FIT2010@Kyushu university

SketchSort: An Efficient Nearest Neighbor Graph Construction Method

Yasuo Tabei
JST Minato Project, Sapporo, Japan

Outline

- Motivation
- Method
- Experiments and Results

Data represented as vector

Text — Vector

x_t=(1, 0, 1, 0, 0, …)

Image

x_i=(0.2, -0.3, -1.3, 1.2, 2.2, …)

Chemical Compound, Protein, DNA/RNA etc

Locality Sensitive Hashing

(Gionis et al,99)

- Mapping vector to binary string (sketch)
- Conserve the distance in the original space
  - Enable to store gigascale data in main memory
  - Speed up learning algorithms

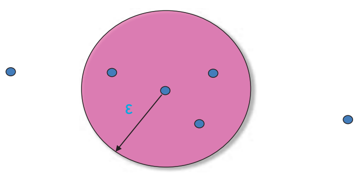
x=(0.2, -0.3, -1.3, 1.2, 2.2, …)

↓

s=101010111010101 01

All Pairs Similarity Search

- Finding all neighbor pairs from sketches
  - Find all pairs (i, j), i < j, $\Delta(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq \epsilon$
- Enable to build a neighborhood graph
  - semi-supervised learning, spectral clustering, ROI detection in images, retrieval of protein sequences



Single Sorting Method (SSM)

- Find neighbors by sorting sketches
  - Various applications ex) google news

(a)Input data	(b) Sort	(c) Scan neighbors
1:101111	7:000000	7:000000
2:110101	4:010000	4:010000
3:110010	8:010110	8:010110
4:010000	10:100100	10:100100
5:101000	5:101000	5:101000
6:111100	1:101111	1:101111
7:000000	3:110010	3:110010
8:010110	2:110101	2:110101
9:110110	9:110110	9:110110
10:100100	6:111100	6:111100

Drawbacks of Single Sorting

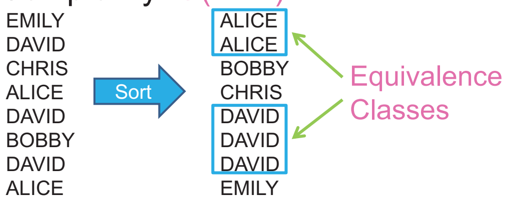
- Need a large number of distance calculation for achieving reasonable accuracy.
- Can not derive an analytic estimate of the fraction of missing neighbors.

Overview of SketchSort

- Employ the multiple sorting method (MSM) as a building block
  - Enumerate all pairs within Hamming distance d from a string pool $S=\{s_1,\dots,s_n\}$
- A number of distance calculation is significantly reduced
- A bound of the expected fraction of missing neighbors can be obtained.

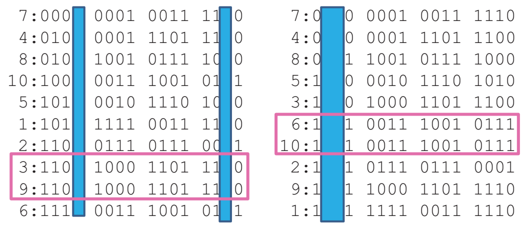
Special case: Finding identical strings(d=0)

- Radix sort, and partition the strings into equivalence classes: O(n)
- Build edges between all pairs in equivalent classes: O(m)
- Complexity: O(n+m)



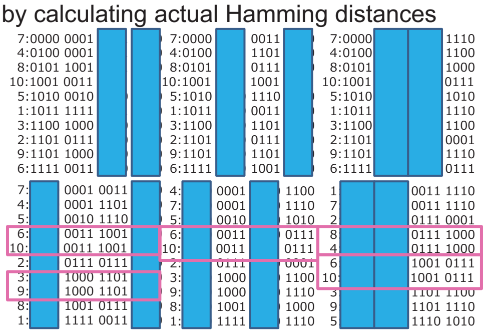
Multiple sorting method (d > 0)

- Mask d characters in all possible ways
- Perform radix sort $\binom{l}{d}$ times
- Time exponential to d, polynomial to the string lengh l
- Still linear to the number of strings!!
- Ex) d=2



Blockwise masking

- Mask d blocks in all possible ways
- The number of sotring operations reduced
- Non-neighbors might be detected
  - Filtered out by calculating actual Hamming distances
- Ex)d=2



Recursive Algorithm

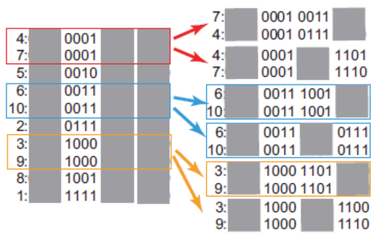


Figure 5: Updating equivalence classes in block concatenation. Strings in a block are sorted and equivalence classes (shown as square frames) are detected. A next block is concatenated to each equivalence class and sorted again.

