Information Processing Letters 136 (2018) 49–58

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Why locality sensitive hashing works: A practical perspective

Kejing Lu [a], Hongya Wang [a,*], Yingyuan Xiao [b], Hui Song [a]

[a] School of Computer Science and Technology, Donghua University, Shanghai, China
[b] School of Computer Science and Engineering, Tianjin University of Technology, Tianjin, China

ARTICLE INFO

Article history:
Received 22 December 2016
Accepted 24 March 2018
Available online 5 April 2018
Communicated by Jinhui Xu

Keywords:
Locality sensitive hashing
Probability of collision
Recall rate
Analysis of variance
Randomized algorithms

ABSTRACT

Locality Sensitive Hashing (LSH) is one of the most efficient approaches to the nearest neighbor search problem in high dimensional spaces. A family $\mathcal{H}$ of hash functions is called locality sensitive if the collision probability $p_{\mathbf{h}}(r)$ of any two points $\langle q, p\rangle$ at distance r over a random hash function $\mathbf{h}$ decreases with r. The classic LSH algorithm employs a data structure consisting of $k * \ell$ randomly chosen hash functions to achieve more desirable collision curves and the collision probability $P_{\mathbf{h}_k^\ell}(r)$ for $\langle q, p\rangle$ is equal to $1-(1-p_{\mathbf{h}}(r)^k)^\ell$. The great success of LSH is usually attributed to the solid theoretical guarantee for $P_{\mathbf{h}_k^\ell}(r)$ and $p_{\mathbf{h}}(r)$.

In practice, however, users are more interested in *recall rate*, i.e., the probability that a random query collides with its r-near neighbor over a fixed LSH data structure h_k^ℓ. Implicitly or explicitly, $P_{\mathbf{h}_k^\ell}(r)$ is often misinterpreted as recall rate and used to predict the performance of LSH. This is problematic because $P_{\mathbf{h}_k^\ell}(r)$ is actually the expectation of recall rates. Interestingly, numerous empirical studies show that, for *most* (if not all) real datasets and a *fixed* sample of random LSH data structure, the recall rate is very close to $P_{\mathbf{h}_k^\ell}(r)$. In this paper, we provide a theoretical justification for this phenomenon. We show that (1) for random datasets the recall rate is asymptotically equal to $P_{\mathbf{h}_k^\ell}(r)$; (2) for arbitrary datasets the variance of the recall rate is very small as long as the parameter k and ℓ are properly chosen and the size of datasets is large enough. Our analysis (1) explains why the practical performance of LSH (the recall rate) matches so well with the theoretical expectation ($P_{\mathbf{h}_k^\ell}(r)$); and (2) indicates that, in addition to the nice theoretical guarantee, the mechanism by which LSH data structures are constructed and the huge amount of data are also the main causes for the success of LSH in practice.

© 2018 Elsevier B.V. All rights reserved.

1. Introduction

The nearest neighbor search (NNS) problem in high dimensional spaces is a fundamental problem that arises in numerous applications such as data mining, information retrieval and image search. While the exact NNS problem seems to suffer from the "curse of dimensionality", many efficient techniques have been devised for finding an approximate solution whose distance from the query point is at most c times the distance between the query and its nearest neighbor. One of the most versatile methods for approximate NNS is based on Locality Sensitive Hashing (LSH). Since introduced by Indyk and Motwani in [6], LSH has been widely applied in a variety of domains, including web clustering, computer vision and computational biology [1,8].

* Corresponding author.
E-mail address: hywang@dhu.edu.cn (H. Wang).

https://doi.org/10.1016/j.ipl.2018.03.018
0020-0190/© 2018 Elsevier B.V. All rights reserved.

The LSH approach to the approximate NNS problem is based on the existence of the locality sensitive hash functions, which will be discussed in details in Section 2. Given a family of LSH functions $\mathcal{H}$, the classic LSH method works as follows [6,4]. For parameters k and ℓ, ℓ functions $g_j(q) = (h_{1,j}(q), \ldots, h_{k,j}(q))$ are chosen, where $h_{i,j}$ $(1 \le i \le k,\ 1 \le j \le \ell)$ is drawn independently and uniformly at random from $\mathcal{H}$ [6]. Given a dataset $\mathcal{D}$, $\forall o \in \mathcal{D}$ the bucket $g_j(o)$ is computed for $j = 1, \ldots, \ell$, and then o is inserted into the corresponding bucket. To process a query q, one has to compute $g_j(q)$, $j = 1, \ldots, \ell$, first, and then retrieve all points that lie in at least one of these buckets. For our purpose, we denote $\mathbf{h}$ and $\mathbf{h}_k^\ell$ as a random LSH function and a random LSH data structure respectively, and the (uniformly drawn) samples of $\mathbf{h}_k^\ell$ and $\mathbf{h}$ are denoted by h_k^ℓ and h, respectively.

LSH and its variants are capable of providing, with some constant success probability, excellent asymptotic performance in terms of space consumption and query cost [6,15,11,5]. The theoretical guarantee relies on the fact that, for any point pair $\langle q, o\rangle$ such that $d(q, o) = r$,[1] the collision probability $p_{\mathbf{h}}(r)$ that $\langle q, o\rangle$ collides over $\mathbf{h}$ decreases monotonically with r [6]. As a quick result, the collision probability of $\langle q, o\rangle$ over $\mathbf{h}_k^\ell$, denoted by $P_{\mathbf{h}_k^\ell}(r)$, can be calculated using Equation (1) since $k * \ell$ hash functions are drawn independently from $\mathcal{H}$.

$$P_{\mathbf{h}_k^\ell}(r) = 1 - (1 - p_{\mathbf{h}}(r)^k)^\ell \tag{1}$$

In real-life applications, however, practitioners are more interested in *recall rate* [13,10,11], which is the probability that a random query q collides with its r-near neighbor o[2] over a concrete LSH data structure. Formally, for a given dataset $\mathcal{D}$ and a sample h_k^ℓ of the random LSH data structure, recall rate $P_{h_k^\ell}(\mathcal{D}_r)$ is defined as $\frac{|\{\langle q,o\rangle \in \mathcal{O}_r | h_k^\ell(q) = h_k^\ell(o)\}|}{|\mathcal{D}_r|}$, where $\mathcal{D}_r = \{\langle q, o\rangle | q, o \in \mathcal{D}, d(q, o) = r\}$.[3] While $P_{\mathbf{h}_k^\ell}(r)$ and $P_{h_k^\ell}(\mathcal{D}_r)$ are conceptually different, practitioners often use $P_{\mathbf{h}_k^\ell}(r)$ to predict the performance of LSH and the consistency between $P_{h_k^\ell}(\mathcal{D}_r)$ and $P_{\mathbf{h}_k^\ell}(r)$ is deemed as an indicator for the effectiveness of the proposed algorithms. Put it another way, $P_{h_k^\ell}(\mathcal{D}_r)$ is implicitly assumed to be (almost) equal to $P_{\mathbf{h}_k^\ell}(r)$ in practice.

$$P_{\mathbf{h}_k^\ell}(r) = \mathbb{E}_{\mathbf{h}_k^\ell}[P_{\mathbf{h}_k^\ell}(\mathcal{D}_r)] \qquad (p_{\mathbf{h}}(r) = \mathbb{E}_{\mathbf{h}}[p_{\mathbf{h}}(\mathcal{D}_r)]) \tag{2}$$

Such an interpretation of $P_{\mathbf{h}_k^\ell}(r)$ is problematic because, as shown in Equation (2), $P_{\mathbf{h}_k^\ell}(r)$ and $p_{\mathbf{h}}(r)$ are actually the expectations of $P_{\mathbf{h}_k^\ell}(\mathcal{D}_r)$ and $p_{\mathbf{h}}(\mathcal{D}_r)$, respectively [12, 16]. For random variable X, using $\mathbb{E}[X]$ to predict a single sample of X usually leads to inaccurate estimation. Interestingly enough, numerous empirical studies show that, for most (if not all) datasets and a *fixed* sample of random LSH data structure, $P_{h_k^\ell}(\mathcal{D}_r)$ is very close to $P_{\mathbf{h}_k^\ell}(r)$ [16,5,11,15, 14]. We conjecture this may be the main reason for the misinterpretation of $P_{\mathbf{h}_k^\ell}(r)$. In this paper, we aim to provide a theoretical justification for this phenomenon.

