Beyond Locality–Sensitive Hashing

Alexandr Andoni, Microsoft Research SVC
Piotr Indyk, MIT
Huy L. Nguyễn, Princeton
Ilya Razenshteyn, MIT

Abstract

We present a new data structure for the c–approximate near neighbor problem (ANN) in the Euclidean space. For n points in $\mathbb{R}^d$, our algorithm achieves $O_c(dn^\rho)$ query time and $O_c(n^{1+\rho} + nd)$ space, where $\rho \le 7/(8c^2) + O(1/c^3) + o_c(1)$. This is the first improvement over the result by Andoni and Indyk (FOCS 2006) and the first data structure that bypasses a locality–sensitive hashing lower bound proved by O'Donnell, Wu and Zhou (ITCS 2011). By a standard reduction we obtain a data structure for the Hamming space and ℓ_1 norm with $\rho \le 7/(8c) + O(1/c^{3/2}) + o_c(1)$, which is the first improvement over the result of Indyk and Motwani (STOC 1998).

1 Introduction

The near neighbor search problem is defined as follows: given a set P of n points in a d-dimensional space, build a data structure that, given a query point q, reports any point within a given distance r to the query (if one exists). The problem is of major importance in several areas, such as databases and data mining, information retrieval, computer vision, databases and signal processing.

Many efficient near(est) neighbor algorithms are known for the case when the dimension d is "low" (e.g., see [Mei93], building on [Cla88]). However, despite decades of effort, the current solutions suffer from either space or query time that are exponential in the dimension d. This phenomenon is often called "the curse of dimensionality". To overcome this state of affairs, several researchers proposed *approximation* algorithms for the problem. In the (c, r)–approximate near neighbor problem (ANN), the data structure is allowed to return any data point whose distance from the query is at most cr, for an approximation factor $c > 1$. Many approximation algorithms for the problem are known, offering tradeoffs between the approximation factor, the space and the query time. See [And09] for an up to date survey.

From the practical perspective, the space used by an algorithm should be as close to linear as possible. If the space bound is (say) sub-quadratic, and the approximation factor c is a constant, the best existing solutions are based on *locality sensitive hashing* [IM98]. The idea of that approach is to hash the points in a way that the probability of collision is much higher for objects which are close (with the distance r) to each other than for those which are far apart (with distance at least cr). Given such hash functions, one can retrieve near neighbors by hashing the query point and retrieving elements stored in buckets containing that point. If the probability of collision is at least p_1 for the close points and at most p_2 for the far points, the algorithm solves the (c, r)–ANN using $n^{1+\rho+o(1)}$ extra space and $dn^{\rho+o(1)}$ query time[1], where $\rho = \log(1/p_1)/\log(1/p_2)$ [HPIM12]. The value of the exponent ρ depends on the distance function and the locality-sensitive hash functions

[1] Assuming that each hash function can be evaluated in $n^{o(1)}$ time, that distances can be computed in $O(d)$ time and that $1/p_1 = n^{o(1)}$.

used. In particular, it is possible to achieve $\rho = 1/c$ for the ℓ_1 norm [IM98], and $\rho = 1/c^2 + o_c(1)$ for the ℓ_2 norm [AI06].

It is known that the above bounds for the value of ρ are *tight.* Specifically, we have that, for all values of c, $\rho \geq 1/c - o_c(1)$ for the ℓ_1 norm[2] [OWZ11]. A straightforward reduction implies that $\rho \geq 1/c^2 - o_c(1)$ for the ℓ_2 norm. Thus, the running time of the simple LSH-based algorithm, which is determined by ρ, cannot be improved.

Results In this paper we show that, despite the aforementioned limitation, the space and query time bounds for ANN can be substantially improved. In particular, for the ℓ_2 norm, we give an algorithm with query time dn^{η} and space $dn + n^{1+\eta}$, where $\eta = \eta(c) \leq 7/(8c^2) + O(1/c^3) + o_c(1)$ that gives an improvement for large enough c. This also implies an algorithm with the exponent $\eta \leq 7/(8c) + O(1/c^{3/2}) + o_c(1)$ for the ℓ_1 norm, by a classic reduction from ℓ_1 to ℓ_2-squared [LLR95]. These results constitute the first improvement to the complexity of the problem since the works of [IM98] and [AI06].

Techniques Perhaps surprisingly, our results are obtained by using essentially the same LSH functions families as described in [AI06] or [IM98]. However, the properties of those hash functions that we exploit, as well as the overall algorithm, are different. On a high-level, our algorithms are obtained by combining the following two observations:

1. After a slight modification, the existing LSH functions can yield better values of the exponent ρ if the search radius r is comparable to the diameter[3] of the point-set. This is achieved by augmenting those functions with a "center point" around which the hashing is performed.

2. We can ensure that the diameter of the point-set is small by applying standard LSH functions to the original point-set P, and building a separate data structure for each bucket.

This approach leads to a two-level hashing algorithm. The *outer hash table* partitions the data sets into buckets of bounded diameter. Then, for each bucket, we build the *inner hash table*, which uses (after some pruning) the center of the minimum enclosing ball of the points in the bucket as a center point. Note that the resulting two-level hash functions cannot be "unwrapped" to yield a standard LSH family, as each bucket uses slightly different LSH functions, parametrized by different center points. That is, the two-level hashing is done in a *data dependent* manner, while the standard LSH functions are chosen from a distribution independent from the data. This enables us to overcome the lower bound of [OWZ11].