SketchSort

- Basic idea: Map vectors to strings and apply MSM
- Not good: Create long strings and apply MSM at once
- Replication:
  - Create Q independent string pools of length l
  - apply MSM to each string pool
- Report the pairs less than a threshold ε

$$\Delta(\boldsymbol{x}_i, \boldsymbol{x}_j) \leq \epsilon$$

Duplication Checks

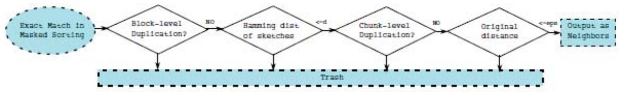


Figure 2: Global flow of our approach.

- Block-level duplication check
  - Define dictionary order of blocks, and take only minimum combinations of blocks.

ex) d=2

(1,2)<(1,3)<(1,4)<(2,3)<(2,4)<(3,4)

- Chunk-level duplication check
  - Take only minimum chunks.

Two types of errors

- True edges E*, Our results E
- Type-I error (false positive): A non-neighbor pair has a Hamming distance within d in at least one replicate

$$F_1 = \{(i,j) \mid (i,j) \in E, (i,j) \notin E^*\}.$$

- Type II-error (false negative): A neighbor pair has a Hamming distance larger than d in all replicates

$$F_2 = \{(i,j) \mid (i,j) \notin E, (i,j) \in E^*\}.$$

Bound of type-II error: Missing edge ratio

- Basically, type-II error is more crucial
  - type-I errors are filtered out by distance calculations
- Missing edge ratio (type-II error) is bounded as

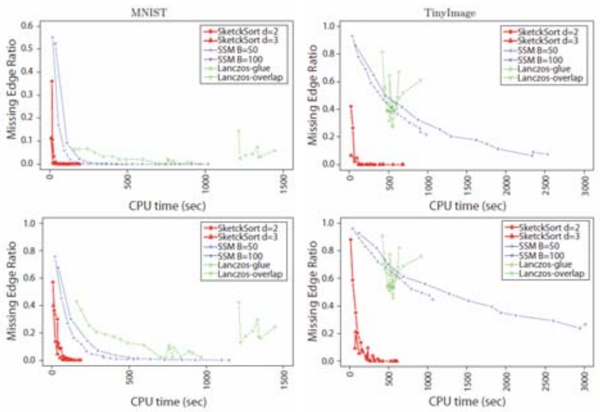
$$E\left[\frac{|F_2|}{|E^*|}\right] \leq \left(1 - \sum_{k=0}^{\lfloor d \rfloor} \binom{\ell}{k} p^k (1-p)^{\ell-k}\right)^Q,$$

where p is an upper bound of the non-collision probability of neighbors

$$p = \frac{\arccos(1-\epsilon)}{\pi}.$$

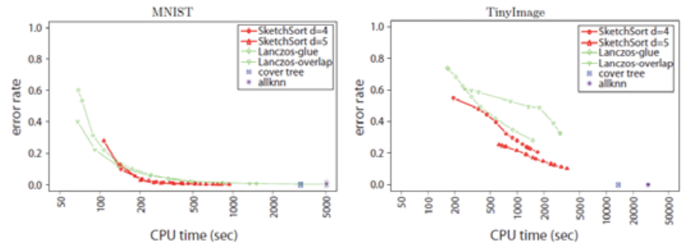
Results for All Pairs Similarity Search

Faster and more accurate than recent methods



All pairs similarity search on MNIST and TinyImage datasets for cosine distance thresholds 0.10Π (top) and 0.15Π (bottom).

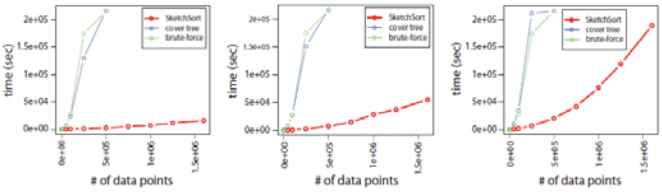
Results for 5-nearest neighbor search



Error rate for 5-nearest neighbor search on MNIST and TinyImage datasets

All Pairs Similarity Search in 1.6 Million Images

- Set parameters so as to keep missing edge ratio no more than 1.0×10^{-6}
- Enable to detect similar pairs nearly exactly
- Take only 4.3 hours for 1.6 million images



Near duplication detection in up to 1.6 million images at threshold 0.05Π (left), 0.10Π (middle) and 0.15Π (right)

A C++ implementation of SketchSort is available from http://code.google.com/p/sketchsort/