# Similarity Search in High Dimensional Spaces: Application to Multimedia

Vincent Oria
New Jersey Institute of Technology
Newark, NJ 07102
USA



# Acknowledgement

- Michael Houle (NII, Tokyo)
  - Search in high dimensional spaces
  - Image annotation through face tag propagation
- Shin'ichi Satoh (NII, Tokyo)
  - Image annotation through face tag propagation
- Jichao Sun (NJIT, USA)
  - Image annotation through face tag propagation

# Outline

- Multimedia
  - Motivating application
    - Knowledge Propagation in Large Image Databases
- Similarity Search and Intrinsic Dimensionality
- Similarity Search and the Curse of Dimensionality
  - Locality Sensitive Hashing (LSH)
  - Cover Tree (CT)
  - Rank Cover Tree (RCT)
- Conclusion



# Multimedia Data

- Multimedia in principle means data of more than one medium
- Commonly used forms of data are numbers, alphanumeric, text, images, audio, and video
- Multimedia denotes a combination of text, audio, and video

# Chronology of Data Types in Computer Science

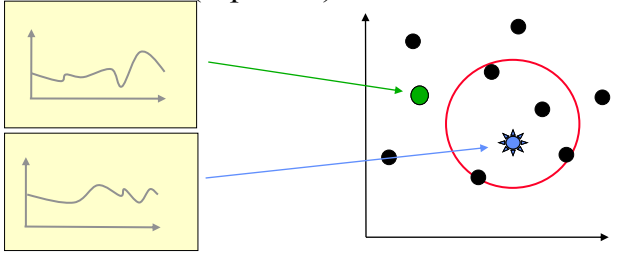
- Numeric Data: scientific computations, early stages of computing
- Alphanumerical Data: Business Applications
  - large volumes of data
  - RDBMS, E-R Model
- Multimedia Data: Novel Applications
  - Text
  - Image
  - Video
  - Audio



# Similarity Searches in Multimedia

1) extract from each object N *numerical features* and map objects into points of a N-dimensional space

2) use a suitable *distance* (e.g., Euclidean) over such a space, and search for "close" objects using a multi-dimensional ("spatial") index

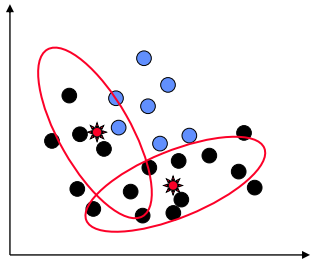


* Slide borrowed from Paolo Ciaccia

# Vector-based Similarity Searches*

- Using the same distance function is not always appropriate

Example: retrieve (only) black points



* Slide borrowed From Paolo Ciaccia



# Outline

- Multimedia
  - Introduction
  - Motivating applications
    - An Example of Music Search Application
- Similarity Search and Intrinsic Dimensionality
  - Similarity Search and the Curse of Dimensionality
  - Experiments
- Conclusion

KProp: Knowledge Propagation in Large Image Databases Using Neighborhood Information

Michael Houle, Vincent Oria, Shin'ichi Satoh,
Jichao Sun

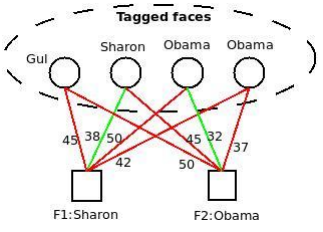
(ACM MM 2011)



# A query-based baseline -- Bestmatch

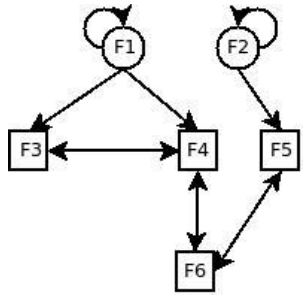
Bestmatch is a simple greedy algorithm which:

- Computes pairwise visual distances of detected objects;
- For each unlabeled object u, find its nearest labeled object v;
- Assign label t to u, where t is the label attached to v.



# Building the influence graph

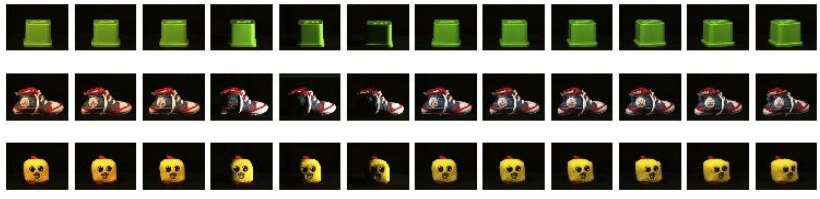
- Object relationships are modeled as a directed influence graph -- G(V, E), with the node set partitioned info $V=V_l \cup V_u$, where $V_l$ and $V_u$ represent the initially-labeled (source) object set and initially-unlabeled (non-source) object set, respectively. E is composed of 3 types of edges:
  - $\forall v \in V_l$, $\langle v,v \rangle \in E$;
  - $\langle v,u \rangle \in E$, whenever $v \in V_l$, $u \in V_u$ and v influences u; and;
  - $\langle u,u' \rangle$, $\langle u',u \rangle \in E$ whenever $u,u' \in V_u$, and either u influences u', or u' influences u (or both)






