)} \int\limits_{-1}^{\cos(n^{-\frac{1}{d}})} \left(1 - x^2\right)^{\frac{d-2}{2}} (\arccos x)^{-d} dx.$$

From Stirling's approximation, we have $\frac{\mu(S^{d-1})}{\mu(S^d)} = \Theta(\sqrt{d})$. After replacing $y = \arccos x$, the integral becomes

$$\int\limits_{n^{-\frac{1}{d}}}^{\pi} y^{-d} \sin^{d-1}(y) dy = \int\limits_{n^{-\frac{1}{d}}}^{\pi} \frac{1}{y}\left(\frac{\sin(y)}{y}\right)^{d-1} dy < \int\limits_{n^{-\frac{1}{d}}}^{\pi} \frac{1}{y} dy = \Theta\left(\frac{\ln n}{d}\right).$$

On the other hand, for $d < \frac{\log n}{2 \log \log n}$:

$$\mathbb{E}\left(\rho(\mathbf{1}, w)^{-d}\right) = \Theta(\sqrt{d}) \int\limits_{n^{-\frac{1}{d}}}^{\pi} \frac{1}{y}\left(\frac{\sin(y)}{y}\right)^{d-1} dy > \Theta(\sqrt{d}) \int\limits_{n^{-\frac{1}{d}}}^{\frac{1}{\sqrt{d}}} \frac{1}{y}\left(\frac{\sin(y)}{y}\right)^{d-1} dy.$$

Since on this interval we have $\left(\frac{\sin(y)}{y}\right)^{d-1} = \Theta(1)$, we can continue:

$$\mathbb{E}\left(\rho(\mathbf{1}, w)^{-d}\right) > \Theta(\sqrt{d}) \int\limits_{n^{-\frac{1}{d}}}^{\frac{1}{\sqrt{d}}} \frac{1}{y} dy = \Theta(\sqrt{d})\left(\frac{\ln n}{d} - \frac{1}{2}\ln d\right) = \Theta\left(\frac{\ln n}{\sqrt{d}}\right).$$

As a result, we get $\mathbb{E}\rho(\mathbf{1}, w)^{-d} = \Theta\left(\frac{\log n}{\sqrt{d}}\right).$ $\qquad\square$

Also, from the proof above it follows that $\mathbb{E}\rho(u, w)^{-2d} < \Theta(\sqrt{d}) \int\limits_{n^{-\frac{1}{d}}}^{\pi} \frac{1}{y^{d+1}} dy = O(\frac{n}{\sqrt{d}})$. Let

$$\text{Den} = \sum_{w: \rho(u,w) > n^{-\frac{1}{d}}} \rho(u, w)^{-d},$$

so $\mathbb{E}\,\text{Den} = \Theta\left(\frac{n\,\log n}{\sqrt{d}}\right)$ and

$$\mathbb{E}\,\text{Den}^2 = n\,\mathbb{E}\rho(u, w)^{-2d} + n(n-1)\left(\mathbb{E}\rho(u, w)^{-d}\right)^2 = O\left(\frac{n^2}{\sqrt{d}} + \frac{n^2 \ln^2 n}{d} - \frac{n \ln^2 n}{d}\right).$$

Finally, from Chebyshev's inequality, we get

$$P\left(|\text{Den} - \mathbb{E}\text{Den}| > \frac{\mathbb{E}\text{Den}}{2}\right) \leq \frac{4\,\text{Var}(\text{Den})}{(\mathbb{E}\text{Den})^2} = O\left(\frac{\sqrt{d}}{\log^2 n}\right) = o(1).$$

So, we may further replace the denominator of Equation (4) by $O\left(\frac{n \ \log n}{\sqrt{d}}\right)$.[2]

We are ready to prove the theorem. We split the search process on a sphere into $\log n$ phases and show that each phase requires $O(\log n)$ steps. Phase j consists of the following nodes: $\{u : t_{j+1} < \rho(u, q) \le t_j\}$, where $t_j = \frac{\pi}{2} t^j = \frac{\pi}{2} \left(1 - \frac{1}{d}\right)^j$.

