Similarity Search in high Dimensions
Searching near and far

February 17, 2016

IT UNIVERSITY OF COPENHAGEN
Parts

1. Locality Sensitive Hashing
2. Furthest Neighbor and the Annulus Query
Nearest Neighbor

Definition (Nearest Neighbor)

Let $S \subseteq \mathbb{R}^d$. Given some $q \in \mathbb{R}^d$ find $x$ with $\min_{x \in S} ||x - q||_2$. 

![Diagram showing a set $S$ in $\mathbb{R}^d$ with points marked to illustrate the Nearest Neighbor definition.]
Nearest Neighbor

Definition (Nearest Neighbor)

Let $S \subseteq \mathbb{R}^d$. Given some $q \in \mathbb{R}^d$ find $x$ with $\min_{x \in S} \|x - q\|_2$. 

$S$

$q$
Nearest Neighbor

Definition (Nearest Neighbor)

Let $S \subseteq \mathbb{R}^d$. Given some $q \in \mathbb{R}^d$ find $x$ with $\min_{x \in S} ||x - q||_2$. 
Low dimension (d=small constant)
Nearest Neighbor in low dimension

$S$
Preprocessing

Find a good partitioning $H$ of the space.

- For $x \in S$ store $H(x)$
Preprocessing

Find a good partitioning $H$ of the space.

- For $x \in S$ store $H(x)$
- At query time look at $H(q)$
Perfect partitioning

**Definition (Voronoi diagram)**

Exact space partitioning. In each cell the cell center is the nearest neighbor.

**Figure:** Voronoi diagram

![Voronoi diagram](image)
Perfect partitioning

Definition (Voronoi diagram)

Exact space partitioning. In each cell the cell center is the nearest neighbor. Space: $O(n^{\left\lceil \frac{1}{2}d \right\rceil})$

Figure: Voronoi diagram
K-d Tree

Definition (K-d Tree)

[Bently ’76] Partition the space, splitting each dimension down a tree. Nearest Neighbor Queries in $O(\log n)$.

Figure: 2d-Tree
K-d Tree

Definition (K-d Tree)

[Bently ’76] Partition the space, splitting each dimension down a tree. Nearest Neighbor Queries in $O(\log n)$. Query time degrades in $k$ as $n^{1-1/d}$.

Figure: 2d-Tree
As $d$ grows...

- Space exponential in $d$
As $d$ grows...

- Space exponential in $d$
- Query time linear in $n$
**Definition (r-Near Neighbor Decision Version)**

Let $S \subseteq \mathbb{R}^d$. Given some $q \in \mathbb{R}^d$ and $r > 0$ if $\exists x \in S$ such that $||x - q||_2 \leq r$ return yes, else return no.
Definition (*r*-Near Neighbor Decision Version)

Let $S \subseteq \mathbb{R}^d$. Given some $q \in \mathbb{R}^d$ and $r > 0$ if $\exists x \in S$ such that $||x - q||_2 \leq r$ return yes, else return no.
Definition (\(r\)-Near Neighbor Decision Version)

Let \(S \subseteq \mathbb{R}^d\). Given some \(q \in \mathbb{R}^d\) and \(r > 0\) if \(\exists x \in S\) such that \(\|x - q\|_2 \leq r\) return yes, else return no.
Seems to be a good reason we have no sub-linear time solution, if you believe in SETH

**Theorem (Sublinear hamming r-NN breaks SETH)**

[Williams ’04],[Alam & Williams ’15]  
Decision r-NN in time $n^{0.99}2^{o(d)}$ implies k-SAT with $n$ variables can be solved in time $\alpha^n$ where $\alpha < 2$. 
Definition \((c, r)\)-Approximate Near Neighbor

Let \( S \subseteq \mathbb{R}^d \). Given some \( q \in \mathbb{R}^d \), \( c > 1 \):

If \( \exists x \in S \) where \( \|x - q\|_2 \leq r \) return some \( x' \) with \( \|x - q\|_2 \leq cr \).
Locality Sensitive Hashing

Definition (Indyk & Motwani 1998)

