Low-quality dimension reduction and high-dimensional Approximate Nearest Neighbor

Ioannis Psarros

Submitted to the Department of Mathematics
Graduate Program in Logic, Algorithms and Computation
$\boldsymbol{\mu} \prod \boldsymbol{\lambda} \forall$

at the

UNIVERSITY OF ATHENS

February 2015

Low-quality dimension reduction and high-dimensional Approximate Nearest Neighbor

by

Ioannis Psarros

Submitted to the Department of Mathematics
Graduate Program in Logic, Algorithms and Computation
$\boldsymbol{\mu}\prod\boldsymbol{\lambda}\forall$

Abstract

The approximate nearest neighbor problem (ANN) in Euclidean settings is a fundamental question, which has been addressed by two main approaches: Data-dependent space partitioning techniques perform well when the dimension is relatively low, but are affected by the curse of dimensionality. On the other hand, locality sensitive hashing has polynomial dependence in the dimension, sublinear query time with an exponent inversely proportional to the error factor ϵ, and subquadratic space requirement.

We generalize the Johnson-Lindenstrauss lemma to define "low-quality" mappings to a Euclidean space of significantly lower dimension, such that they satisfy a requirement weaker than approximately preserving all distances or even preserving the nearest neighbor. This mapping guarantees, with arbitrarily high probability, that an approximate nearest neighbor lies among the k approximate nearest neighbors in the projected space. This leads to a randomized tree based data structure that avoids the curse of dimensionality for $(1+\epsilon)$-ANN. Our algorithm, given n points in dimension d, achieves space usage in $O(dn)$, preprocessing time in $O(dn\log n)$, and query time in $O(dn^{\rho}\log n)$, where ρ is proportional to $1-1/\ln\ln n$, for fixed $\epsilon \in (0,1)$. It employs a data structure, such as BBD-trees, that efficiently finds k approximate nearest neighbors. The dimension reduction is larger if one assumes that pointsets possess some structure, namely bounded expansion rate.

Thesis Supervisor: Ioannis Z. Emiris
Title: Professor

Acknowledgments

First of all I would like to thank my parents, as well as the rest of my family for their continued support. I would also like to thank my advisor Ioannis Z. Emiris for his advices and all the stimulating conversations we have. Furthermore, special thanks to all members of the Erga lab for patiently answering my questions and creating more.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nearest neighbor searching is a fundamental computational problem. Let X be a set of n points in $\mathbb{R}^d$ and let $d(p, p')$ be the (Euclidean) distance between any two points p and p'. The problem consists in reporting, given a query point q, a point $p \in X$ such that $d(p, q) \leq d(p', q)$, for all $p' \in X$ and p is said to be a "nearest neighbor" of q. For this purpose, we preprocess X into a data structure. However, an exact solution to high-dimensional nearest neighbor search, in sublinear time, requires prohibitively heavy resources. Thus, many techniques focus on the less demanding task of computing the approximate nearest neighbor (ANN). Given a parameter $\epsilon \in (0, 1)$, a $(1 + \epsilon)$-approximate nearest neighbor to a query q is a point p in X such that $d(q, p) \leq (1 + \epsilon) \cdot d(q, p')$, $\forall p' \in X$. Hence, under approximation, the answer can be any point whose distance from q is at most $(1 + \epsilon)$ times larger than the distance between q and its nearest neighbor. We can naturally have more than one approximate nearest neighbors and the nearest neighbor is also an approximate nearest neighbor as depicted in Figure 1-1.

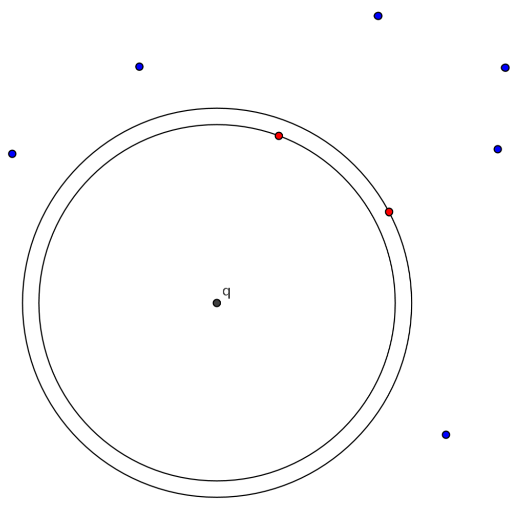

Figure 1-1: Two approximate nearest neighbors.

1.1 Existing work

As it was mentioned above, an exact solution to high-dimensional nearest neighbor search, in sublinear time, requires heavy resources. One notable solution to the problem [Mei93] shows that nearest neighbor queries can be answered in $O(d^5 \log n)$ time, using $O(n^{d+\delta})$ space, for arbitrary $\delta > 0$.

One class of methods for ϵ-ANN may be called data-dependent, since the decisions taken for partitioning the space are affected by the given data points. In [AMN+98] they introduced the Balanced Box-Decomposition (BBD) trees. The BBD-trees data structure achieves query time $O(c \log n)$ with $c \leq d/2\lceil 1 + 6d/\epsilon \rceil^d$, using space in $O(dn)$, and preprocessing time in $O(dn \log n)$. BBD-trees can be used to retrieve the $k \geq 1$ approximate nearest-neighbors at an extra cost of $O(d \log n)$ per neighbor. BBD-trees have proved to be very practical, as well, and have been implemented in software library `ANN`.

Another data structure is the Approximate Voronoi Diagrams (AVD). They are shown to establish a tradeoff between the space complexity of the data structure and the query time it supports [AMM09]. With a tradeoff parameter $2 \leq \gamma \leq \frac{1}{\epsilon}$, the query time is $O(\log(n\gamma) + 1/(\epsilon\gamma)^{\frac{d-1}{2}})$ and the space is $O(n\gamma^{d-1} \log \frac{1}{\epsilon})$. They are implemented on a hierarchical quadtree-based subdivision of space into cells, each storing a number of representative points, such that for any query point lying in the cell, at least one of the representatives is an approximate nearest neighbor. Further improvements to the space-time trade offs for ANN, are obtained in [AdFM11].

One might directly apply the celebrated Johnson-Lindenstrauss lemma and map the points to $O(\frac{\log n}{\epsilon^2})$ dimensions with distortion equal to $1 + \epsilon$ in order to improve space requirements. In particular, AVD combined with the Johnson-Lindenstrauss lemma require $n^{O(\log \frac{1}{\epsilon}/\epsilon^2)}$ space which is prohibitive if $\epsilon \ll 1$ and query time polynomial in $\log n$, d and $1/\epsilon$. On the other hand BBD-trees combined with the JL lemma require $O(dn)$ space but query time superlinear in n. Notice that we relate the approximation error with the distortion for simplicity. Our approach (Theorem 10) requires $O(dn)$ space and has query time sublinear in n and polynomial in d.