Related work In this paper we assume worst case input. If the input is generated at random, it is known that one can achieve better running times. Specifically, assume that all points are generated uniformly at random from $\{0,1\}^d$, and the query point is "planted" at distance $d/(2c)$ from its near neighbor. In this setting, the work of [CR93, GPY94, KWZ95, PRR95] gives an exponent of roughly $\frac{1}{\ln 4 \cdot c} \approx \frac{1}{1.39c}$.

[2] Assuming $1/p_1 = n^{o(1)}$.

[3] In the analysis we use a notion that is weaker than the diameter. However, we ignore this detail for now for the sake of clarity.

Even better results are known for the problem of finding the *closest pair* of points in a dataset. In particular, the algorithm of [Dub10] for the closest pair has an exponent of $1 + \frac{1}{2c-1}$.[4] More recently, [Val12] showed how to obtain an algorithm with a runtime exponent < 1.79 for any approximation $c = 1 + \varepsilon$ in the random case. Moreover, [Val12] also gives an algorithm for the worst-case closest pair problem with a runtime exponent of $2 - \Omega(\sqrt{\varepsilon})$ for $c = 1 + \varepsilon$ approximation.

There are also two related lines of lower bounds for ANN. First, the work of [MNP06] showed that LSH for Hamming space must have $\rho \geq 1/(2c) - O(1/c^2) - o_c(1)$, and [OWZ11] improved the lower bound to $\rho \geq 1/c - o_c(1)$. Second, [PTW08, PTW10] have given cell-probe lower bounds for ℓ_1 and ℓ_2, roughly showing that any randomized ANN algorithm must either use space $n^{1+\Omega(1/(tc))}$ or more than t cell–probes. We note that the LSH lower bound of $\rho \geq 1/(2c)$ from [MNP06] might more naturally predict lower bounds for ANN because it induces a "hard distribution" that corresponds to the aforementioned "random case" . In contrast, if one tries to generalize the LSH lower bound of [OWZ11] into a near neighbor hard distribution, one obtains a dataset with special structure, which one can exploit (and our algorithm will indeed exploit such structure). In fact, the LSH lower bound of [MNP06] has been used (at least implicitly) in the data structure lower bounds from [PTW08, PTW10].

Notation In the text we denote the ℓ_2 norm by $\|\cdot\|$. When we use $O(\cdot)$, $o(\cdot)$, $\Omega(\cdot)$ or $\omega(\cdot)$ we explicitly write all the parameters that the corresponding constant factors depend on as subscripts.

2 Preliminaries

Definition 1. *The* (c, r)–approximate near neighbor problem (ANN) *with failure probability f is to construct a data structure over a set of points P in metric space (X, D) supporting the following query: given any fixed query point $q \in X$, if there exists $p \in P$ with $D(p, q) \leq r$, then report some $p' \in P$ such that $D(p', q) \leq cr$, with probability at least $1 - f$.*

Remark: note that we allow preprocessing to be randomized as well, and we measure the probability of success over the random coins tossed during *both* preprocessing and query phases.

Definition 2 ([HPIM12])**.** *For a metric space (X, D) we call a family of hash functions $\mathcal{H}$ on X* (r_1, r_2, p_1, p_2)-sensitive*, if for every $x, y \in X$ we have*

- *if $D(x, y) \leq r_1$, then $\Pr_{h\sim\mathcal{H}}[h(x) = h(y)] \geq p_1$;*
- *if $D(x, y) \geq r_2$, then $\Pr_{h\sim\mathcal{H}}[h(x) = h(y)] \leq p_2$.*

Remark: for $\mathcal{H}$ to be useful we should have $r_1 < r_2$ and $p_1 > p_2$.

Definition 3. *If $\mathcal{H}$ is a family of hash functions on a metric space X, then for any $k \in \mathbb{N}$ we can define a family of hash function $\mathcal{H}^{\otimes k}$ as follows: to sample a function from $\mathcal{H}^{\otimes k}$ we sample k functions $h_1, h_2, \ldots, h_k$ from $\mathcal{H}$ independently and map $x \in X$ to $(h_1(x), h_2(x), \ldots, h_k(x))$.*

Lemma 4. *If $\mathcal{H}$ is (r_1, r_2, p_1, p_2)-sensitive, then $\mathcal{H}^{\otimes k}$ is (r_1, r_2, p_1^k, p_2^k)-sensitive.*

[4] Note that a near neighbor search algorithm with query time n^ρ and space/preprocessing time of $n^{1+\rho}$ naturally leads to a solution for the closest pair problem with the runtime of $n^{1+\rho}$.

Theorem 5 ([HPIM12]). *Suppose there is a (r, cr, p_1, p_2)-sensitive family $\mathcal{H}$ for (X, D), where $p_1, p_2 \in (0, 1)$ and let $\rho = \ln(1/p_1)/\ln(1/p_2)$. Then there exists a data structure for (c, r)–ANN over a set $P \subseteq X$ of at most n points, such that:*

- *the query procedure requires at most $O(n^\rho/p_1)$ distance computations and at most $O(n^\rho/p_1 \cdot \lceil \log_{1/p_2} n \rceil)$ evaluations of the hash functions from $\mathcal{H}$ or other operations;*
- *the data structure uses at most $O(n^{1+\rho}/p_1)$ words of space, in addition to the space needed to store the set P.*

The failure probability of the data structure can be made to be arbitrarily small constant.

Remark: this theorem says that in order to construct a good data structure for the (c, r)–ANN it is sufficient to have a (r, cr, p_1, p_2)-sensitive family $\mathcal{H}$ with small $\rho = \ln(1/p_1)/\ln(1/p_2)$ and not too small p_1.