A hash-family $\mathcal{H}$ is called $(r, cr, P_1, P_2)$-sensitive LSH if

$$Pr_{\mathcal{H}}[h(q) = h(x)] \geq P_1 \text{ when } D(x, q) \leq r$$  \hspace{1cm} (1)

$$Pr_{\mathcal{H}}[h(q) = h(x)] \leq P_2 \text{ when } D(x, q) \geq cr$$  \hspace{1cm} (2)

And $P_1 > P_2$. 

$$Pr[h(x) = h(q)]$$

$S$

$||x - q||$

$1$

$r$

$cr$
Locality Sensitive Hashing

Definition (Indyk & Motwani 1998)

A hash-family $\mathcal{H}$ is called $(r, cr, P_1, P_2)$-sensitive LSH if

$$Pr_{\mathcal{H}}[h(q) = h(x)] \geq P_1 \text{ when } D(x, q) \leq r$$ \hspace{1cm} (1)

$$Pr_{\mathcal{H}}[h(q) = h(x)] \leq P_2 \text{ when } D(x, q) \geq cr$$ \hspace{1cm} (2)

And $P_1 > P_2$. 

![Diagram showing locality sensitive hashing](image)

**J. Sivertsen Theory Lunch talk - Searching near and far**

Locality Sensitive Hashing 13/42
Locality Sensitive Hashing

Definition (Indyk & Motwani 1998)

A hash-family \( \mathcal{H} \) is called \((r, cr, P_1, P_2)\)-sensitive LSH if

\[
Pr_{\mathcal{H}}[h(q) = h(x)] \geq P_1 \quad \text{when} \quad D(x, q) \leq r
\]

\[
Pr_{\mathcal{H}}[h(q) = h(x)] \leq P_2 \quad \text{when} \quad D(x, q) \geq cr
\]

And \( P_1 > P_2 \).
Example, Indyk, Motwani 1998

\[ S \subseteq \{0, 1\}^d \]

\[ S = \\
010111100110101010010101010 \]
\[ 010100010101011101101101011 \]
\[ 11110110010100101010101010 \]
\[ 010101010101101100001010 \]
\[ 00000111010101110101001010 \]
\[ 11010100110101010100100101 \]
\[ 0101010101011101001010100 \]
Example, Indyk, Motwani 1998

\[ S = \]
\[
\begin{align*}
0101111001101010010101010 & \\
0101000101010110111101101011 & \\
111110110010101001010101010 & \\
0101010101011101100110001010 & \\
000001110101011101010010101 & \\
110101001001101010101001010 & \\
01010101010111010010101000 & \\
\end{align*}
\]

\[ h_1(x) = x_4, \text{ 2 buckets} \]
Example, Indyk, Motwani 1998

\[
S = \\
0101111001101010010101010 \\
0101000101010111011011011011 \\
111110110010101001010101010 \\
0101010101011101100001010 \\
000001110101011101010010101 \\
110101001101101010101001010 \\
010101010101011101001010100 \\
h_1(x) = x_4, 2 buckets \\
P_1 = 1 - r/d \\
P_2 = 1 - cr/d
\]
Example hash function

**Definition (Indyk, Motwani 1998)**

\[ P_1 = 1 - \frac{r}{d} \]
\[ P_2 = 1 - \frac{cr}{d} \]
For \( c > 1 \) we have \( P_1 > P_2 \)

"Quality" measure \( \rho = \frac{\log \frac{1}{P_1}}{\log \frac{1}{P_2}} = \frac{1}{c} \).
Example, Indyk, Motwani 1998

\[ S = \\
01011100110101010010101010 \\
010100010101011011101101011 \\
111110110010101001010101010 \\
010101010101101100001010 \\
00000111010101111010010101 \\
1101010011011001010101001010 \\
010101010101101001010100100100 \\
01010101010110100100101001001000 \\
h_1(x) = x_4, \text{ 2 buckets} \]
Example, Indyk, Motwani 1998

\[ S = \]
\[
\begin{array}{c}
010111001101010100101010
0101000101010111011101101011
111110110010101001010101010
010101010101011101100001010
000001110101011101010010101
110101001001101010101001010
010101010101010111010010100
\end{array}
\]

\[ g(x) = \{ h_1(x), h_2(x), \ldots, h_k(x) \}, \quad 2^k \text{ buckets} \]

