APPROXIMATE NEAREST NEIGHBOR QUERIES WITH A TINY INDEX
Outline

- Overview of Our Research
- SRS: c-Approximate Nearest Neighbor with a tiny index [PVLDB 2015]
- Conclusions
Research Projects

- Similarity Query Processing
- Keyword search on (Semi-) Structured Data
- Graph
- Succinct Data Structures
NN and c-ANN Queries

Definitions

- A set of points, \( D = \bigcup_{i=1}^{n} \{O_i\} \), in \( d \)-dimensional Euclidean space (\( d \) is large, e.g., hundreds)
- Given a query point \( q \), find the closest point, \( O^* \), in \( D \)
- Relaxed version:
  - Return a c-ANN point: i.e., its distance to \( q \) is at most \( c^* \text{Dist}(O^*,q) \)
  - May return a c-ANN point with at least constant probability

\[ d \text{-dimensional space} \]

\[ \text{c-ANN} = x \]

aka. \((1+\epsilon)\text{-ANN}\)
Applications and Challenges

Applications

- Feature vectors: Data Mining, Multimedia DB
- Fundamental geometric problem: “post-office problem”
- Quantization in coding/compression
- ...

Challenges

- Curse of Dimensionality / Concentration of Measure
  - Hard to find algorithms sub-linear in n and polynomial in d
- Large data size: 1KB for a single point with 256 dims
Existing Solutions

- **NN:** Linear scan is (practically) the best approach using linear space & time
  - $O(d^5 \log(n))$ query time, $O(n^{2d} + \delta)$ space
  - $O(dn^{1-\varepsilon(d)})$ query time, $O(dn)$ space
  - Linear scan: $O(dn/B)$ I/Os, $O(dn)$ space

- **(1 + \varepsilon)-ANN**
  - $O(\log(n) + 1/\varepsilon^{(d-1)/2})$ query time, $O(n^* \log(1/\varepsilon))$ space
  - Probabilistic test $\Rightarrow$ remove exponential dependency on $d$
    - Fast JLT: $O(d^* \log(d) + \varepsilon^{-3} \log^2(n))$ query time, $O(n^{\max(2, \varepsilon^{-2})})$ space
    - LSH-based: $\tilde{O}(dn^{\rho + o(1)})$ query time, $\tilde{O}(n^{1 + \rho + o(1)} + nd)$ space
      - $\rho = 1/(1 + \varepsilon) + o_c(1)$

*LSH is the best approach using sub-quadratic space*

*Linear scan is (practically) the best approach using linear space & time*
Approximate NN for Multimedia Retrieval

- Cover-tree
- Spill-tree
- Reduce to NN search with Hamming distance
- Dimensionality reduction (e.g., PCA)
- Quantization-based approaches (e.g., CK-Means)
Locality Sensitive Hashing (LSH)

- **Equality search**
  - Index: store \( o \) into bucket \( h(o) \)
  - Query: retrieve every \( o \) in the bucket \( h(q) \), verify if \( o = q \)

- **LSH**
  - \( \forall h \in LSH\text{-family}, \Pr[ h(q) = h(o) ] \propto 1/\text{Dist}(q, o) \)
  - \( h :: \mathbb{R}^d \rightarrow \mathbb{Z} \)
  - technically, dependent on \( r \)
  - “Near-by” points (blue) have more chance of colliding with \( q \) than “far-away” points (red)

**LSH** is the **best** approach using **sub-quadratic** space
LSH: Indexing & Query Processing

Index

- For a fixed $r$
  - $\text{sig}(o) = \langle h_1(o), h_2(o), \ldots, h_k(o) \rangle$
  - store $o$ into bucket $\text{sig}(o)$
- Iteratively increase $r$

Query

- Search with a fixed $r$
  - Retrieve and “verify” points in the bucket $\text{sig}(q)$
  - Repeat this $L$ times (boosting)
- Galloping search to find the first good $r$

Reduce Query Cost

Incurs additional cost + only $c^2$ quality guarantee
Locality Sensitive Hashing (LSH)

- **Standard LSH**
  - $c^2$-ANN $\Rightarrow$ binary search on $R(c^i, c^{i+1})$-NN problems