A natural approach to this problem is to analyze the variance of $P_{\mathbf{h}_k^\ell}(\mathcal{D}_r)$ and/or $p_{\mathbf{h}}(\mathcal{D}_r)$. Unfortunately, obtaining general close-formed expressions for them is intractable due to their data-dependent nature. As a workaround, we first consider the case of random datasets and derive $p_h(\mathcal{D}_r)$ analytically for random inputs. It is shown that, for the Euclidean space, noticeable $\mathbb{D}[p_{\mathbf{h}}(\mathcal{D}_r)]$ exists because $p_h(\mathcal{D}_r)$ is a function of the length of random vectors. To reduce $\mathbb{D}[p_{\mathbf{h}}(\mathcal{D}_r)]$, we propose an enhanced LSH family in which the random vectors are all normalized to length $\sqrt{n}$ first.[4] Not only owns the same collision probability as the existing LSH family, the proposed LSH family also provides better performance in terms of $\mathbb{D}[p_{\mathbf{h}}(\mathcal{D}_r)]$.

For arbitrary datasets, deriving $p_h(\mathcal{D}_r)$ analytically is nearly impossible. Therefore, we switch our attention to discuss the relation between $\mathbb{D}[P_{\mathbf{h}_k^\ell}(\mathcal{D}_r)]$ and $\mathbb{D}[p_{\mathbf{h}}(\mathcal{D}_r)]$. We show that the variance of $P_{\mathbf{h}_k^\ell}(\mathcal{D}_r)$ is upperbounded by a function of k and $\mathbb{D}[p_{\mathbf{h}}(\mathcal{D}_r)]$. More importantly, numerical calculation indicates that, for large datasets, $\mathbb{D}[P_{\mathbf{h}_k^\ell}(\mathcal{D}_r)]$ is two to three order of magnitude less than $\mathbb{D}[p_{\mathbf{h}}(\mathcal{D}_r)]$ for a variety of k and ℓ values adopted in real-life applications. As a result, practical $\mathbb{D}[P_{\mathbf{h}_k^\ell}(\mathcal{D}_r)]$ tends to be very small as long as the parameters k, ℓ are chosen properly and the size of datasets is large enough, even if $\mathbb{D}[p_{\mathbf{h}}(\mathcal{D}_r)]$ is moderately large. Thus, by Chebyshev inequality the difference between $P_{\mathbf{h}_k^\ell}(r)$ and $P_{\mathbf{h}_k^\ell}(\mathcal{D}_r)$ is bounded by a small constant with high probability.

The results in this paper explain why the practical performance of LSH (recall rate) matches so well with the theoretical expectation ($P_{\mathbf{h}_k^\ell}(r)$). Moreover, our analysis reveals that, in addition to the nice theoretical guarantee, other factors such as the mechanism by which LSH data structures are constructed, proper parameter settings and the huge amount of data are also the main causes for the great success of LSH.

The rest of this article is organized as follows. A brief overview of LSH is provided in Section 2. The derivation of $p_h\mathcal{D}_r)$ and the discussion on its relation with $p_{\mathbf{h}}(r)$ for l_2 distance are presented in Section 3. Section 4 discusses how $\mathbb{D}[P_{\mathbf{h}_k^\ell}(\mathcal{D}_r)]$ varies with $\mathbb{D}[p_{\mathbf{h}}(\mathcal{D}_r)]$ and we conclude this paper with a summary of our results in Section 5.

2. Preliminaries

To solve the r-near neighbor search problem, Indyk and Motwani introduced the concept of Locality Sensitive Hashing in their influential paper [6]. The idea of random projection, however, can be traced back to much earlier work in [9,2]. The rationale behind LSH is that, by using specific hashing functions, we can hash the points such that the probability of collision for data points which are close to each other is much higher than that for those which are far apart. In the rest of this paper, we use $\mathcal{H}$

[1] $d(q, o)$ is the distance between q and o.
[2] The points that are at distance r from q.
[3] $p_h(\mathcal{D}_r)$ can be defined in a similar vein.
[4] n is the dimension of the Euclidean space.

to denote a family of hash functions mapping $\Re^n$ to some universe $\mathcal{U}$. For any two points o and q and a hash function h that is chosen from $\mathcal{H}$ uniformly at random, the family $\mathcal{H}$ is called locality sensitive if the probability that these two points collide ($h(q) = h(o)$) satisfies the following condition. For an LSH family to be useful, it has to satisfy $p^1 > p^2$.

Definition 1 *([6]).* A family $\mathcal{H}$ of hash functions is called (r, cr, p^1, p^2)-sensitive if for any two points o, $q \in \Re^n$:

- if $\|q - o\| \leq r$ then $Pr_{\mathcal{H}}[h(q) = h(o)] \geq p^1$
- if $\|q - o\| \geq cr$ then $Pr_{\mathcal{H}}[h(q) = h(o)] \leq p^2$

In this paper, we focus on the LSH family designed the Euclidean space and results for Hamming distance and Arccos similarity are omitted due to space limitation.

l_2 distance. For the Euclidean space, [4] proposes the following LSH family. Pick a random vector $\vec{a}$ in $\Re^n$ and project o onto $\vec{a}$. The 1-dimensional line ($\vec{a}$) is then chopped into segments of length w. The number of segment which o falls into, after shifted by a random value $b \in [0, w)$, is the hash value of o. Formally, $h_{a,b}(o) = \left\lfloor \frac{\vec{a}\cdot\vec{o}+b}{w} \right\rfloor$. Each coordinate of $\vec{a}$ is drawn uniformly at random following the Gaussian distribution. The probability of collision that any two points at distance r collides over the random vector $\vec{a}$ is as follows

$$p^e_{\mathbf{h}}(r) = \int_0^w \frac{1}{r} g(\frac{t}{r})(1 - \frac{t}{w})dt \tag{3}$$

where $g(x) = 2f(x)$ and $f(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$.

3. The analysis of $p_h(\mathcal{D}_r)$ for l_2 distance

As discussed in Section 1, $p_{\mathbf{h}}(r)$ is only the expectation of recall rate $p_h(\mathcal{D}_r)$, which is of more interest for users. In this section, we first derive the probability of collision that a random pair of points collides over any single hash function, i.e., $p_h(\mathcal{D}_r)$, in the case of random datasets. The relation between $p_h(\mathcal{D}_r)$ and $p_{\mathbf{h}}(r)$ is discussed. As a byproduct, an enhanced LSH family $\mathcal{H}'$ is proposed for the Euclidean space, which can offer smaller $\mathbb{D}[p_{\mathbf{h}'}(\mathcal{D}_r)]$ compared with the original LSH family.

3.1. Collision probability

In this section, we derive the probability of collision, denoted by $p^e_h(\mathcal{D}_r)$, for any single hash function on random input under l_2 distance. To do this, we first need the following Lemma.

Suppose $\vec{v}$ and $\vec{w}$ are two vectors drawn uniformly at random from an n-dimensional unit hypersphere, the probability density function $p^{(n)}(\theta)$ of the angle θ between $\vec{v}$ and $\vec{w}$ is shown in Lemma 1, where $\Gamma(n)$ is the Gamma function of n.