Setting \( k \) we define \( \mathcal{G} \).
Example, Indyk, Motwani 1998

\[ S = \]
\[
01011100110101010010101010 \\
0101000101011101101101101011 \\
1111101100101001010101010100 \\
0101010101010111011000010100 \\
0000011101010111010100101011 \\
1101010010011010101010010101 \\
0101010101011101001010100100 \\
0101010101010111010101010100 \\
0101010101011101010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 \\
0101010101010111010101010100 
\]

\[ g(x) = \{ h_1(x), h_2(x), \ldots, h_k(x) \}, \text{ 2}^k \text{ buckets} \]

Setting \( k \) we define \( \mathcal{G} \).

- If \( x \) is a \( r \)-near. \( Pr[g(x) = g(p)] \geq P_1^k \).
- Using \( L \) hash functions from \( \mathcal{G} \), at least one collision with \( \text{Prb.} \geq 1 - (1 - P_1^k)^L \).
$r$-NN with $\mathcal{H}$

**Theorem (Indyk, Motwani, Gionis ’99)**

Set $L = n^\rho$:

1. Hash all points with $L$ functions from $\mathcal{G}$.
2. Query: Find buckets from $g_1(q), \ldots, g_L(q)$ in $O(n^\rho)$.
3. Look at up to $3L$ points from those buckets in $O(dn^\rho)$.

Given $H$ this solves $(c, r)$-NN in $O(dn^\rho)$ time with error $\lambda < 1$. 
Theorem (Indyk, Motwani, Gionis ’99)

Set $L = n^\rho$:

1. Hash all points with $L$ functions from $G$.
$r$-NN with $\mathcal{H}$

Theorem (Indyk, Motwani, Gionis '99)

Set $L = n^\rho$:

1. Hash all points with $L$ functions from $\mathcal{G}$.
2. Query: Find buckets from $g_1(q), \ldots, g_L(q)$ in $O(n^\rho)$. 

Given $H$ this solves $(c, r)$−$NN$ in $O(n^\rho)$ time with error $\lambda < 1$. 
Theorem (Indyk, Motwani, Gionis '99)

Set $L = n^\rho$:

1. Hash all points with $L$ functions from $\mathcal{G}$.
2. Query: Find buckets from $g_1(q), \ldots, g_L(q)$ in $O(n^\rho)$.
3. Look at up to $3L$ points from those buckets in $O(dn^\rho)$.

Given $\mathcal{H}$ this solves $(c, r) - \text{NN}$ in $O(dn^\rho)$ time with error $\lambda < 1$. 
Developments

Smaller $\rho \Rightarrow$ larger $P1/P2$ gap.

<table>
<thead>
<tr>
<th>Reference</th>
<th>$\rho$</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear search</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Indyk &amp; Motwani STOC 1998</td>
<td>$\frac{1}{c}$</td>
<td>Worst case upper bound</td>
</tr>
<tr>
<td>O’Donnell, Wu &amp; Zhou ITCS 2010</td>
<td>$\frac{1}{c}$</td>
<td>Worst case, lower bound</td>
</tr>
</tbody>
</table>

Dubiner Trans. Inf. Theory 2010

Andoni & Razenshteyn STOC 2015

Kapralov PODS 2015

Worst case upper bound

Data Dependent, Upper and lower bound

Linear space upper bound

$c$
Developments

Smaller $\rho \Rightarrow$ larger $P_1/P_2$ gap.

<table>
<thead>
<tr>
<th>Reference</th>
<th>$\rho$</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear search</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Indyk &amp; Motwani STOC 1998</td>
<td>$\frac{1}{c}$</td>
<td>Worst case upper bound</td>
</tr>
<tr>
<td>O’Donnell, Wu &amp; Zhou ITCS 2010</td>
<td>$\frac{1}{c}$</td>
<td>Worst case, lower bound</td>
</tr>
<tr>
<td>Dubiner Trans. Inf. Theory 2010</td>
<td>$\frac{1}{2c-1}$</td>
<td>Assuming random data, upper bound</td>
</tr>
</tbody>
</table>
Developments

Smaller $\rho \Rightarrow$ larger $P1/P2$ gap.

