S2JSD-LSH: A Locality-Sensitive Hashing Schema for Probability Distributions

Xian-Ling Mao,♠ Bo-Si Feng,♠ Yi-Jing Hao,♠ Liqiang Nie,♡ Heyan Huang,♠*, Guihua Wen♣
♠Department of Computer Science, Beijing Institute of Technology, China
♡Department of Computing, National University of Singapore, Singapore
♣Department of Computer Science and Technology, South China University of Technology, China
{maoxl, 2120160986, 2220150504, hhy63}@bit.edu.cn
nieliqiang@gmail.com, crghwen@scut.edu.cn

Abstract

To compare the similarity of probability distributions, the information-theoretically motivated metrics like Kullback-Leibler divergence (KL) and Jensen-Shannon divergence (JSD) are often more reasonable compared with metrics for vectors like Euclidean and angular distance. However, existing locality-sensitive hashing (LSH) algorithms cannot support the information-theoretically motivated metrics for probability distributions. In this paper, we first introduce a new approximation formula for S2JSD-distance, and then propose a novel LSH scheme adapted to S2JSD-distance for approximate nearest neighbors search in high-dimensional probability distributions. We define the specific hashing functions, and prove their local-sensitivity. Furthermore, extensive empirical evaluations well illustrate the effectiveness of the proposed hashing schema on six public image datasets and two text datasets, in terms of mean Average Precision, Precision@N and Precision-Recall curve.

In the past decade, we have witnessed an explosive growth of data on the Internet. Billions of data are publicly available on the Web, and it brings both challenges and opportunities to traditional algorithms developed on small to median scale data sets. Particularly, nearest neighbor search has become a key ingredient in many large-scale machine learning and data management tasks. In large-scale applications, it is usually time-consuming or impossible to return the exact nearest neighbors to a given query. Fortunately, approximate nearest neighbors (ANN) (Indyk and Motwani 1998; Liu et al. 2014b) are enough to achieve satisfactory performance in many applications, such as the image retrieval task in search engines. Moreover, ANN search is usually more efficient than exact nearest neighbor search to solve large-scale problems. Hence, ANN search has attracted more and more attention in this big data era (Heo et al. 2015).

Due to the low storage cost and fast retrieval speed, hashing is one of the popular solutions for ANN search (Liu et al. 2014a; Andoni et al. 2014; Zhen et al. 2016). The hashing techniques used for ANN search are usually called similarity-preserving hashing, and its basic idea is to transform the data points from the original feature space into a binary-code Hamming space, where the similarity in the original space is preserved. More specifically, the Hamming distance between the binary codes of two points should be small if these two points are similar in the original space. Otherwise, the Hamming distance should be as large as possible. With the binary-code representation, the storage cost can be substantially reduced and the query speed can be dramatically improved for ANN search (Indyk and Motwani 1998; ODonnell, Wu, and Zhou 2014). For example, if we encode each image with 256 bits, we can store a data set of 1 million images with only 32M memory.

The existing hashing methods can be mainly divided into two categories (ODonnell, Wu, and Zhou 2014; Andoni and Razenshteyn 2015): data-independent methods and data-dependent methods. Data-dependent hashing methods (Liu et al. 2014a; Zhang et al. 2015) learn hash functions from the training data; Data-independent hashing methods like locality sensitive hashing (LSH), use simple random projections which are independent of the training data for hash functions. More details refer to (Wang et al. 2014; 2016) for a brief survey. Compared with the data-dependent methods, data-independent methods are dynamic, and allow dynamically updates to the point set. The dynamic feature of data-independent methods facilitates many tasks such as image retrieval where crawled images are feed persistently.