Lemma 1. $p^{(n)}(\theta) = \frac{\Gamma(n/2)}{\Gamma((n-1)/2)} \cdot \frac{\sin^{n-2}(\theta)}{\sqrt{\pi}}$

Proof. The volume of an n-dimensional Euclidean ball of radius r is

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

The surface area of the $n-1$-dimensional sphere of radius r in $\Re^n$ is

$$S_{n-1}(r) = \frac{dV_n(r)}{dr} = \frac{2\pi^{n/2}}{\Gamma(n/2)} r^{n-1}$$

Then we have

$$p^{(n)}(\theta)d\theta = \frac{dS_{n-1}(r)}{S_{n-1}(r)} = \frac{S_{n-2}(r\sin\theta)rd\theta}{S_{n-1}(r)}$$

which leads us to

$$p^{(n)}(\theta) = \frac{\Gamma(n/2)}{\Gamma((n-1)/2)} \cdot \frac{\sin^{n-2}(\theta)}{\sqrt{\pi}} \quad \square \tag{4}$$

Recall that the LSH function for l_2 distance is $h_{a,b}(o) = \left\lfloor \frac{\vec{a}\cdot\vec{o}+b}{w} \right\rfloor$. A geometric interpretation of $h_{a,b}$ is illustrated in Fig. 1(a) for the case of $n = 2$. The dot product $\vec{a} \cdot \vec{o_1}$ is the projection of point o_1 onto line $\vec{a}$, namely, point A in the figure. In a similar way o_2 is projected to point B on line $\vec{a}$. The effect of $\vec{a} \cdot \vec{o_1} + b$ is to shift A by a distance b (along the line) to point o'_1. Finally, we partition the line into segments of length w and the hash value $h(o)$ is the ID of the segment covering o'_1. As shown in Fig. 1, o_1 and o_2 collide over hash function $h_{a,b}$ since o'_1 and o'_2 are located in the same segment.

While deriving $p^e_{\mathbf{h}}(r)$, $\vec{a}$ is viewed as an n-dimensional random vector and the pair of data points are fixed. In our model, however, we regard $\vec{a}$ as a fixed vector as illustrated in Fig. 1(b) and Fig. 1(c). For ease of presentation, we assume $\|\vec{a}\| = 1$ for now and will consider the general case later.

To model a pair of random data points (o_1, o_2) with distance r, one can think o_1 as a random data point and o_2 as the point randomly chosen from the hypersphere of radius r centering around o_1, which is illustrated in Fig. 1(b) and Fig. 1(c). Since the shift of the hypersphere along the direction perpendicular to $\vec{a}$ will not affect the collision probability, we only need to consider the movement of the hypersphere along the direction in parallel with $\vec{a}$, which is equivalent to the random choice of o_1.

Assume the range of space is unlimited, then $p^e_h(\mathcal{D}_r)$ is invariable for all segments and the random shift b doesn't affect $p^e_h(\mathcal{D}_r)$ as well. Thus, we only need to focus on a specific segment, say AC, as illustrated in Fig. 1(b) and Fig. 1(c). In this figure, AB and CD are two hyperplanes going through A and C respectively and perpendicular to $\vec{a}$. The probability of o_1 and o_2 falling into segment AC is equal to the ratio of the surface area bounded by hyperplane AB and CD to the surface area of the whole hypersphere. This ratio varies as the hypersphere moves along KL, which is in parallel with $\vec{a}$. Obviously, the ratio is a function of x, the distance between o_1 and K. By summing up all the ratios obtained during the movement of o_1 from point K to L and dividing it by the length of the segment,

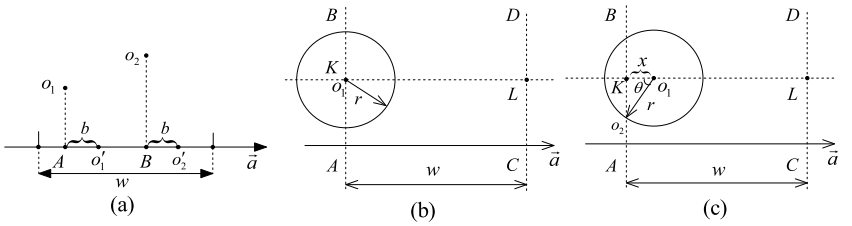

Fig. 1. A geometric interpretation of $h_{a,b}(o)$ and Case 1 for deriving $p_h^e(\mathcal{D}_r)$.

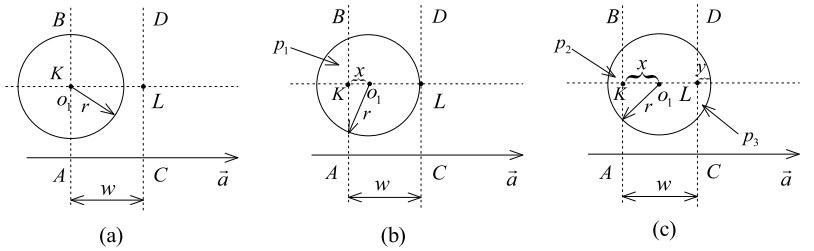

Fig. 2. Case 2 for deriving $p_h^e(\mathcal{D}_r)$.

we will get the expected probability of collision, that is, $p_h^e(\mathcal{D}_r)$.

Actually, we only need to move o_1, the center of the hypersphere, to the midpoint of KL by symmetry. As calculating the surface area outside the space bounded by the two hyperplanes is easier, we first compute the probability that two random points do not collide, and then subtract it from 1 to get $p_h^e(\mathcal{D}_r)$. There are three cases to consider according to the relation between r and w.

Case 1: $2r < w$ (Fig. 1(b) and (c))

According to the discussion above, Equation (5) gives the expected collision probability in this case. $p^{(n)}(\theta)$ is used to compute the fraction of the surface area of the hypersphere falling outside the space between the two hyperplanes.

$$p_h^e(\mathcal{D}_r) = 1 - \frac{\int_0^r \int_0^{\arccos \frac{x}{r}} p^{(n)}(\theta) d\theta dx}{w/2} \tag{5}$$

Case 2: $\frac{r}{2} < w$ (Fig. 2)

In this case, computing $p_h^e(\mathcal{D}_r)$ is somewhat complicated because, as illustrated in Fig. 2(c), the right part of the hypersphere will intersect with hyperplane CD during the movement of o_1. Equation (6) gives $p_h^e(\mathcal{D}_r)$ in this case

$$p_h^e(\mathcal{D}_r) = 1 - \frac{(p_1 + p_2 + p_3)}{w/2} \tag{6}$$

where

$$p_1 = \int_0^{w-r} \int_0^{\arccos \frac{x}{r}} p^{(n)}(\theta) d\theta dx,$$

$$p_2 = \int_{w-r}^{w/2} \int_0^{\arccos \frac{x}{r}} p^{(n)}(\theta) d\theta dx$$

and

$$p_3 = \int_0^{r-w/2} \int_0^{\arccos \frac{r-y}{r}} p^{(n)}(\theta) d\theta dy$$

Please note that p_3 is derived by considering the hypersphere falling on the right side of hyperplane CD. Thus, by symmetry we have

$$p_3 = \int_{w/2}^{r} \int_0^{\arccos \frac{x}{r}} p^{(n)}(\theta) d\theta dx$$

By substituting p_1, p_2 and p_3 into Equation (6), we can see that $p_h^e(\mathcal{D}_r)$ in Case 2 is the same as the one presented in Equation (5).

Case 3: $r \geq w$ (Fig. 3)

As illustrated in Fig. 3, p_4 and p_5 are obtained by considering the cap located on the left side of AB and the right side of CD, respectively. Thus, $p_h^e(\mathcal{D}_r)$ can be calculated using Equation (7)

$$p_h^e(\mathcal{D}_r) = 1 - \frac{p_4 + p_5}{w/2} \tag{7}$$

where

$$p_4 = \int_0^{w/2} \int_0^{\arccos \frac{x}{r}} p^{(n)}(\theta) d\theta dx$$

and

$$p_5 = \int_0^{r-w} \int_0^{\arccos \frac{r-y}{r}} p^{(n)}(\theta) d\theta dy$$

Similarly, by symmetry we have

$$p_5 = \int_{w/2}^{w} \int_0^{\arccos \frac{x}{r}} p^{(n)}(\theta) d\theta dx$$

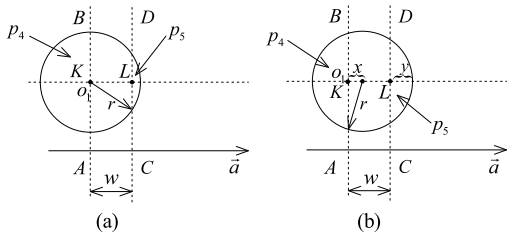

Fig. 3. Case 3 for deriving $p_h^e(\mathcal{D}_r)$.