We use the LSH family crafted in [AI06]. The properties of this family that we need are summarized in the following theorem.

Theorem 6 ([AI06]). *For every sufficiently large d and n there exists a family $\mathcal{H}$ of hash functions for ℓ_2^d such that*

- *a function from $\mathcal{H}$ can be sampled in time, stored in space, and computed in time $t^{O(t)} \cdot \log n + O(dt)$, where $t = \log^{2/3} n$;*
- *the collision probability of $\mathcal{H}$ for two points $u, v \in \mathbb{R}^d$ depends only on the distance between u and v; let us denote it by $p(\|u - v\|)$;*
- *one has the following inequalities for $p(\cdot)$:*

$$\begin{aligned} & p(1) \geq L, \\ \forall c > 1 \quad & p(c) \leq U(c), \end{aligned}$$

where

$$\begin{aligned} L &= \frac{A}{2\sqrt{t}} \cdot \frac{1}{(1 + \varepsilon + 8\varepsilon^2)^{t/2}}, \\ U(c) &= \frac{2}{(1 + c^2\varepsilon)^{t/2}}, \end{aligned}$$

where A is an absolute positive constant that is less than 1, and $\varepsilon = 1/(4t^{1/2})$.

Combining Theorem 5 and Theorem 6 one has the following corollary.

Corollary 7. *There exists a data structure for (c, r)–ANN for ℓ_2^d with preprocessing time and space $O_c(n^{1+1/c^2+o_c(1)} + nd)$ and query time $O_c(dn^{1/c^2+o_c(1)})$.*

Proof. By rescaling one can assume wlog that $r = 1$. Then, it is left to check that $L = n^{-o_c(1)}$ and

$$\ln(1/L)/\ln(1/U(c)) \leq 1/c^2 + o_c(1).$$

These computations can be found in [AI06]. □

We use the following standard estimate on tails of Gaussians (see, e.g., [KMS98]).

Lemma 8 ([KMS98])**.** *For every $t > 0$*

$$\frac{1}{\sqrt{2\pi}} \cdot \left(\frac{1}{t} - \frac{1}{t^3}\right) \cdot e^{-t^2/2} \le \mathrm{Pr}_{X \sim N(0,1)}[X \ge t] \le \frac{1}{\sqrt{2\pi}} \cdot \frac{1}{t} \cdot e^{-t^2/2}.$$

We use Johnson–Lindenstrauss dimension reduction procedure.

Theorem 9 ([JL84], [DG03])**.** *For every $d \in \mathbb{N}$ and $\varepsilon, \delta > 0$ there exists a distribution over linear maps $A\colon \mathbb{R}^d \to \mathbb{R}^{O(\log(1/\delta)/\varepsilon^2)}$ such that for every $x \in \mathbb{R}^d$ one has*

$$\mathrm{Pr}_A[\|Ax\| \in (1 \pm \varepsilon)\|x\|] \ge 1 - \delta.$$

Moreover, such a map can be sampled in time $O(d \log(1/\delta)/\varepsilon^2)$.

3 Gaussian LSH

In this section we present and analyze a $(1, c, p_1, p_2)$–sensitive family of hash functions for the ℓ_2 norm that gives an improvement upon [AI06] for the case, when all the points and queries lie on a spherical shell of radius $O(c)$ and width $O(1)$. The construction is similar to an SDP rounding scheme from [KMS98].

First, we present an "idealized" family. In the following theorem we do not care about time and space complexity and assume that all points lie on a *sphere* of radius $O(c)$.

Theorem 10. *For a sufficiently large c, every $\nu \ge 1/2$ and $1/2 \le \eta \le \nu$ there exists an LSH family for $\eta c \cdot S^{d-1} = \left\{x \in \mathbb{R}^d \mid \|x\| = \eta c\right\}$ with the ℓ_2 norm that is $(1, c, p_1, p_2)$–sensitive, where*

- *$p_1 = \exp(-o_{c,\nu}(d))$;*
- *one has*
$$\rho = \frac{\ln(1/p_1)}{\ln(1/p_2)} = \left(1 - \frac{1}{4\eta^2}\right) \cdot \frac{1}{c^2} + O_\nu\left(\frac{1}{c^3}\right) + o_{c,\nu}(1).$$

Proof. Let $\varepsilon > 0$ be a positive parameter that depends on d as follows: $\varepsilon = o(1)$ and $\varepsilon = \omega(d^{-1/2})$. Let $\mathcal{H}$ be a family of hash functions described by Algorithm 1 (the pseudocode describes how to sample $h \sim \mathcal{H}$).

Clearly for $u, v \in \eta c \cdot S^{d-1}$ with angle α between them

$$\begin{aligned}\mathrm{Pr}_{h \sim \mathcal{H}}[h(u) = h(v)] &= \frac{\mathrm{Pr}_{w \sim N(0,1)^d}[\langle u, w\rangle \ge \eta c \cdot \varepsilon\sqrt{d} \wedge \langle v, w\rangle \ge \eta c \cdot \varepsilon\sqrt{d}]}{\mathrm{Pr}_{w \sim N(0,1)^d}[\langle u, w\rangle \ge \eta c \cdot \varepsilon\sqrt{d} \vee \langle v, w\rangle \ge \eta c \cdot \varepsilon\sqrt{d}]} \\ &= \Theta(1) \cdot \frac{\mathrm{Pr}_{X,Y \sim N(0,1)}[X \ge \varepsilon\sqrt{d} \wedge \cos\alpha \cdot X - \sin\alpha \cdot Y \ge \varepsilon\sqrt{d}]}{\mathrm{Pr}_{X \sim N(0,1)}[X \ge \varepsilon\sqrt{d}]} \\ &= \Theta(\varepsilon\sqrt{d}) \cdot \frac{\mathrm{Pr}_{X,Y \sim N(0,1)}[X \ge \varepsilon\sqrt{d} \wedge \cos\alpha \cdot X - \sin\alpha \cdot Y \ge \varepsilon\sqrt{d}]}{e^{-\varepsilon^2 d/2}}. \quad (1)\end{aligned}$$

In the last equality we use Lemma 8 and the fact that $\varepsilon = \omega(d^{-1/2})$.