# Sample images of the datasets

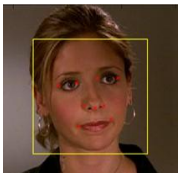
- ALOI-100



- Google-23

# Feature Descriptor and Distance Measure

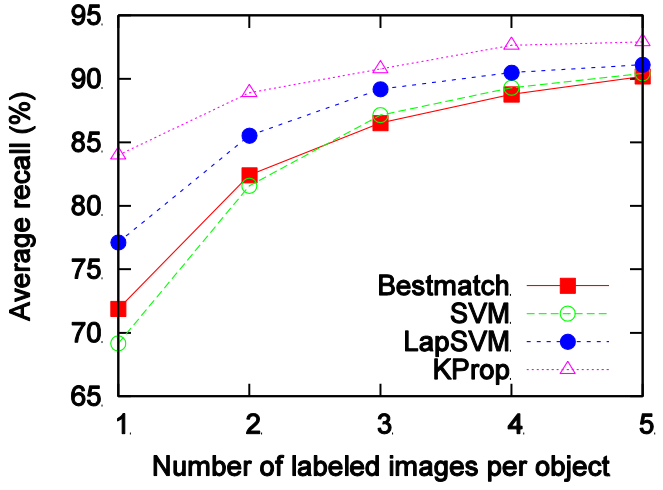
- Google-23 Face Set
  - Frontal faces are detected by the face detector of OpenCV 1.0
  - Feature descriptors are computed by the Oxford Matlab code:
    - 13 (9 detected +4 inferred) interest points
    - 149-D vector computed around each interest point
    - 1937-D vector for each face
  - Euclidean distance (L2) is used as distance measure



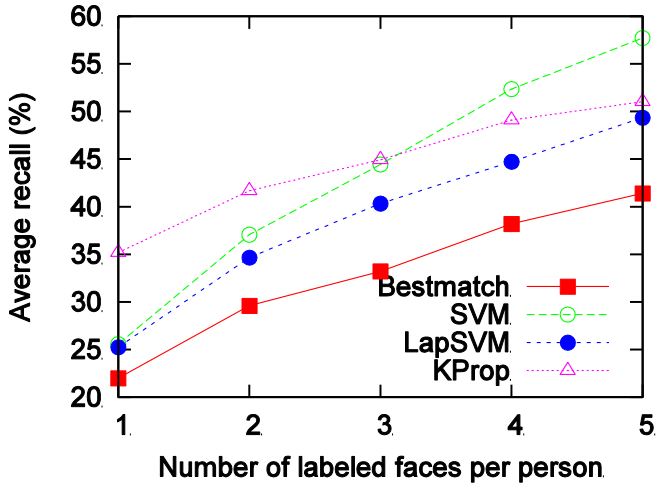
- ALOI
  - Each image is represented by a 641-D vector based on color and texture histograms
  - Again L2 distance is used as distance measure



# Experimental results – ALOI-100



# Experimental results – Google-23



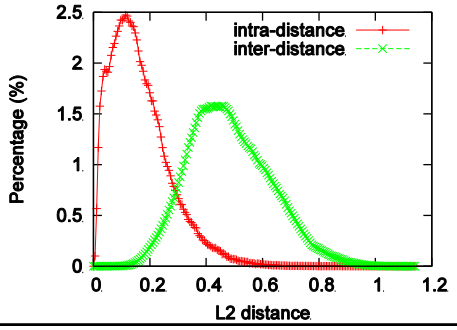


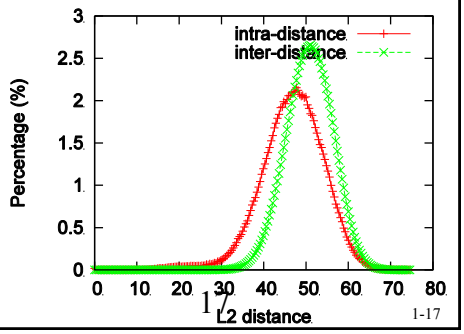

# Discussion

- All four methods perform better on ALOI-100 than on Google-23.
- LapSVM is not consistently better than SVM --- it beats SVM on ALOI-100 but loses to SVM on Google-23 --- since it is sensible to labeled data and usually needs to be well tuned.
- KProp has much better performance than all other methods especially when the number of labeled sample is small (say 1, 2 or 3). It is always better than SVM on ALOI-100 but SVM overtakes KProp on Google-23 when more than 3 faces are labeled per person. This can be explained by the transitivity of object relationships.

# Discussion (cont'd)

- Distance distributions of the two datasets (from left to right: ALOI-100 and Google-23).
- It can be seen from the figures that, it is much difficult to tell whether two faces belong to a same person by their distance.






# Outline

- Multimedia
  - Motivating application
    - Knowledge Propagation in Large Image Databases
- Similarity Search and Intrinsic Dimensionality
  - Similarity Search and the Curse of Dimensionality
  - Locality Sensitive Hashing (LSH)
  - Cover Tree (CT)
  - Rank Cover Tree (RCT)
- Conclusion

# Spatial Indexing

- Traditional approaches:
  - Data drawn from a real vector space $\mathbf{R}^m$.
  - Distance function $d$: $\mathbf{R}^m \rightarrow \mathbf{R}$.
  - Structure makes use of data representation.
  - Organization often depends on hierarchical decomposition of the domain → costs typically exponential in $m$.
  - Triangle inequality used for pruning of search paths.
- Examples:
  - R-tree (Guttman 1984).
  - SR-tree (Katayama & Satoh 1997).
  - Quadtree (Finkel & Bentley 1974), Octree.
  - *k-d* tree (Bentley 1975).
  - Many, many more…

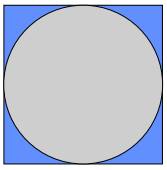


# Spatial Indexing

- Traditional approaches:
  - Data drawn from a real vector space Rm.
  - Distance function d: Rm □ R.
  - Structure makes use of data representation.
  - Organization often depends on hierarchical decomposition of the domain □ costs typically exponential in m
- Examples:
  - R-tree (Guttman 1984).
  - SR-tree (Katayama & Satoh 1997).
  - Quadtree (Finkel & Bentley 1974), Octree.
  - k-d tree (Bentley 1975).
  - Many, many more…

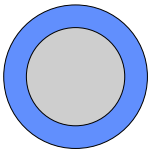
# The Curse of Dimensionality

- Spatial effect of dimensionality:
  - Exponential increase in volume associated with increase in dimensionality.
  - Distances concentrate around their mean values → indistinguishable.
  - Variances tend to zero as a proportion of the mean.
  - Points tend to concentrate along region boundaries.
- Implications for search:
  - Search paths begin to look identical.
  - Modelling of data becomes more difficult.

