Kernelized Locality-Sensitive Hashing for Scalable Image Search

Brian Kulis and Kristen Grauman
LSH overview [Charikar ’02]

• Caveat: Many different types of LSH algorithm. This is not really the same as the standard [Indyk ’99] algorithm.

• In the end, no hash table is involved! Very close connections to a bunch of techniques known as min-hash.

• Task: $r$-similarity search (range search). Given a set of points, $X$ in $R^d$, and a query point, $q$, return all points, $P$, such that:

$$\forall p \in P, \quad sim(q, p) \geq r$$
LSH overview [Charikar ’02]

LSH can “hash” the points using a function, \( h : \mathbb{R}^d \rightarrow \mathbb{Z} \), such that:

\[
P[h(q) = h(p)] = sim(q, p)
\]

Charikar showed that for a similarity:

\[
sim(x, y) = 1 - \frac{1}{\pi} \cos^{-1}\left(\frac{x^T y}{||x|| ||y||}\right)
\]

The following \( h \) satisfies the hashing condition:

\[
h_r(x) = \text{sign}(x^T r)
\]

Where \( r \) is a random vector drawn iid from a Gaussian.
Using arccos expansion:

\[ \arccos(x) \approx \frac{\pi}{2} - x \]

And so (assuming \( \|x\| = \|y\| = 1 \)):

\[ \sim(x, y) \approx \frac{1}{2} + \frac{x^T y}{\pi} \]

This paper assumes that \( \sim(x, y) = \frac{1}{2} + \frac{1}{2}x^T y \) which isn’t quite right...
LSH overview [Charikar ’02]

Comparing actual similarity (blue) to desired (red)
LSH overview [Charikar ’02]

This probability is quite bad with lots of points. Choose the conjunction, $H$, of many different hash functions, so that:

$$P[H(q) = H(p)] = (\text{sim}(q, p))^p$$
• Instead of hashing the points, compute several random permutations of the hash bits and sort them in a big list.
• Search then becomes a binary search over several sorted lists.
• Search computes the distance to $O(n^{1/(1+\epsilon)})$ points to compute $(1 + \epsilon)$ near-neighbours.
• In practice a pretty poor bound – for say $\epsilon = 0.1$ we have to compute $O(n^{0.909})$ similarities.