By substituting p_4 and p_5 into Equation (7), we have $p_h^e(\mathcal{D}_r)$ in this case.

$$p_h^e(\mathcal{D}_r) = 1 - \frac{\int_0^w \int_0^{\arccos\frac{x}{r}} p^{(n)}(\theta)d\theta dx}{w/2} \tag{8}$$

3.2. Relation between $p_{\mathbf{h}}^e(r)$ and $p_h^e(\mathcal{D}_r)$

While $p_{\mathbf{h}}^e(r)$ and $p_h^e(\mathcal{D}_r)$ look different, we will show that there is intrinsic connection between them actually.

In the previous section we assume $\|\vec{a}\| = 1$. By substituting r with $\|\vec{a}\| r$ into Equation (5) and Equation (8), we have the generalized random-input-based collision probability for l_2 distance as follows.

$$p_h^e(\mathcal{D}_r) = \begin{cases} 1 - \frac{2}{w} \cdot \int_0^{\|\vec{a}\|r} \int_0^{\arccos\frac{x}{\|\vec{a}\|r}} p^{(n)}(\theta)d\theta dx & w > \|\vec{a}\| r \\ 1 - \frac{2}{w} \cdot \int_0^{w} \int_0^{\arccos\frac{x}{\|\vec{a}\|r}} p^{(n)}(\theta)d\theta dx & w \le \|\vec{a}\| r \end{cases} \tag{9}$$

By substituting $\|\vec{a}\|$ with $\sqrt{n}$ into Equation (9) we have

$$p_h^e(\mathcal{D}_r) = \begin{cases} 1 - \frac{2}{w} \cdot \int_0^{\sqrt{n}r} \int_0^{\arccos\frac{x}{\sqrt{n}r}} p^{(n)}(\theta)d\theta dx & w > \sqrt{n}r \\ 1 - \frac{2}{w} \cdot \int_0^{w} \int_0^{\arccos\frac{x}{\sqrt{n}r}} p^{(n)}(\theta)d\theta dx & w \le \sqrt{n}r \end{cases} \tag{10}$$

For $p_h^e(\mathcal{D}_r)$ shown in Equation (10), we have the following Theorem.

Theorem 1. $\lim_{n\to\infty} p_h^e(\mathcal{D}_r) = p_{\mathbf{h}}^e(r)$

Proof. $p_h^e(\mathcal{D}_r)$ is a piece-wise function, thus we need to consider two cases as follows.

Case 1: $\sqrt{n}r \ge w$

Recall that $p_{\mathbf{h}}^e(r) = \int_0^w \frac{1}{r} g(\frac{t}{r})(1 - \frac{t}{w})dt$ where $g(x)$ denotes the probability density function of the absolute value of the normal distribution. Obviously, $\int_0^w \int_0^{\frac{\pi}{2}} p^{(n)}(\theta)d\theta dx = w/2$ per definition. Thus, to prove the theorem we only need to show that

$$\lim_{n\to\infty} \int_0^w \int_{\arccos\frac{x}{\sqrt{n}r}}^{\frac{\pi}{2}} p^{(n)}(\theta)d\theta dx = \int_0^w \frac{1}{r} f(\frac{t}{r})(w-t)dt \tag{11}$$

where $f(x)$ is the probability density function of the normal distribution.

For fixed radius r, the probability of collision can be viewed as a function of w. Let $a(w)$ and $a_0(w)$ be equal to the expressions on the left and right side of Equation (11), respectively

$$a(w) = \int_0^w \int_{\arccos\frac{x}{\sqrt{n}r}}^{\frac{\pi}{2}} p^{(n)}(\theta)d\theta dx$$

and

$$a_0(w) = \int_0^w \frac{1}{r} f(\frac{t}{r})(w-t)dt$$

For continuous functions $a(w)$ and $a_0(w)$, it is easy to see that $a(w) = a_0(w)$ if $a(w_0) = a_0(w_0)$ for some constant w_0 and $a'(w) = a_0'(w)$. Given the fact that $a(0) = 0$ and $a_0(0) = 0$, to prove $a(w) = a_0(w)$ we only need to show

$$\lim_{n\to\infty} \int_{\arccos\frac{w}{\sqrt{n}r}}^{\frac{\pi}{2}} p^{(n)}(\theta)d\theta = \int_0^w \frac{1}{r} f(\frac{t}{r})dt \tag{12}$$

Please note that $a'(w) = \int_{\arccos\frac{w}{\sqrt{n}r}}^{\frac{\pi}{2}} p^{(n)}(\theta)d\theta$ and $a_0'(w) = \int_0^w \frac{1}{r} f(\frac{t}{r})dt$

In a similar vein, given the fact that $a'(0) = 0$ and $a_0'(0) = 0$, to prove $a'(w) = a_0'(w)$ we only need to show

$$\lim_{n\to\infty} \frac{1}{\sqrt{1 - \frac{w^2}{nr^2}}} \times \frac{1}{\sqrt{n}r} \times p^{(n)}(\arccos\frac{w}{\sqrt{n}r}) = \frac{1}{r} f(\frac{w}{r}) \tag{13}$$

where $a''(w) = \frac{1}{\sqrt{1-\frac{w^2}{nr^2}}} \times \frac{1}{\sqrt{n}r} \times p^{(n)}(\arccos\frac{w}{\sqrt{n}r})$ and $a_0''(w) = \frac{1}{r} f(\frac{w}{r})$.

Clearly, $\lim_{n\to\infty} \frac{1}{\sqrt{1-\frac{w^2}{nr^2}}} = 1$, thus we only need to prove the following equation by substituting $p^{(n)}(\theta)$ and $f(x)$ into Equation (13).

$$\lim_{n\to\infty} \frac{\Gamma(\frac{n}{2})}{\Gamma(\frac{n-1}{2})} \times \frac{1}{\sqrt{n}} \times (1 - \frac{w^2}{nr^2})^{\frac{n-2}{2}} = \frac{1}{\sqrt{2}} e^{-\frac{w^2}{2r^2}} \tag{14}$$

It is easy to see that Equation (14) holds given the facts that $\lim_{n\to\infty} (1 - \frac{1}{n})^n = \frac{1}{e}$ and $\lim_{n\to\infty} \frac{\Gamma(n+\alpha)}{\Gamma(n)n^\alpha} = 1, \alpha \in R$. Therefore, we proved this theorem for Case 1.

Case 2: $\sqrt{n}r < w$

Similar to Case 1, we view the collision probability as a function of w and let

$$b(w) = w/2 - \int_0^{\sqrt{n}r} \int_0^{\arccos\frac{x}{\sqrt{n}r}} p^{(n)}(\theta)d\theta dx \tag{15}$$

Table 1
The impact of the length of random vectors on the probability of collision.

	$r=0.4$	$r=0.5$	$r=0.6$	$r=0.7$	$r=0.8$	$r=0.9$	$r=1.0$
$p_h^e(\mathcal{D}_r)$ ($\|\vec{a}\|=\sqrt{n}/4$)	0.9202	0.9002	0.8803	0.8603	0.8404	0.8204	0.8005
Experimental Collision Prob.	0.9243	0.9056	0.8869	0.8687	0.8510	0.8335	0.8169
$p_h^e(\mathcal{D}_r)$ ($\|\vec{a}\|=\sqrt{n}/2$)	0.8404	0.8005	0.7606	0.7211	0.6823	0.6449	0.6093
Experimental Collision Prob.	0.8341	0.7948	0.7586	0.7220	0.6854	0.6491	0.6135
$p_h^e(\mathcal{D}_r)$ ($\|\vec{a}\|=2\sqrt{n}$)	0.4424	0.3685	0.3142	0.2732	0.2413	0.2159	0.1952
Experimental Collision Prob.	0.4409	0.3671	0.3135	0.2731	0.2418	0.2169	0.1964
$p_h^e(\mathcal{D}_r)$ ($\|\vec{a}\|=4\sqrt{n}$)	0.2413	0.1952	0.1637	0.1409	0.1236	0.1100	0.0991
Experimental Collision Prob.	0.2414	0.1931	0.1629	0.1405	0.1241	0.1110	0.1003

To prove the theorem in this case, we only need to show $b(w)=a_0(w)$. Since $b(w)$ and $a_0(w)$ are both continuous functions of w, we can safely replace the constraint $\sqrt{n}r<w$ with $\sqrt{n}r\le w$ without affecting the correctness of our analysis.