The following two lemmas allow us to estimate the numerator of the right–hand side of (1).

Algorithm 1 Gaussian partitioning

$\mathcal{P} \leftarrow \emptyset$ ▷ eventually, $\mathcal{P}$ will be a partition of $\eta c \cdot S^{d-1}$
while $\bigcup \mathcal{P} \neq \eta c \cdot S^{d-1}$ **do** ▷ we denote $\bigcup \mathcal{P}$ the union of all sets that belong to $\mathcal{P}$
  sample $w \sim N(0,1)^d$
  $S \leftarrow \left\{ u \in \eta c \cdot S^{d-1} \mid \langle u, w \rangle \geq \eta c \cdot \varepsilon \sqrt{d} \right\} \setminus \bigcup \mathcal{P}$
  if $S \neq \emptyset$ **then**
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{S\}$
  end if
end while
define h to be the function that maps a point $u \in \eta c \cdot S^{d-1}$ to the part of $\mathcal{P}$ that it belongs to

Lemma 11.

$$\Pr_{X,Y \sim N(0,1)}[X \geq \varepsilon\sqrt{d} \wedge \cos\alpha \cdot X - \sin\alpha \cdot Y \geq \varepsilon\sqrt{d}] = O\left(\frac{e^{-\varepsilon^2 d \cdot (1+\tan^2 \frac{\alpha}{2})/2}}{\varepsilon\sqrt{d}}\right).$$

Proof.

$$\begin{aligned}
\Pr_{X,Y \sim N(0,1)}&[X \geq \varepsilon\sqrt{d} \wedge \cos\alpha \cdot X - \sin\alpha \cdot Y \geq \varepsilon\sqrt{d}] \\
&\leq \Pr_{X,Y \sim N(0,1)}[(1+\cos\alpha) \cdot X - \sin\alpha \cdot Y \geq 2\varepsilon\sqrt{d}] \\
&= \Pr_{Z \sim N(0,1)}[\sqrt{(1+\cos\alpha)^2 + \sin^2\alpha} \cdot Z \geq 2\varepsilon\sqrt{d}] \\
&= \Pr_{Z \sim N(0,1)}[\sqrt{2 \cdot (1+\cos\alpha)} \cdot Z \geq 2\varepsilon\sqrt{d}] = O\left(\frac{e^{-\varepsilon^2 d \cdot (1+\tan^2 \frac{\alpha}{2})/2}}{\varepsilon\sqrt{d}}\right)
\end{aligned}$$

In the last equality we used Lemma 8, the fact that $\varepsilon = \omega(d^{-1/2})$ and the identity

$$\frac{2}{1+\cos\alpha} = 1 + \tan^2\frac{\alpha}{2}.$$

□

Lemma 12. *If* $0 \leq \alpha < \alpha_0$ *for some* constant $0 < \alpha_0 < \pi/2$*, then*

$$\Pr_{X,Y \sim N(0,1)}[X \geq \varepsilon\sqrt{d} \wedge \cos\alpha \cdot X - \sin\alpha \cdot Y \geq \varepsilon\sqrt{d}] = \Omega\left(\frac{e^{-\varepsilon^2 d \cdot (1+\tan^2 \frac{\alpha_0}{2})/2}}{\varepsilon^2 d \cdot \tan\frac{\alpha_0}{2}}\right).$$

Proof.

$$\begin{aligned}
\Pr_{X,Y \sim N(0,1)}&[X \geq \varepsilon\sqrt{d} \wedge \cos\alpha \cdot X - \sin\alpha \cdot Y \geq \varepsilon\sqrt{d}] \\
&\geq \Pr_{X,Y \sim N(0,1)}[X \geq \varepsilon\sqrt{d} \wedge Y \leq -\tan\frac{\alpha}{2} \cdot \varepsilon\sqrt{d}] \\
&= \Pr_{X \sim N(0,1)}[X \geq \varepsilon\sqrt{d}] \cdot \Pr_{Y \sim N(0,1)}[Y \geq \tan\frac{\alpha}{2} \cdot \varepsilon\sqrt{d}] \\
&\geq \Pr_{X \sim N(0,1)}[X \geq \varepsilon\sqrt{d}] \cdot \Pr_{Y \sim N(0,1)}[Y \geq \tan\frac{\alpha_0}{2} \cdot \varepsilon\sqrt{d}] = \Omega\left(\frac{e^{-\varepsilon^2 d \cdot (1+\tan^2 \frac{\alpha_0}{2})/2}}{\varepsilon^2 d \cdot \tan\frac{\alpha_0}{2}}\right).
\end{aligned}$$

In the first inequality we use that for $\alpha < \alpha_0 < \pi/2$ the right–hand side event implies the left–hand side event. Indeed,

$$\cos\alpha \cdot X - \sin\alpha \cdot Y \geq \cos\alpha \cdot \varepsilon\sqrt{d} + \sin\alpha \cdot \tan\frac{\alpha}{2} \cdot \varepsilon\sqrt{d} = \varepsilon\sqrt{d}.$$

In the last equality we used Lemma 8, the fact that α_0 is constant and $\varepsilon = \omega(d^{-1/2})$. □

Thus, combining (1), Lemma 11 and Lemma 12, we have the following estimates on the probability of collision.

