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.

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/

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.