<table>
<thead>
<tr>
<th>Reference</th>
<th>$\rho$</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear search</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Indyk &amp; Motwani STOC 1998</td>
<td>$\frac{1}{c}$</td>
<td>Worst case upper bound</td>
</tr>
<tr>
<td>O’Donnell, Wu &amp; Zhou ITCS 2010</td>
<td>$\frac{1}{c}$</td>
<td>Worst case, lower bound</td>
</tr>
<tr>
<td>Dubiner Trans. Inf. Theory 2010</td>
<td>$\frac{1}{2c-1}$</td>
<td>Assuming random data, upper bound</td>
</tr>
<tr>
<td>Andoni &amp; Razenshteyn STOC 2015</td>
<td>$\frac{1}{2c-1}$</td>
<td>Data Dependent, Upper and lower bound</td>
</tr>
</tbody>
</table>
Developments

Smaller $\rho \Rightarrow$ larger $P1/P2$ gap.

<table>
<thead>
<tr>
<th>Reference</th>
<th>$\rho$</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear search</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Indyk &amp; Motwani STOC 1998</td>
<td>$\frac{1}{c}$</td>
<td>Worst case upper bound</td>
</tr>
<tr>
<td>O’Donnell, Wu &amp; Zhou ITCS 2010</td>
<td>$\frac{1}{c}$</td>
<td>Worst case, lower bound</td>
</tr>
<tr>
<td>Dubiner Trans. Inf. Theory 2010</td>
<td>$\frac{1}{2c-1}$</td>
<td>Assuming random data, upper bound</td>
</tr>
<tr>
<td>Andoni &amp; Razenshteyn STOC 2015</td>
<td>$\frac{1}{2c-1}$</td>
<td>Data Dependent, Upper and lower bound</td>
</tr>
<tr>
<td>Kapralov PODS 2015</td>
<td>$\frac{4}{c+1}$</td>
<td>Linear space upper bound</td>
</tr>
</tbody>
</table>
Part 2

- Annulus Query
- Furthest Neighbor
Annulus query

**Definition**

Given $S \subseteq \mathbb{R}^d$ a query point $q$ and parameters $r, w > 1$ return $x$ such that $\frac{r}{w} \leq \|x - q\|_2 \leq wr$. 

Applications in recommender systems.
Annulus query

**Definition**

Given $S \subseteq \mathbb{R}^d$ a query point $q$ and parameters $r, w > 1$ return $x$ such that $\frac{r}{w} \leq ||x - q||_2 \leq wr$.

Applications in recommender systems.
The Furthest Neighbor Problem

Joint work with Rasmus Pagh, Matthew Skala and Francesco Silvestri, SISAP ’15.

Definition

Let \( S \subseteq \mathbb{R}^d \). Given some \( q \in \mathbb{R}^d \) find \( x \) with \( \max ||x - q||_2 \).
Sublinear Furthest Neighbor

► For $q \in \{0, 1\}^d$: 
Sublinear Furthest Neighbor

- For \( q \in \{0, 1\}^d \):
- Furthest Neighbor \(-q = \text{Nearest Neighbor } q \) [Goel et al., '09]

Sublinear time Nearest Neighbor breaks SETH [Williams, '04], [Alman & Williams, '15]
Sublinear Furthest Neighbor

- For $q \in \{0, 1\}^d$:
  - Furthest Neighbor $-q = \text{Nearest Neighbor } q$ [Goel et al.,'09]
  - Sublinear time Nearest Neighbor breaks SETH [Williams,'04], [Alman & Williams,'15]
Approximate Furthest Neighbor

**Definition**

Let $S \subseteq \mathbb{R}^d$. Given some $q \in \mathbb{R}^d$, let $x$ be the furthest neighbor. Return $x'$ such that $\|x' - q\| \geq \frac{\|x - q\|}{c}$. We call this $c$-FN.
Approximate Furthest Neighbor

**Definition**

Let $S \subseteq \mathbb{R}^d$. Given some $q \in \mathbb{R}^d$, let $x$ be the furthest neighbor. Return $x'$ such that $||x' - q|| \geq \frac{||x - q||}{c}$. We call this $c$-FN.
Approximate Furthest Neighbor

Definition

Let $S \subseteq \mathbb{R}^d$. Given some $q \in \mathbb{R}^d$, let $x$ be the furthest neighbor. Return $x'$ such that $||x' - q|| \geq \frac{||x-q||}{c}$. We call this $c$-FN.
Related work

Though not nearly as popular as Nearest Neighbor there is notable prior work on Furthest neighbor.