We start at a distance at most $\frac{\pi}{2}$, this corresponds to $j = 0$. Recall that the nearest neighbor (in the dense regime) is at a distance about $2^{-\frac{\log n}{d}}$. Then, the number of phases needed to reach the nearest neighbor is

$$k \sim -\frac{1}{\log\left(1 - \frac{1}{d}\right)} \cdot \frac{\log n}{d} \sim \log n \,.$$

Suppose we are at some node belonging to a phase j. Let us prove the following inequality for the probability of making a step to a phase with a larger number:

$$\text{P(make a step to a closer phase)} > \frac{\Theta(1)}{\log n} \,.$$

In the polar coordinates, we can express this probability as

$$\text{P(make a step to a closer phase)} = \frac{(d-1)\sqrt{d}}{2\pi \ \log n} \int_{\cos t_{i+1}}^{1} \int_{-\arccos \frac{\cos(t_{i+1})}{r}}^{\arccos \frac{\cos(t_{i+1})}{r}} \left(\sqrt{1 - r^2}\right)^{d-3}$$
$$\cdot (\arccos(\sin(t_i) r \sin(\phi) + \cos(t_i) r \cos(\phi)))^{-d} r \ d\phi \ dr$$
$$= \frac{\Theta(d^{\frac{3}{2}})}{\log n} \int_{\cos t_{i+1}}^{1} \int_{-\arccos \frac{\cos(t_{i+1})}{r}}^{\arccos \frac{\cos(t_{i+1})}{r}} \left(\sqrt{1 - r^2}\right)^{d-3} (\arccos(r \cos(t_i - \phi)))^{-d} r \ d\phi \ dr.$$

Let $r = \cos(\psi)$ and $\phi = t_j - \phi$, then the integral becomes

$$\int_0^{t_{j+1}} \cos\psi \, (\sin\psi)^{d-2} \int_{t_j - \arccos \frac{\cos(t_{j+1})}{\cos\psi}}^{t_j + \arccos \frac{\cos(t_{j+1})}{\cos\psi}} (\arccos(\cos\psi \cos(\phi)))^{-d} d\phi d\psi \,.$$

From convexity of $\log \cos \sqrt{x}$, it follows that $\forall \psi, \phi \in [0, \frac{\pi}{2}]$ we have

$$\arccos(\cos\psi \cos\phi) \le \sqrt{\psi^2 + \phi^2} \,,$$
$$\arccos \frac{\cos(t_{j+1})}{\cos\psi} \ge \sqrt{t_{j+1}^2 - \psi^2} \,,$$
$$\sin(\psi) \ge \psi - \frac{\psi^3}{6} \,.$$

We use these bounds since we need a lower bound for the integral. Also, we replace the upper limit of the inner integral with t_j and consider $\psi = t_j \psi$ and $\phi = t_j \phi$:

$$\int_0^t F(t_j, \psi) \psi^{d-2} \int_{1 - \sqrt{t^2 - \psi^2}}^{1} \left(\sqrt{\psi^2 + \phi^2}\right)^{-d} d\phi \ d\psi \,,$$

where $F(t_j, \psi) = \cos(t_j \ \psi) \left(1 - \frac{t_j^2 \psi^2}{6}\right)^{d-2}$.

---

[2]More formally, our analysis below is conditioned on the fact that the denominator is less than $\frac{C n \log n}{\sqrt{d}}$ for some constant $C > 0$. The probability that it does not hold is $o(1)$ and for such nodes we can just assume that we do not use long edges.

14

Consider the inner integral:

$$\int\limits_{1-\sqrt{t^2-\psi^2}}^{1} (\sqrt{\psi^2+\phi^2})^{-d}\frac{1}{2\phi}d\phi^2 > \frac{1}{2}\int\limits_{1-2\sqrt{t^2-\psi^2}+t^2-\psi^2}^{1} (\psi^2+x)^{-\frac{d}{2}}\,dx$$

$$= \frac{1}{d-2}\left((1-2\sqrt{t^2-\psi^2}+t^2)^{-\frac{d-2}{2}}-(1+\psi^2)^{-\frac{d-2}{2}}\right).$$

Substitute the second term to the original integral and estimate it from above:

$$\int_0^t F(t_j,\psi)\psi^{d-2}(1+\psi^2)^{-\frac{d-2}{2}}\,d\psi \leq \int_0^t \left(\frac{\psi^2}{1+\psi^2}\right)^{\frac{d-2}{2}}\,d\psi = o\left(\frac{1}{\sqrt{d}}\right).$$

Now we estimate from below the second term $\int\limits_0^t \left(\frac{\psi^2}{1-2\sqrt{t^2-\psi^2}+t^2}\right)^{\frac{d-2}{2}}d\psi$.