In high dimensional spaces, data dependent data structures are affected by the curse of dimensionality. This means that, when the dimension increases, either the query time or the required space increases exponentially.

It is known [HIN12] that the $(1+\epsilon)$-ANN problem reduces to the $(1+\epsilon, R)$-Approximate Near Neighbor problem with a roughly logarithmic increase in storage requirement and query time. A data structure which solves the $(1+\epsilon, R)$-ANN problem answers queries of the following form: for some query q if there exists a point in distance $\leq R$ return a point in distance $\leq (1+\epsilon)R$. The $(1+\epsilon, R)$-ANN problem is already hard when the dimension is high.

An important method conceived for the $(1+\epsilon, R)$-ANN problem for high dimensional data is locality sensitive hashing (LSH). LSH induces a data independent space partition and is dynamic, since it supports insertions and deletions. It relies on the existence of locality sensitive hash functions, which are more likely to map similar objects to the same bucket. The existence of such functions depends on the metric space. In general, LSH requires roughly $O(dn^{1+\rho})$ space and $O(dn^{\rho})$ query time for some parameter $\rho \in (0, 1)$. In

[AI08] they show that in the Euclidean case, one can have $\rho \leq \frac{1}{(1+\epsilon)^2}$ which matches the lower bound of hashing algorithms proved in [OWZ11]. Lately, it was shown that it is possible to overcome this limitation with an appropriate change in the scheme which achieves $\rho \leq \frac{7}{8(1+\epsilon)^2} + O(\frac{1}{(1+\epsilon)^3}) + o(1)$ [AINR14]. One approach with better space requirement is also achieved in [And09]. In particular, they present a data structure for the $(1+\epsilon, c)$-ANN problem achieving near-linear space and $dn^{O(1/(1+\epsilon)^2)}$ query time. One different approach [Pan06] achieves near linear space but query time proportional to $O(dn^{\frac{2}{1+\epsilon}})$. For comparison, in Theorem 10 we show that it is possible to use $O(dn)$ space, with query time roughly $O(dn^{\rho})$ where $\rho < 1$ is now higher than the one appearing in LSH.

Exploiting the structure of the input is an important way to improve the complexity of nearest neighbor search. In particular, significant amount of work has been done for pointsets with low doubling dimension. In [HPM05], they provide an algorithm for ANN with expected preprocessing time $O(2^{\dim(X)} n \log n)$, space $O(2^{\dim(X)} n)$ and query time $O(2^{\dim(X)} \log n + \epsilon^{-O(\dim(X))})$ for any finite metric space X of doubling dimension $\dim(X)$. In [IN07] they provide randomized embeddings that preserve nearest neighbor with constant probability, for points lying on low doubling dimension manifolds in Euclidean settings. Naturally, such an approach can be easily combined with any known data structure for ANN.

In [DF08] they present random projection trees which adapt to pointsets of low doubling dimension. Like kd-trees, every split partitions the pointset into subsets of roughly equal cardinality; in fact, instead of splitting at the median, they add a small amount of "jitter". Unlike kd-trees, the space is split with respect to a random direction, not necessarily parallel to the coordinate axes. Classic *kd*-trees also adapt to the doubling dimension of randomly rotated data [Vem12]. However, for both techniques, no related theoretical arguments about the efficiency of ϵ-ANN search were given.

In [KR02], they introduce a different notion of intrinsic dimension for an arbitrary metric space, namely its expansion rate c; it is formally defined in section 3.1. The doubling dimension is a more general notion of intrinsic dimension in the sense that, when a finite metric space has bounded expansion rate, then it also has bounded doubling dimension, but the converse does not hold [GKL03]. Several efficient solutions are known for metrics with bounded expansion rate, including for the problem of exact nearest neighbor. In [KL04], they present a data structure which requires $c^{O(1)}n$ space and answers queries in $c^{O(1)} \ln n$. Cover Trees [BKL06] require $O(n)$ space and each query costs $O(c^{12} \log n)$ time for exact nearest neighbors. In Theorem 13, we provide a data structure for the ϵ-ANN problem with linear space and $O((C^{1/\epsilon^3} + \log n) d \log n/\epsilon^2)$ query time, where C depends on c. The result concerns pointsets in the d-dimensional Euclidean space.

1.2 Our contribution

Tree based space partitioning techniques perform well when the dimension is relatively low, but are affected by the curse of dimensionality. To that end, randomized methods like Locality Sensitive Hashing are more efficient when the dimension is high. One may also apply the Johnson-Lindenstrauss Lemma to improve upon standard space partitioning techniques, but the properties guaranteed are stronger than what is required for efficient approximate nearest neighbor search.

We define a "low-quality" mapping to a Euclidean space of dimension $O(\log \frac{n}{k}/\epsilon^2)$, such that an approximate nearest neighbor lies among the k approximate nearest neighbors in the projected space. This leads to our main Theorem 10 which offers a new randomized algorithm for approximate nearest neighbor search with the following complexity. Given n points in $\mathbb{R}^d$, the data structure which is based on Balanced Box-Decomposition (BBD) trees, requires $O(dn)$ space, and reports an $(1+\epsilon)^2$-approximate nearest neighbor in time $O(dn^\rho \log n)$, where function $\rho < 1$ is proportional to $1 - 1/\ln \ln n$ for fixed $\epsilon \in (0,1)$ and shall be specified in Section 2.2. The total preprocessing time is $O(dn \log n)$. For each query $q \in \mathbb{R}^d$, the preprocessing phase succeeds with probability $> 1 - \delta$ for any constant $\delta \in (0,1)$. The low-quality embedding is extended to pointsets with bounded expansion rate c (see section 3.1 for exact definitions). The pointset is now mapped to a Euclidean space of dimension roughly $O(\log c/\epsilon^2)$ for large enough k.

One part of this work also appears in [AEP15]. Comparison of our result with previous

	Space	Query
BBD-trees	$O(dn)$	$O((\frac{d}{\epsilon})^d \log n)$
AVD	$\tilde{O}(\frac{n}{\epsilon^d})$	$O(\log n)$
BBD-trees+JL	$O(dn)$	$\omega(n)$
AVD +JL	$n^{O(\log \frac{1}{\epsilon}/\epsilon^2)}$	$O(\log n)$
LSH	$\tilde{O}(dn^{1+\frac{1}{(1+\epsilon)^2}})$	$\tilde{O}(dn^{\frac{1}{(1+\epsilon)^2}})$
multi-probe LSH	$\tilde{\tilde{O}}(dn)$	$\tilde{O}(dn^{\frac{2}{(1+\epsilon)}})$
Our approach	$O(dn)$	$\tilde{O}(dn^{1-\epsilon^2/C \log\log n})$

Table 1.1: Comparison with other results.

work can be viewed in Table 1.1.