Lemma 13. *One has*

$$\ln\frac{1}{\Pr_{h\sim\mathcal{H}}[h(u) = h(v)]} \geq \frac{\varepsilon^2 d}{2} \cdot \tan^2\frac{\alpha}{2} - O(1);$$

and if $\alpha < \alpha_0$ for some constant $0 < \alpha_0 < \pi/2$, then

$$\ln\frac{1}{\Pr_{h\sim\mathcal{H}}[h(u) = h(v)]} \leq \frac{\varepsilon^2 d}{2} \cdot \tan^2\frac{\alpha_0}{2} + \ln\left(\varepsilon\sqrt{d} \cdot \tan\frac{\alpha_0}{2}\right) + O(1).$$

Since

$$\tan^2\frac{\alpha}{2} = \frac{\|u-v\|^2/(\eta c)^2}{4 - \|u-v\|^2/(\eta c)^2},$$

by setting $\varepsilon = d^{-1/4}$ and invoking Lemma 13 for the angles that correspond to distances 1 and c, we have

$$\begin{aligned} \ln\frac{1}{p_1} &\leq \frac{\sqrt{d}}{2} \cdot \frac{1/(\eta c)^2}{4 - 1/(\eta c)^2} + O_{c,\nu}(\ln d), \\ \ln\frac{1}{p_2} &\geq \frac{\sqrt{d}}{2} \cdot \frac{1/\eta^2}{4 - 1/\eta^2} - O(1). \end{aligned}$$

Note that here we use that c is large enough, since we must have $\alpha_0 < \pi/2$ in order to be able to apply Lemma 13.

Thus, we have $p_1 = \exp(-o_{c,\nu}(d))$. A similar estimate holds for p_2 provided that η is separated from $1/2$. Therefore

$$\rho = \frac{\ln(1/p_1)}{\ln(1/p_2)} = \frac{4 - 1/\eta^2}{4 - 1/(\eta c)^2} \cdot \frac{1}{c^2} + o_{c,\nu}(1) = \left(1 - \frac{1}{4\eta^2}\right) \cdot \frac{1}{c^2} + O_\nu\left(\frac{1}{c^3}\right) + o_{c,\nu}(1).$$

□

Remark: we could have had $O_\nu(1/c^4)$ term in the expression for ρ, but we state the theorem with $O_\nu(1/c^3)$ in order to be consistent with the next theorem.

Now we show how to convert this "idealized" family to a real one.

Theorem 14. *For a sufficiently large c, every $\nu \geq 1/2$ and $1/2 \leq \eta \leq \nu$ there exists an LSH family $\mathcal{H}$ for*

$$\left\{x \in \mathbb{R}^d \mid \|x\| \in [\eta c - 1; \eta c + 1]\right\}$$

with the ℓ_2 norm such that

- *it satisfies the conclusion of Theorem 10;*
- *for every $k \in \mathbb{N}$ one can sample a function from $\mathcal{H}$ in time $\exp(o(d))$, store it in space $\exp(o(d))$ and query in time $\exp(o(d))$.*

Proof. We use the family from the proof of Theorem 10, but with two modifications. First, if we want to compute $h(x)$ for $h \sim \mathcal{H}$, then before doing so, we normalize x to the length ηc. Second, in Algorithm 1 instead of checking the condition $\bigcup \mathcal{P} = \eta c \cdot S^{d-1}$, we simply run the partitioning process for $\exp(o(d))$ steps. Namely, we require that after the end the probability of the event $\bigcup \mathcal{P} = \eta c \cdot S^{d-1}$ is at least $1 - \exp(-d)$ (one can see that this will be the case after $\exp(o(d))$ steps by a standard ε-net argument). Such a high probability means that this LSH family achieves the same parameters as the one from Theorem 10. Clearly, such a function can be stored in space $\exp(o(d))$ and queried in time $\exp(o(d))$.

It is left to argue that normalizing a vector before computing h does not affect the quality (namely, we are interested in p_1, p_2 and ρ) by a lot.

Lemma 15. *For any vectors u and v,*

$$\|u/\|u\| - v/\|v\|\|^2 = \frac{1}{\|u\| \cdot \|v\|} \Big((\|u - v\|^2 - |\|u\| - \|v\||^2) \Big)$$

Proof.

$$\begin{aligned} \|u/\|u\| - v/\|v\|\|^2 &= 2 - \frac{2\langle u, v\rangle}{\|u\| \cdot \|v\|} \\ &= \frac{1}{\|u\| \cdot \|v\|} \Big((\|u - v\|^2 - |\|u\| - \|v\||^2) \Big) \end{aligned}$$

□

By the above lemma, one can check that for $u, v \in \Big\{ x \in \mathbb{R}^d \mid \|x\| \in [\eta c - 1; \eta c + 1] \Big\}$

- if $\|u - v\| \le 1$, then

$$\begin{aligned} (\eta c \cdot \|u/\|u\| - v/\|v\|\|)^2 &\le \frac{(\eta c)^2}{(\eta c - 1)^2} \\ &\le 1 + O_\nu\left(\frac{1}{c}\right) \end{aligned}$$

- if $\|u - v\| \ge c$, then

$$\begin{aligned} (\eta c \cdot \|u/\|u\| - v/\|v\|\|)^2 &\ge \frac{(\eta c)^2}{(\eta c + 1)^2}(c^2 - 4) \\ &\ge c^2 \cdot \left(1 - O_\nu\left(\frac{1}{c}\right)\right). \end{aligned}$$

Clearly, from these inequalities we can see that the conclusion of Theorem 10 is still true for our case. □

4 Two–level hashing

We now describe our near neighbor data structure. The data structure is composed of several independent data structures, where each one is a two-level hashing scheme, described next. We will conclude with proving our main theorem for ANN search.