Note that if $\psi = \frac{2}{\sqrt{d}}$ and $t = 1-\frac{1}{d}$, then

$$\left(\frac{\psi^2}{1-2\sqrt{t^2-\psi^2}+t^2}\right)^{\frac{d-2}{2}} = \left(\frac{\frac{4}{d}}{1-2\sqrt{1-\frac{2}{d}+\frac{1}{d^2}-\frac{4}{d}}+1-\frac{2}{d}+\frac{1}{d^2}}\right)^{\frac{d-2}{2}}$$

$$> \left(\frac{\frac{4}{d}}{\frac{4}{d}+\frac{11}{d^2}+\frac{1}{d^2}}\right)^{\frac{d-2}{2}} = e^{-\frac{3}{2}}(1+o(1)).$$

Similarly, it can be shown that if $\psi = \frac{3}{\sqrt{d}}$, then

$$\left(\frac{\psi^2}{1-2\sqrt{t^2-\psi^2}+t^2}\right)^{\frac{d-2}{2}} > \Theta(1).$$

So, for $\psi \in [\frac{2}{\sqrt{d}}, \frac{3}{\sqrt{d}}]$ this fraction is greater than some constant (as well as $F(t_j,\psi)$) and the derivative does not change the sign on this segment. As a result,

$$\int_0^t F(t_j,\psi)\left(\frac{\psi^2}{1-2\sqrt{t^2-\psi^2}+t^2}\right)^{\frac{d-2}{2}}d\psi > \frac{\Theta(1)}{\sqrt{d}}.$$

And finally,

$$\mathrm{P}(\text{make a step in to a closer phase}) > \frac{\Theta(1)}{\log n}.$$

To sum up, there are $O(\log n)$ phases and the number of steps in each phase is geometrically distributed with the expected value $O(\log n)$. From this the theorem follows.

C.3 Proof of Corollary 2

We have

$$\mathrm{P}(\text{short-cut step within }\log n\text{ trying}) = 1-(1-P)^{\log n} = \left(1-\frac{1}{e}\right)(1-o(1)), \qquad (5)$$

where P is the probability corresponding to one long-range edge, which is estimated in the proof above.

Also, since $d \gg \log\log n$, we have $\frac{M^d}{\sqrt{d}} > \log n$, so the step complexity is the same.

It is easy to see that increasing the number of shortcut edges does not improve the asymptotic complexity, since the probability in (5) is already constant.

15

C.4 Proof of Lemma 1 (effect of pre-sampling)

For convenience, in this proof we assume that the overall number of elements is $n + 1$ instead of n which does not affect the analysis.

Let v be the k-th neighbor for the source node u. For the initial distribution we have:

$$P(\text{edge from } u \text{ to } v) \sim \frac{1}{k \ln n}.$$

By pre-sampling of n^φ nodes, we modify this probability to

$$P(\text{edge from } u \text{ to } v | v \text{ is sampled}) \cdot P(v \text{ is sampled}). \tag{6}$$

The second term above is equal to $\frac{n^\varphi}{n}$. Assuming that $k = n^\alpha > n^{1-\varphi}$, we can estimate the probability above. Below by l we denote the rank of v in the selected subset and obtain:

$$P(\text{edge from } u \text{ to } v | v \text{ is sampled}) \cdot P(v \text{ is sampled})$$

$$= \frac{n^\varphi}{n} \sum_{l=1}^{\min(n^\alpha, n^\varphi)} \binom{n^\varphi - 1}{l - 1} \left(\frac{n^\alpha - 1}{n - 1}\right)^{l-1} \left(\frac{n - n^\alpha}{n - 1}\right)^{n^\varphi - l} \frac{1}{l} \frac{1}{\ln n^\varphi}$$

$$= \frac{1}{\varphi \, n \, \ln n} \sum_{l=1}^{\min(n^\alpha, n^\varphi)} \binom{n^\varphi}{l} \left(\frac{n^\alpha - 1}{n - 1}\right)^{l-1} \left(\frac{n - n^\alpha}{n - 1}\right)^{n^\varphi - l}$$

$$= \frac{\Theta(1)}{\varphi \, n^\alpha \ln n} \sum_{l=1}^{\min(n^\alpha, n^\varphi)} \binom{n^\varphi}{l} \left(\frac{n^\alpha - 1}{n - 1}\right)^{l} \left(\frac{n - n^\alpha}{n - 1}\right)^{n^\varphi - l}.$$