Existing data-independent hashing methods depend on two crucial elements: 1) Data type; 2) Distance metric. For vector-type data, we can use Euclidean distance (l_2), Manhattan distance (l_1), and angular metric ($arccos$) etc., to measure the distance between two vectors. Based on these metrics, various hashing methods are developed. Particularly, based on l_p distance with $p \in [0, 2)$, lots of LSH methods have been proposed, such as p-stable LSH (Datar et al. 2004), Leech lattice LSH (Andoni and Indyk 2006), Spherical LSH (Terasawa and Tanaka 2007), and Beyond LSH (Andoni et al. 2014). Also, angular metric ($\arccos$) is a popular measure for vectors, and many LSH methods based on angular metric have been developed, e.g. Random Projection (PR) (Charikar 2002; Andoni and Indyk 2006), Super-bit LSH (Ji et al. 2012), Kernel LSH (Kulis and Grauman 2012), Concomitant LSH (Eshghi and Rajaram 2008), and Hyperplane hashing (Jain, Vijayanarasimhan, and Grauman 2010). Moreover, Chi-squared Distance and Bregman divergence have also been used as similarity functions to develop corresponding hashing algorithms for vectors (Gorisse, Cord,

[*] Corresponding author.

and Precioso 2012; Mu and Yan 2010). For set-type data, Jaccard Coefficient based LSH include Min-hash (Broder et al. 1997), K-min Sketch (Li, Owen, and Zhang 2012), Min-max hash (Ji et al. 2013), B-bit minwise hashing (Li, Konig, and Gui 2010), and Sim-min-hash (Zhao, Jégou, and Gravier 2013) etc.

However, as far as we know, few prior LSH work is devoted to the distance for probability distributions. Probability-distribution-type data is widespread, such as topics in topic modeling (Chen et al. 2015), color histogram or normalized bag of visual words in image processing (Karpathy and Fei-Fei 2015). Intuitively, we can simply use the existing LSH methods for vectors to process the probability distributions if taking the probability distributions as general vectors. For example, the work (Krstovski et al. 2013) first reducts Hellinger divergence to Euclidean distance, and then use existing ANN techniques, such as LSH and k-d tree, to accelerate search in the probability simplex. However, the solution does not consider the special attributions of probability distributions, such as Non-negative and Sum-equal-one. Thus, it is not optimal solution. Furthermore, when comparing the similarity of probability distributions, the information-theoretically motivated metrics like Kullback-Leibler divergence (KL), Jensen-Shannon divergence (JSD) and S2JSD (Endres and Schindelin 2003) are often more reasonable compared with metrics for vectors like Euclidean (l_2) and angular (θ) distance. For example, in the K nearest neighbors (KNN) search task, p@5 results of five metrics on four public datasets in the form of probability distribution by brute-force search are as follows

Datasets	Distance Measures				
	θ	l_2	KL	JSD	S2JSD
Local-Patch	0.848	0.850	0.832	**0.852**	**0.852**
CIFAR100-100	0.198	0.194	0.212	**0.218**	**0.218**
CIFAR100-20	0.342	0.322	**0.348**	0.346	0.346
CIFAR10	0.493	0.478	0.522	**0.528**	**0.528**

The results shows that the information-theoretically motivated metrics over probability distributions perform better than metrics for vectors.

Most the information-theoretically motivated metrics, such as JSD and KL, are not the well-defined distance metrics, which do not satisfy triangle inequality; Lemma 1 in the paper (Charikar 2002) says: "for any similarity function that admits a locality sensitive hash function family, its distance function satisfies triangle inequality.", thus LSH for JSD or KL does not exist since there isn't a triangular inequality. Fortunately, Endres and Schindelin (2003) have introduced a new metric for probability distributions, called *S2JSD*. *S2JSD* has been proved that it satisfies the symmetry, non-negativity and triangle inequality, i.e. it is a distance metric. Thus, in this paper, we will study the hashing schema based on S2JSD-distance for probability distributions by developing a new approximation formula $S2JSD_{aprx}^{new}$.

We make the following contributions in this paper:

- We propose a new approximation formula $S2JSD_{aprx}^{new}$ for S2JSD-distance, tailored for the S2JSD hashing method. It is symmetric, and has better approximation performance.
- We develop a novel similarity-preserving hashing schema for $S2JSD_{aprx}^{new}$, which can be applied to probability distributions.
- We have released our codes to facilitate other researchers to repeat our experiments and validate their own ideas [1].