In Case 1 we have shown that $b(w)=a_0(w)$ when $\sqrt{n}r=w$. Next, we only need to show $b'(w)=a_0'(w)$. Since $b'(w)=1/2$, what we really have to prove is

$$\lim_{n\to\infty}\int_0^w \frac{1}{r}f(\frac{t}{r})dt=1/2 \tag{16}$$

By substitution we have $\int_0^w \frac{1}{r}f(\frac{t}{r})dt=\int_0^{\frac{w}{r}} f(t)dt$. As n goes to infinity, w/r approaches infinity as well under the constraint $w/r\ge\sqrt{n}$ for fixed r. Based on the fact that $\int_0^{+\infty} f(t)dt=1/2$, we prove the theorem. □

3.3. An enhanced LSH family for l_2 distance

$p_h^e(\mathcal{D}_r)$ is, in addition to w and r, a function of dimensionality n and the length of random vector $\|\vec{a}\|$. Actually, from Equation (9) one can see that the length of random vectors has significant impact on the probability of collision. To verify this observation, we calculated $p_h^e(\mathcal{D}_r)$ using Equation (9) for different vector lengths. The lengths of $\|\vec{a}\|$ are set to 0.25, 0.5, 2 and 4 times as much as $\sqrt{784}$.[5] Then, we conduct an experimental study with Minst dataset following the methodology in [16]. Minst data set contains 10K 784-dimensional data points, each of which representing a picture with 28*28 pixels.[6] Interested readers are referred to [16] for more details about the experimental study. Both the analytical and empirical results are listed in Table 1. As we can see in Table 1, $p_h^e(\mathcal{D}_r)$ do vary, both analytically and empirically, with the length of hash functions. Please note that here $p_h^e(\mathcal{D}_r)$ is averaged over 100 randomly chosen hash functions.

Recall that $\vec{a}$ is an n-dimensional vector and each component of $\vec{a}$ is drawn uniformly at random following the normal distribution. It is well known that $\|\vec{a}\|^2=|a_1|^2+|a_2|^2+\cdots+|a_n|^2$ follows χ^2 distribution with degree of freedom n. As a result, it is likely for users to select a "bad" hash functions in the sense that $p_h^e(\mathcal{D}_r)$ deviates drastically from $p_{\mathbf{h}}^e(r)$. This motivates us to propose an enhanced LSH family $\mathcal{H}'$, where the random vector $\vec{a}$ is normalized to length $\sqrt{n}$ first. Please note that $\|\vec{a}\|$ and r are symmetric in Eqn. (9), which suggests that we can view the fixed hash function as a pair of data points and the random pairs of data points as a random vectors on the hypersphere. Thus, by Theorem 1, we have $p_{\mathbf{h}'}^e(r)=p_{\mathbf{h}}^e(r)$ and $\mathbb{D}[p_{\mathbf{h}'}^e(\mathcal{D}_r)]=0$. $\mathcal{H}'$ may be of practical interest since $\mathbb{D}[p_{\mathbf{h}'}^e(\mathcal{D}_r)]$ is expected to be smaller than $\mathbb{D}[p_{\mathbf{h}}^e(\mathcal{O}_r)]$ for real datasets.

4. The relation between $\mathbb{D}[P_{\mathbf{h}_k^\ell}(\mathcal{D}_r)]$ and $\mathbb{D}[p_{\mathbf{h}}(\mathcal{D}_r)]$ for arbitrary datasets

In this section, we examine the variance of $P_{\mathbf{h}_k^\ell}(\mathcal{D}_r)$ and discuss its relation with $\mathbb{D}[p_{\mathbf{h}}(\mathcal{D}_r)]$ for datasets with arbitrary distributions. Please note the results presented here are also applicable for random datasets.

The analysis of $\mathbb{D}[P_{\mathbf{h}_k^\ell}(\mathcal{D}_r)]$ will be conducted in a two-step way by considering two kinds of randomness, that is, the hash functions and the data distributions. First, we define a random variable (with respect to $\mathbf{h}$) $\mathcal{F}_{\mathbf{h}_k^\ell}(\mathcal{D}_r)=1-\prod_{j=1}^{\ell}(1-\prod_{i=1}^{k}p_{h_{i,j}}(\mathcal{D}_r))$ and show that its variance is upperbounded by a function of k and $\mathbb{D}[p_{\mathbf{h}}(\mathcal{D}_r)]$. Numerical calculation reveals that $\mathbb{D}[\mathcal{F}_{\mathbf{h}_k^\ell}(\mathcal{D}_r)]$ is two to three order of magnitude less than $\mathbb{D}[p_{\mathbf{h}}(\mathcal{D}_r)]$ in a variety of application scenarios. Then we show, by reasonable approximation, that $\mathbb{D}[\mathcal{F}_{\mathbf{h}_k^\ell}(\mathcal{D}_r)]$ is very close to $\mathbb{D}[P_{\mathbf{h}_k^\ell}(\mathcal{D}_r)]$ with high probability as long as the size of datasets is large enough, which finally justify our claim that the difference between $P_{\mathbf{h}_k^\ell}(\mathcal{D}_r)$ and $P_{\mathbf{h}}(r)$ is bounded by a small constant with high probability.

4.1. Relation between $\mathbb{D}[\mathcal{F}_{\mathbf{h}_k^\ell}(\mathcal{D}_r)]$ and $\mathbb{D}[p_{\mathbf{h}}(\mathcal{D}_r)]$

In practice, users often specify an expected recall rate α and a search radius r first, and then determine appropriate k and ℓ values according to Equation (1). To simplify notations, let $X=p_{\mathbf{h}}(\mathcal{D}_r)$, $Y=\mathcal{F}_{\mathbf{h}_k^\ell}(\mathcal{D}_r)$ and thus $Y=1-\prod_{j=1}^{\ell}(1-\prod_{i=1}^{k}X_{i,j})$. The following theorem gives an upper bound of Y.

Theorem 2. $\mathbb{D}(Y)\le-\ln(1-\alpha)\cdot\gamma^k$ *where* $\gamma=\frac{\mathbb{D}(X^2)-\mathbb{E}^2(X)}{\mathbb{E}(X)}$.

[5] 784 is the dimensionality of Minst dataset.
[6] http://yann.lecun.com/exdb/minist/.