Let us analyze the sum above. First, it is easy to see that it is less than 1. Second, if $n^\varphi \leq n^\alpha$, then the sum is "almost equal" to 1 (without one term corresponding to $l = 0$, which we analyze below). Otherwise, we know that for a binomial distribution its median cannot lie too far away from the mean (see, e.g., [3]). Since $\alpha > 1 - \varphi$, we have

$$\min(n^\alpha, n^\varphi) \geq \frac{n^\varphi(n^\alpha - 1)}{n - 1} + 1 = \mathbb{E}\operatorname{Bin}(n^\varphi, \frac{n^\alpha - 1}{n - 1}) + 1 > \operatorname{median}(\varphi, \alpha).$$

Hence,

$$\sum_{l=0}^{\min(n^\alpha, n^\varphi)} \binom{n^\varphi}{l} \left(\frac{n^\alpha - 1}{n - 1}\right)^{l} \left(\frac{n - n^\alpha}{n - 1}\right)^{n^\varphi - l} > \frac{1}{2}.$$

Note that we added one term corresponding to $l = 0$, but it is easy to see that in the worst case it is about $\frac{1}{e}$. Namely, for $l = 0$:

$$\binom{n^\varphi}{0} \left(\frac{n^\alpha - 1}{n - 1}\right)^{0} \left(\frac{n - n^\alpha}{n - 1}\right)^{n^\varphi - 0} = \left(\frac{n - n^\alpha}{n - 1}\right)^{n^\varphi} < \left(1 - \frac{n^{1-\varphi} - 1}{n - 1}\right)^{n^\varphi} = \frac{1}{e}(1 + o(1)).$$

D   Proof of Theorem 4 (effect of beam search)

First, let us prove the following lemma.

Lemma 15. *Under the conditions of Theorem 4 in the main text, if we start from x_0 such that $\rho(q, x_0) \leq \arcsin\left(R\,n^{-1/d}\right)$, then $G(L)$-based NNS solves the exact NN problem with probability $1 - o(1)$.*

Proof. Properties of a wide class of random graphs (including the random geometric graphs) are well studied [2, 6]. It is known that in many cases connectivity follows from the fact that there are

no isolated nodes in the graph. Namely, [7] expanded this criterion to a wide class of "connection" functions (rules for connection the nodes, which in our case is just a threshold function) and different spaces (see Theorem 2.2 in [7]).

We use this result directly: our estimates of intersections of spherical caps (Lemma 2) and the conditions on L and R stated in the theorem guarantee (see the proof below) that the subgraph induced by $\{x_i | \rho(x_i, q) \leq \arcsin(R\,n^{-1/d})\}$ is connected. Also, according to Lemma 1, for sufficiently large C we have enough candidates to store all these elements. From this the lemma follows.

Let us prove the connectedness. For each element under consideration we can estimate the probability that it is isolated. The maximum of such probabilities is achieved if $\rho(x_i, q) = \arcsin(R\,n^{-1/d})$.

Let us denote $S^2 = L^2\left(1 - \frac{L^2}{4R^2}\right)$. In the worst case, from Lemma 2 and Lemma 7, we know that the probability of an isolation is

$$(1 - W(R, L, \arcsin R\delta))^n = (e + o(1))^{-W(R,L,\arcsin R\delta)n} < e^{-\Theta\left(\frac{S^d}{d}\right)}.$$

Thus, taking into account that we have $\frac{R^d}{\sqrt{d}}$ elements, we want $\frac{R^d}{\sqrt{d}}\,e^{-\Theta\left(\frac{S^d}{d}\right)} \to 0$. In other words,

$$d\ln R - \frac{1}{2}\ln d - \Theta\left(\frac{S^d}{d}\right) \to -\infty.$$

That is true, for $S > 1$ and $d \to \infty$. The fact that $S > 1$ follows from the statement of Theorem 4. $\qquad\square$

Note that for $G(L)$ the conditions of Lemma 9 are also satisfied. This means we do not get stuck in a local minimum until we reach the query's R-neighborhood. After that, we can find the nearest element according to Lemma 15. So, the NNS problem is indeed solved with probability $1 - o(1)$.

