Similarity Search in High Dimensions

Piotr Indyk
MIT
Definitions

• Given: a set $P$ of $n$ points in $\mathbb{R}^d$

• Nearest Neighbor: for any query $q$, returns a point $p \in P$ minimizing $\|p-q\|$

• $r$-Near Neighbor: for any query $q$, returns a point $p \in P$ s.t. $\|p-q\| \leq r$ (if it exists)
The case of d=2

- Compute Voronoi diagram
- Given q, perform point location
- Performance:
  - Space: $O(n)$
  - Query time: $O(\log n)$
High-dimensional near(est) neighbor: applications

- Machine learning: nearest neighbor rule
  - Find the closest example with known class
  - Copy the class label
- Near-duplicate Retrieval

Dimension=number of pixels

(..., 1, ..., 4, ..., 2, ..., 2, ...)
(..., 6, ..., 1, ..., 3, ..., 6, ...)
(..., 1, ..., 3, ..., 7, ..., 5, ...)
(..., 2, ..., 2, ..., 1, ..., 1, ...)

Dimension=number of words
The case of $d>2$

- Voronoi diagram has size $n^{[d/2]}$
  - [Dobkin-Lipton’78]: $n^{2^{(d+1)}}$ space, $f(d) \log n$
  - [Clarkson’88]: $n^{[d/2](1+\varepsilon)}$ space, $f(d) \log n$ time
  - [Meiser’93]: $n^{O(d)}$ space, $(d+ \log n)^{O(1)}$ time

- We can also perform a linear scan: $O(dn)$ time

- Or parametrize by intrinsic dimension

- In practice:
  - kd-trees work “well” in “low-medium” dimensions
Approximate Nearest Neighbor

• c-Approximate Nearest Neighbor: build data structure which, for any query \( q \)
  – returns \( p \in P, \|p-q\| \leq cr \),
  – where \( r \) is the distance to the nearest neighbor of \( q \)
Approximate Near Neighbor

- **c-Approximate r-Near Neighbor**: build data structure which, for any query q:
  - If there is a point \( p \in P \), \( \|p - q\| \leq r \)
  - It returns \( p' \in P \), \( \|p - q\| \leq cr \)

- **Most algorithms randomized**: 
  - For each query \( q \), the probability (over the randomness used to construct the data structure) is at least 90%

- **Reductions and variants**:
  - c-Approx Nearest Neighbor reduces to c-Approx Near Neighbor
Approximate algorithms

• **Space/time exponential in** $d$ [Arya-Mount’93], [Clarkson’94], [Arya-Mount-Netanyahu-Silverman-Wu’98] [Kleinberg’97], [Har-Peled’02], ….

• **Space/time polynomial in** $d$ [Indyk-Motwani’98], [Kushilevitz-Ostrovsky-Rabani’98], [Indyk’98], [Gionis-Indyk-Motwani’99], [Charikar’02], [Datar-Immorlica-Indyk-Mirrokni’04], [Chakrabarti-Regev’04], [Panigrahy’06], [Ailon-Chazelle’06]…

<table>
<thead>
<tr>
<th>Space</th>
<th>Time</th>
<th>Comment</th>
<th>Norm</th>
<th>Ref</th>
</tr>
</thead>
<tbody>
<tr>
<td>$dn + n^{O(1/\varepsilon^2)}$</td>
<td>$d \times \log n / \varepsilon^2$ (or 1)</td>
<td>$c=1+\varepsilon$</td>
<td>Hamm, $l_2$</td>
<td>[KOR’98, IM’98]</td>
</tr>
<tr>
<td>$n^{\Omega(1/\varepsilon^2)}$</td>
<td>$O(1)$</td>
<td></td>
<td></td>
<td>[AIP’06]</td>
</tr>
<tr>
<td>$dn + n^{1+\rho(c)}$</td>
<td>$dn^{\rho(c)}$</td>
<td>$\rho(c)=1/c$</td>
<td>Hamm, $l_2$</td>
<td>[IM’98], [GIM’98],[Cha’02]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\rho(c)&lt;1/c$</td>
<td>$l_2$</td>
<td>[DIIM’04]</td>
</tr>
<tr>
<td>$dn \times \log s$</td>
<td>$dn^{\sigma(c)}$</td>
<td>$\sigma(c)=O(\log c/c)$</td>
<td>Hamm, $l_2$</td>
<td>[Ind’01]</td>
</tr>
<tr>
<td>$dn + n^{1+\rho(c)}$</td>
<td>$dn^{\rho(c)}$</td>
<td>$\rho(c)=1/c^2 + o(1)$</td>
<td>$l_2$</td>
<td>[AI’06]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\sigma(c)=O(1/c)$</td>
<td>$l_2$</td>
<td>[Pan’06]</td>
</tr>
</tbody>
</table>
$n^{O(1/\varepsilon^2)}$ space, $d \times \log n / \varepsilon^2$ query time, Hamming distance
For the curious
Hamming distance sketches