Chapter 2

Randomized Embeddings and Approximate Nearest Neighbor Search

2.1 Low Quality Randomized Embeddings

This chapter examines standard dimensionality reduction techniques and extends them to approximate embeddings optimized to our setting. In the following, we denote by $\|\cdot\|$ the Euclidean norm and by $|\cdot|$ the cardinality of a set.

Let us start with the classic Johnson and Lindenstrauss lemma:

Proposition 1. [JL84] *For any set $X \subset \mathbb{R}^d$, $\epsilon \in (0,1)$ there exists a distribution over linear mappings $f : \mathbb{R}^d \longrightarrow \mathbb{R}^{d'}$, where $d' = O(\log |X|/\epsilon^2)$, such that for any $p, q \in X$,*

$$(1-\epsilon)\|p-q\|^2 \leq \|f(p)-f(q)\|^2 \leq (1+\epsilon)\|p-q\|^2.$$

In the initial proof [JL84], they show that this can be achieved by orthogonally projecting the pointset on a random linear subspace of dimension d'. In [DG02], they provide a proof based on elementary probabilistic techniques. In [IM98], they prove that it suffices to apply a gaussian matrix G on the pointset. G is a $d \times d'$ matrix with each of its entries independent random variables given by the standard normal distribution $N(0,1)$. Instead of a gaussian matrix, we can apply a matrix whose entries are independent random variables with uniformly distributed values in $\{-1, 1\}$ [Ach03].

However, it has been realized that this notion of randomized embedding is somewhat stronger than what is required for approximate nearest neighbor searching. The following definition has been introduced in [IN07] and focuses only on the distortion of the nearest neighbor.

Definition 2. *Let (Y, d_Y), (Z, d_Z) be metric spaces and $X \subseteq Y$. A distribution over mappings $f : Y \to Z$ is a* nearest-neighbor preserving embedding *with distortion $D \geq 1$ and probability of correctness $P \in [0, 1]$ if, $\forall c \geq 1$ and $\forall q \in Y$, with probability at least P, when $x \in X$ is such that $f(x)$ is an ϵ-ANN of $f(q)$ in $f(X)$, then x is a $(D \cdot (1 + \epsilon))$-approximate nearest neighbor of q in X.*

While in the ANN problem we search one point which is approximately nearest, in the k approximate nearest neighbors problem (kANNs) we seek an approximation of the k nearest points, in the following sense. Let X be a set of n points in $\mathbb{R}^d$, let $q \in \mathbb{R}^d$ and $1 \leq k \leq n$. The problem consists in reporting a sequence $S = \{p_1, \cdots, p_k\}$ of k distinct points such that the i-th point is an $(1 + \epsilon)$ approximation to the i-th nearest neighbor of q. Furthermore, the following assumption is satisfied by the search routine of tree-based data structures such as BBD-trees.

Assumption 3. *Let $S' \subseteq X$ be the set of points visited by the $(1 + \epsilon)$-kANNs search such that $S = \{p_1, \cdots, p_k\} \subseteq S'$ is the set of points which are the k nearest to the query point q among the points in S'. We assume that $\forall x \in X \setminus S'$, $d(x, q) > d(p_k, q)/(1 + \epsilon)$.*

Assuming the existence of a data structure which solves ϵ-kANNs, we can weaken Definition 2 as follows.

Definition 4. *Let (Y, d_Y), (Z, d_Z) be metric spaces and $X \subseteq Y$. A distribution over mappings $f : Y \to Z$ is a* locality preserving embedding *with distortion $D \geq 1$, probability of correctness $P \in [0, 1]$ and locality parameter k, if $\forall c \geq 1$ and $\forall q \in Y$, with probability at least P, when*

$S = \{f(p_1), \cdots, f(p_k)\}$ is a solution to $(1+\epsilon)$-kANNs for q, under Assumption 3 then there exists $f(x) \in S$ such that x is a $D \cdot c$ approximate nearest neighbor of q in X.

According to this definition we can reduce the problem of $(1+\epsilon)$-ANN in dimension d to the problem of computing k approximate nearest neighbors in dimension $d' < d$.

We use the Johnson-Lindenstrauss dimensionality reduction technique.

Lemma 5. [DG02] *There exists a distribution over linear maps $A : \mathbb{R}^d \to \mathbb{R}^{d'}$ s.t., for any $p \in \mathbb{R}^d$ with $\|p\| = 1$:*

- *if $\beta < 1$ then $\Pr[\|Ap\|^2 \leq \beta^2 \cdot \frac{d'}{d}] \leq exp(\frac{d'}{2}(1 - \beta^2 + 2\ln\beta)$,*
- *if $\beta > 1$ then $\Pr[\|Ap\|^2 \geq \beta^2 \cdot \frac{d'}{d}] \leq exp(\frac{d'}{2}(1 - \beta^2 + 2\ln\beta)$.*

We prove the following lemma which will be useful.

Lemma 6. *For all $i \in \mathbb{N}$, $\epsilon \in (0,1)$, the following holds:*

$$\frac{(1+\epsilon/2)^2}{(2^i(1+\epsilon))^2} - 2\ln\frac{(1+\epsilon/2)}{2^i(1+\epsilon)} - 1 > 0.05(i+1)\epsilon^2.$$

Proof. For $i = 0$, it can be checked that if $\epsilon \in (0,1)$ then, $\frac{(1+\epsilon/2)^2}{(1+\epsilon)^2} - 2\ln\frac{1+\epsilon/2}{1+\epsilon} - 1 > 0.05\epsilon^2$. This is our induction basis. Let $j \geq 0$ be such that the induction hypothesis holds, namely $\frac{(1+\epsilon/2)^2}{(2^j(1+\epsilon))^2} - 2\ln\frac{(1+\epsilon/2)}{2^j(1+\epsilon)} - 1 > 0.05(j+1)\epsilon^2$. Then, to prove the induction step

$$\frac{1}{4}\frac{(1+\epsilon/2)^2}{(2^j(1+\epsilon))^2} - 2\ln\frac{(1+\epsilon/2)}{2^j(1+\epsilon)} + 2\ln 2 - 1 > 0.05(j+1)\epsilon^2 - \frac{3}{4}\frac{(1+\epsilon/2)^2}{(2^j(1+\epsilon))^2} + 2\ln 2 >$$

$$> 0.05(j+1)\epsilon^2 - \frac{3}{4} + 2\ln 2 > 0.05(j+2)\epsilon^2,$$

since $\epsilon \in (0,1)$. □

A simple calculation shows the following.

