CLSH: Cluster-based Locality-Sensitive Hashing

Xiangyang Xu
Tongwei Ren
Gangshan Wu

Multimedia Computing Group, State Key Laboratory for Novel Software Technology, Nanjing University

xiangyang.xu@smail.nju.edu.cn
Outline

- Background
- Approach
- Experiment
- Conclusion
Outline

- Background
- Approach
- Experiment
- Conclusion
Nearest neighbor search

- Search over millions, even billions of data
  - Images, local features, other media objects, ...

- Applications
  - Image retrieval, computer vision, machine learning, ...
Challenges

- **Query precision and recall**
  - Basic requirements in nearest neighbor search
- **Query speed**
  - For high-dimensional spaces, there is no any generic exact algorithm that is faster than linear search [M. Muja, 2013]
  - \(O(n)\) complexity is prohibitive
- **Memory cost**
  - Increase in number of dimensions leads to rapid increase in volume

Challenges

- **Query precision and recall**  
  - Basic requirements in nearest neighbor search

- **Query speed**  
  - For high-dimensional spaces, there is no any generic exact algorithm that is faster than linear search [M. Muja, 2013]  
  - $O(n)$ complexity is prohibitive

- **Memory cost**  
  - Increase in number of dimensions leads to rapid increase in volume

Hashing-based methods

- $O(1)$ search time for single bucket
- Each bucket stores an inverted list
- Reranking may be needed
- LSH, spectral hashing, semi-supervised hashing, weakly-supervised hashing and kernelized LSH, etc.
Motivation and Contribution

■ Cluster-based
  ● Clustering algorithm
  ● Index is carried out on a distributed cluster

■ Centralized settings → distributed settings
  ● CLSH can cope with larger scale feature dataset
    □ Clustering and hashing
  ● The generated clusters can guide feature dataset
    automatic mappings to a distributed cluster
    □ One node cover one cluster
  ● Search time is significantly reduced
    □ Parallel searching on multiple computing nodes
Motivation and Contribution

- Cluster-based
  - Clustering algorithm
  - Index is carried out on a distributed cluster

- Centralized settings $\rightarrow$ distributed settings
  - CLSH can cope with large scale feature dataset

Efficiency & Scalability

- One node cover one cluster
- Search time is significantly reduced
  - Parallel searching on multiple computing nodes
Outline

- Background
- Approach
- Experiment
- Conclusion
Approach

- Index construction
- Nearest neighbor searching

Queries
1. Find the nearest ClusterID(s)
2. Use LSH to find NNs

Two criteria for choosing nearest ClusterID(s)

Feature points

Clustering

Locality-Sensitive Hashing

Writing to Index Files

n-dimensions

k-dimensions
Indexing construction

- Clustering the feature dataset
  - $k$-means

- LSH is employed in each cluster

Prob(hash code collision) is proportional to data similarity

\[ P \{ H(x) = H(y) \} = l \cdot \left[ 1 - \frac{\cos^{-1} x^\top y}{\pi} \right]^K \]

$l$: # hash tables, $K$: hash bits per table
Nearest neighbor searching

- Query near the cluster boundary
  - Search fixed number of clusters
  - **Search the clusters:**

\[
\frac{d_i}{\min\{d_i\}} \leq T (i = 1, \ldots, k)
\]
Outline

- Background
- Approach
- Experiment
- Conclusion
Experiments

- Experiment settings
  - Dataset
    - INRIA BIGANN (10K 128-d SIFT, 1M SIFT, 1M 960-d GIST)
  - LSH is a filter-and-refine framework, only recall is employed for measurement
# Results

### Table 1: Comparison on Recall

<table>
<thead>
<tr>
<th>Dataset</th>
<th>SIFT10K</th>
<th>SIFT1M</th>
<th>GIST1M</th>
</tr>
</thead>
<tbody>
<tr>
<td>E2LSH</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$s = 1$</td>
<td>0.8704</td>
<td>0.8926</td>
<td>0.7732</td>
</tr>
<tr>
<td>$s = 2$</td>
<td>0.9667</td>
<td>0.9494</td>
<td>0.9514</td>
</tr>
<tr>
<td>$s = 3$</td>
<td>0.9741</td>
<td>0.9494</td>
<td>0.9647</td>
</tr>
<tr>
<td>$T = 1.1$</td>
<td>0.9518</td>
<td>0.9319</td>
<td>0.8953</td>
</tr>
<tr>
<td>$T = 1.2$</td>
<td>0.9741</td>
<td>0.9494</td>
<td>0.9640</td>
</tr>
<tr>
<td>$T = 1.3$</td>
<td>0.9741</td>
<td>0.9494</td>
<td>0.9647</td>
</tr>
</tbody>
</table>

### Table 2: Comparison on the detailed distance evaluation times

<table>
<thead>
<tr>
<th>Dataset</th>
<th>SIFT10K</th>
<th>SIFT1M</th>
<th>GIST1M</th>
</tr>
</thead>
<tbody>
<tr>
<td>E2LSH</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$s = 1$</td>
<td>95.03</td>
<td>9,854.27</td>
<td>53,021.7</td>
</tr>
<tr>
<td>$s = 2$</td>
<td>124.64</td>
<td>13,318.5</td>
<td>91,421.2</td>
</tr>
<tr>
<td>$s = 3$</td>
<td>134.5</td>
<td>14,639.4</td>
<td>106,805</td>
</tr>
<tr>
<td>$T = 1.1$</td>
<td>108.17</td>
<td>11,078.8</td>
<td>75,891.2</td>
</tr>
<tr>
<td>$T = 1.2$</td>
<td>119.32</td>
<td>12,753.2</td>
<td>93,990</td>
</tr>
<tr>
<td>$T = 1.3$</td>
<td>128.46</td>
<td>13,467</td>
<td>107,738</td>
</tr>
</tbody>
</table>
Results (cntd.)

- Search time in our settings
  - 6 computing nodes (64-bit 2.00GHz, 8GB RAM each)

Table 3: Comparison on total search time (s)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>E2LSH</th>
<th>CLSH</th>
<th>Max{T_{ci}}</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>s = 1</td>
<td>s = 2</td>
</tr>
<tr>
<td>SIFT10K</td>
<td>0.00031</td>
<td>0.00021</td>
<td>0.00022</td>
</tr>
<tr>
<td>SIFT1M</td>
<td>0.01531</td>
<td>0.00813</td>
<td>0.00907</td>
</tr>
<tr>
<td>GIST1M</td>
<td>0.59721</td>
<td>0.25116</td>
<td>0.25832</td>
</tr>
</tbody>
</table>
Outline

- Background
- Approach
- Experiment
- Conclusion
Conclusion

- A distributed scalable framework for large-scale high-dimensional datasets indexing and searching
- Clustering is applied and the generated clusters are treated as a guideline to automatically deliver the feature dataset to a distributed cluster
- The search time is significantly reduced in CLSH framework
- Data-adaptive hashing function
- Extend our work to further applications
References

Thank you!