Proof. Since $X_{i,j}$ are independent random variables, by the properties of expectation we have

$$\mathbb{E}(Y^2) = \mathbb{E}[1 - 2 \cdot \mathbb{E}(X)^k + (\mathbb{D}(X) + \mathbb{E}(X)^2)^k]^\ell + 1 - 2 \cdot (1 - \mathbb{E}(X)^k)^\ell$$

$$\mathbb{E}^2(Y) = [1 - 2 \cdot \mathbb{E}(X)^k + \mathbb{E}(X)^{2k}]^\ell + 1 - 2 \cdot (1 - \mathbb{E}(X)^k)^\ell$$

For ease of presentation, let $x = \mathbb{D}(X)$, $y = \mathbb{E}(X)^2$, $z = 1 - 2 \cdot \mathbb{E}(X)^k$. Then we have

$$\mathbb{D}(Y) = \mathbb{E}(Y^2) - \mathbb{E}(Y)^2 = [(x+y)^k + z]^\ell - [y^k + z]^\ell \tag{17}$$

For any real number a, b and positive integer m, if $a \geq b > 0$, the following inequality holds:

$$(a-b)^m = (a-b)(a^{m-1} + a^{m-2}b + \cdots b^{m-1}) \leq (a-b) \cdot a^{m-1} \cdot m \tag{18}$$

By Equation (17) and (18), it follows

$$\mathbb{D}(Y) \leq (x+y)^k \cdot [(x+y)^k + z]^{\ell-1} \cdot \ell \tag{19}$$

Since $0 < X < 1$, it's clear that $\mathbb{E}(X^2) < \mathbb{E}(X)$. Then we have

$$\begin{aligned}[(x+y)^k + z] &= [\mathbb{E}(X)^2 + \mathbb{D}(X)]^k + 1 - 2\mathbb{E}(X)^k \\ &\leq 1 - [\mathbb{E}(X)^2 + \mathbb{D}(X)]^k \\ &= 1 - (x+y)^k \end{aligned} \tag{20}$$

Let $\beta \cdot (x+y)^k = \frac{1}{\ell}$. By substituting (20) into (19) we have

$$\begin{aligned}\mathbb{D}(Y) &\leq \frac{1}{\beta} \cdot (1 - \frac{1}{\beta\ell})^{\ell-1} \\ &\leq \frac{1}{\beta} \\ &= \ell \cdot (1 - \sqrt[\ell]{1-\alpha}) \cdot \gamma^k \\ &\leq -\ln(1-\alpha) \cdot \gamma^k \end{aligned} \tag{21}$$

Note that $(1 - \sqrt[\ell]{1-\alpha}) \cdot \gamma^k \leq -\ln(1-\alpha)$ holds because $\ell \cdot (1 - \sqrt[\ell]{1-\alpha})$ is an increasing function of ℓ and $\lim_{\ell \to \infty} \ell \cdot (1 - \sqrt[\ell]{1-\alpha}) = -\ln(1-\alpha)$ □

α and $\mathbb{E}(X)$ are constant once users specified the expected recall rate and the search radius r, thus $\mathbb{D}(Y)$ is upperbounded by a function of k and $\mathbb{D}(X)$. Since $\mathbb{D}(X)$ is data dependent, the only way to reduce the upper bound is to choose large k. This is not always feasible because ℓ increases with k, which means larger index size.

The variance of $\mathcal{F}_{\mathbf{h}_k^\ell}(\mathcal{D}_r)$ under practical settings is far less than the aforementioned upperbound. Next, we will examine the variance of $\mathcal{F}_{\mathbf{h}_k^\ell}(\mathcal{D}_r)$ using some practical parameter settings. The parameter k and ℓ are often tuned based on the data and application characteristics to achieve the best performance. In this paper, we choose five typical settings for different real datasets reported in [7,3]. The recall rate α is set to 0.9 for all settings. Although $\mathbb{D}[p_{\mathbf{h}}(\mathcal{D}_r]$ varies with datasets, we fixed it to 0.01 because even for small real datasets (less than 10k), $\mathbb{D}[p_{\mathbf{h}}(\mathcal{D}_r]$s is no greater than 0.01 as reported in [16]. $p_{\mathbf{h}}(r)$ is computed using Equation (1) by substituting α, k and l in. $\mathbb{D}[\mathcal{F}_{\mathbf{h}_k^\ell}(\mathcal{D}_r)]$ is computed using Equation (17) and results are listed in Table 2. We can find that the real variances of $\mathcal{F}_{\mathbf{h}_k^\ell}(\mathcal{D}_r)$ for all datasets are much smaller (two or three order of magnitude less) than original variance 0.01 after concatenating $k * \ell$ random LSH functions.

Table 2
The variance of $\mathcal{F}_{\mathbf{h}_k^\ell}(\mathcal{D}_r)$ under practical parameter settings.

dataset	k	L	variance
β-globin in [3]	7	383	4.81e-005
TCR in [3]	11	258	6.08e-005
chromosome in [3]	13	583	3.21e-005
HSV/SIFT in [7]	13	253	6.45e-005
DIPOLE in [7]	15	36	3.9e-004

4.2. Relation between $\mathbb{D}[\mathcal{F}_{h_k^\ell}(\mathcal{D}_r)]$ and $\mathbb{D}[P_{h_k^\ell}(\mathcal{D}_r)]$

In this subsection, we discuss the relation between $\mathbb{D}[\mathcal{F}_{h_k^\ell}(\mathcal{D}_r)]$ and $\mathbb{D}[P_{h_k^\ell}(\mathcal{D}_r)]$. Recall that h_k^ℓ denotes any randomly generated but then fixed LSH data structure. Let $\mathcal{S}_{h_k^\ell}(\mathcal{D}_r) = \{\mathcal{D}_r^* | \mathcal{D}_r^* = \mathcal{D}_r, p_{h_{i,j}}(\mathcal{D}_r^*) = p_{h_{i,j}}(\mathcal{D}_r), 1 \leq i \leq k, 1 \leq j \leq \ell\}$ where $\exists h_{i,j}, \langle q^*, v^* \rangle \in \mathcal{D}_r^*$ *s.t.* $(h_{i,j}(q^*) = h_{i,j}(v^*) \wedge h_{i,j}(q) \neq h_{i,j}(v)) \vee (h_{i,j}(q^*) \neq h_{i,j}(v^*) \wedge h_{i,j}(q) = h_{i,j}(v))$. Informally, $\mathcal{S}_{h_k^\ell}(\mathcal{D}_r)$ is the set of all possible data distributions $\mathcal{D}_r^*$ under the constraints of $\mathcal{D}_r^* = \mathcal{D}_r$ and $p_{h_{i,j}}(\mathcal{D}_r^*) = p_{h_{i,j}}(\mathcal{D}_r)$. By definition it is easy to prove that $\mathbb{E}_{\mathcal{D}_r^*}[P_{h_k^\ell}(\mathcal{D}_r^*)] = \mathcal{F}_{h_k^\ell}(\mathcal{D}_r)$. Thus, to show that $\mathcal{F}_{h_k^\ell}(\mathcal{D}_r)$ is very close to $P_{h_k^\ell}(\mathcal{D}_r)$ with high probability, we only need to prove that $\mathbb{D}_{\mathcal{D}_r^*}[P_{h_k^\ell}(\mathcal{D}_r^*)]$ is bounded by a small constant, where the randomness comes from the choice of possible data distributions.

For $\mathcal{D}_r^*$ drawn uniformly and independently from $\mathcal{S}_{h_k^\ell}(\mathcal{D}_r)$, we can approximately model whether the ith, $1 \leq i \leq N$, data pair in $\mathcal{D}_r^*$ collides over $h_{i,j}$ as a random variable $Z_{i,j}$ following Bernoulli distribution $B(1, p_{h_{i,j}}(\mathcal{D}_r))$. In high dimensional spaces, randomly picked vectors are very likely to be orthogonal to each other and thus we can assume (by approximation) that $Z_{i,j}$ are independent random variables. Then whether the ith, $1 \leq i \leq N$, data pair in $\mathcal{D}_r^*$ collides over h_k^ℓ follows Bernoulli distribution $B(1, \mathcal{F}_{h_k^\ell}(\mathcal{D}_r))$. As a result, the number of N data pairs in $\mathcal{D}_r^*$ that collides over h_k^ℓ follows Binomial distribution $B(N, \mathcal{F}_{h_k^\ell}(\mathcal{D}_r))$ and $P_{h_k^\ell}(\mathcal{D}_r^*)$ is approximately equal to $B(N, \mathcal{F}_{h_k^\ell}(\mathcal{D}_r))/N$. It is easy to prove that $\mathbb{D}[Pr_{h_k^\ell}(\mathcal{D}_r^*)] = \mathbb{D}[B(N, \mathcal{F}_{h_k^\ell}(\mathcal{D}_r))/N)] = O(\frac{1}{4N})$. Then we come to the conclusion that the difference between $P_{h_k^\ell}(\mathcal{D}_r)$ and $\mathcal{F}_{h_k^\ell}(\mathcal{D}_r)$ is very small with high probability as long as N is large enough.