Construction

We want to solve $(c, 1)$–ANN for ℓ_2^d. As a first step, we apply Johnson–Lindenstrauss transform (Thereom 9) and reduce our problem to $(c-1, 1)$–ANN for $\ell_2^{O_c(\log n)}$ by increasing the failure probability by an arbitrarily small constant. This means that all quantities of order $\exp(o(d))$ are now $n^{o_c(1)}$. Abusing notation, let us assume that we are solving $(c, 1)$–ANN in $\ell_2^{O_c(\log n)}$.

Preprocessing

Let $\tau > 1$ be a constant parameter that we will choose later. We consider the following *two-level* hashing scheme. It consists of an *outer hash table* and several *inner hash tables*.

First, let us construct an *outer hash table*. We hash all the points from P using a function from $\mathcal{H}_1^{\otimes k_0}$, where $\mathcal{H}_1$ is the hash family from Theorem 6, and k_0 is the smallest positive integer such that

$$\left(\frac{U(\tau c - 1)}{L}\right)^{k_0} \leq \frac{1}{100n} \tag{2}$$

(L and $U(\cdot)$ are from Theorem 6). Then, for every non–empty bucket we do the following. While there exist a pair of points in a bucket with distance more than τc, we remove both of them. Let us call all the removed points *filtered out*. If there are no points left, we proceed to the next bucket. Otherwise, we find a $(1 + 1/c)$–approximation to the minimum enclosing ball for the remaining points. Such a ball can be found in time $O_c(n \log n)$ using the algorithm from [BC03]. Note that by Jung's theorem (for a modern treatment see Exercise 1.3.5 in [Mat02]) the radius of this ball is at most

$$\left(1 + \frac{1}{c}\right) \cdot \frac{\tau c}{\sqrt{2}},$$

since the diameter of the set of the remaining points is at most τc. For every non–empty bucket we store the center of its ball. In addition to it, we store a remaining point that is closest to the center.

For a point $u \in \mathbb{R}^d$ let $B(u) \subseteq P$ be the points from the bucket of u that are not filtered out, and let $p_0(u)$ be the center of the corresponding ball (provided that $B(u)$ is non–empty). Let $s(u)$ denote the closest to $p_0(u)$ point from $B(u)$.

Second, let us show how to construct *inner hash tables*. We consider buckets one by one. Let p_0 be a center of a ball that corresponds to a non–empty bucket whose set of points we denote by R. For every integer $0 \leq l \leq \left\lceil (1 + 1/c)\tau c/\sqrt{2} - c/2 \right\rceil$ we consider a set

$$P_l = \{p - p_0 \mid p \in R, \|p - p_0\| \in [c/2 + l - 1; c/2 + l + 1]\}.$$

We hash every P_l using $\mathcal{H}_2^{\otimes k}$, where $\mathcal{H}_2$ is the LSH family from Theorem 14 and k is the smallest positive integer such that for every $u, v \in \mathbb{R}^d$ with $\|u\|, \|v\| \in [c/2+l-1; c/2+l+1]$ and $\|u-v\| \geq c$ we have

$$U(c)^{k_0} \cdot \mathrm{Pr}_{h \sim \mathcal{H}_2^{\otimes k}}[h(u) = h(v)] \leq \frac{1}{3n}. \tag{3}$$

Query

Suppose that $q \in \mathbb{R}^d$ is a query point. To query the two-level data structure we do the following. If $B(q)$ is empty, we stop. If $\|q - s(q)\| \leq c$, then we output $s(q)$ and stop. Otherwise, we locate all non-empty P_l's such that $[c/2 + l - 1; c/2 + l + 1] \ni \|q - p_0(q)\|$, consider the $(q - p_0(q))$'s buckets in the corresponding inner hash tables and enumerate all the points from these buckets. If we find a point $p \in P$ such that $\|p - q\| \leq c$, then we output it and stop.

Analysis

Let q be a query point. We want to analyze collision probabilities for a point $p \in P$ and q. Let $\mathcal{A}$ be the event "p and q collide in the outer hash table", $\mathcal{B}$ be the event "every point from P that is more than $\tau c - 1$ apart from q does not collide with q in the outer hash table", $\mathcal{C}$ be the event "p and q collide in an inner hash table, where we search for $q - p_0(q)$" and $\mathcal{D}$ be the event "$\|q - s(q)\| \leq c$".