- **LSH on external memory**
  - **LSB-forest** [SIGMOD’09, TODS’10]:
    - A different reduction from $c^2$-ANN to a $R(c^i, c^{i+1})$-NN problem
  - **C2LSH** [SIGMOD’12]:
    - Do not use composite hash keys
    - Perform fine-granular counting number of collisions in $m$ LSH projections

- **SRS** (Ours)
  - $O(n/B)$ query, $O(n/B)$ space

$O((dn/B)^{0.5})$ query, $O((dn/B)^{1.5})$ space
Weakness of Ext-Memory LSH Methods

- Existing methods use super-linear space
  - Thousands (or more) of hash tables needed if rigorous
  - People resorts to hashing into binary code (and using Hamming distance) for multimedia retrieval
- Can only handle $c = x^2$, for integer $x \geq 2$
  - To enable reusing the hash table (merging buckets)
- Valuable information lost (due to quantization)
- Update? (changes to $n$, and $c$)

<table>
<thead>
<tr>
<th>Dataset Size</th>
<th>LSB-forest</th>
<th>C2LSH$^+$</th>
<th>SRS (Ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio, 40MB</td>
<td>1500 MB</td>
<td>127 MB</td>
<td>2 MB</td>
</tr>
</tbody>
</table>
SRS: Our Proposed Method

- Solving c-ANN queries with $O(n)$ query time and $O(n)$ space with constant probability
  - Constants hidden in $O()$ is very small
  - Early-termination condition is provably effective

- Advantages:
  - Small index
  - Rich-functionality
  - Simple

- Central idea:
  - c-ANN query in $d$ dims $\Rightarrow$ kNN query in $m$-dims with filtering
  - Model the distribution of $m$ “stable random projections”
Let $\mathcal{D}$ be the 2-stable random projection = standard Normal distribution $\mathcal{N}(0, 1)$

For two i.i.d. random variables $A \sim \mathcal{D}$, $B \sim \mathcal{D}$, then $x^*A + y^*B \sim (x^2+y^2)^{1/2} * \mathcal{D}$

Illustration:

$\mathbf{v} \cdot \mathbf{r}_1 \sim \mathcal{N}(0, \|v\|)$

$\mathbf{v} \cdot \mathbf{r}_2 \sim \mathcal{N}(0, \|v\|)$
Dist(O) and ProjDist(O) and Their Relationship

\[ z_1 := \langle V, r_1 \rangle \sim \mathcal{N}(0, \|v\|) \]
\[ z_2 := \langle V, r_2 \rangle \sim \mathcal{N}(0, \|v\|) \]
\[ z_1^2 + z_2^2 \sim \|v\|^2 * \chi^2_m \]

- z_1 := \langle V, r_1 \rangle \sim \mathcal{N}(0, \|v\|)
- z_2 := \langle V, r_2 \rangle \sim \mathcal{N}(0, \|v\|)
- \( z_1^2 + z_2^2 \sim \|v\|^2 * \chi^2_m \)

\[ \Psi_m(x) \]: cdf of the standard \( \chi^2_m \) distribution

- i.e., scaled Chi-squared distribution of m degrees of freedom

\( m \)-2-stable random projections

\( O \) in d dims

(\( z_1, \ldots, z_m \)) in m dims
LSH-like Property

- **Intuitive idea:**
  - If \( \text{Dist}(O_1) \ll \text{Dist}(O_2) \) then \( \text{ProjDist}(O_1) < \text{ProjDist}(O_2) \) with high probability.
  - But the inverse is NOT true.
    - NN object in the projected space is most likely **not** the NN object in the original space with few projections, as
    - Many far-away objects projected before the NN/cNN objects.
    - But we can bound the expected number of such cases! (say \( T \)).