Preliminary

In this section, we present an overview of the locality-sensitive hashing (LSH) schema. The LSH algorithm was first introduced in (Indyk and Motwani 1998), to solve the near neighbor search problem. It is based on the definition of LSH family $\mathscr{H}$, a family of hash functions mapping similar input items to the same hash code with higher probability than dissimilar items. Let *S* be the domain of the objects and D the distance measure between objects. Formally, an LSH family is defined as follows:

Definition 1 (Locality-sensitive hashing). *A function family $\mathscr{H} = \{h : S \to U\}$ is called (r_1, r_2, p_1, p_2)-sensitive, with $r_1 < r_2$ and $p_1 > p_2$, for D if for any v, q $\in$ S*

- *if $D(q, v) \leq r_1$ then $P_H[h(q) = h(v)] \geq p_1$,*
- *if $D(q, v) > r_2$ then $P_H[h(q) = h(v)] \leq p_2$.*

Intuitively, the definition states that nearby objects (those within distance r_1) are more likely to collide ($p_1 > p_2$) than objects that are far apart (those with a distance greater than r_2).

Given an LSH family $\mathscr{H}$, the LSH scheme amplifies the gap between the high probability p_1 and the low probability p_2 by concatenating several functions. In particular, for a given integer *K*, let us define a new function family $\mathscr{G} = \{g : S \to U^K\}$ such that $g(v) = (h_1(v), ..., h_K(v))$, where $h_i \in \mathscr{H}$.

Distance Metric for Probability Distributions

First, we investigate a distance measure for probability distributions, which is suitable to develop a Locality Sensitive Hashing method.

Endres and Schindelin (2003) introduced a metric for probability distributions, which is bounded, information-theoretically motivated, and it is a close relative of the capacitory discrimination and Jensen-Shannon divergence (JSD) (Endres and Schindelin 2003). The distance measure is the square root of two times the Jensen-Shannon divergence, called *S2JSD*, as follows:

$$\begin{aligned} S2JSD(P,Q) &= \sqrt{2JSD} \\ &= \sqrt{\sum_{i=1}^{N}(p_i \log \frac{2p_i}{p_i + q_i} + q_i \log \frac{2q_i}{p_i + q_i})} \qquad (1) \end{aligned}$$

where *P* and *Q* are two known distributions, N is the dimensionality of *P* and *Q*, and p_i and q_i are respectively the values of the i^{th} component of *P* and *Q*. *S2JSD* has been proved that it satisfies the symmetry, non-negativity and triangle inequality, i.e. it is a distance metric. Also, Endres and Schindelin (2003) have proved that *S2JSD* distance can be approximated asymptotically by

[1] https://www.dropbox.com/s/2yral5h23lwzipp/src.zip?dl=0

$$S2JSD \approx S2JSD_{aprx}^{ES} = \sqrt{\frac{1}{4}\sum_{j=1}^{N}\frac{(p_j - q_j)^2}{q_j}} \tag{2}$$

Obviously, the approximation breaks the symmetry of *S2JSD*, which shows the approximation is not a real distance metric. To overcome the shortcoming, in this paper, we will propose a new approximation of S2JSD distance, which satisfies the symmetry.

First, we can expand S2JSD-distance by a term-by-term Taylor series to yield

$$\begin{aligned} S2JSD &= \sqrt{\sum_{i=1}^{N} p_i \log\frac{2p_i}{p_i+q_i} + q_i \log\frac{2q_i}{p_i+q_i}} \\ &= \sqrt{\sum_{i=1}^{N}\sum_{v=1}^{\inf}\frac{1}{2v(2v-1)}(p_i+q_i)(\frac{p_i-q_i}{p_i+q_i})^{2v}} \\ &= \sqrt{\sum_{v=1}^{\inf}\sum_{i=1}^{N}\frac{1}{2v(2v-1)}(p_i+q_i)(\frac{p_i-q_i}{p_i+q_i})^{2v}} \end{aligned} \tag{3}$$

where v is the index of Taylor series expansion, N is the dimension of a probability distribution. Then, by using first order approximation, we can obtain

$$S2JSD \approx S2JSD_{aprx}^{new} = \sqrt{\frac{1}{2}\sum_{i=1}^{N}\frac{(p_i - q_i)^2}{p_i + q_i}} \tag{4}$$

Unlike $S2JSD_{aprx}^{ES}$, our proposed approximation $S2JSD_{aprx}^{new}$ does not break the symmetry of *S2JSD*; Meanwhile, it's easy to prove that $S2JSD_{aprx}^{new}$ satisfies triangle inequality. Furthermore, we will observe that our proposed approximation $S2JSD_{aprx}^{new}$ is closer to S2JSD than the approximation $S2JSD_{aprx}^{ES}$ with large probability for high-dimensional probability distributions by following simulation.