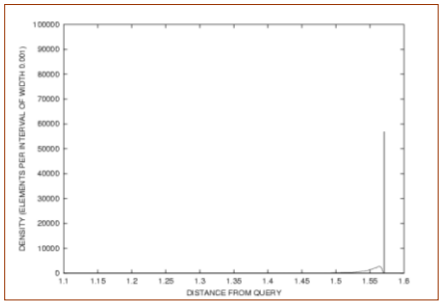


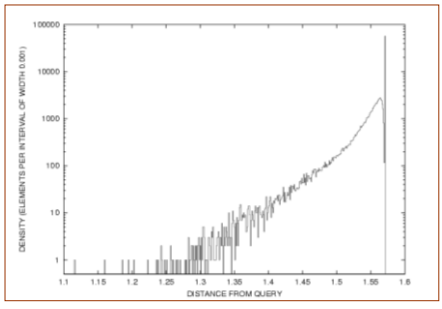
$$VS_d = 2^d r^d$$

$$VB_d = \frac{2\pi^{d/2}}{\Gamma(1+d/2)} r^d$$








# The Curse of Dimensionality

Searching high-dimensional data:

- Exact similarity queries require close to linear time.
- Data organization is a major challenge.
- 2D and 3D intuition does not apply!

But!... This doesn't mean that neighborhoods are meaningless!

Example: LA-Times 127738x6590 text data set, vector angle metric

# Intrinsic Dimension and Search

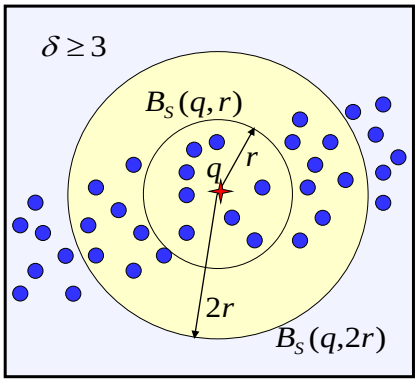
- Idea: performance analysis in terms of a characterization of the data, not the space.
- Question: what dimension does the data appear to be in?
- Many measures have been proposed, including:
  - Fractal dimension.
  - Doubling dimension.
  - Expansion dimension.
- We will look at two approaches with analysis based on expansion dimension:
  - Distance-based: Cover Tree [Beygelzimer et al, 2006].
  - Rank-based: Rank Cover Tree [H. and Nett, submitted].



# Expansion Dimension

- Expansion rate of set *S*:
  - Introduced by Karger & Ruhl (2002).
  - Used to measure of the cost of an expanding search in the vicinity of a query point *q*.
  - Maximum ratio d of the number of points in two balls centered at *q*.
- Expansion dimension: $D = \log_2 \delta$
  - Measure of intrinsic dimensionality of *S*.
  - If representational dimension is *m*…
  - Doubling the radius of a sphere → volume increases by factor $2^m$.

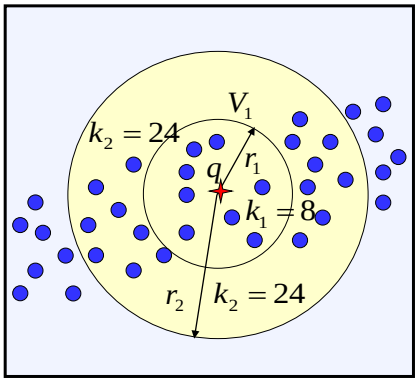
$$|B_S(q,r)| \geq b$$
$$\Rightarrow |B_S(q,2r)| \leq \delta \cdot |B_S(q,r)|$$



# Generalization of expansion dimension

- Generalization of expansion dimension.
- Choose any two spheres of positive, unequal radii.
- If volumes are known, can compute representational dimension.
- Volumes are not known, so...
- ...estimate using numbers of points captured by the spheres.
- Two sets of measurements allows for assessment of local intrinsic dimensionality.
- Can characterize data sets according to average stereological dimension.

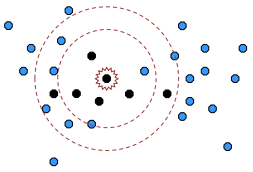
$$d = \frac{\log \frac{V_2}{V_1}}{\log \frac{r_2}{r_1}} \quad \Rightarrow \quad D \approx \frac{\log \frac{k_2}{k_1}}{\log \frac{r_2}{r_1}}$$






# Coping with the Curse: *k*-NN Approximation Methods

- Trade exactness of similarity query for efficiency.
- Older approximation methods of mainly theoretical interest.
- Performance typically depends on:
  - Dimension.
  - Relative distance error.
  - Probability of correctness.
- Distance error approximation is much less effective in high dimensions!



# $k$-NN Approximation Methods

- Several results claiming speedups of 1-2 orders of magnitude, over sequential search:
  - Metric data: M-Tree (Zezula et al., 1998).
  - Vector data: LSH (Indyk and Motwani, 1998; with Gionis, 1999).
  - Vector data: clustering based approximation (Ferhatosmanoglu et al., 2001).
  - ...
- Data sets typically of the order of $10^3$ -$10^5$ elements and 50-200 attributes.
- Time / accuracy tradeoff difficult to manage in practice.
- No consensus on how to measure accuracy.