Lemma 7. *For all $x > 0$, it holds:*

$$\frac{(1+x)^2}{(1+2x)^2} - 2\ln(\frac{1+x}{1+2x}) - 1 < (1+x)^2 - 2\ln(1+x) - 1. \qquad (2.1)$$

Theorem 8. *Under the notation of Definition 4, there exists a randomized mapping $f : \mathbb{R}^d \to \mathbb{R}^{d'}$ which satisfies Definition 4 for $d' = O(\log \frac{n}{\delta k}/\epsilon^2)$, distortion $D = 1+\epsilon$ and probability of success $1-\delta$, for any constant $\delta \in (0,1)$.*

Proof. Let X be a set of n points in $\mathbb{R}^d$ and consider map

$$f : \mathbb{R}^d \to \mathbb{R}^{d'} : v \mapsto \sqrt{d/d'} \cdot A\ v,$$

where A is a matrix as in Definition 4. Wlog the query point q lies in the origin and its nearest neighbor u lies at distance 1 from q. We denote by $c \geq 1$ the approximation ratio guaranteed by the assumed data structure. That is the assumed data structure solves the c-kANNs problem. For each point x, $L_x = \|Ay\|^2$ where $y = x/\|x\|$. Let N be the random variable whose value indicates the number of "bad" candidates, that is

$$N = |\,\{x \in X \ :\ \|x-q\| > \gamma \ \wedge\ L_x \leq \frac{\beta^2}{\gamma^2} \cdot \frac{d'}{d}\}\,|,$$

where we define $\beta = c(1+\epsilon/2)$, $\gamma = c(1+\epsilon)$. Hence, by Lemma 5,

$$\mathbb{E}[N] \leq n \cdot exp(\frac{d'}{2}(1 - \frac{\beta^2}{\gamma^2} + 2\ln\frac{\beta}{\gamma})).$$

By Markov's inequality,

$$\Pr[N \geq k] \leq \frac{\mathbb{E}[N]}{k} \implies \Pr[N \geq k] \leq n\ \cdot\ exp(\frac{d'}{2}(1 - \frac{\beta^2}{\gamma^2} + 2\ln\frac{\beta}{\gamma}))/k.$$

The event of failure is defined as the disjunction of two events:

$$[\,N \geq k\,]\ \vee\ [\,L_u \geq (\beta/c)^2\frac{d'}{d}\,],$$

and its probability is at most equal to

$$\Pr[N \geq k] + exp(\frac{d'}{2}(1 - (\beta/c)^2 + 2\ln(\beta/c))),$$

by applying again Lemma 5. Now, we bound these two terms. For the first we choose d' such that

$$d' \geq 2\,\frac{\ln \frac{2n}{\delta k}}{\frac{\beta^2}{\gamma^2} - 1 - 2\ln\frac{\beta}{\gamma}}. \tag{2.2}$$

Therefore,

$$\frac{exp(\frac{d'}{2}(1 - \frac{\beta^2}{\gamma^2} + 2\ln\frac{\beta}{\gamma}))}{k} \leq \frac{\delta}{2n} \implies \Pr[N \geq k] \leq \frac{\delta}{2}. \tag{2.3}$$

Notice that $k \leq n$ and due to expression (2.1), we obtain $(\beta/\gamma)^2 - 2\ln(\beta/\gamma) - 1 < (\beta/c)^2 - 2\ln(\beta/c) - 1$. Hence, inequality (2.2) implies inequality (2.4):

$$d' \geq 2\frac{\ln\frac{2}{\delta}}{(\beta/c)^2 - 1 - 2\ln(\beta/c)}. \tag{2.4}$$

Therefore,

$$exp(\frac{d'}{2}(1 - (\beta/c)^2 + 2\ln(\beta/c))) \leq \frac{\delta}{2}. \tag{2.5}$$

Inequalities (2.3), (2.5) imply that the total probability of failure is at most δ.

Using Lemma 6 for $i = 0$, we obtain, that there exists d' such that

$$d' = O(\log\frac{n}{\delta k}/\epsilon^2)$$

and with probability of success at least $1 - \delta$, these two events occur:

- $\|f(q) - f(u)\| \leq (1 + \frac{\epsilon}{2})\|u - q\|$.
- $|\{p \in X | \|p - q\| > c(1+\epsilon)\|u - q\| \implies \|f(q) - f(p)\| \leq c(1 + \epsilon/2)\|u - q\|\}| < k$.

Now consider the case when the random experiment succeeds and let $S = \{f(p_1), ..., f(p_k)\}$ a solution of the c-kANNs problem in the projected space, given by a data-structure which satisfies Assumption 3. We have that $\forall f(x) \in f(X) \setminus S'$, $\|f(x) - f(q)\| > \|f(p_k) - f(q)\|/c$ where S' is the set of all points visited by the search routine.

Now, if $f(u) \in S$ then S contains the projection of the nearest neighbor. If $f(u) \notin S$ then if $f(u) \notin S'$ we have the following:

$$\|f(u) - f(q)\| > \|f(p_k) - f(q)\|/c \implies \|f(p_k) - f(q)\| < c(1 + \epsilon/2)\|u - q\|,$$

which means that there exists at least one point $f(p^*) \in S$ s.t. $\|q - p^*\| \leq c(1 + \epsilon)\|u - q\|$. Finally, if $f(u) \notin S$ but $f(u) \in S'$ then

$$\|f(p_k) - f(q)\| \leq \|f(u) - f(q)\| \implies \|f(p_k) - f(q)\| \leq (1 + \epsilon/2)\|u - q\|,$$

which means that there exists at least one point $f(p^*) \in S$ s.t. $\|q - p^*\| \leq c(1 + \epsilon)\|u - q\|$.

Hence, f satisfies Definition 4 for $D = 1 + \epsilon$.

□

2.2 Approximate Nearest Neighbor Search

This section combines tree-based data structures which solve $(1 + \epsilon)$-kANNs with the results above, in order to obtain an efficient randomized data structure which solves $(1 + \epsilon)$-ANN.

BBD-trees [AMN+98] require $O(dn)$ space, and allow computing k points, which are $(1+\epsilon)$-approximate nearest neighbors, within time $O((\lceil 1+6\frac{d}{\epsilon}\rceil^d + k) d \log n)$. The preprocessing time is $O(dn \log n)$. Notice, that BBD-trees satisfy the Assumption 3. The algorithm for the $(1 + \epsilon)$-kANNs search, visits cells in increasing order with respect to their distance from the query point q. If the current cell lies in distance more than r_k/c where r_k is the current distance to the kth nearest neighbor, the search terminates. We apply the random projection for distortion $D = 1 + \epsilon$, thus relating approximation error to the allowed distortion; this is not required but simplifies the analysis.

Moreover, $k = n^\rho$; the formula for $\rho < 1$ is determined below. Our analysis then focuses on the asymptotic behaviour of the term $O(\lceil 1 + 6\frac{d'}{\epsilon}\rceil^{d'} + k)$.