As we know, if parameter $\alpha = 1$, we can obtain randomly probability distributions through symmetry Dirichlet distribution, i.e. Dir(α =1). For each dimensionality d ($d \in \{2, 3, ..., 1000\}$), we sample respectively 1,000,000 <P, Q> pairs of d-dimensional probability distributions from Dir(α =1), then we compute the value of $S2JSD$ (Eq.(1)), Endres' approximation $S2JSD_{aprx}^{ES}$ (Eq.(2)) and our approximation $S2JSD_{aprx}^{new}$ (Eq.(4)) for each pair. For a <P, Q> pair, if $(|S2JSD_{aprx}^{ES} - S2JSD| > |S2JSD_{aprx}^{new} - S2JSD|)$, the $S2JSD_{aprx}^{new}$ is closer to S2JSD than $S2JSD_{aprx}^{ES}$, which means the proposed new approximation is better, and the pair is called as **Intended Pair**. We count the number of the Intended Pairs, then divide it by total number of pairs, called **Intended Pair Ratio**. Figure 1 shows the values of the Intended Pair Ratio at each dimensionality when the dimensionality d changes from 2 to 1000. From the figure, we observe that the proposed new approximation $S2JSD_{aprx}^{new}$ is better than Endres' approximation $S2JSD_{aprx}^{ES}$ with large probability, especially when the dimensionality is high. For example, when the dimensionality is 100, the Intended Pair Ratio is 98.455%, which means the proposed approximation $S2JSD_{aprx}^{new}$ is closer to S2JSD than $S2JSD_{aprx}^{ES}$ with about 98.455% probability.

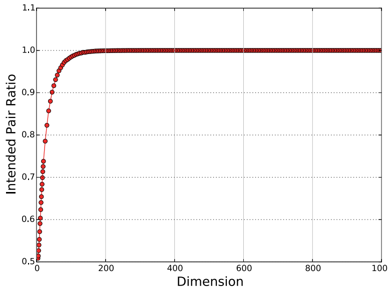

Figure 1: Simulation Results over 1,000,000 sampling <P, Q> pairs respectively at each dimension.

In a word, compared with $S2JSD_{aprx}^{ES}$, the proposed new approximation $S2JSD_{aprx}^{new}$ has two merits: (i) Symmetry; (ii) Better approximation. Furthermore, without $\log$ function, $S2JSD_{aprx}^{new}$ is more appropriate than $S2JSD$ to develop a LSH algorithm. Thus, in this paper, we will develop a novel locality sensitive hashing algorithm based on $S2JSD_{aprx}^{new}$ distance while keeping the same effectiveness as S2JSD distance.

The Proposed Hashing Algorithm

Borrowing the idea in the p-stable LSH (Datar et al. 2004; Wang et al. 2014), the line $l_{\mathbf{a}}$ is chosen as the projection-space. This line is obtained by projecting all points on a random vector **a** whose each entry is the absolute value of a sample chosen independently from a standard normal distribution. We uniformly partition this line with respect to the $S2JSD_{aprx}^{new}$ distance, i.e. each partition interval $[Y_{i-1}, Y_i]$ has the same length, W:

$$\forall i \in N, \quad S2JSD_{aprx}^{new}(Y_{i-1}, Y_i) \overset{def}{=} \sqrt{\frac{1}{2}\frac{(Y_{i-1} - Y_i)^2}{Y_{i-1} + Y_i}} = W \tag{5}$$

We can rewrite Eq.(5):

$$Y_i = Y_{i-1} + W^2(\sqrt{\frac{4Y_{i-1}}{W^2} + 1} + 1), i \in N \tag{6}$$

If Y_0 = 0, we get:

$$Y_i = f(i) = i(i+1)W^2, i \in N \tag{7}$$

Now, we want to define hash functions $h_{\mathbf{a}}$ such that:

$$\forall \mathbf{p} \in K^{+d}, \qquad h_{\mathbf{a}}(\mathbf{p}) = i \iff Y_{i-1} \le \mathbf{a} \cdot \mathbf{p} < Y_i \tag{8}$$

where K^{+d} is the point set, where each point is a probability distribution with d dimensionality. For any y = $\mathbf{a} \cdot \mathbf{p}$, we have to calculate the integer i such that i $\le \lfloor f^{-1}(y) \rfloor <$ i + 1. From Eq.(7), we get the inverse function:

$$i = g_w(Y_i) = f^{-1}(Y_i) = \frac{\sqrt{\frac{4Y_i}{W^2} + 1} - 1}{2}, i \in N, \tag{9}$$