# Some Measures of Accuracy

Distance Based

$$A_1(q,U) = \frac{\sum_{i=1}^{|U|} dist(q,u_i)}{\sum_{i=1}^{|U|} r_i}$$

$$A_2(q,U) = \frac{1}{|U|}\sum_{i=1}^{|U|} \frac{dist(q,u_i)}{r_i}$$

Rank Based

$$A(q,U) = \max_{U' \subseteq U} \frac{|U'|}{|U|}$$

Definitions

- $U = \{u_1, u_2, \ldots\}$ : approx NN set.
- $U'$ : subset of some (unknown) exact $k$-NN set.
- $r_i$ : (unknown) distance from $q$ to exact $i$-th NN.

# Outline

- Multimedia
  - Motivating application
    - Knowledge Propagation in Large Image Databases
- Similarity Search and Intrinsic Dimensionality
  - Similarity Search and the Curse of Dimensionality
  - Locality Sensitive Hashing (LSH)
  - Cover Tree (CT)
  - Rank Cover Tree (RCT)
- Conclusion

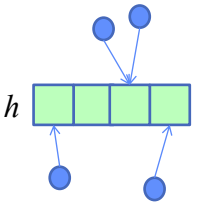


# Locality Sensitive Hashing

- Indyk and Motwani (1998), with Gionis (1999)
  - "Locality-sensitive" hashing technique.
  - Hash data so that similar items are mapped to the same bucket with high probability.
  - Data modeled as vectors
  - Hamming distance with bit sampling.
- Advantages
  - 1-2 orders of magnitude speedup possible, for $n \sim 10^5$, $d \sim 60$.
  - General technique extensible – very popular!
- Drawbacks
  - Accuracy measured according to distances but not rank..

# LSH Families

- LSH family $F$ of hash functions for metric space $M$:
  - Distance function $d : M \rightarrow R^{\geq 0}$.
  - Hash table $T$.
  - Hash functions $h \in F : M \rightarrow T$.
  - Distance threshold $r$.
  - Approximation factor $c > 1$.
  - Randomly-selected hash function $h \in F$.

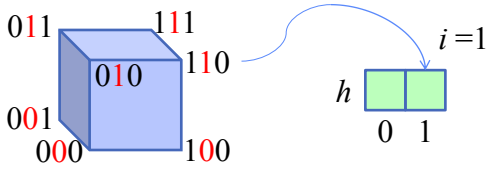



- $F$ is a $(r,cr,p,q)$-sensitive family when:
  - Close points likely map to the same bucket:
    $d(x,y) \leq r \Rightarrow h(x) = h(y)$ with probability at least $p$.
  - Far points likely map to different buckets:
    $d(x,y) \geq cr \Rightarrow h(x) = h(y)$ with probability at most $q$.
  - Interesting when $p > q$.



# Examples of LSH Families

- Bit-sampling LSH:
  - Each point represented as an $m$-dimensional bit vector.
  - Hamming distance $d : \{0,1\}^m \rightarrow R^{\geq 0}$, number of differing bits.
  - LSH family: $h_i(x)$ selects $i$-th bit of $x$.
  - Choose some distance threshold $r$ and approximation factor $c > 1$.
  - $(r,cr,p,q)$-sensitive for $p = 1 - r/m$ and $q = 1 - cr/m$.

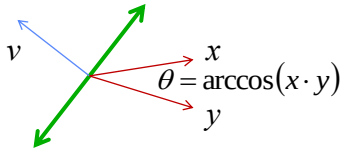


# Examples of LSH Families

- Separating hyperplanes (random projections):
  - LSH family: $h(x)$ associated with hyperplane with normal vector $\boldsymbol{v}$.
  - Random hash function $h(x)$ = **sign** $(\boldsymbol{v} \bullet x) = \pm 1$ , for random $\boldsymbol{v}$.
  - Depends on which side of the hyperplane $x$ lies.
  - For uniformly distributed data:

$$\Pr[h(x) = h(y)] = 1 - \arccos(x \cdot y)/\pi$$

  - Sensitive LSH family for vector angle distance metric **arccos**.






# Approximate 1-NN Search Using LSH

- Basic idea:
  - Start with any LSH family $F$.
  - Construct a new LSH family $G$ by concatenation of $w$ hash functions from $F$.

$$g(x) = [h_1(x), \cdots, h_w(x)]$$

  - Choose $u$ random hash functions from $G$ .
  - Preprocessing step: hash all points of the dataset into the $u$ hash tables.
- Processing of query $q$ :
  - For each hash function $g$, search through the buckets indexed by $g(q)$.
  - Stop once we find a point $x$ for which $d(q,x) \leq cr$ .

# Approximate 1-NN Search Using LSH

- Performance:
  - $t_h$ : time to evaluate hash functions of $F$.
  - $t_d$ : time to evaluate distance function.
  - $n$ : number of points in the data set.
  - Other design choices: $\rho = \frac{\log p}{\log q}; \quad u = n^{\rho}; \quad w = \frac{\log n}{\log 1/q}$
  - Preprocessing time: $O(nuwt_h) \rightarrow O(n^{1+\rho} wt_h)$
  - Additional space: $O(nu) \rightarrow O(n^{1+\rho})$
  - Query time: $O(uwt_h + nuq^w t_d) \rightarrow O(n^{\rho} wt_h + n^{\rho} t_d)$
  - Prob. of finding neighbour within distance $cr$ : $\Omega(up^w) \rightarrow \Omega(1)$

35



# Approximate 1-NN Search Using LSH

- Requirements for scalability:
  - Small approximation factor $c$.
  - Probability $p$ must be much larger than probability $q$.
  - If the representational dimension is high, the distance computation time $t_d$ must not depend on it.
- Conclusion:
  - LSH has intriguing possibilities for data mining, but ...
  - ... the family of hash functions must be quite sensitive!
  - Hashing typically depends on the representational dimension.
  - Better practical performance by abandoning theoretical guarantees → heuristics!

| $p$ | $q$ | $\rho = \log p / \log q$ |
|---|---|---|
| 0.80 | 0.20 | 0.1386 |
| 0.70 | 0.30 | 0.2962 |
| 0.60 | 0.40 | 0.5575 |
| 0.55 | 0.45 | 0.7487 |
| 0.50 | 0.50 | 1.0000 |