<table>
<thead>
<tr>
<th>Paper</th>
<th>c</th>
<th>Query Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Bespamyatnikh '96</td>
<td>$c &gt; 1$</td>
<td>$O((1 + \frac{1}{c-1})^{d-1})$</td>
</tr>
</tbody>
</table>

1. Split tree
Related work

Though not nearly as popular as Nearest Neighbor there is notable prior work on Furthest neighbor.

<table>
<thead>
<tr>
<th>Paper</th>
<th>$c$</th>
<th>Query Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bespamyatnikh '96</td>
<td>$c &gt; 1$</td>
<td>$O((1 + \frac{1}{c-1})^{d-1})$</td>
</tr>
<tr>
<td>Goel et al. '01</td>
<td>$c &gt; \sqrt{2}$</td>
<td>$O(d^2)$</td>
</tr>
</tbody>
</table>

1. Split tree
2. Minimum enclosing ball
Related work

Though not nearly as popular as Nearest Neighbor there is notable prior work on Furthest neighbor.

<table>
<thead>
<tr>
<th>Paper</th>
<th>c</th>
<th>Query Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bespamyatnikh '96</td>
<td>$c &gt; 1$</td>
<td>$O((1 + \frac{1}{c-1})^{d-1})$</td>
</tr>
<tr>
<td>Goel et al. '01</td>
<td>$c &gt; \sqrt{2}$</td>
<td>$O(d^2)$</td>
</tr>
<tr>
<td>Indyk et al. '03</td>
<td>$c &gt; 1$</td>
<td>$O(n^{1/c^2} d \log^{1+(1-1/c^2)/2}(n) \log_{1+\delta}(d) \log \log_{1+\delta} d)$</td>
</tr>
</tbody>
</table>

1. Split tree
2. Minimum enclosing ball
3. Random Projections, binary search
Related work

Though not nearly as popular as Nearest Neighbor there is notable prior work on Furthest neighbor.

<table>
<thead>
<tr>
<th>Paper</th>
<th>$c$</th>
<th>Query Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bespamyatnikh '96</td>
<td>$c &gt; 1$</td>
<td>$O((1 + \frac{1}{c-1})^{d-1})$</td>
</tr>
<tr>
<td>Goel et al. '01</td>
<td>$c &gt; \sqrt{2}$</td>
<td>$O(d^2)$</td>
</tr>
<tr>
<td>Indyk et al. '03</td>
<td>$c &gt; 1$</td>
<td>$O(n^{1/c^2} d \log^{1+(1-1/c^2)/2}(n) \log_{1+\delta}(d) \log \log_{1+\delta} d)$</td>
</tr>
<tr>
<td>This paper</td>
<td>$c &gt; 1$</td>
<td>$O(n^{1/c^2} \log^{\frac{c^2}{2}} - \frac{1}{3}(n)(d + \log n))$</td>
</tr>
</tbody>
</table>

1. Split tree
2. Minimum enclosing ball
3. Random Projections, binary search
4. Random Projections, single query
Random Projections

Fix two points in $\mathbb{R}^d$. 

\[ x_1 \quad \quad \quad x_2 \]
Lemma (Distance preservation)

\[ a_i \cdot (x_1 - x_2) \sim \mathcal{N}(0, 1) \| x_1 - x_2 \|_2 \quad (3) \]

\[ a_i = \{g_1, g_2, \ldots, g_d\} \text{ where } g_j \sim \mathcal{N}(0, 1) \]
Lemma (Distance preservation)

\[ a_i \cdot (x_1 - x_2) \sim \mathcal{N}(0, 1) \|x_1 - x_2\|_2 \]  

(3)

\[ a_i = \{g_1, g_2, \ldots, g_d\} \text{ where } g_j \sim \mathcal{N}(0, 1) \]
Random Projections

Lemma (Distance preservation)

\[ a_i \cdot (x_1 - x_2) \sim \mathcal{N}(0, 1) \|x_1 - x_2\|_2 \]  \hspace{1cm} (3)

\[ a_i = \{g_1, g_2, \ldots, g_d\} \text{ where } g_j \sim \mathcal{N}(0, 1) \]
Preprocessing

We would like to examine the points with the largest $|a_i \cdot (x - q)|$. 
Preprocessing

We would like to examine the points with the largest $a_i \cdot (x - q)$.