For any $\mathbf{a} \cdot \mathbf{p}$, if let Y_i = $\mathbf{a} \cdot \mathbf{p}$, Eq.(9) be reformed as:

$$\forall \mathbf{a} \cdot \mathbf{p} \in R^+, \qquad i = \lfloor g_w(\mathbf{a} \cdot \mathbf{p}) \rfloor = \left\lfloor \frac{\sqrt{\frac{4\mathbf{a}\cdot\mathbf{p}}{W^2} + 1} - 1}{2} \right\rfloor \tag{10}$$

i.e. we can define the hash functions as follows:

$$h_{\mathbf{a}}(\mathbf{p}) = \lfloor g_w(\mathbf{a}\cdot\mathbf{p})\rfloor = \left\lfloor \frac{\sqrt{\frac{4\mathbf{a}\cdot\mathbf{p}}{W^2}+1}-1}{2} \right\rfloor \tag{11}$$

The previous construction of the hash functions holds by setting Y_0 = b because all points are shifted by b, i.e.,

$$h_{\mathbf{a},b}(\mathbf{p}) = \lfloor g_w(\mathbf{a}\cdot\mathbf{p}) + b\rfloor = \left\lfloor \frac{\sqrt{\frac{4\mathbf{a}\cdot\mathbf{p}}{W^2}+1}-1}{2} + b \right\rfloor \tag{12}$$

where $b \in Unif(0,1)$. The family of such hash functions be denoted as $\mathscr{H}$, called *S2JSD-LSH*.

S2JSD-LSH: A Locality Sensitive Function

According the results in (Datar et al. 2004), for a fixed **a**, b, if the hash function $h_{\mathbf{a},b}$ has the form of $h_{\mathbf{a},b}(\mathbf{v}) = \lfloor \frac{\mathbf{a}\cdot\mathbf{v}+b}{r} \rfloor$, it follows the proposition below.

Proposition 1 (p-stable distribution property)**.** *Let* $f_p(t)$ *(*$p \in (0,1]$*) denote the probability density function of the absolute value of the p-stable distribution, b is a real number chosen uniformly from the range* $[0,r]$*. Given two vectors* $\mathbf{v_1}$*,* $\mathbf{v_2}$*, and a random vector* $\mathbf{a}$ *where each entry is drawn from a p-stable distribution,* $\mathbf{a}\cdot(\mathbf{v_1}-\mathbf{v_2})$ *is distributed as* cX *where* $c = \|\mathbf{v_1}-\mathbf{v_2}\|_p$ *and* X *is a random variable drawn from a p-stable distribution. It follows that:*

$$P = p(c) = \int_0^r \frac{1}{c} f_p(\frac{t}{c})(1-\frac{t}{r})dt \tag{13}$$

For a fixed parameter r, P decreases monotonically with c.

We now illustrate that the original LSH schema (Definition 1) still holds for S2JSD-LSH hash family $\mathscr{H}$.

Theorem 1 (S2JSD-LSH sensitivity)**.** *The S2JSD-LSH hash function family* $\mathscr{H}$*, defined in Eq.(12), is* (r_1, r_2, p_1, p_2)*-sensitive.*

Proof. Let us define P as the probability of the hash functions to be locality-sensitive:

$$\begin{aligned} P &= P_{\mathscr{H}}[h_{\mathbf{a},b}(\mathbf{p}) = h_{\mathbf{a},b}(\mathbf{q})] \\ &= P_{\mathbf{a},b}\begin{bmatrix} \exists n, n \le g_w(\mathbf{a}\cdot\mathbf{p}) + b < n+1 \\ n \le g_w(\mathbf{a}\cdot\mathbf{q}) + b < n+1 \end{bmatrix} \end{aligned}$$