- **Solution**
  - Perform incremental k-NN search on the projected space till accessing \( T \) objects.
  - + Early termination test.
Finding the minimum \( m \)

- **Input**
  - \( n, c, T \) = max # of points to access by the algorithm

- **Output**
  - \( m \) : # of 2-stable random projections
  - \( T' \leq T \): a better bound on \( T \)
  - \( m = O(n/T) \). We use \( T = O(n) \), so \( m = O(1) \) to achieve linear space index

Generate \( m \) 2-stable random projections  \( \Rightarrow \) \( n \) projected points in a \( m \)-dimensional space

Index these projections using any index that supports *incremental kNN search*, e.g., R-tree

Space cost: \( O(m \times n) = O(n) \)
SRS- $\alpha \beta (T, c, p_\tau)$

- Compute $\text{proj}(Q)$
- Do incremental kNN search from $\text{proj}(Q)$
  
  ```
  for k = 1 to T
      Compute $\text{Dist}(O_k)$
      Maintain $O_{\text{min}} = \arg\min_{1 \leq i \leq k} \text{Dist}(O_i)$
      If early-termination test $(c, p_\tau) = \text{TRUE}$
      \>
      \> BREAK
  
  Return $O_{\text{min}}$
  ```

**Early-termination test:**

\[
\psi_m \left( \frac{c^2 \times \text{ProjDist}^2(o_k)}{\text{Dist}^2(o_{\text{min}})} \right) > p_\tau
\]

Main Theorem:

SRS- $\alpha \beta$ returns a $c$-NN point with probability $p_\tau - f(m, c)$ with $O(n)$ I/O cost.

---

$c = 4$, $d = 256$, $m = 6$, $T = 0.00242n$, $B = 1024$, $p_\tau = 0.18$

Index = 0.0059n, Query = 0.0084n, succ prob = 0.13
Variations of SRS- $\alpha \beta (T, c, p_\tau)$

- Compute $\text{proj}(Q)$
- Do incremental $k$NN search from $\text{proj}(Q)$
  - for $k = 1$ to $T$
    - Compute $\text{Dist}(O_k)$
    - Maintain $O_{\text{min}} = \text{argmin}_{1 \leq i \leq k} \text{Dist}(O_i)$
    - If early-termination test $(c, p_\tau) = \text{TRUE}$
      - BREAK
  - Return $O_{\text{min}}$

// stopping condition $\alpha$

// stopping condition $\beta$

1. SRS- $\alpha$
   Better quality; query cost is $O(T)$

2. SRS- $\beta$
   Best quality; query cost bounded by $O(n)$; handles $c = 1$

3. SRS- $\alpha \beta (T, c', p_\tau)$
   Better quality; query cost bounded by $O(T)$

All with success probability at least $p_\tau$
Can be easily extended to support top-k c-ANN queries (k > 1)

No previous known guarantee on the correctness of returned results

We guarantee the correctness with probability at least $p_\tau$, if SRS- $\alpha \beta$ stops due to early-termination condition

$\approx 100\%$ in practice (97\% in theory)
Analysis
Stopping Condition \( \alpha \)

- “near” point: the NN point \( \Rightarrow \) its distance \( = r \)
- “far” points: points whose distance \( > c \ast r \)
- Then for any \( \kappa > 0 \) and any \( o \):
  - \( \Pr[\text{ProjDist}(o) \leq \kappa \ast r \mid o \text{ is a near point}] \geq \psi_m(\kappa^2) \)
  - \( \Pr[\text{ProjDist}(o) \leq \kappa \ast r \mid o \text{ is a far point}] \leq \psi_m(\kappa^2/c^2) \)
    - Both because \( \text{ProjDist}^2(o)/\text{Dist}^2(o) \sim \chi^2_m \)
  - \( \Pr[\text{the NN point projected before } \kappa \ast r] \geq \psi_m(\kappa^2) \)
  - \( \Pr[\text{# of bad points projected before } \kappa \ast r < T] > (1 - \psi_m(\kappa^2/c^2)) \ast (n/T) \)
  - Choose \( \kappa \) such that \( P1 + P2 - 1 > 0 \)
    - Feasible due to good concentration bound for \( \chi^2_m \)
Choosing $\kappa$

*Probability (y-axis) of Seeing an Object at Distance (x-axis)*

- Let $c = 4$
- Mode $= m - 2$
- Blue: 4
- Red: $4*(c^2) = 64$
Consider cases where both conditions hold (re. near and far points) \( \Rightarrow P_1 + P_2 - 1 \) probability.