Lemma 9. *With the above notation, there exists $k > 0$ s.t., for fixed $\epsilon \in (0, 1)$, it holds that $\lceil 1 + 6\frac{d'}{\epsilon}\rceil^{d'} + k = O(n^\rho)$, where $\rho \leq 1 - \epsilon^2/\hat{c}(\epsilon^2 + \ln(\max(\frac{1}{\epsilon}, \ln n))) < 1$ for some appropriate* *constant $\hat{c} > 1$.*

Proof. Recall that $d' \leq \frac{\tilde{c}}{\epsilon^2} \ln \frac{n}{k}$ for some appropriate constant $\tilde{c} > 0$. The constant δ is hidden in $\tilde{c}$. Since $(\frac{d'}{\epsilon})^{d'}$ is a decreasing function of k, we need to choose k s.t. $(\frac{d'}{\epsilon})^{d'} = \Theta(k)$. Let $k = n^{\rho}$. Obviously $\lceil 1 + 6\frac{d'}{\epsilon} \rceil^{d'} \leq (c'\frac{d'}{\epsilon})^{d'}$, for some appropriate constant $c' \in (1, 7)$. Then, by substituting d', k we have:

$$(c'\frac{d'}{\epsilon})^{d'} = n^{\frac{\tilde{c}(1-\rho)}{\epsilon^2} \ln(\frac{\tilde{c}c'(1-\rho)\ln n}{\epsilon^3})}. \tag{2.6}$$

We assume $\epsilon \in (0, 1)$ is a fixed constant. Hence, it is reasonable to assume that $\frac{1}{\epsilon} < n$. We consider two cases when comparing $\ln n$ to ϵ:

- $\frac{1}{\epsilon} \leq \ln n$. Substituting $\rho = 1 - \frac{\epsilon^2}{2\tilde{c}(\epsilon^2 + \ln(c' \ln n))}$ into equation (2.6), the exponent of n is bounded as follows:

$$\frac{\tilde{c}(1-\rho)}{\epsilon^2} \ln(\frac{\tilde{c}c'(1-\rho)\ln n}{\epsilon^3}) =$$

$$= \frac{\tilde{c}}{2\tilde{c}(\epsilon^2 + \ln(c' \ln n))} \cdot [\ln(c' \ln n) + \ln \frac{1}{\epsilon} - \ln(2\epsilon^2 + 2\ln(c' \ln n))] < \rho.$$

- $\frac{1}{\epsilon} > \ln n$. Substituting $\rho = 1 - \frac{\epsilon^2}{2\tilde{c}(\epsilon^2 + \ln \frac{c'}{\epsilon})}$ into equation(2.6), the exponent of n is bounded as follows:

$$\frac{\tilde{c}(1-\rho)}{\epsilon^2} \ln(\frac{\tilde{c}c'(1-\rho)\ln n}{\epsilon^3}) =$$

$$= \frac{\tilde{c}}{2\tilde{c}(\epsilon^2 + \ln \frac{c'}{\epsilon})} \cdot [\ln \ln n + \ln \frac{c'}{\epsilon} - \ln(2\epsilon^2 + 2\ln \frac{c'}{\epsilon})] < \rho.$$

□

Notice that for both cases $d' = O(\frac{\log n}{\epsilon^2 + \log\log n})$.

Combining Theorem 8 with Lemma 9 yields the following main theorem.

Theorem 10 (Main). *Given n points in $\mathbb{R}^d$, there exists a randomized data structure which requires $O(dn)$ space and reports an $(1+\epsilon)^2$-approximate nearest neighbor in time*

$$O(dn^{\rho} \log n), \text{ where } \rho \leq 1 - \epsilon^2/\hat{c}(\epsilon^2 + \ln(\max(\frac{1}{\epsilon}, \ln n)))$$

for some appropriate constant $\hat{c} > 1$. The preprocessing time is $O(dn \log n)$. For each query $q \in \mathbb{R}^d$, the preprocessing phase succeeds with any constant probability.

Proof. The space required to store the dataset is $O(dn)$. The space used by BBD-trees is $O(d'n)$ where d' is defined in Lemma 9. We also need $O(dd')$ space for the matrix A as specified in Theorem 8. Hence, since $d' < d$ and $d' < n$, the total space usage is bounded above by $O(dn)$.

The preprocessing consists of building the BBD-tree which costs $O(d'n \log n)$ time and sampling A. Notice that we can sample a d'-dimensional random subspace in time $O(dd'^2)$ as follows. First, we sample in time $O(dd')$, a $d \times d'$ matrix where its elements are independent random variables with the standard normal distribution $N(0, 1)$. Then, we orthonormalize using Gram-Schmidt in time $O(dd'^2)$. Since $d' = O(\log n)$, the total preprocessing time is bounded by $O(dn \log n)$.

For each query we use A to project the point in time $O(dd')$. Next, we compute its n^ρ approximate nearest neighbors in time $O(d'n^\rho \log n)$ and we check its neighbors with their real coordinates in time $O(dn^\rho)$. Hence, each query costs $O(d \log n + d'n^\rho \log n + dn^\rho) = O(dn^\rho \log n)$ because $d' = O(\log n)$, $d' = O(d)$. Thus, the query time is dominated by the time required for ϵ-kANNs search and the time to check the returned sequence of k approximate nearest neighbors. □

To be more precise, the probability of success, which is the probability that the random projection succeeds according to Theorem. 8, is greater than $1 - \delta$, for any constant $\delta \in (0, 1)$. Notice that the preprocessing time for BBD-trees has no dependence on ϵ.

Chapter 3

Exploiting hidden structure

3.1 Bounded Expansion Rate

This section models the structure that the data points may have so as to obtain more precise bounds.

The bound on the dimension obtained in Theorem 8 is quite pessimistic. We expect that, in practice, the space dimension needed in order to have a sufficiently good projection is less than what Theorem 8 guarantees. Intuitively, we do not expect to have instances where all points in X, which are not approximate nearest neighbors of q, lie at distance almost equal to $(1+\epsilon)d(q,X)$ as in Figure 3-1. To this end, we consider the case of pointsets with bounded expansion rate.

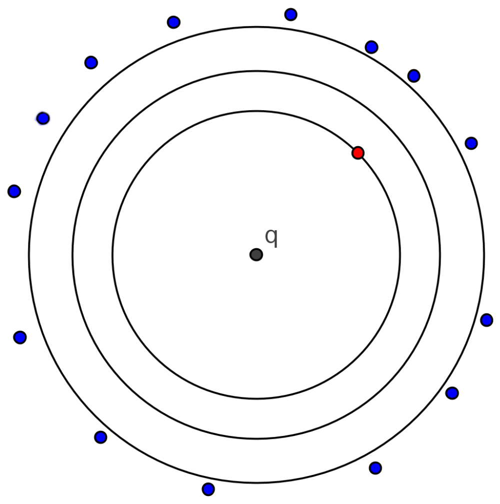

Figure 3-1: Bad case.