For all **a**, without loss of generality and for the sake of demonstration clarity, let us consider $\mathbf{a}\cdot\mathbf{p} \le \mathbf{a}\cdot\mathbf{q}$, **a** and b are independent. Then P may be computed using marginalization over b with the following integral bounds. From the two inequalities above, we have: $n \le g_w(\mathbf{a}\cdot\mathbf{p}) + b \le g_w(\mathbf{a}\cdot\mathbf{q}) + b < n+1$, so that bounds on b are:

$$n - g_w(\mathbf{a}\cdot\mathbf{p}) \le b < n + 1 - g_w(\mathbf{a}\cdot\mathbf{q}) \tag{14}$$

and we also have:

$$0 \le g_w(\mathbf{a}\cdot\mathbf{q}) - g_w(\mathbf{a}\cdot\mathbf{p}) \le 1 \tag{15}$$

Integrating on the random variable b leads to:

$$\int_{n-g_w(\mathbf{a}\cdot\mathbf{p})}^{n+1-g_w(\mathbf{a}\cdot\mathbf{q})} db = 1 - (g_w(\mathbf{a}\cdot\mathbf{q}) - g_w(\mathbf{a}\cdot\mathbf{p})) \tag{16}$$

P can be rewrited as:

$$P = P_{\mathbf{a}}\big[0 \le 1 - (g_w(\mathbf{a}\cdot\mathbf{q}) - g_w(\mathbf{a}\cdot\mathbf{p})) \le 1\big] \tag{17}$$

Considering the expression Eq.(7) and the relation between hash function $h_{\mathbf{a},b}$ and interval bounds Y_n in Eq.(8), we can then rewrite,

$$P = P_{\mathbf{a}}\Big[0 \le \mathbf{a}(\mathbf{p}-\mathbf{q}) \le 2(n+1)W^2\Big] \tag{18}$$

Because each entry of **a** in S2JSD-LSH is from standard normal distribution which is a stable distribution, we can make use of Proposition 1 (r = $2(n+1)W^2$) to get:

$$P = p(c) = \int_0^{2(n+1)W^2} \frac{1}{c} f(\frac{t}{c})(1-\frac{t}{2(n+1)W^2})dt \tag{19}$$

At the same time, P decreases monotonically with respect to c, reminding that $r_1 < r_2$, if we set $p_1 = p(r_1)$ and $p_2 = p(r_2)$, then $p_2 < p_1$. This concludes the proof of Theorem 1: The S2JSD-LSH hash function family $\mathscr{H}$ is (r_1, r_2, p_1, p_2)-sensitive. □

Experiments

Data Sets and Evaluation Protocols

Six publicly available image datasets, namely CIFAR10, CIFAR100-20, CIFAR100-100, Local-Patch, MNIST and COVTYPE, and two crawled text datasets are used to compare the proposed approach against state-of-the-art methods. **CIFAR10**[2] dataset consists of 60K 32x32 colour images in 10 classes. Every image is represented by a 512-dimensional GIST feature vector. CIFAR-100 is just like the CIFAR-10, except that it has 20 "coarse" and 100 "fine" superclasses, denoted as **CIFAR100-20**[2] and **CIFAR100-100**[2]. **Local-Patch**[3] contains roughly 300K 32x32 image patches from photos of Trevi Fountain (Rome), Notre Dame (Paris) and Half Dome (Yosemite). For each image patch, we compute a 128-dimensional SIFT vector as the holistic descriptor. **MNIST**[4] consists of a total of 70000 handwritten digit samples, each with 780 features. **COVTYPE**[5] is a common benchmark featuring 54 dimensions. The feature vectors of all image datasets are probability distributions by L_1-normalization.

Moreover, probability distributions with labels are generated by a topic model over two labeled text datasets. First, we crawled nearly all the questions and associated answer pairs (QA pairs) of two top categories of Yahoo! Answers: *Computers* & *Internet* and *Health*. This produced forty-three sub-categories from 2005.11 to 2008.11, and an archive of 6,345,786 QA documents. We refer the Yahoo! Answers data as **Y_Ans**. The second dataset contains 2.1G microblogs with 3503 hashtags, which removed microblogs without hashtag and hashtags whose idf is less than 50, downloaded from Twitter website in six days, denoted as **TW**. We built

[2] http://www.cs.toronto.edu/ kriz/cifar.html

[3] http://phototour.cs.washington.edu/

[4] http://yann.lecun.com/exdb/mnist/

[5] https://archive.ics.uci.edu/ml/datasets/Covertype

Table 1: mAP on eight datasets. The best mAP is shown in bold face.