[Kushilevitz-Ostrovsky-Rabani’98]

• Let $x, y$ in $\{0,1\}^d$, $r>1$, $\varepsilon>0$, $0<\delta<1$

• Want: a distribution over mappings $sk: \{0,1\}^d \rightarrow \{0,1\}^t$ and a function $F$ such that given $sk(x), sk(y)$:
  – If $H(x,y) > (1+\varepsilon)r$, we have $F(sk(x), sk(y)) = \text{NO}$
  – If $H(x,y) < (1-\varepsilon)r$, we have $F(sk(x), sk(y)) = \text{YES}$

with probability $> 1-\delta$

• In fact
  
  $F(sk(x), sk(y)) = \text{YES}$ iff $H(sk(x), sk(y)) < R$

  for some $R$

• How low $t$ can we get?

• Will see $t = O(\log(1/\delta)/\varepsilon^2)$ suffices
Sketch: single bit

- **Setup:**
  - Choose a random set $S$ of coordinates
    - For each $i$, we have $\Pr[i \in S] = 1/r$
  - Choose a random vector $u$ in $\{0,1\}^d$
- **Sketch:** $\sum_S(x) = \sum_{i \in S} x_i u_i \mod 2$
- **Estimation algorithm:**
  - $B = \sum_S(x) + \sum_S(y) \mod 2$
  - YES, if $B = 0$
  - NO, if $B = 1$
- **Analysis:**
  - We have $B = \sum_S(z)$ where $z = x \ XOR \ y$
  - Let $D$ be the number of non-zeros in $z$
  - $\Pr[B=1] = \frac{1}{2} \times \Pr[z_S \neq 0]$
    - $= \frac{1}{2} \times \left[1 - \Pr[z_S = 0]\right]$
    - $= \frac{1}{2} \times \left[1 - (1 - 1/r)^D\right]$
  - For $r$ large enough: $(1 - 1/r)^D \approx e^{-D/r}$, so
    - If $D > (1+\epsilon)r$, then $e^{-(1+\epsilon)} < 1/e - \epsilon/3$ and $\Pr > 1/2(1-1/e + \epsilon/3)$
    - If $D < (1-\epsilon)r$, then $e^{-(1-\epsilon)} > 1/e + \epsilon/3$ and $\Pr < 1/2(1-1/e - \epsilon/3)$
Sketch: $t$ bits

- We have that, for $Pr=Pr[\text{Sum}_S(x)\leftrightarrow\text{Sum}_S(y)]$:
  - If $D > (1+\varepsilon)r$ then $Pr > 1/2(1-1/e + \varepsilon/3)$
  - If $D < (1-\varepsilon)r$ then $Pr < 1/2(1-1/e - \varepsilon/3)$

- Choose $S_1 \ldots S_t$, $t=(\log(1/\delta)/\varepsilon^2)$ and define
  $$sk(x)=\text{Sum}_{S_1}(x) \ldots \text{Sum}_{S_t}(x)$$

- Set $R=1/2(1-1/e)t$

- By Chernoff bound we have that
  - If $H(x,y) > (1+\varepsilon)r$, we have $H(sk(x), sk(y)) > R$
  - If $H(x,y) < (1-\varepsilon)r$, we have $H(sk(x), sk(y)) < R$

with probability $>1-\delta$
Sketch is good

• Data structure (for $P$, $r>1$, $\varepsilon>0$)
  – Compute $\text{sk}: \{0,1\}^d \to \{0,1\}^t$, $t=O(\log(1/\delta)/\varepsilon^2)$ for $\delta=1/n^{O(1)}$
    • Sketch works (with high probability) for fixed query $q$ and all points $p$ in $P$
  – Exhaustive storage trick:
    • Compute
      $$ S=\{u \text{ in } \{0,1\}^t : H(u,p)<R \text{ for some } p \text{ in } P \} $$
    • Store $S$ (space: $2^t=n^{O(1/\varepsilon^2)}$)

• Query: check whether $\text{sk}(q)$ in $S$