ProjDist\((O_T)\): Case 1

- **NN point**
- Less than \( n \) "far" points

Distance to \( q \) in \( d \)-dims:

- \( 0 \) to \( r \)
- \( c \cdot r \)

Less than \( T \) "far" points

- \( 0 \) to \( \kappa \cdot r \)
- ProjDist\((o)\) in \( m \)-dims

\( O_{\text{min}} = \text{the NN point} \)
Consider cases where both conditions hold (re. near and far points) \( \Rightarrow P_1 + P_2 - 1 \) probability

\[
\text{ProjDist}(O_T): \text{Case II}
\]

- **NN point**
- **less than n "far" points**
- Distance to q in d-dims

\[
\text{less than T "far" points}
\]

\[
\text{ProjDist}(o) \text{ in m-dims}
\]

\[
\text{ProjDist}(O_T) \quad \text{min} = a \text{cNN point}
\]
Early-termination Condition ($\beta$)

- Omit the proof here
  - Also relies on the fact that the squared sum of $m$ projected distances follows a scaled $\chi^2_m$ distribution

- Key to
  - Handle the case where $c = 1$
    - Returns the NN point with guaranteed probability
    - **Impossible** to handle by LSH-based methods
  - Guarantees the correctness of top-$k$ cANN points returned when stopped by this condition
    - **No** such guarantee by any previous method
Experiment Setup

- **Algorithms**
  - LSB-forest [SIGMOD’09, TODS’10]
  - C2LSH [SIGMOD’12]
  - SRS-* [VLDB’15]

- **Data**

- **Measures**
  - Index size, query cost, result quality, success probability
Table 5: Statistics of the Datasets and Index Sizes (in Megabytes) (Italic numbers in parentheses are conservative estimates, as the indexes are too large to be built on the PC used in the experiment)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Statistics</th>
<th>Index Sizes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>n</td>
<td>d</td>
</tr>
<tr>
<td>Small</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Audio</td>
<td>54,287</td>
<td>192</td>
</tr>
<tr>
<td>SUN</td>
<td>80,006</td>
<td>512</td>
</tr>
<tr>
<td>Enron</td>
<td>95,863</td>
<td>1,369</td>
</tr>
<tr>
<td>Medium</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tiny</td>
<td>8,288,062</td>
<td>384</td>
</tr>
<tr>
<td>Large</td>
<td>ANN_SIFT450M</td>
<td>450,000,000</td>
</tr>
</tbody>
</table>

5.6PB  369GB  16GB
Tiny Image Dataset (8M pts, 384 dims)

- Fastest: SRS-$\alpha \beta$
- Slowest: C2LSH
- Quality the other way around
- SRS-$\alpha$ has comparable quality with C2LSH yet has much lower cost.
- SRS-* dominates LSB-forest
Approximate Nearest Neighbor

- Empirically better than the theoretic guarantee
- With 15% I/Os of linear scan, returns NN with probability 71%
- With 62% I/Os of linear scan, returns NN with probability 99.7%
Large Dataset (0.45 Billion)

Table 7: Our Algorithms on ANN_SIFT450M

<table>
<thead>
<tr>
<th>Alg.</th>
<th>SRS-12</th>
<th>SRS-12(^+) ((c' = 1.5))</th>
<th>SRS-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>Ratio</td>
<td>I/O</td>
<td>Ratio</td>
</tr>
<tr>
<td>300M</td>
<td>1.408</td>
<td>2,695</td>
<td>1.138</td>
</tr>
<tr>
<td>350M</td>
<td>1.410</td>
<td>2,897</td>
<td>1.135</td>
</tr>
<tr>
<td>400M</td>
<td>1.408</td>
<td>3,373</td>
<td>1.135</td>
</tr>
<tr>
<td>450M</td>
<td>1.412</td>
<td>3,722</td>
<td>1.138</td>
</tr>
</tbody>
</table>
Summary

- c-ANN queries in arbitrarily high dim space $\Rightarrow$ kNN query in low dim space
- Our index size is approximately $d/m$ of the size of the data file
- Opens up a new direction in c-ANN queries in high-dimensional space
  - Find efficient solution to kNN problem in 6-10 dimensional space
Q&A

Similarity Query Processing Project Homepage:
http://www.cse.unsw.edu.au/~weiw/project/simjoin.html