	Local-Patch			**CIFAR100-100**		
#Bits	sblsh	l_2	S2JSD-LSH	sblsh	l_2	S2JSD-LSH
8	0.3369	0.3246	**0.3849**	0.0114	0.0101	**0.0122**
16	0.3485	0.3246	**0.3588**	0.0129	0.0101	**0.0139**
32	0.3522	0.3246	**0.3628**	**0.0169**	0.0101	0.0151
64	0.3640	0.3246	**0.3685**	0.0153	0.0101	**0.0203**
128	**0.3786**	0.3246	0.3755	**0.0214**	0.0101	0.0203
256	0.3804	0.3246	**0.3825**	0.0228	0.0101	**0.0232**
	CIFAR100-20			**CIFAR10**		
#Bits	sblsh	l_2	S2JSD-LSH	sblsh	l_2	S2JSD-LSH
8	0.0533	0.0502	**0.0563**	0.1035	0.1002	**0.1128**
16	0.0546	0.0502	**0.0554**	0.1072	0.1002	**0.1167**
32	**0.0606**	0.0502	0.0559	0.1112	0.1002	**0.1244**
64	**0.0615**	0.0502	0.0604	0.1283	0.1002	**0.1333**
128	0.0657	0.0502	**0.0667**	0.1391	0.1002	**0.1453**
256	0.0689	0.0502	**0.0717**	0.1413	0.1002	**0.1430**
	MNIST			**COVTYPE**		
#Bits	sblsh	l_2	S2JSD-LSH	sblsh	l_2	S2JSD-LSH
8	0.1125	0.1004	**0.1429**	0.4016	0.3649	**0.4078**
16	**0.1818**	0.1004	0.1654	0.4171	0.3649	**0.4197**
32	**0.2431**	0.1004	0.2227	0.4214	0.3649	**0.4221**
64	0.2971	0.1004	**0.2993**	**0.4220**	0.3649	0.4192
128	0.3123	0.1004	**0.3242**	0.4343	0.3649	**0.4357**
256	0.3664	0.1004	**0.3710**	0.4390	0.3649	**0.4396**
	Y_Ans			**TW**		
#Bits	sblsh	l_2	S2JSD-LSH	sblsh	l_2	S2JSD-LSH
8	0.8793	0.8712	**0.9031**	0.8513	0.8334	**0.8647**
16	0.8817	0.8712	**0.9085**	0.8582	0.8334	**0.8691**
32	0.8835	0.8712	**0.9140**	0.8645	0.8334	**0.8711**
64	0.8846	0.8712	**0.9238**	0.8693	0.8334	**0.8725**
128	0.8907	0.8712	**0.9274**	0.8736	0.8334	**0.8777**
256	0.8963	0.8712	**0.9323**	0.8748	0.8334	**0.8882**

"probability distribution-label" database over the two text data sets above by Labeled LDA algorithm (Ramage et al. 2009), which can automatically obtain a topic (i.e. a probability distribution) for a label. Specifically, to increase the number of points with same labels in probability space, we split separately the two datasets into 12 pieces (**TW**) and 30 pieces (**Y_Ans**), and train separately Labeled LDA over each piece. After training, **Y_Ans** consists of 1,380 points in 46 classes, and every point is represented by a 153,827-dimensional probability distribution; **TW** consists of 12,139 points in 3,503 classes, and every point is represented by a 189,841-dimensional probability distribution.

In this paper, the state-of-the-art methods, Super-bit LSH ($sblsh$) (Ji et al. 2012) and p-stable LSH (l_2) (Datar et al. 2004; Wang et al. 2016), are chosen to evaluate the effectiveness of the proposed methods. Super-bit LSH ($sblsh$) is based on angular distance for vectors, while p-stable LSH (l_2) is based on Euclidean distance for vectors. For all the baselines, we set the parameters by following the suggestions in the corresponding papers.