Definition 11. *Let M a metric space and $X \subseteq M$ a finite pointset and let $B_p(r) \subseteq X$ denote the points of X lying in the closed ball centered at p with radius r. We say that X has (ρ, c)-expansion rate if and only if, $\forall p \in M$ and $r > 0$,*

$$|B_p(r)| \geq \rho \implies |B_p(2r)| \leq c \cdot |B_p(r)|.$$

Theorem 12. *Under the notation introduced in the previous definitions, there exists a randomized mapping $f : \mathbb{R}^d \to \mathbb{R}^{d'}$ which satisfies Definition 4 for dimension $d' = O(\frac{\log(c+\frac{\rho}{\delta k})}{\epsilon^2})$, distortion $D = 1 + \epsilon$ and probability of success $1 - \delta$, for any constant $\delta \in (0, 1)$, for pointsets with (ρ, c)-expansion rate.*

Proof. We proceed in the same spirit as in the proof of Theorem 8, and using the notation from that proof. Let r_0 be the distance to the $\rho-$th nearest neighbor, excluding neighbors at distance $\leq 1 + \epsilon$. For $i > 0$, let $r_i = 2 \cdot r_{i-1}$ and set $r_{-1} = 1 + \epsilon$. Clearly,

$$\begin{aligned}\mathbb{E}[N] &\leq \sum_{i=0}^{\infty} |B_p(r_i)| \cdot exp(\frac{d'}{2}(1 - \frac{(1+\epsilon/2)^2}{r_{i-1}^2} + 2\ln\frac{1+\epsilon/2}{r_{i-1}})) \\ &\leq \sum_{i=0}^{\infty} c^i \rho \cdot exp(\frac{d'}{2}(1 - \frac{(1+\epsilon/2)^2}{2^{2i}(1+\epsilon)^2} + 2\ln\frac{1+\epsilon/2}{2^i(1+\epsilon)})).\end{aligned}$$

Now, using Lemma 6,

$$\mathbb{E}[N] \leq \sum_{i=0}^{\infty} c^i \rho \cdot exp(-\frac{d'}{2} 0.05(i+1)\epsilon^2),$$

and for $d' \geq 40 \cdot \ln(c + \frac{2\rho}{k\delta})/\epsilon^2$,

$$\mathbb{E}[N] \leq \rho \cdot \sum_{i=0}^{\infty} c^i \cdot (\frac{1}{c + \frac{2\rho}{k\delta}})^{i+1} = \rho \cdot \sum_{i=0}^{\infty} c^i \cdot (\frac{1}{c})^{i+1} \cdot (\frac{1}{1 + \frac{2\rho}{kc\delta}})^{i+1} = \frac{\rho}{c} \cdot \sum_{i=0}^{\infty} (\frac{1}{1 + \frac{2\rho}{kc\delta}})^{i+1} = \frac{k\delta}{2}.$$

Finally,

$$\Pr[N \geq k] \leq \frac{\mathbb{E}[N]}{k} \leq \frac{\delta}{2}.$$

□

Employing Theorem 12 we obtain a result analogous to Theorem 10 which is weaker than those in [KL04, BKL06] but underlines the fact that our scheme shall be sensitive to structure in the input data, for real world assumptions.

Theorem 13. *Given n points in $\mathbb{R}^d$ with $(\ln n, c)$-expansion rate, there exists a randomized data structure which requires $O(dn)$ space and reports an $(1+\epsilon)^2$-approximate nearest neighbor in time $O((C^{1/\epsilon^3} + \log n)d \log n/\epsilon^2)$, for some constant C depending on c. The preprocessing time is $O(dn \log n)$. For each query $q \in \mathbb{R}^d$, the preprocessing phase succeeds with any constant probability.*

Proof. Set $k = \ln n$. Then $d' = O(\frac{\ln c}{\epsilon^2})$ and $(\frac{d'}{\epsilon})^{d'} = O(c^{\frac{1}{\epsilon^2}\ln[\frac{\ln c}{\epsilon^3}]})$. Now the query time is

$$O((c^{\frac{1}{\epsilon^2}\ln[\frac{\ln c}{\epsilon^3}]} + \ln n) d \frac{\ln c}{\epsilon^2} \ln n) = O((C^{1/\epsilon^3} + \log n) d \frac{\ln n}{\epsilon^2}),$$

for some constant C such that $c^{\ln(\ln c/\epsilon^3)/\epsilon^2} = O(C^{1/\epsilon^3})$. □

3.2 Doubling dimension

In this section, we generalize our idea for pointsets with bounded doubling dimension.

Definition 14. *The doubling dimension of a metric space M is the smallest positive integer $ddim(M)$ such that every set S with diameter D_S can be covered by $2^{ddim(M)}$ (the doubling constant) sets of diameter $D_S/2$.*

Now, let $X \subset \mathbb{R}^d$ s.t. $|X| = n$ and X has doubling constant λ_X. Now let $S_i \subseteq X$ with diameter $2r_i$. Then we need $\lambda_X^{\log \frac{8r_i}{\epsilon}}$ tiny balls $b_\epsilon \subseteq X$ of diameter $\epsilon/4$ in order to cover S_i. However under the approximate nearest neighbor setting, it is impossible to bound the number of points per tiny ball which is needed in order to generalize our idea. To that end, we focus on the approximate near neighbor problem. Recall that in the (c, R)-ANN problem we build a data structure which answers queries of the following form: given a query point q, if there exists a point in X in distance $\leq R$ from q then return a point in distance $\leq cR$, where $c > 1$. We extend this definition in order to define the (c, R)-kANNs problem in which a data structure answers queries of the following form: given a query point q, if there exist k points in X in distance $\leq R$ from q then return k points in distance $\leq cR$, where $c > 1$. The (c, R)-ANN problem is already hard in high dimensions.

We can assume that $R = 1$, since we can scale X. The idea is that we first compute $X' \subseteq X$ which satisfies the following two properties:

- $\forall p, q \in X'$ $\|p - q\| > \epsilon/8$,
- $\forall q \in X \exists p \in X'$ s.t. $\|p - q\| \leq \epsilon/8$.

The obvious naive algorithm computes X' in $O(n^2)$ time.

```
• $Y = \emptyset$
• for all $x \in X \setminus Y$:
    – $X' \leftarrow X' \cup \{x\}$
    – $Y \leftarrow Y \cup \{x\}$
    – for all $y \in X \setminus Y$ then
        ∗ if $\|x - y\| < \epsilon/8$ then $Y \leftarrow Y \cup \{y\}$
```

Then, for X' we know that each $S_i \subseteq X'$ contains $\leq \lambda_X^{\log \frac{8r_i}{\epsilon}}$ points.