Lemma 16. *If $\|p - q\| \geq c$, then*

$$\Pr[\mathcal{C}] \leq \frac{1}{n}.$$

Proof. Since $\mathcal{C}$ implies $\mathcal{A}$,

$$\Pr[\mathcal{C}] = \Pr[\mathcal{A}] \cdot \Pr[\mathcal{C} \mid \mathcal{A}]$$

By (3) and the fact that we query at most 3 inner hash tables, the latter quantity is at most $1/n$. ☐

Lemma 17. *If $\|p - q\| \leq 1$, then*

$$\Pr[\mathcal{C} \vee \mathcal{D}] \geq Q = n^{-\left(1 - \frac{1}{2\tau^2} + \frac{1}{2\tau^4}\right) \cdot \frac{1}{c^2} + O_\tau\left(\frac{1}{c^3}\right) + o_{\tau,c}(1)}.$$

Proof. Since $\mathcal{C}$ and $\mathcal{D}$ are disjoint, we have

$$\begin{aligned}\Pr[\mathcal{C} \vee \mathcal{D}] \geq \Pr[\mathcal{A} \wedge \mathcal{B} \wedge (\mathcal{C} \vee \mathcal{D})] &= \Pr[\mathcal{A}] \cdot \Pr[\mathcal{B} \mid \mathcal{A}] \cdot \Pr[\mathcal{C} \vee \mathcal{D} \mid \mathcal{A}, \mathcal{B}] \\ &\geq \Pr[\mathcal{A}] \cdot \Pr[\mathcal{B} \mid \mathcal{A}] \cdot \Pr[\mathcal{C} \mid \mathcal{A}, \mathcal{B}, \neg\mathcal{D}]\end{aligned}$$

(in the last inequality we use that if $\mathcal{U}$ and $\mathcal{V}$ are two disjoint events, then $\Pr[\mathcal{U} \vee \mathcal{V}] \geq \Pr[\mathcal{U} \mid \neg\mathcal{V}]$). Let us go through these probabilities one by one. From Theorem 6 we have $\Pr[\mathcal{A}] \geq L^{k_0}$.

Lemma 18.

$$\Pr[\mathcal{C} \mid \mathcal{A}, \mathcal{B}, \neg\mathcal{D}] \geq \left(\frac{1}{3n} \cdot \frac{1}{U(c)^{k_0}}\right)^{\left(1 - \frac{1}{2\tau^2}\right) \cdot \frac{1}{c^2} + O_\tau\left(\frac{1}{c^3}\right) + o_{\tau,c}(1)}$$

Proof. First, observe that if $\mathcal{A}$ and $\mathcal{B}$ hold, then p is not filtered out. Second, since we condition on $\|s(q) - q\| > c$ we have $\|p - p_0(q)\|, \|q - p_0(q)\| > (c-1)/2$, so p will be in some S_l and thus from Theorem 14 (invoked for $\nu = \tau/\sqrt{2} + O_\tau(1/c)$) and (3) we have the desired bound. ☐

Now let us bound $\Pr[\mathcal{B} \mid \mathcal{A}]$ from below.

Lemma 19.

$$\Pr[\mathcal{B} \mid \mathcal{A}] \geq 0.99$$

Proof. We will prove that $\Pr[\neg\mathcal{B} \mid \mathcal{A}] \le 0.01$. Clearly,

$$\begin{aligned}
\Pr[\neg\mathcal{B} \mid \mathcal{A}] &\le \sum_{\substack{p' \in P: \\ \|q-p'\| > \tau c - 1}} \Pr[q \text{ and } p' \text{ collide in the outer hash table} \mid \mathcal{A}] \\
&\le \sum_{\substack{p' \in P: \\ \|q-p'\| > \tau c - 1}} \frac{\Pr[q \text{ and } p' \text{ collide in the outer hash table}]}{\Pr[\mathcal{A}]} \\
&\le n \cdot \left(\frac{U(\tau c - 1)}{L}\right)^{k_0} \le 0.01.
\end{aligned}$$

The last inequality is due to (2). □

In order to combine the estimates and prove the lemma it is left to estimate L^{k_0} and $U(c)^{k_0}$ (using (2)). Because for any constant x, $\frac{\ln 1/L}{\ln 1/U(x)} = (1+o(1))x^{-2}$, we have

$$U(\tau c - 1) \le L^{(1-o(1))(\tau c-1)^2}$$
$$U(c) \le L^{(1-o(1))c^2}$$

By the definition of k_0 and the fact that $k_0 = \omega(1)$,

$$\frac{1}{100n} \le \left(\frac{U(\tau c - 1)}{L}\right)^{k_0 - 1} \le L^{(1-o(1))(\tau^2 c^2 - 2\tau c)k_0}$$

In other words,

$$L^{k_0} \ge n^{-(1+o(1))/(\tau^2 c^2 - 2\tau c)}$$

Combining all these estimates, we can finally bound $\Pr[\mathcal{C} \vee \mathcal{D}]$.

$$\begin{aligned}
\Pr[\mathcal{C} \vee \mathcal{D}] &\ge n^{-(1-\frac{1}{2\tau^2})\cdot\frac{1}{c^2} - O_\tau\left(\frac{1}{c^3}\right) - o_{\tau,c}(1)} L^{k_0(1-(1-o(1))(1-\frac{1}{2\tau^2}) - O_\tau\left(\frac{1}{c}\right) - o_{\tau,c}(1))} \\
&\ge n^{-(1-\frac{1}{2\tau^2})\cdot\frac{1}{c^2} - O_\tau\left(\frac{1}{c^3}\right) - o_{\tau,c}(1)} L^{k_0(1+o(1))\frac{1}{2\tau^2}} \\
&\ge n^{-(1-\frac{1}{2\tau^2})\cdot\frac{1}{c^2} - \frac{1}{2\tau^2(\tau^2c^2 - 2\tau c)} - O_\tau\left(\frac{1}{c^3}\right) - o_{\tau,c}(1)} \\
&\ge n^{-(1-\frac{1}{2\tau^2}+\frac{1}{2\tau^4})\cdot\frac{1}{c^2} - O_\tau\left(\frac{1}{c^3}\right) - o_{\tau,c}(1)}
\end{aligned}$$

□

Finally, we are ready to prove our main result.