Now it is easy to see that time complexity is $\Theta\left(d^{1/2} \cdot \log n \cdot L^d\right) + O\left(d \cdot R^d \cdot L^d \cdot \log R\right)$, where the first term corresponds to $\log n$ steps before we reach the R-neighborhood and the second one corresponds to the remaining steps. For $d > \log\log n$ the first term is negligible compared to the second one, so the required complexity follows.

E Comparison with the results of [4]

Here we extend the related work from the main text and discuss in more detail how our research differs from the results of [4].

Laarhoven [4] analyzes time and space complexity for graph-based NNS in sparse regime when $d \gg \log n$. He considers plain NN graphs and allows multiple restarts. In contrast, we consider both regimes and assume only one iteration of a graph-based search. We do not consider multiple restarts since it is non-trivial to rigorously prove that restarts can be assumed "almost independent" (see Section A.3, proof of Lemma 15, [4]). As a result, for sparse datasets, we consider a slightly weaker setting with only one iteration, but all results are formally proven. Our result for sparse regime (Theorem 2 in the main text) corresponds to the case $\rho_q = \rho_s$ from [4].

Also, in Section A, we state new bounds for the volumes of spherical caps' intersections, which are needed for the rigorous analysis in both sparse and dense regimes. We could not use the results of [1] since parameters defining spherical caps are assumed to be constant there, while they can tend to 0 or 1 in dense and sparse regimes.

We also address the problem of possible dependence between consecutive steps of the algorithm (Lemma 10). While we prove that it can be neglected, it is important for rigorous analysis.

Most importantly, we analyze the dense regime and additional techniques (shortcuts and beam search), which are essential for the effective graph-based search. Interestingly, shortcut edges are useful only in the dense regime.
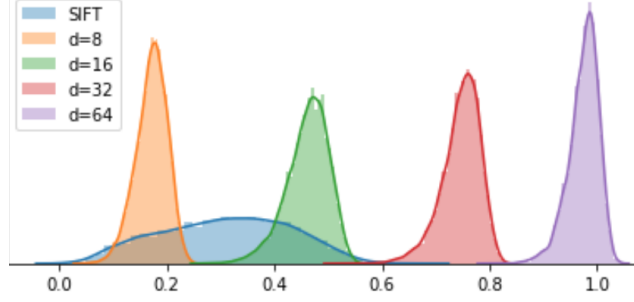
Figure 6: The distribution of the distance to the nearest neighbor for the SIFT dataset and synthetic uniform data for different dimensions and the same size (1M)

F Additional experiments

F.1 Dense vs sparse setting

Let us discuss our intuition on why real datasets are "more similar" to dense rather than sparse synthetic ones.

In the sparse regime, all elements are almost at the same distance from each other, and even in the moderate regime ($d \propto \log(n)$), the distance to the nearest neighbor must be close to a certain constant. In contrast, the dense regime implies high proximity of the nearest objects. While real datasets are always finite and asymptotic properties cannot be formally verified, we still can compare the properties of real and synthetic datasets. We plotted the distribution of the distance to the nearest neighbor (see Figure 6) and see that for the SIFT dataset, the obtained distribution is more similar to the ones in the dense regime. This is further supported by the literature which estimates the intrinsic dimension of real data. For example, for the SIFT dataset with 128-dimensional vectors, the estimated intrinsic dimension is 16 [5]. Thus, we conclude that the analysis of the dense regime is important.

F.2 Parameters of algorithms

In this section, we specify additional hyperparameters used in our experiments.

The number of edges used in KL, when is not explicitly specified, is equal to 15, which is close to $\ln n$.

The number of edges in KNN graphs is dynamic when the beam search is not used. When the beam search is used, the number of edges for synthetic datasets is 8 for $d = 2$, 10 for $d = 4$, 16 for $d = 8$, 20 for $d = 16$, and 25 for all real datasets.

The dimension we use for DIM-RED is 64 for GIST, 32 for SIFT, 48 for DEEP, 128 for GloVe.