Theorem 15. *The (c, R)-ANN problem in $\mathbb{R}^d$ reduces to the (c, R)-kANNs problem in $\mathbb{R}^{d'}$, where $d' = O(\frac{\log(1/\epsilon)\log\lambda_X + \log(1+\frac{2}{k\delta})}{\epsilon^2})$, by a randomized algorithm which succeeds with probability at least $1-\delta$. Preprocessing costs an additional of $O(n^2)$ time and the delay in query time is proportional to k.*

Proof. Once again we proceed in the same spirit as in the proof of Theorem 8. Let $r_i = 2^{i+1}(1+\epsilon)$ for $i \geq -1$ and let $B_p(r) \subseteq X$ denote the points of X lying in the closed ball centered at p with radius r.

$$\begin{aligned}
\mathbb{E}[N] \leq &\sum_{i=0}^{\infty} |B_p(r_i)| \cdot exp(\frac{d'}{2}(1 - \frac{(1+\epsilon/2)^2}{r_{i-1}^2} + 2\ln\frac{1+\epsilon/2}{r_{i-1}})) \\
\leq &\sum_{i=0}^{\infty} \lambda_X^{\log\frac{8r_i}{\epsilon}} \cdot exp(\frac{d'}{2}(1 - \frac{(1+\epsilon/2)^2}{2^{2i}(1+\epsilon)^2} + 2\ln\frac{1+\epsilon/2}{2^i(1+\epsilon)})).
\end{aligned}$$

Now, using Lemma 6,

$$\mathbb{E}[N] \leq \sum_{i=0}^{\infty} \lambda_X^{\log(\frac{2^{i+4}(1+\epsilon)}{\epsilon})} \cdot exp(-\frac{d'}{2}0.05(i+1)\epsilon^2),$$

$$= \sum_{i=0}^{\infty} \lambda_X^{(i+4+\log(\frac{(1+\epsilon)}{\epsilon})} \cdot exp(-\frac{d'}{2}0.05(i+1)\epsilon^2),$$

and for $d' \geq \ 40 \cdot ((3 + \log(\frac{(1+\epsilon)}{\epsilon})\ln\lambda_X - \ln(\frac{1}{1+2/k\delta}))/\epsilon^2$,

$$\mathbb{E}[N] \leq \frac{k\delta}{2}$$

and

$$\Pr[N \geq k] \leq \frac{\mathbb{E}[N]}{k} \leq \frac{\delta}{2}.$$

Now if $\exists p \in X$ s.t. $\|p - q\| \leq 1$ then $\exists p' \in X'$ s.t. $\|p' - q\| \leq 1 + \epsilon/8$. Hence, if $\alpha = 1 + \epsilon/8$

and $\beta = 1 + \epsilon/2$, it suffices $d' \geq 2\frac{\ln \frac{2}{\delta}}{(\beta/\alpha)^2 - 1 - 2\ln(\beta/\alpha)}$

$$\Pr[\|f(p') - f(q)\| > (1 + \epsilon/2)\|p' - q\|] \leq \frac{\delta}{2}.$$

Hence, it suffices $d' = O(\frac{\log(1/\epsilon)\log\lambda_X + \log(1+\frac{2}{k\delta})}{\epsilon^2})$. □

Notice that if $k = 1/\delta$ then $d' = O(\frac{\log(1/\epsilon)\log\lambda_X}{\epsilon^2})$. This bound improves upon [IN07] by a factor of $\log 1/\delta$ which is useful in case we need $\delta \ll 1$ even if the query time increases by a factor of $1/\delta$.

Chapter 4

Experiments

Our method was implemented and tested[1] in [AEP15]. The following chapter summarizes our experimental results. We generated our own synthetic datasets and query points to verify our results. First of all, as in [DI04], we followed the “planted nearest neighbor model” for our datasets. This model guarantees for each query point q the existence of a few approximate nearest neighbors while keeping all others points sufficiently far from q. The benefit of this approach is that it represents a typical ANN search scenario, where for each point there exist only a handful approximate nearest neighbors. In contrast, in a uniformly generated dataset, all the points will tend to be equidistant to each other in high dimensions, which is quite unrealistic.

More precisely, in our scenario each query q has a nearest neighbor at distance R and the rest of the points lie at distance $> (1+\epsilon)R$. Moreover a significant amount of points lie close to the boundary of the ball centered at q with radius $(1+\epsilon)R$.

[1]Credit to Evangelos Anagnostopoulos

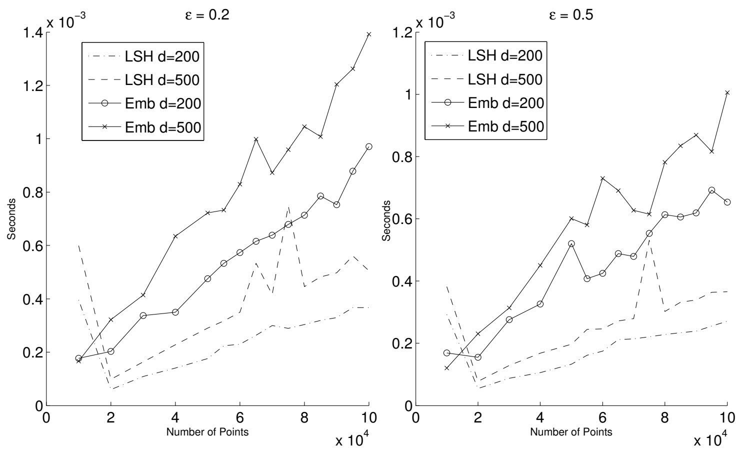

Figure 4-1: Our approach against E2LSH: query time.

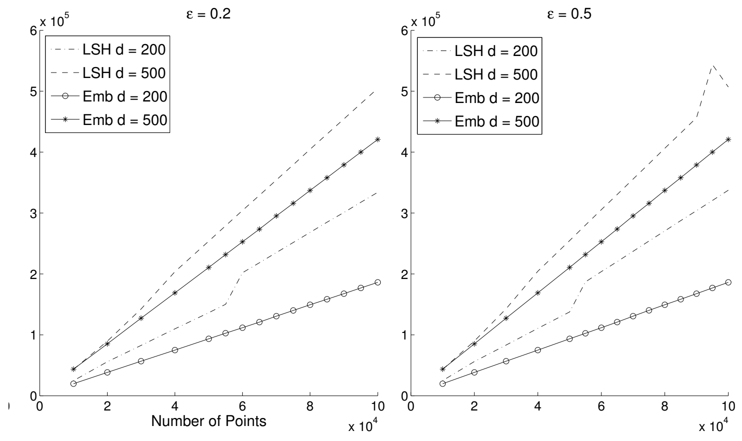

Figure 4-2: Our approach against E2LSH: memory usage.