# Outline

- Multimedia
  - Motivating application
    - Knowledge Propagation in Large Image Databases
- Similarity Search and Intrinsic Dimensionality
  - Similarity Search and the Curse of Dimensionality
  - Locality Sensitive Hashing (LSH)
  - Cover Tree (CT)
  - Rank Cover Tree (RCT)
  - Experiments
- Conclusion



# Cover Tree

- Introduced by Beygelzimer, Kakade & Langford (2006).
- Tree index for exact similarity search.
  - (1+e)-distance approximation also possible using early termination.
- Metric space data
  - Triangle inequality satisfied.
  - No assumed knowledge of data representation or dimension.
- Worst-case performance optimal in *n*:
  - *O* (*n* log *n*) construction, *O* (log *n*) search, insertion, deletion.
  - No explicit dependence on representational dimension, BUT...
  - Strong dependence on a measure of intrinsic dimensionality.

# Cover Tree Properties

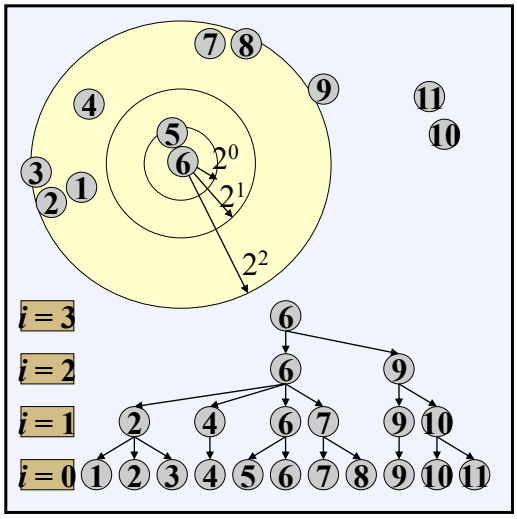
- Let *u* be a node at level *i*-1.
- Let *v* be the parent of *u* at level *i*.
- Let *w* be another node at level *i*.
- Covering tree condition:

$$d(u,v) < 2^i$$

- Separation condition:

$$d(v,w) \geq 2^i$$

- Closest pair assumed to have distance 1.



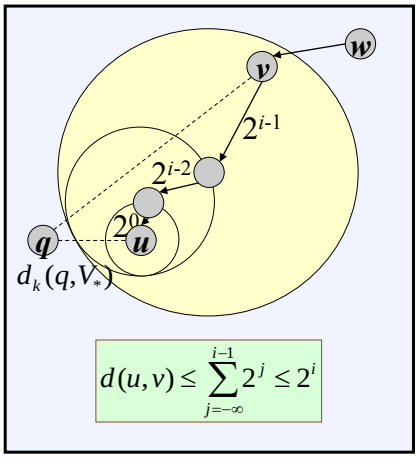



# Cover Sets

- Assume that we have:
  - Query item *q*.
  - At level $i > 0$, a set of nodes $V_i$ containing ancestors of every *k*-NN *u* of *q* (Cover Set).
  - Let $V_*$ be the set of all children of $V_i$.
  - Distance to *k*-th NN of $V_*$: $d_k(q,V_*)$
- Implication:
  - Can form new cover set at level *i* -1 as

$$V_{i-1} = \{v \in V_* : d(q,v) \leq d_k(q,V_*) + 2^i\}$$

  - All children of *v* are within distance $2^i$ of *v*, so no ancestor of neighbor *u* is discarded.



# Cover Tree Search

- Find *k*-Nearest (cover tree *T*, query point *q*, number of neighbors *k*):
  - Set $V_h = L_h$ to be the initial cover set (all nodes are descendants).
  - For $i = h\text{-}1$ down to 1:
    - Set $V_*$ be the set of all children of $V_i$.
    - Form new cover set $V_{i-1} = \{v \in V_* : d(q,v) \le d_k(q,V_*) + 2^i\}$
  - Return the *k* items of $V_0$ closest to *q*.
- Insertion and deletion resemble search:
  - Insertion by local modification of tree structure after search.
  - Only the highest-level copy of the item is explicitly inserted.
  - Deletion slightly more complex.
- Construction by successive insertion.



# Cover Tree Performance

- Let the expansion dimension be $D = \log_2 d$.

| Operation | WC Cost (in $\delta$) | WC Cost (in $D$) |
|---|---|---|
| Construction (Space) | $n$ | $n$ |
| Construction (Time) | $\delta^6 n \log_2 n$ | $2^{6D} n \log_2 n$ |
| Insert / Delete | $\delta^6 \log_2 n$ | $2^{6D} \log_2 n$ |
| 1-NN Query | $\delta^{12} \log_2 n$ | $2^{12D} \log_2 n$ |

- Practical speedups over sequential search: variable!
  - Datasets from KDD and UCI machine learning archives (among others).
  - Speedups of between 10-100 times is common.
  - Some sets achieved 1000 times speedup, others essentially no speedup.
  - Substantial speedups coincide with smallest expansion dimensions (< 20).
  - However, very large real datasets can have expansion dimensions in the thousands.