F.3 Additional experimental results

In Figure 7, we show that several approximations discussed in the main text do not affect the quality of graph-based NNS significantly (in the uniform case). Namely,

- Connecting a node to other nodes at a distance smaller than some constant (THRNN) and to the fixed number of nearest neighbors (KNN) lead to graph-based algorithms with similar performance;
- Pre-sampling of $\sqrt{n}$ nodes when adding shortcut edges (SAMPLE) lowers the quality, but not substantially;
- Rank-based probabilities for shortcut edges (KL-RANK) can lead to even better quality than distance-based (KL-DIST).

In Figure 8, we illustrate how the number of long-range edges affects the quality of the algorithm. Let us note that 16 is close to $\log n$ discussed in Corollary 2 of the main text. Figure 8 shows that this value is indeed close to being optimal, especially for the high-accuracy regime, which is a focus of
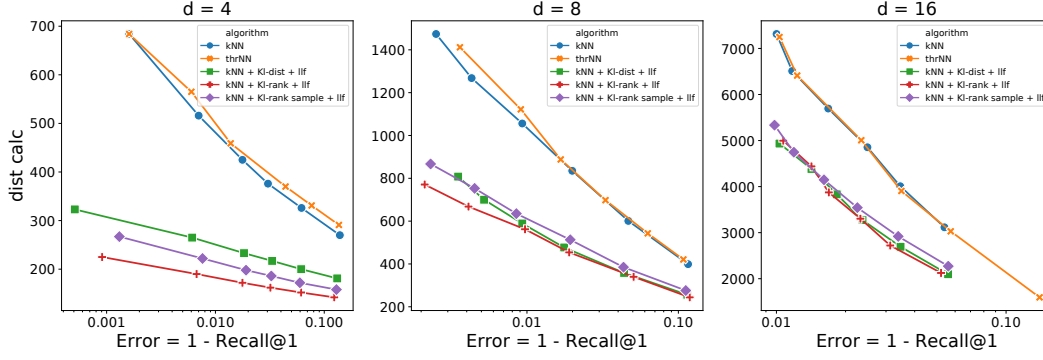
18

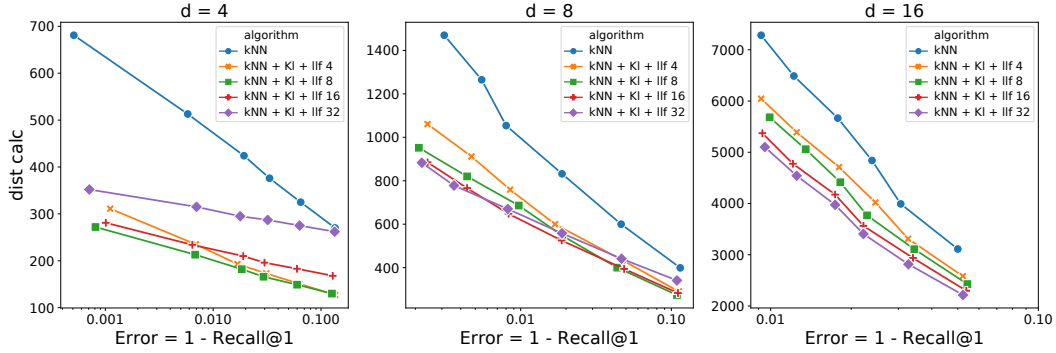Figure 7: The effect of κNN and KL approximations

Figure 8: The effect of the number of long-range edges

the current research. However, it also seems that the optimal number of long edges may depend on d: on Figure 8, the relative performance of graphs with 32 long edges is improving as d grows.

References

[1] A. Becker, L. Ducas, N. Gama, and T. Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 10–24, 2016.

[2] B. Bollobás and F. R. K. Chung. The diameter of a cycle plus a random matching. *SIAM Journal on discrete mathematics*, 1(3):328–333, 1988.

[3] K. Hamza. The smallest uniform upper bound on the distance between the mean and the median of the binomial and poisson distributions. *Statistics & Probability Letters*, 23(1):21–25, 1995.

[4] T. Laarhoven. Graph-based time-space trade-offs for approximate near neighbors. In *34th International Symposium on Computational Geometry (SoCG 2018)*, 2018.

[5] E. Levina and P. J. Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in neural information processing systems*, pages 777–784, 2005.

[6] M. Penrose et al. *Random geometric graphs*, volume 5. Oxford university press, 2003.

[7] M. D. Penrose et al. Connectivity of soft random geometric graphs. *The Annals of Applied Probability*, 26(2):986–1028, 2016.

[8] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.