5. Conclusion

LSH-based algorithms are capable of providing probabilistic guarantee of the collision ratio for any two points at distance r. Such probability of collision, however, is only the expectation of the practical recall rates that practition-

ers are interested in. In this paper, we provide a theoretical justification for why recall rate is so close to its expectation. Our results may be of practical interest because, in addition to the theoretical guarantee, we identify what other factors affect the variance of the recall rate, i.e., its consistency with the theoretical probability of collision.

Acknowledgements

The work reported in this paper is partially supported by NSFC under grant numbers 61370205 and 91646117, NSF of Shanghai under grant number 13ZR1400800, Science Foundation of Tianjin under grant number 17JCYBJC15200, Tianjin Science and Technology Correspondent Project under grant number 16JCTPJC53600 and the Fundamental Research Funds for the Central Universities.

Appendix A

In this appendix, we consider the probabilities of collision $p_h^b(\mathcal{D}_r)$ and $p_h^c(\mathcal{D}_\theta)$ for the Hamming distance and Arccos similarity, respectively. Their relations with $p_{\mathbf{h}}^b(r)$ and $p_{\mathbf{h}}^c(\theta)$ are also discussed.

Hamming distance. Without loss of generality, assume the kth bit is chosen as the hash function. For two random data point o_1 and o_2 in Hamming space. Let A be the event that o_1 collides with o_2, i.e., the kth bits of o_1 and o_2 are identical. Let B be the event that the distance of o_1 and o_2 is r. Obviously, $p_h^b(\mathcal{D}_r) = Pr(A|B)$. Recall that $p_{\mathbf{h}}^b(r) = 1 - r/n$. Theorem 3 presents the relation between them.

Theorem 3. $Pr(A|B) = 1 - r/n$

Proof. By Bayes' theorem, we have

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{22}$$

In the n-dimensional Hamming space, the number of different data pairs is $2^n \times (2^n - 1)$. Provided the kth bits are identical, the number of different data pairs is $2 \times 2^{n-1} \times (2^{n-1} - 1)$. Therefore, $P(A)$ can be computed as follows

$$P(A) = 2 \times \frac{2^{n-1}}{2^n} \times \frac{2^{n-1} - 1}{2^n - 1} = \frac{2^{n-1} - 1}{2^n - 1} \tag{23}$$

The distance between o_1 and o_2 is r, which means there are r bits on which o_1 and o_2 disagree. Thus, $P(B)$ is as follows

$$P(B) = \frac{C_n^r \times 2^r \times 2^{n-1}/2}{C_{2^n}^2} \tag{24}$$

The probability of two data points with distance r provided the kth bits of them are identical is

$$P(B|A) = \frac{C_{c-1}^r \times 2^r \times 2^{n-r-1}/2}{C_{2^{n-1}}^2} \tag{25}$$

By substituting Equation (23) to (25) into Equation (22), we have

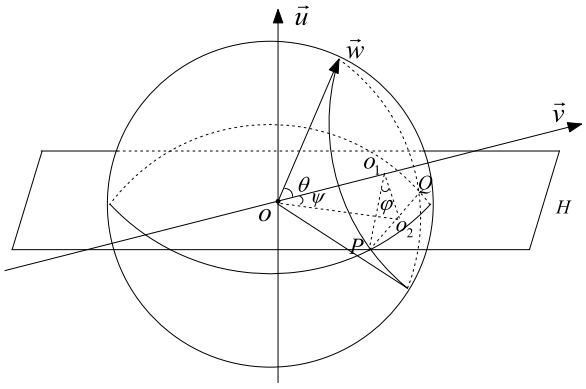

Fig. 4. An illustrative example for deriving $p_h^c(\theta)$.

$$P(A|B) = \frac{n-r}{r} \times \frac{2^n \times (2^n - 1)}{2^{n-1} \times (2^{n-1} - 1) \times 2} \times \frac{2^{n-1} - 1}{2^n - 1} = 1 - \frac{r}{n}$$

We prove the theorem. □

Arccos similarity. The idea for deriving $p_h^c(\mathcal{D}_\theta)$ is illustrated in Fig. 4. Vector $\vec{u}$ is the given hash function and the zero-axial hyperplane H is perpendicular to $\vec{u}$. Let $A(\vec{v}, \vec{w})$ denote the angle between vectors $\vec{v}$ and $\vec{w}$. To model two random vectors $\vec{v}$ and $\vec{w}$ with $A(\vec{v}, \vec{w}) = \theta$, one can view $\vec{v}$ as a random vector and $\vec{w}$ as a random generatrix of cone $C(\vec{v}, \theta)$, of which the axis is $\vec{v}$ and the half-angle is θ $(0 < \theta < \frac{\pi}{2})$. The apex of $C(\vec{v}, \theta)$ is the origin and, without loss of generality, suppose the length of $\vec{u}$ and the length of generatrices of $C(\vec{v}, \theta)$ are both equal to 1.

As $\vec{v}$ is a random vector, it is clear that in some cases $C(\vec{v}, \theta)$ has no intersection with H, which means $\vec{v}$ and $\vec{w}$ collide over $\vec{u}$. To compute $Pr_h^c(\theta)$ we only need to focus on the case where $C(\vec{v}, \theta)$ intersects with H. Observation 1 shows that in which condition $C(\vec{v}, \theta)$ has no intersection with H.

Observation 1. *If $A(\vec{v}, H) > \theta$, then $C(\vec{v}, \theta)$ and H have no intersection except for origin o, where $A(\vec{v}, H)$ is the angle ψ between $\vec{v}$ and H.*

Next we first consider the case where $\theta \in (0, \frac{\pi}{2})$ and $A(\vec{v}, H) \in [0, \theta)$, that is, $C(\vec{v}, \theta)$ intersects with H. For ease of presentation, we first confine our analysis to the 3-dimensional space as illustrated in Fig. 4. As shown in this figure, the bottom of $C(\vec{v}, \theta)$ is split into two parts by H and PQ is the intersection line. Similar to the analysis for l_2 distance, the fraction of the perimeter of $C(\vec{v}, \theta)$'s bottom under H is equal to the probability that $\vec{v}$ and $\vec{w}$ do not collide. To compute such fraction using $p^{(n)}(\theta)$, we need to compute φ ($\angle Po_1o_2$) first, which is equal to $\arccos(\frac{\tan(\psi)}{\tan\theta})$ as illustrated. $p_h^c(\mathcal{D}_\theta)$ is the weighted average *w.r.t.* ψ of such probability. Please note that the variation of ψ is equivalent to the random movement of $\vec{v}$.

When the dimensionality n is greater than 3, the probability that $\vec{v}$ and $\vec{w}$ do not collide is the fraction of the surface area of $C(\vec{v}, \theta)$'s bottom under H. The bottom of

$C(\vec{v},\theta)$ in this case is an $n-1$ dimensional ball with radius $\sin(\theta)$ and the computation of φ is similar to that in 3-dimensional space. In the case of $\theta \in (\frac{\pi}{2},\pi)$, we consider vector $\vec{v}'$ that is opposite to $\vec{v}$. $C(\vec{v}',\theta)$ can be seen as the complement of $C(\vec{v},\pi-\theta)$. Since the probability that $\vec{v}'$ and $\vec{v}$ are on the same side of H is 0, we have $p_h^c(\mathcal{D}_\theta) = 1 - p_h^c(\mathcal{D}_{\pi-\theta})$, $\theta \in (\frac{\pi}{2},\pi)$. Thus, the probability of collision that two random vectors with angle θ collide over $\vec{u}$ can be calculated as follows.

$$p_h^c(\mathcal{D}_\theta) = \begin{cases} 1 - 2\cdot\int_0^\theta [p^{(n)}(\frac{\pi}{2}-\psi) \\ \quad \times \int_0^{\arccos\frac{\tan\psi}{\tan\theta}} p^{(n-1)}(\varphi)d\varphi]d\psi \quad (1) \\ 2\cdot\int_0^{\pi-\theta}[p^{(n)}(\frac{\pi}{2}-\psi) \\ \quad \times \int_0^{\arccos\frac{\tan\psi}{\tan(\pi-\theta)}} p^{(n-1)}(\varphi)]d\psi \quad (2) \end{cases} \tag{26}$$

where $0 < \theta < \frac{\pi}{2}$ in case (1) and $\frac{\pi}{2} < \theta < \pi$ in case (2). Recall that $p_{\mathbf{h}}^c(\theta) = 1 - \frac{\theta}{\pi}$, Theorem 4 shows the asymptotic equivalence of $p_h^c(\mathcal{D}_\theta)$ and $p_{\mathbf{h}}^c(\theta)$