- $a_i \cdot (x - q)$ not known till query time
We would like to examine the points with the largest $a_i \cdot (x - q)$.

- $a_i \cdot (x - q)$ not known till query time
- $a_i \cdot x \geq a_i \cdot y \Rightarrow a_i(x - q) \geq a_i(y - q)$
Preprocessing

We would like to examine the points with the largest $a_i \cdot (x - q)$.

- $a_i \cdot (x - q)$ not know till query time
- $a_i \cdot x \geq a_i \cdot y \Rightarrow a_i(x - q) \geq a_i(y - q)$
- $\forall x \in S$ we can preprocess $a_i \cdot x$
Preprocessing

We would like to examine the points with the largest $a_i \cdot (x - q)$.

- $a_i \cdot (x - q)$ not know till query time
- $a_i \cdot x \geq a_i \cdot y \Rightarrow a_i(x - q) \geq a_i(y - q)$
- $\forall x \in S$ we can preprocess $a_i \cdot x$

Algorithm:

1. $\forall a_i :$ store all $x$ in order of $a_i \cdot x$
We would like to examine the points with the largest $a_i \cdot (x - q)$.

- $a_i \cdot (x - q)$ not know till query time
- $a_i \cdot x \geq a_i \cdot y \Rightarrow a_i(x - q) \geq a_i(y - q)$
- $\forall x \in S$ we can preprocess $a_i \cdot x$

Algorithm:

1. $\forall a_i$: store all $x$ in order of $a_i \cdot x$
2. Query: Examine all points with $a_i \cdot (x - q)$ larger than some threshold.
Finding the threshold

\[ X = a_i \cdot (x - q) \]

is a normal distributed variable:

\[ a_i (x - q) \]

\[ ||x - q|| \]

\[ \frac{||x - q||}{c} \]
Lemma (See Lemma 7.4 in Karger, Motwani, and Sudan ’98)

For every $t > 0$, if $X \sim N(0, 1)$ then

$$\frac{1}{\sqrt{2\pi}} \cdot \left( \frac{1}{t} - \frac{1}{t^3} \right) \cdot e^{-t^2/2} \leq \Pr[X \geq t] \leq \frac{1}{\sqrt{2\pi}} \cdot \frac{1}{t} \cdot e^{-t^2/2}$$
Finding the threshold

Lemma (See Lemma 7.4 in Karger, Motwani, and Sudan ’98)

For every \( t > 0 \), if \( X \sim N(0, 1) \) then

\[
\frac{1}{\sqrt{2\pi}} \cdot \left( \frac{1}{t} - \frac{1}{t^3} \right) \cdot e^{-t^2/2} \leq \Pr[X \geq t] \leq \frac{1}{\sqrt{2\pi}} \cdot \frac{1}{t} \cdot e^{-t^2/2}
\]

We want \( O(n^{1/c^2}) \)

Set \( t \) so \( \Pr[X > t] \geq (1 - o(1)) \frac{1}{n^{1/c^2}} \).
Crossing the threshold

Lemma (Threshold projection)

\[ \Delta = \frac{rt}{c} \]

\[
\Pr \left[ a \cdot (x - q) \geq \Delta \right] \geq (1 - o(1)) \frac{1}{n^{1/c^2}}, \text{ for } \|x - q\|_2 \geq \frac{r}{c}
\]

\[
\Pr \left[ a \cdot (x - q) \geq \Delta \right] \leq \frac{\log^{c^2/2 - 1/3} n}{n}, \text{ for } \|x - q\|_2 < \frac{r}{c}
\]
Crossing the threshold