# Outline

- Multimedia
  - Motivating application
    - Knowledge Propagation in Large Image Databases
- Similarity Search and Intrinsic Dimensionality
  - Similarity Search and the Curse of Dimensionality
  - Locality Sensitive Hashing (LSH)
  - Cover Tree (CT)
  - Rank Cover Tree (RCT)
  - Experiments
- Conclusion

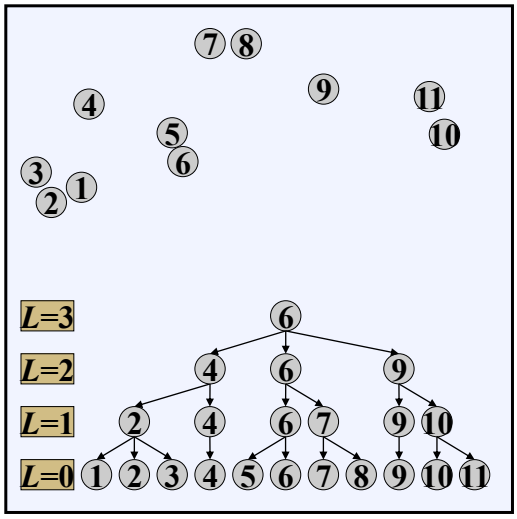


# Rank Cover Tree

- New tree index for similarity search based on the Cover Tree.
  - Design based on neighborhood ranks to the query, instead of distances to the query.
  - Computes exact *k*-NN similarity queries with extremely high probability.
  - Can accelerate performance at the expense of exactness.
- Metric space data
  - Triangle inequality satisfied.
  - No assumed knowledge of data representation or dimension.

# Rank Cover Tree

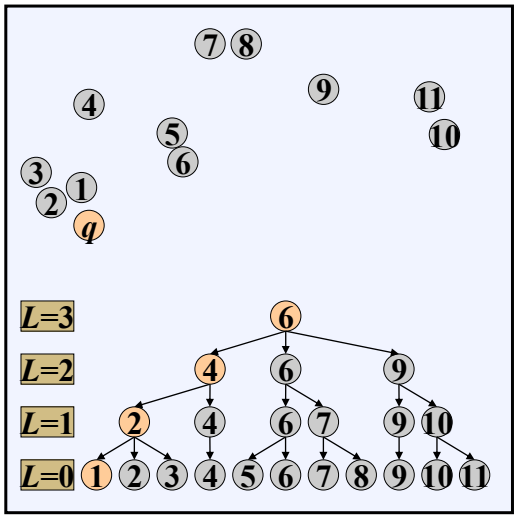
- Random leveling.
  - Copy promoted to higher level with probability 1/Δ.
  - Expected number of nodes at level $L$ is $n / \Delta^L$.
  - Expected height of the tree is ~ $\log_\Delta n$.
  - Similar to skip list index.
- Well-formed condition:
  - For each node $u$ at level $L$-1, its parent $v$ is the nearest neighbor of $u$ from level $L$.






# RCT Strategy

- Cover Tree vs RCT.
  - As search descends the CT, cover distance drops by a factor of 2.
  - In the RCT a random leveling, the 1-NN neighbor within the current level has an expected NN rank with respect to the full set, AND...
  - Expected rank drops by factor of 2.
- Implication :
  - Can form sets at each level that cover all $k$-NNs with very high probability.
  - The cardinality of the cover set does not depend on $2^k$, and only sublinearly on $n$.



# RCT Search

- Find *k*-Nearest (RCT *T*, query point *q*, number of neighbors *k*, coverage parameter $\psi$):
  - Set $V_{h-1}=L_{h-1}$ to be the initial cover set (all nodes are descendants).
  - For $j = h-2$ down to 0:
    - Set $V_*$ be the set of all children of $V_{j+1}$.
    - Let the cover set size quota at level *j* be:

      $$k_j = \psi \cdot \max\left\{\frac{k}{\Delta^j}, 1\right\}$$

    - Set $V_j$ to be the set of items of $V_*$ attaining the smallest $k_j$ distances from *q* (or all items of $V_*$ if its size is less than $k_j$).
  - Return the *k* items of $V_0$ closest to *q*.
- Construction by level-order insertion – parent of *u* is the 1-NN from among the items of the level immediately above *u*.



# Coverage Parameter

- How should we choose the coverage parameter $\psi$?
  - Can show: if $\psi$ is chosen to be sufficiently large, the query is exact with high probability.
  - Smaller choices allow for speedup at the expense of accuracy.
  - Expected time cost of queries can be controlled through the choice of $\psi$.
- Outcome of analysis:
  - Assume that the coverage parameter is chosen as:

    $$\psi = (ch + \max\{2h, e\Delta\}) \cdot \delta^{\lfloor \log_\psi(\sqrt{5}h) \rfloor}$$

  - Then with probability at least $1 - 1/n^c$
    - ... RCT construction produces a well-formed tree in expected time at most $O(\psi\delta\Delta nh)$
    - ... when the tree is well-formed, RCT similarity search produces a correct result in expected time at most $O(\psi\delta\Delta(k+h))$

# Complexities

- If the sampling rate $\Delta$ is fixed:
  - $h = \log_{\Delta} n$.
  - Construction time is in $O\left(c\delta^{2.68+1.441\log_2 \log_{\Delta} n} \cdot n \log^2 n\right)$
  - Search time is in $O\left(c\delta^{2.68+1.441\log_2 \log_{\Delta} n} \cdot (k+\log n)\log n\right)$
- If the tree height $h$ is fixed:
  - $\Delta = n^{1/h}$.
  - Construction time is in $O\left(c\delta^{2.68+1.441\log_2 h} \cdot n^{1+(2/h)}\right)$
  - Search time is in $O\left(c\delta^{2.68+1.441\log_2 h} \cdot kn^{2/h}\right)$
- All with very high probability $1-1/n^c$



# Comparison of RCT with CT

- CT is exact, RCT is correct with very high probability.
- RCT achieves much smaller dependence on the intrinsic dimensionality while still being sublinear in $n$.
- CT real cost involves keeping track of nodes that lie in regions of diameters of very large length (exponential in 2) $\rightarrow$ for some distance measures, all data points could lie in these bounds until the very lowest levels of the search!
- RCT real costs are decided through the explicit choice of the coverage parameter $\psi$.