Theorem 4. $\lim_{n\to\infty} p_h^c(\mathcal{D}_\theta) = 1 - \frac{\theta}{\pi}$

Proof. We focus on the case $\theta \in [0,\pi/2]$ and the other case can be proved similarly. It is easy to see that $\lim_{\theta\to 0} p_h^c(\mathcal{D}_\theta) = 0$ and $\lim_{\theta\to\pi/2} p_h^c(\mathcal{D}_\theta) = \frac{1}{2}$. Thus, we can extend the domain of $p_h^c(\mathcal{D}_\theta)$ to $[0,\pi]$. Since $p_{\mathbf{h}}^c(0) = 1$ and $p_{\mathbf{h}}^c(\frac{\pi}{2}) = \frac{1}{2}$, we know that $p_h^c(\mathcal{D}_0) = p_{\mathbf{h}}^c(0)$ and $p_h^c(\mathcal{D}_{\frac{\pi}{2}}) = p_{\mathbf{h}}^c(\frac{\pi}{2})$.

Let

$$c(\theta) = \int_0^\theta [p^{(n)}(\frac{\pi}{2}-x) \int_0^{\arccos\frac{\tan x}{\tan\theta}} p^{(n-1)}(\varphi)d\varphi]dx \tag{27}$$

We have

$$c'(\theta) = \int_0^\theta [p^{(n)}(\frac{\pi}{2}-x)\cdot p^{(n-1)}(\arccos\frac{\tan x}{\tan\theta}) \cdot \frac{\tan x}{\sin^2\theta\cdot\sqrt{1-\frac{\tan^2 x}{\tan^2\theta}}}]dx \tag{28}$$

To prove the Theorem, we only need to show $\lim_{n\to\infty} c'(\theta) = \frac{1}{2\pi}$ given the fact that $c(\theta)$ and $c'(\theta)$ are both continuous functions of $\theta \in [0,\pi/2]$.

Given the fact $\lim_{n\to\infty} \frac{\Gamma(n-\frac{1}{2})}{\Gamma(n)n^{-\frac{1}{2}}} = 1$, by substituting $p^n(\theta)$ into $c'(\theta)$ we have

$$\lim_{n\to\infty} c'(\theta) = \lim_{n\to\infty} \frac{1}{2\pi}\cdot\frac{\int_0^\theta [(\cos x\cdot\sqrt{1-\frac{\tan^2 x}{\tan^2\theta}})^n]\cdot\frac{\sin x\cdot\cos x}{\sin^2\theta}]dx}{1/n} \tag{29}$$

By substituting $\sin^2 x$ with y, we have

$$\lim_{n\to\infty} c'(\theta) = \lim_{n\to\infty}\frac{1}{2\pi}\cdot\frac{\int_0^{\sin^2\theta}(1-y-\frac{y}{\tan^2\theta})^{\frac{n}{2}}dy}{2\sin^2\theta/n} \tag{30}$$

Thus we get the result

$$\begin{aligned}&\lim_{n\to\infty} c'(\theta) \\ &= \frac{1}{2\pi}\cdot\lim_{n\to\infty} -\frac{(1-y-\frac{y}{\tan^2\theta})^{\frac{n}{2}+1}\Big|_0^{\sin^2\theta}}{(\frac{n}{2}+1)\cdot(1+\frac{1}{\tan^2\theta})\cdot(2\sin^2\theta/n)} \\ &= \frac{1}{2\pi}\end{aligned} \tag{31}$$

We prove the theorem. □

Proof of Lemma 1.

Proof. The volume of an n-dimensional Euclidean ball of radius r is

$$V_n(r) = \frac{\pi^{n/2}}{\Gamma(n/2+1)}r^n \tag{32}$$

The surface area of the $n-1$-dimensional sphere of radius r in $\Re^n$ is

$$S_{n-1}(r) = \frac{dV_n(r)}{dr} = \frac{2\pi^{n/2}}{\Gamma(n/2)}r^{n-1} \tag{33}$$

Then we have

$$p^{(n)}(\theta)d\theta = \frac{dS_{n-1}(r)}{S_{n-1}(r)} = \frac{S_{n-2}(r\sin\theta)rd\theta}{S_{n-1}(r)} \tag{34}$$

which leads us to

$$p^{(n)}(\theta) = \frac{\Gamma(n/2)}{\Gamma((n-1)/2)}\cdot\frac{\sin^{n-2}(\theta)}{\sqrt{\pi}} \quad \square \tag{35}$$

References

[1] Alexandr Andoni, Piotr Indyk, Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions, Commun. ACM 51 (1) (2008) 117–122.

[2] Andrei Z. Broder, Steven C. Glassman, Mark S. Manasse, Geoffrey Zweig, Syntactic clustering of the web, Comput. Netw. 29 (8–13) (1997) 1157–1166.

[3] Jeremy Buhle, Efficient large-scale sequence comparison by locality-sensitive hashing, in: Bioinformatics, 2001, pp. 419–428.

[4] Mayur Datar, Nicole Immorlica, Piotr Indyk, Vahab S. Mirrokni, Locality-sensitive hashing scheme based on p-stable distributions, in: SoCG, 2004, pp. 253–262.

[5] Junhao Gan, Jianlin Feng, Qiong Fang, Wilfred Ng, Locality-sensitive hashing scheme based on dynamic collision counting, in: SIGMOD, 2012, pp. 541–552.

[6] Piotr Indyk, Rajeev Motwani, Approximate nearest neighbors: towards removing the curse of dimensionality, in: STOC, 1998, pp. 604–613.

[7] Alexis Joly, Olivier Buisson, A posteriori multi-probe locality sensitive hashing, in: ACM Multimedia, 2008, pp. 209–218.

[8] Yan Ke, Rahul Sukthankar, Larry Huston, An efficient parts-based near-duplicate and sub-image retrieval system, in: ACM Multimedia, 2004, pp. 869–876.

[9] Jon M. Kleinberg, Two algorithms for nearest-neighbor search in high dimensions, in: STOC, 1997, pp. 599–608.

[10] Hongrae Lee, Raymond T. Ng, Kyuseok Shim, Similarity join size estimation using locality sensitive hashing, Proc. VLDB Endow. 4 (6) (2011) 338–349.
[11] Qin Lv, William Josephson, Zhe Wang, Moses Charikar, Kai Li, Multi-probe LSH: efficient indexing for high-dimensional similarity search, in: VLDB, 2007, pp. 950–961.
[12] Rajeev Motwani, Assaf Naor, Rina Panigrahy, Lower bounds on locality sensitive hashing, SIAM J. Discrete Math. 21 (4) (2007) 930–935.
[13] Venu Satuluri, Srinivasan Parthasarathy, Bayesian locality sensitive hashing for fast similarity search, Proc. VLDB Endow. 5 (5) (2012) 430–441.
[14] Narayanan Sundaram, Aizana Turmukhametova, Nadathur Satish, Todd Mostak, Piotr Indyk, Samuel Madden, Pradeep Dubey, Streaming similarity search over one billion tweets using parallel locality-sensitive hashing, Proc. VLDB Endow. 6 (14) (2013) 1930–1941.
[15] Yufei Tao, Ke Yi, Cheng Sheng, Panos Kalnis, Quality and efficiency in high dimensional nearest neighbor search, in: SIGMOD, 2009, pp. 563–576.
[16] Hongya Wang, Jiao Cao, LihChyun Shu, Davood Rafiei, Locality sensitive hashing revisited: filling the gap between theory and algorithm analysis, in: CIKM, 2013, pp. 1969–1978.