Lecture #6

Last Time: The Johnson-Lindenstrauss lemma, and sparse/fast variants

unbiased estimator of $l_2$-distance via Gaussians

Today: Nearest neighbor search and LSH

Setup: Given a set $P$ of $n$ points in $d$-dimensions

1. Construct data structure

   - be able to

2. Answer query point $q \in \mathbb{R}^d$, with closest point in $P$

   e.g. spam classification

Some initial ideas:

Approach #1: No preprocessing, on query, search through $P$

- space: $O(dn)$
- every time: $O(dn)$

Can we improve the query time? And at what cost?

Approach #2 ($d=1$): Binary search tree

- $O \quad O \quad O \quad O \quad O$

   - more generally, think of $P$ as Partitions space

   - construct binary search tree on $P$, on query search for $q$

   - space: $O(n)$
   - every time: $O(\log n)$
Approach #2 (d=2) Voronoi Diagram

Partition of plane into regions based on which queries map to given point.

As dimension increases, becomes much more complex.

Other kd-trees [Bentley, 1975]

All approaches in high-dimension have exponential space or query time or both.

"Curse of dimensionality" computational geometry coined by Bellman

We will study relaxation.

c-ANN

Setup: Given a set \( P \) of \( n \) points in \( d \)-dimensions construct data structure s.t. on query \( q \):

Return a point \( p \in P \) with

\[ d(p, q) \leq c \min_{p' \in P} d(p', q) \leq c \cdot \text{approx nearest neighbor} \]
Even easier:

**setup:** Given a set $P$ of $n$ points in $d$-dimensions and radii $r_1, r_2$, construct data structure $S$ on query $q$.

1. If $\exists p \in P$ s.t. $d(p, q) \leq r_1$ return any $p'$ with $d(p', q) \leq r_2$

2. If $\nexists p \in P$ s.t. $d(p, q) \leq r_2$ return **NO**

We don't care what the algorithm does if closest point is between $R$ and $(1+\varepsilon)R$.

Let $D_{\text{max}}/D_{\text{min}}$ be max/min interpoint distances in $P$.

**lemma:** If for every $r$, there is a data structure with space $S$ and query time $T$ that solves $(r, (1+\varepsilon)r)$-PLEB then there is an algorithm for $(1+\varepsilon)^2$-ANN with space $O(S \log^{\frac{1}{1+\varepsilon}} \frac{D_{\text{max}}}{D_{\text{min}}})$ and query time $O(T \log^{\frac{1}{1+\varepsilon}} \frac{D_{\text{max}}}{D_{\text{min}}})$.

**pf:** Construct a data structure for $-\text{PLEB}$ for radii $r$.

\[ \frac{D_{\text{min}}}{2}, \ (1+\varepsilon) \frac{D_{\text{min}}}{2}, \ (1+\varepsilon)^2 \frac{D_{\text{min}}}{2}, \ ... \ \frac{D_{\text{max}}}{(r, (1+\varepsilon)r)} \]

Use binary search to find minimum $r$ s.t. $-\text{PLEB}$ returns YES, let $p$ be returned point and let $r^*$ be radius. Then

1. $d(p, q) \leq (1+\varepsilon) r^*$ since $(r', (1+\varepsilon)r^*)-\text{PLEB}$ said YES

2. $\forall p', d(p', q) \geq \frac{r^*}{(1+\varepsilon)}$ since $(r, (1+\varepsilon)r^*)-\text{PLEB}$ said NO

Thus $P$ is a $(1+\varepsilon)^2$-ANN.
More efficient reduction in [Indyk, Motwani, 1998]
and [Har-Peled, 2001] $r, (r, 1 + r) - PEB \Rightarrow (1 + \varepsilon) - ANN$

So far collisions = bad, but today we will exploit them.