Lemma (Threshold projection)

\[ \Delta = \frac{rt}{c} \]

\[
\Pr_a \left[ a \cdot (x - q) \geq \Delta \right] \geq (1 - o(1)) \frac{1}{n^{1/c^2}}, \text{ for } \|x - q\|_2 \geq \frac{r}{c}
\]

\[
\Pr_a \left[ a \cdot (x - q) \geq \Delta \right] \leq \frac{\log^{c^2/2-1/3} n}{n}, \text{ for } \|x - q\|_2 < \frac{r}{c}
\]

What is \( r \)?
Two ways to fail

1. No ”far” points have any $a_i \cdot (x - q) \geq \Delta$. We will cross the threshold without seeing a valid candidate.
Two ways to fail

1. No ”far” points have any $a_i \cdot (x - q) \geq \Delta$. We will cross the threshold without seeing a valid candidate.
2. Too many ”close” points have $a \cdot (x - q) \geq \Delta$. We will not cross the threshold.
Corollary (Failure probability)

Using \( \ell \) random projections.

\[
\Pr[\text{Missing the far point}] \leq (1 - 1/n^{1/c^2})^\ell
\]
Corollary (Failure probability)

Using $\ell = 2n^{1/c^2}$ random projections.

$$\Pr[\text{Missing the far point}] \leq (1 - 1/n^{1/c^2})^\ell \leq 1/e^2$$
Bounding the error

**Corollary (Failure probability)**

*Using $\ell = 2n^{1/c^2}$ random projections.*

\[
\Pr[\text{Missing the far point}] \leq (1 - 1/n^{1/c^2})^\ell \leq 1/e^2
\]

\[
\Pr[\text{Too many close points}] \leq \Pr[\ell \log^{c^2/2 - 1/3} n > m]
\]
Bounding the error

Corollary (Failure probability)

Using $\ell = 2n^{1/c^2}$ random projections.

\[
\Pr[\text{Missing the far point}] \leq (1 - 1/n^{1/c^2})^\ell \leq 1/e^2
\]

\[
\Pr[\text{Too many close points}] \leq \Pr[\ell \log^{c^2/2-1/3} n > m] \leq 1/e^2
\]

For $m = e^2 \ell \log^{c^2/2-1/3} n$.

The combined failure probability is $\leq \frac{2}{e^2}$.
Data structure

Points are stored in their projection order. We use $\ell = 2n^{1/c^2}$ projections and store the top $m$ points in each.

$S_{i \in [\ell]}$

$O(\ell m d) \approx dn^{2/c^2}$ space. (With some large constants).
Query procedure

- Create an empty priority queue $PQ$.

$S_{i \in [\ell]}$

$PQ$

$\ell = 2n - 1/c^2$

Points are added with priority $a_i \cdot x - a_i \cdot q$.

Take out the top priority element and examine its distance to $q$.

Take its neighbor on the projection it came from, add it to $PQ$.

We will look at $m = e^{2\ell} \log \frac{c^2}{2} - \frac{1}{3}n$ points.

Time $O(\ell + m(d + \log \ell))$ vs. $O(\ell md)$.
Query procedure

- Create an empty priority queue $PQ$.
- Add the $\ell = 2n^{1/c^2}$ points.
- Points are added with priority $a_i \cdot x - a_i \cdot q$. 

Diagram:

$S_{i \in [\ell]}$

$\begin{align*}
  &x_5 \quad x_2 \quad x_3 \quad x_1 \\
\end{align*}$

$\begin{align*}
  &x_5 \quad x_3 \quad x_1 \quad x_2 \\
\end{align*}$

$\begin{align*}
  &x_2 \quad x_3 \quad x_1 \quad x_5 \\
\end{align*}$

$\begin{align*}
  &a_1^T x \\
\end{align*}$

$\begin{align*}
  &a_2^T x \\
\end{align*}$

$\begin{align*}
  &a_3^T x \\
\end{align*}$

$m$
Query procedure

- Create an empty priority queue $PQ$.
- Add the $\ell = 2^{n^{1/c^2}}$ points.
- Points are added with priority $a_i \cdot x - a_i \cdot q$.
- Take out the top priority element and examine its distance to $q$.
- Take its neighbor on the projection it came from, add it to $PQ$.

