Nearest Neighbour Searching in High Dimensional Metric Space
A COMP6702 Research Project

David Marshall

Advisor: Dr. Shyjan Mahamud (NICTA)

November 10, 2006
Object Classification

Parametric
- Other
  - Data Mining: SPSS Clementine [SPSS Inc.]
  - Computer Vision
    - Nudity Filtering of EMail [Astaro]

Non-Parametric
- Other
  - Computer Vision
    - Multi Target Tracking [Cai et al. 2006]
  - Other
    - Plagiarism Detection in Documents [Finkel et al. 2002]
Computer Vision

SmartGate

Safe-T-Cam
What’s in a Title?

- **Nearest Neighbour Searching**: the point with minimum distance. (dissimilarity)
- in **High Dimensional**: more than 10 ...in this case 128.
- **Metric Space**: a distance measure exists and satisfies the triangle inequality.
Defining the Problem

<table>
<thead>
<tr>
<th></th>
<th>exact</th>
<th>approximate</th>
</tr>
</thead>
<tbody>
<tr>
<td>nearest</td>
<td>KD-Tree</td>
<td></td>
</tr>
<tr>
<td>near</td>
<td>LSH</td>
<td></td>
</tr>
</tbody>
</table>

Diagram: A circle with points $p_1$, $q$, $p_2$, and $p_3$. The point $q$ is inside the circle, and $p_1$ is outside.
What are the Neighbours?
SIFT: A More Complex Feature

- Scale Invariant Feature Transform
- Find extrema of intensity in “scale space”.
- Measure intensity gradients.
- Assign scale and orientation.
- 128 dimensional descriptor.
The Curse of Dimensionality

- Many algorithms try to improve upon $O(n)$ time by partitioning points into a binary tree.
- But they suffer from the “Curse of Dimensionality”.
- ...the exponentially growing difficulty of performing various types of spatial analysis in high dimensions.
- ...the tendency for points in high dimensional space to become equi-distant from each other.
A Trade-Off

- High dimensional features can be powerful at discriminating between classes.
- But... they also suffer from the curse of dimensionality.
- Relaxing the problem by considering “near” vs “nearest” and “approximate” vs “exact”.
Locality Sensitive Hashing (LSH)

- a point within distance $r_1$ should be hashed to the same bucket as query point $q$ with a high probability.
- a point with distance greater than $r_2$ should be hashed to the same bucket as query point $q$ with a low probability.
Spill Trees

- Extends a basic KD or M-Tree with “spilling”.
- Points near a partition will be stored on both sides.
- Search is more efficient, at the expense of storage space.
Rationale of LSHB

LSH: two 2-bit hashes

LSHB: one 2-bit hash
Spilling

A binary hash in LSH.

A binary hash in LSHB.
Effects of Buffer Size

![Graphs showing the effects of buffer size on error, query time, and structure size.](image-url)
The Speed Advantage

![Graph showing query time (nsec) vs hash sizes (bits) for LSH and LSHB methods. The query time decreases as hash sizes increase.]
The Space Trade-Off

A graph showing the relationship between Hash Sizes (bits) and Structure Size (# points stored) for LSH and LSHB.
Conclusions

- Where sufficient memory is available (2-3 times or more), LSHB outperforms LSH on SIFT data query speed by 10% to 20%.
- Both LSH and LSHB are appropriate near-neighbour search algorithms in the context of SIFT data.
Future Work

• Compare LSH and LSHB performance via an actual object classifier implementation.
• Investigate the selection of better performing hash functions, rather than random selection.
Questions?