| Operation | CT Cost | RCT Cost ($h$=3) | RCT Cost ($h$=4) | RCT Cost ($h$=8) |
|---|---|---|---|---|
| 1-NN Query | $\delta^{12}\log_2 n$ | $\delta^{4.97}n^{2/3}$ | $\delta^{5.57}n^{1/2}$ | $\delta^{7.01}n^{1/4}$ |

# Outline

- Multimedia
  - Introduction
  - Motivating applications
    - An Example of Music Search Application
    - Knowledge Propagation in Large Image Databases
- Similarity Search and Intrinsic Dimensionality
  - Similarity Search and the Curse of Dimensionality
  - Locality Sensitive Hashing (LSH)
  - Cover Tree (CT)
  - Rank Cover Tree (RCT)
  - Experiments
- Conclusion

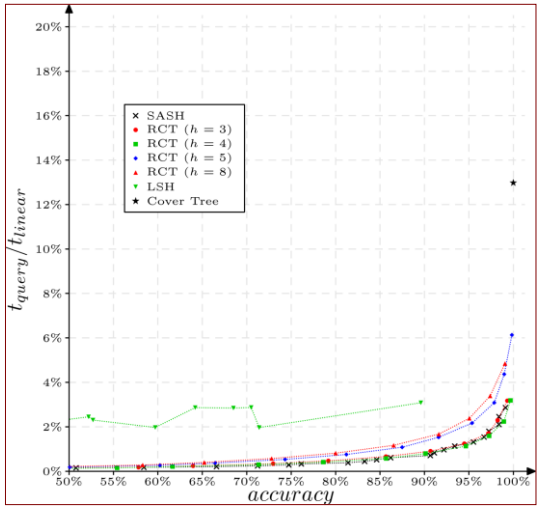


# Experimental Results

- 100-NN queries, averaged over 100 different query points & 10 builds.
- RCT and SASH: time versus accuracy plots.
- Cover Tree: exact time.
- LSH:
  - E2LSH tool – implementation performs range queries.
  - For *k*-NN queries, must expand range until the desired number of neighbors is obtained
  - In our experimentations, we give it the true *k*-NN distance.
  - Tremendous advantage over RCT, SASH, Cover Tree!

# Experimental Results

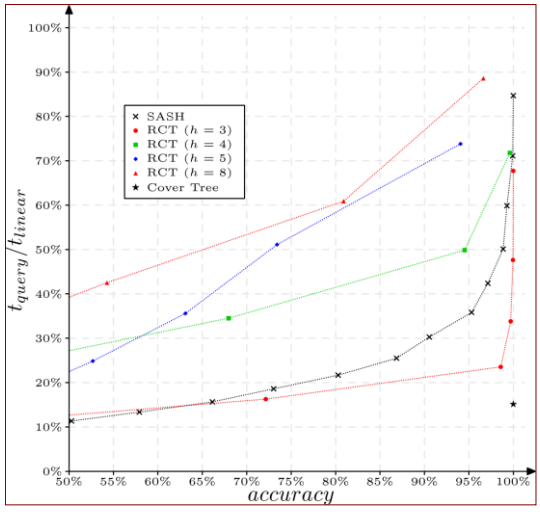
- ALOI (Amsterdam Library of Object Images)
- 110,250 images, 641 features (data prepared by INRIA-Rocquencourt).
- Average expansion dimension (up to *k*=200): 6.7





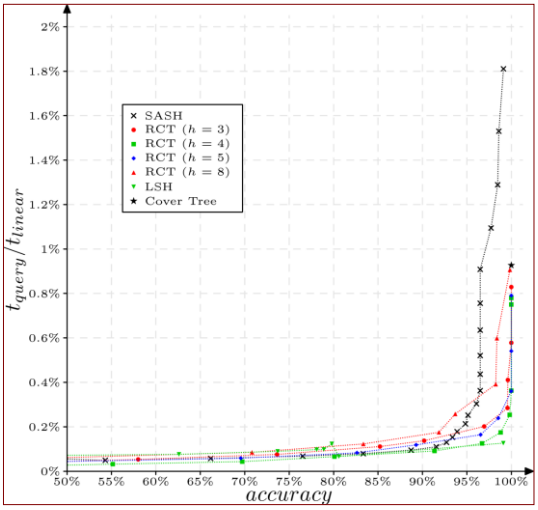

# Experimental Results

- Chess
- Database of 28,056 endgame positions (King + Rook vs. King).
- 6 features.



# Experimental Results

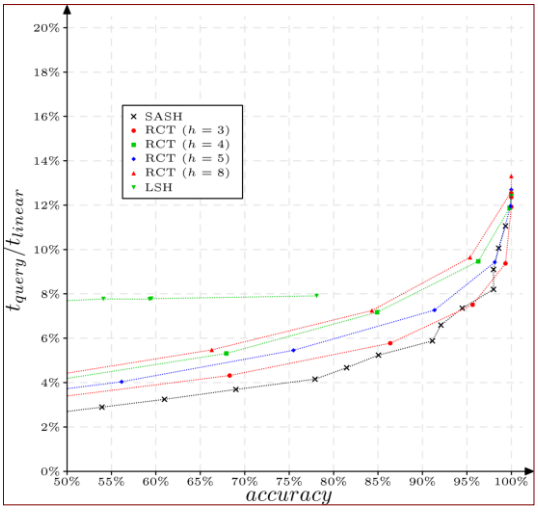
- Covertype
- Topographical information on forest cover.
- 580,012 forest cells of 900 sq. meters each.
- 54 attributes.