$$m \in [\ell]$$

$$S_i \in [\ell]$$

- $x_5 \xrightarrow{a_1^T x} x_2 \xrightarrow{a_2^T x} x_3 \xrightarrow{a_3^T x} x_1 \xrightarrow{a_1^T x} a_1^T x$$
- $x_5 \xrightarrow{a_2^T x} x_3 \xrightarrow{a_3^T x} x_1 \xrightarrow{a_2^T x} a_2^T x$
Query procedure

- Create an empty priority queue $PQ$.
- Add the $\ell = 2n^{1/c^2}$ points.
- Points are added with priority $a_i \cdot x - a_i \cdot q$.
- Take out the top priority element and examine its distance to $q$.
- Take its neighbor on the projection it came from, add it to $PQ$.
- We will look at at $m = e^{2\ell} \log^{c^2/2 - 1/3} n$ points.

![Diagram](image)
Query procedure

- Create an empty priority queue $PQ$.
- Add the $\ell = 2n^{1/c^2}$ points.
- Points are added with priority $a_i \cdot x - a_i \cdot q$.
- Take out the top priority element and examine its distance to $q$.
- Take its neighbor on the projection it came from, add it to $PQ$.
- We will look at at $m = e^2 \ell \log^{c^2/2 - 1/3} n$ points.
- Time $O(\ell + m(d + \log \ell))$ vs. $O(\ell md)$.

![Diagram of query procedure]
Approximate Furthest Neighbor

Theorem (Approximate Furthest Neighbor)

There exists a datastructure for $c$ – FN over any set $S \in \mathbb{R}^d$ of at most $n$ points, such that:

- Queries take $\tilde{O}(n^{1/c^2} d)$ time.
- The data structure uses $\tilde{O}(n^{2/c^2} d)$ space.

With probability of success at least $1 - 2/e^2 \geq 0.72$
Annulus query

\[ q \quad r/w \quad rw \]
Approximate Annulus query
Annulus Query Datastructure

Natural combination of random projections and LSH
\[ \forall x \in S \]
- \[ \forall i \in [\ell] \text{ calculate } p_i(x)x = a_i \cdot x \]
Annulus Query Datastructure

Natural combination of random projections and LSH

\( \forall x \in S \)

- \( \forall i \in [\ell] \) calculate \( p_i(x)x = a_i \cdot x \)
- \( \forall l \in [L] \) calculate \( g_l(x) \)

\( \text{Store non-empty hash-buckets in order of } p_i \text{ with PQ query structure.} \)

\( \text{Query by PQ design. Early termination now possible} \)
Annulus Query Datastructure

Natural combination of random projections and LSH

∀x ∈ S

- ∀i ∈ [ℓ] calculate \( p_i(x)x = a_i \cdot x \)
- ∀l ∈ [L] calculate \( g_l(x) \)
- Store non-empty hash-buckets in order of \( p_i \) with PQ query structure.
Annulus Query Datastructure

Natural combination of random projections and LSH
\( \forall x \in S \)

- \( \forall i \in [\ell] \) calculate \( p_i(x)x = a_i \cdot x \)
- \( \forall l \in [L] \) calculate \( g_l(x) \)
- Store non-empty hash-buckets in order of \( p_i \) with PQ query structure.
- Query by PQ design. Early termination now possible
Combining these techniques with LSH gets:

**Theorem (Annulus query)**

There exists a data structure for \((c, w, r)\)-AAQ over any set \(S \in \mathbb{R}^d\) of at most \(n\) points, such that:

- Queries can be answered in time \(\tilde{O}(n^{\rho+1/c^2})\)
- The data structure takes space \(\tilde{O}(n^{1+\rho+1/c^2})\) in addition to storing \(S\).

Failure probability \(\lambda < 1\).
Open problems

► Expand the random projection technique to general metric space?
Open problems

- Expand the random projection technique to general metric space?
- Use furthest neighbor to improve LSH output sensitivity?
Open problems

- Expand the random projection technique to general metric space?
- Use furthest neighbor to improve LSH output sensitivity?
- Direct Annulus Query hash functions?
Thank you!
Thank you!

Questions?