We built a BBD-tree data structure on the projected space using the `ANN` library with the default settings. Next, we measured the average time needed for each query q to find its ϵ-kANNs, for $k = \sqrt{n}$, using the BBD-Tree data structure and then to select the first point at distance $\leq R$ out of the k in the original space. We compare these times to the average times reported by E2LSH range queries, given the range parameter R, when used from its default script for probability of success 0.95. The script first performs an estimation of the best parameters for the dataset and then builds its data structure using these parameters. We required from the two approaches to have accuracy > 0.90, which in our case means that in at least 90 out of the 100 queries they would manage to find the approximate nearest neighbor.

It is clear from Figure 4-1 that E2LSH is faster than our approach by a factor of 3. However in Figure 4-2, where we present the memory usage comparison between the two approaches, it is obvious that E2LSH requires more space. Figure 4-2 also validates the linear space dependency of our embedding method.

Chapter 5

Open Questions

In terms of practical efficiency it is obvious that checking the real distance to the neighbors while performing an kANNs search in the reduced space, is more efficient in practice than naively scanning the returned sequence of k-approximate nearest neighbors and looking for the best in the initial space. Moreover, we do not exploit the fact that BBD-trees return a sequence and not simply a set of neighbors.

Our embedding possibly has further applications. One possible application is the problem of computing the k-th approximate nearest neighbor. The problem may reduce to computing all neighbors between the i-th and the j-th nearest neighbors in a space of significantly smaller dimension for some appropriate $i < k < j$. Other possible applications include computing the approximate minimum spanning tree or the closest pair of points.

It is also worth mentioning that our ideas may apply in other norms. For example it is known that there exist JL-type random projections from l_2 to l_1 norm. However solving the ANN problem in l_1 appears to be at least as difficult as solving it in l_2.

Bibliography

[Ach03] D. Achlioptas. Database-friendly random projections: Johnson-Lindenstrauss with binary coins. In *J. Computer and System Sciences*, pages 671–687, 2003.

[AdFM11] Sunil Arya, Guilherme D. da Fonseca, and David M. Mount. Approximate polytope membership queries. In *Proc. ACM Symposium on Theory of Computing*, pages 579–586, 2011.

[AEP15] E. Anagnostopoulos, I. Z. Emiris and I. Psarros, Low-quality dimension reduction and high-dimensional Approximate Nearest Neighbor. In arXiv:1412.1683, to appear in *SoCG* 2015.

[And09] Alexandr Andoni. Nearest Neighbor Search: the Old, the New, and the Impossible. PhD thesis, Massachusetts Institute of Technology, 2009.

[AI05] Alexandr Andoni and Piotr Indyk. E^{2}LSH 0.1 User Manual, 2005, Implementation of LSH: E2LSH, http://www.mit.edu/ andoni/LSH

[AI08] Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Commun. ACM*, pages 117–122, 2008.

[AINR14] Alexandr Andoni, Piotr Indyk, H.L. Nguy En and Ilya Razenshteyn, Beyond Locality-Sensitive Hashing. In *Proc. Symp. of Discrete Algorithms (SODA)*, 2014.

[AMM09] Sunil Arya, Theocharis Malamatos, and David M. Mount. Space-time tradeoffs for approximate nearest neighbor searching. In *J. ACM*, pages 1:1–1:54, 2009.

[AMN+98] S. Arya, D.M. Mount, N.S. Netanyahu, R. Silverman and A.Y. Wu. An Optimal Algorithm for Approximate Nearest Neighbor Searching Fixed Dimensions. In *J. ACM*, 1998.

[BKL06] Alina Beygelzimer, Sham Kakade and John Langford, Cover Trees for Nearest Neighbor, In *Proc. of the 23rd Inter. Conf. on Machine Learning*, pages 97–104, 2006.

[DF08] Sanjoy Dasgupta and Yoav Freund, Random projection trees and low dimensional manifolds, In *Proc. ACM Symposium on Theory of Computing*, pages 537–546. ACM, 2008.

[DG02] Sanjoy Dasgupta and Anupam Gupta, An Elementary Proof of a Theorem of Johnson and Lindenstrauss, In *Random Struct. Algorithms*, 2003.

[DI04] Mayur Datar, Nicole Immorlica, Piotr Indyk and Vahab S. Mirrokni Locality-sensitive hashing scheme based on p-stable distributions, In *Proc. of the 20th annual Symp. on Computational Geometry*, pages 253–262.

[GKL03] Anupam Gupta and Robert Krauthgamer and James R. Lee, Bounded Geometries, Fractals, and Low-Distortion Embeddings, In *Proc. 44th Annual IEEE Symp. Foundations of Computer Science*, pages 534–, 2003.

[HIN12] Sariel Har-Peled and Piotr Indyk and Rajeev Motwani, Approximate Nearest Neighbor: Towards Removing the Curse of Dimensionality, Theory of Computing, 2012.

[HPM05] Sariel Har-Peled and Manor Mendel. Fast construction of nets in low dimensional metrics, and their applications. In *Proc. Symp. on Computational Geometry*, pages 150–158. 2005.

[IM98] Piotr Indyk and Rajeev Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality, In *Proc. 30th ACM Symp. on Theory of Computing*, pages 604–613, 1998.

[IN07] Piotr Indyk and Assaf Naor. Nearest-neighbor-preserving embeddings. In *ACM Trans. Algorithms*, 3(3), 2007.

[JL84] William B. Johnson and Joram Lindenstrauss, Extensions of Lipschitz mappings into a Hilbert space, In *Contemp. Math* 26 (1984), 189–206.

[KL04] Robert Krauthgamer and James R. Lee, Navigating Nets: Simple Algorithms for Proximity Search, In *Proc. 15th Annual ACM-SIAM Symp. on Discrete Algorithms*, pages 798–807, 2004.

[KR02] David R. Karger and Matthias Ruhl, Finding Nearest Neighbors in Growth-restricted Metrics, In *Proc. 34th annual ACM Symp. on Theory of Computing*, pages 741–750, 2002.

[Mei93] Stefan Meiser. Point location in arrangements of hyperplanes. *Information and Computation*, pages 286–303, 1993.

[ML09] Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *Proc. Intern. Conf. Computer Vision Theory and Applications*, pages 331–340, 2009.

[Mou10] David M. Mount. ANN Programming Manual, 2010, Implementation of ANN: http://www.cs.umd.edu/ mount/ANN/

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

[Pan06] , Rina Panigrahy, Entropy Based Nearest Neighbor Search in High Dimensions, Proc. 17th Annual ACM-SIAM Symp. Discrete Algorithms, SODA'06.

[Vem12] Santosh Vempala. Randomly-oriented k-d Trees Adapt to Intrinsic Dimension, In *Proc. Foundations of Software Technology and Theoretical Computer Science*, pages 48–57, 2012.