**Locality Sensitive Hashing:** [Indyk, Motwani, 1998]
similar items are more likely to map to same bucket.

**Setup:** Hash family $\mathcal{H} = \{h : U \rightarrow S\}$ is called $(r_1, r_2, \rho_1, \rho_2)$-locality sensitive if for any $p, p' \in U$

1. If $d(p, p') \leq r_1$, then $Pr[h(p) = h(p')] \geq \rho_1$
2. If $d(p, p') > r_2$, then $Pr[h(p) = h(p')] \leq \rho_2$

We will always work with $\rho_1 > \rho_2$ and $r_1 < r_2$.

**Theorem [Indyk, Motwani]:** Suppose there is a $(r_1, r_2, \rho_1, \rho_2)$-locality sensitive hash family $\mathcal{H}$.
Then there is an algorithm for $(r_1, r_2)$-PEB when $

\text{space: } O(dn + n^{1+2})$ $\text{query time: } O(n^2) \text{ evaluations of hash function}

\text{where } p = \frac{\ln \frac{1}{\rho_1}}{\ln \frac{1}{\rho_2}} \text{ and succeeds with constant probability}
\text{for fixed } p, \alpha.

The space is polynomial, but query time is sublinear.
Let $k, \varepsilon$ be parameters chosen later. Let

\[ G = \{ g : U \to \mathbb{S}^k \} \]

where $g = (h_1(p), h_2(p), \ldots, h_{12}(p))$ and each $h_i \in \mathcal{A}$

**Preprocessing:**
1. Choose $g_1, g_2, \ldots, g_\varepsilon$ independently, uniformly at random.
2. For each $p \in P$, store it in buckets $g_1(p), g_2(p), \ldots, g_\varepsilon(p)$.
3. Discard empty buckets.

On query, search $g_1(q), g_2(q), \ldots, g_\varepsilon(q)$ and yes.

Return any point $p$ found with $d(p, q) \leq r_2$. Else, return NO.

**Analysis:** We want to show with constant probability,

For some $j$:

1. If $\exists p \in P$ with $d(p, q) \leq r$, then $g_j(p) = g_j(q)$.

There are at most $2\varepsilon$ points.

2. $n$ points $p \in P$ with $d(p, q) > r_2$ and $g_j(p) = g_j(q)$ for some $j$.

Let $k = \log_{1/2} \log \frac{1}{\varepsilon} n$, then the expected number of points satisfying the conditions in (2) is at most

\[ n (P_1)^k = n (P_2)^{\log_{1/2} \log \frac{1}{\varepsilon} n} \leq 1 \]

Thus by Markov, the probability (2) does not hold is at most $\frac{1}{2}$.

Fix $j$: The probability of $g_j(p) = g_j(q)$ in (1)

\[ P_j = n (P_1)^{\log_{1/2} \log \frac{1}{\varepsilon} n} \leq n^{-1/2} \]

Recall $P_1 = \log_{1/2} \log \frac{1}{\varepsilon} n$.
\[ \ln(1+x) = x - \frac{x^2}{2} - \ldots \quad \text{thus} \]
\[
\frac{\ln \frac{1}{1-c}}{\ln \frac{1}{1-c_0}} = \frac{\ln(1+c)}{\ln(1+c_0)} \approx \frac{c}{c_0} = \frac{1}{c}
\]

Thus the probability that (1) holds is at least
\[
1 - (1 - n^{-c})^d \geq 1 - \frac{1}{e} > \frac{1}{2}
\]
and setting \( d = n^{\delta_0} \) gives.

Thus with constant probability both (1) and (2) hold, which implies the algorithm succeeds in solving \((r_1, r_2)\)-PLEB.

**Corollary:** There is an algorithm for \((r, cr)\)-PLEB in \(H_1^d\) that uses space \(O(dn + n^{1-\frac{c}{2}})\) and for each query needs \(O(n^c)\) evaluations of the hash function, each of which takes \(O(d)\) time.

\[ \text{i.e. for the Hamming distance, } P \leq \frac{1}{c} \]

**Theorem [Andoni, Indyk, 2006] for Euclidean distance, } P \leq \frac{1}{c^2} \]

These bounds are tight \([O' Donnell, Wu, Zhou, 2011]\)