Theorem 20. *There exists a data structure for $(c,1)$–ANN for ℓ_2^d that has preprocessing time and space $O_c(n^{1+\rho} + d \cdot \log n)$ and query time $O_c(n^\rho + d \cdot \log n)$, where*

$$\rho \le \frac{7/8}{c^2} + O\left(\frac{1}{c^3}\right) + o_c(1).$$

Proof. We set $\tau = \sqrt{2}$ and then proceed exactly as in the proof of Theorem 5 (see [HPIM12]). The details are included for completeness.

The data structure consists of $1/Q$ independent copies of the two-level hashing scheme, where Q is the lower bound of $\Pr[\mathcal{C} \vee \mathcal{D}]$ in Lemma 17. Given a query q, the algorithm looks for a near neighbor in all the copies and stops when a near neighbor is found or more than $3/Q+1$ points have been examined. Observe that the space of the algorithm is $O_c(n^{1+o_c(1)}Q + d\log n)$ and the query time is $O_c(n^{o_c(1)}Q + d\log n)$ (the $O(d\log n)$ terms come from storing and applying the matrix of the Johnson-Lindenstrauss transform). It remains to argue that the algorithm succeeds with constant probability. Assume that there exists p^* such that $\|q - p^*\| \leq 1$.

By Lemma 16, in each copy, the expected number of points in that are more than c apart from q colliding with q in some inner hash table is at most 1. The expected number of collisions in $1/Q$ tables is at most $1/Q$. Thus, by Markov's inequality, the probability that it exceeds $3/Q$ is at most $1/3$.

Next, we compute the probability that p^* collides with q or the event $\mathcal{D}$ happens (in which case we succeed as well) in some copy. It is bounded from below by $1 - (1-Q)^{1/Q} \geq 1 - 1/e$.

Therefore, with probability at least $1 - 1/3 - 1/e$, after examining $3/Q + 1$ points, the algorithm finds some point within distance c from q. □

References

[AI06] Alexandr Andoni and Piotr Indyk. Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions. In *FOCS*, pages 459–468, 2006.

[And09] Alexandr Andoni. *Nearest Neighbor Search: the Old, the New, and the Impossible.* PhD thesis, MIT, 2009.

[BC03] Mihai Badoiu and Kenneth L. Clarkson. Smaller core-sets for balls. In *SODA*, pages 801–802, 2003.

[Cla88] Kenneth L. Clarkson. A Randomized Algorithm for Closest-Point Queries. *SIAM J. Comput.*, 17(4):830–847, 1988.

[CR93] Andrea Califano and Isidore Rigoutsos. FLASH: a fast look-up algorithm for string homology. In *International Conference on Intelligent Systems for Molecular Biology*, pages 56–64, 1993.

[DG03] Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Struct. Algorithms*, 22(1):60–65, 2003.

[Dub10] Moshe Dubiner. Bucketing coding and information theory for the statistical high-dimensional nearest-neighbor problem. *IEEE Transactions on Information Theory*, 56(8):4166–4179, 2010.

[GPY94] Daniel H. Greene, Michal Parnas, and F. Frances Yao. Multi-index hashing for information retrieval. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 722–731, 1994.

[HPIM12] Sariel Har-Peled, Piotr Indyk, and Rajeev Motwani. Approximate nearest neighbor: Towards removing the curse of dimensionality. *Theory Of Computing*, 8:321–350, 2012.

[IM98] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613. ACM, 1998.

[JL84] William B. Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *Conference in modern analysis and probability (New Haven, Conn., 1982)*, volume 26 of *Contemp. Math.*, pages 189–206. Amer. Math. Soc., 1984.

[KMS98] David R. Karger, Rajeev Motwani, and Madhu Sudan. Approximate Graph Coloring by Semidefinite Programming. *J. ACM*, 45(2):246–265, 1998.

[KWZ95] Richard M. Karp, Orli Waarts, and Geoffrey Zweig. The bit vector intersection problem. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 621–630, 1995.

[LLR95] Nathan Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.

[Mat02] Jiří Matoušek. *Lectures on Discrete Geometry.* Springer-Verlag New York, Inc., 2002.

[Mei93] Stefan Meiser. Point Location in Arrangements of Hyperplanes. *Inf. Comput.*, 106(2):286–303, 1993.

[MNP06] Rajeev Motwani, Assaf Naor, and Rina Panigrahy. Lower bounds on locality sensitive hashing. In *Proceedings of the ACM Symposium on Computational Geometry (SoCG)*, pages 253–262, 2006.

[OWZ11] Ryan O'Donnell, Yi Wu, and Yuan Zhou. Optimal lower bounds for locality sensitive hashing (except when q is tiny). In *ICS*, pages 275–283, 2011.

[PRR95] Ramamohan Paturi, Sanguthevar Rajasekaran, and John Reif. The light bulb problem. *Information and Computation*, 117(2):187–192, 1995.

[PTW08] Rina Panigrahy, Kunal Talwar, and Udi Wieder. A geometric approach to lower bounds for approximate near-neighbor search and partial match. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 414–423, 2008.

[PTW10] Rina Panigrahy, Kunal Talwar, and Udi Wieder. Lower bounds on near neighbor search via metric expansion. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, 2010.

[Val12] Gregory Valiant. Finding correlations in subquadratic time, with applications to learning parities and juntas with noise. In *FOCS. Preliminary version as ECCC TR12-006*, 2012.