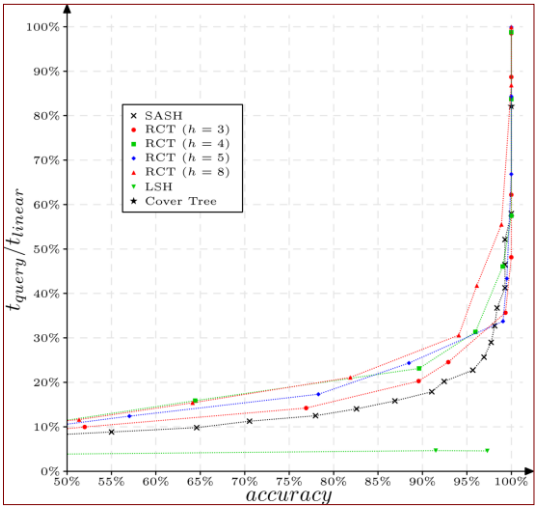

# Experimental Results

- Gisette
- 13,500 recordings of hand-written digits 4 and 9.
- 5,000 numerical attributes.



# Experimental Results

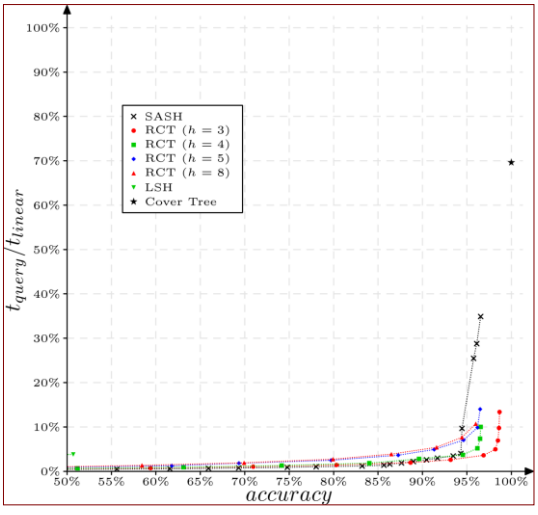
- Isolet
- 7797 recordings of spoken letters.
- 617 attributes (spectral coefficients, contour features, sonorant features, etc.)
- Average stereological dimension (up to $k$=200): 11.9





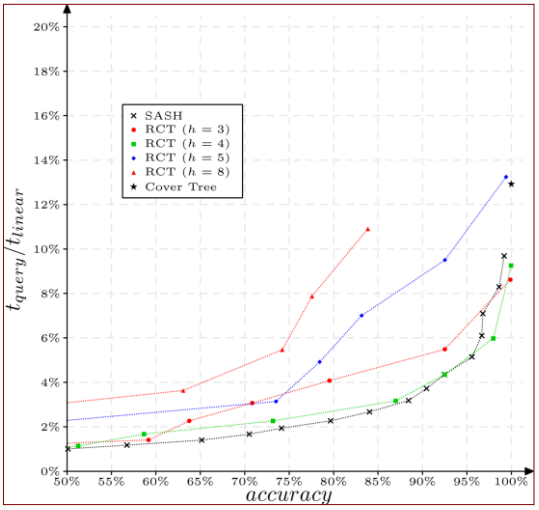

# Experimental Results

- MNIST
- Database of hand-written digits.
- 70,000 instances written by 500 individuals.
- 784 feature dimensions.



# Experimental Results

- Poker
- 1,025,100 hands of 5 cards.
- 10 attributes.

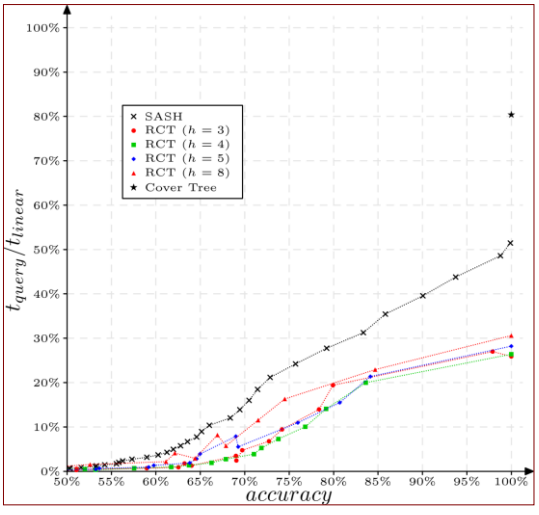



59



# Experimental Results

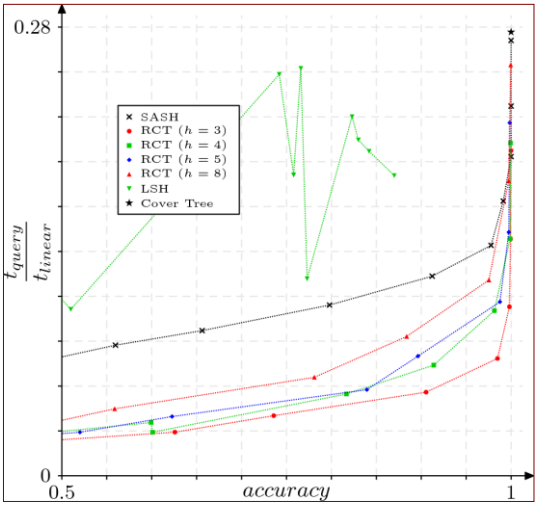
- Reuters
- 487,000 news articles (roughly half the data set).
- Approx. 300,000 keywords, no dimensional reduction.
- Bag-of-words vectors with TF-IDF weighting.
- Average stereological dimension (up to *k*=200): 21.6




60

# Experimental Results

- Spambase
- E-mail spam, frequency of certain words.
- 4601 messages.
- 57 attributes (keywords).




61



# Experimental Results

- Wikimedia Commons faces.
- 200,000 faces, 2580 feature dimensions.
- Vectorization performed by Toshiba.
- Sequential search performance is excellent.

?

62

# Experimental Results

- Wikimedia Commons faces.
- 200,000 faces, 2580 feature dimensions.
- Vectorization performed by Toshiba.
- Sequential search performance is excellent.
- All indices failed miserably!
- Average stereological dimension (up to $k$=200): 150.1