All the experimental results are averaged over 10 random training/test partitions. For each partition, we randomly select 100 points with their tags as queries, and the remaining points and tags as reference database. We use **mean Average Precision** (mAP), **p@N** and **Precision-Recall curve** to illustrate performances of different methods.

All experiments are conducted on our workstation with Intel(R) Xeon(R) CPU X7560@2.27GHz and 32G memory.

Table 2: p@5 on eight datasets. The best p@5 is shown in bold face.

	Local-Patch			**CIFAR100-100**		
#Bits	sblsh	l_2	S2JSD-LSH	sblsh	l_2	S2JSD-LSH
8	0.342	0.333	**0.335**	0.012	0.014	**0.018**
16	**0.393**	0.333	0.387	**0.030**	0.014	0.016
32	0.475	0.333	**0.522**	**0.038**	0.014	0.032
64	0.648	0.333	**0.652**	0.044	0.014	**0.064**
128	0.740	0.333	**0.757**	0.096	0.014	**0.098**
256	**0.825**	0.333	0.792	0.106	0.014	**0.120**
	CIFAR100-20			**CIFAR10**		
#Bits	sblsh	l_2	S2JSD-LSH	sblsh	l_2	S2JSD-LSH
8	0.058	0.058	**0.059**	0.086	0.104	**0.146**
16	**0.072**	0.058	0.066	0.122	0.104	**0.160**
32	0.094	0.058	**0.112**	**0.222**	0.104	0.212
64	0.126	0.058	**0.138**	0.228	0.104	**0.246**
128	0.166	0.058	**0.170**	0.294	0.104	**0.310**
256	0.240	0.058	**0.258**	0.374	0.104	**0.380**
	MNIST			**COVTYPE**		
#Bits	sblsh	l_2	S2JSD-LSH	sblsh	l_2	S2JSD-LSH
8	0.130	0.104	**0.246**	0.332	0.054	**0.364**
16	0.370	0.104	**0.506**	0.462	0.054	**0.472**
32	0.596	0.104	**0.600**	**0.550**	0.054	0.514
64	**0.760**	0.104	0.748	0.658	0.054	**0.674**
128	**0.882**	0.104	0.876	0.804	0.054	**0.848**
256	0.902	0.104	**0.922**	0.858	0.054	**0.889**
	Y_Ans			**TW**		
#Bits	sblsh	l_2	S2JSD-LSH	sblsh	l_2	S2JSD-LSH
8	**0.648**	0.612	0.639	0.530	0.520	**0.530**
16	0.681	0.612	**0.697**	0.603	0.520	**0.628**
32	0.763	0.612	**0.795**	**0.671**	0.520	0.655
64	0.788	0.612	**0.807**	0.713	0.520	**0.748**
128	0.826	0.612	**0.838**	0.796	0.520	**0.819**
256	0.894	0.612	**0.923**	0.880	0.520	**0.907**

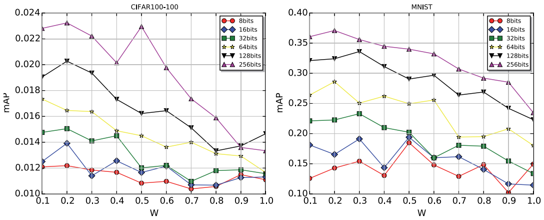

Figure 2: Impact of parameter W in S2JSD-LSH over CIFAR100-100 and MNIST datasets at code size of 8, 16, 32, 64, 128 and 256 bits.

Parameter W

Figure 2 shows the effect of the partition interval W in S2JSD-LSH hash functions (Eq.(11)) at different code size on the **CIFAR100-100** and **MNIST**. As we can see, the trend of mAP values decreases when W changes from 0.1 to 1.0, and our method can achieve the best accuracy synthetically when W = 0.2 on both datasets. Similar trends have been observed over other datasets. In the following experiments, we set parameter W = 0.2 for S2JSD-LSH.

Experimental Results

The mAP values for different methods with different code sizes on eight datasets are shown in Table 1. The value of each entry in the tables is the mAP of a combination of a method under a specific code size. The best mAP among *sblsh*, l_2 and *S2JSD-LSH* under the same setting is shown in bold face. From the table, we can make following observa-

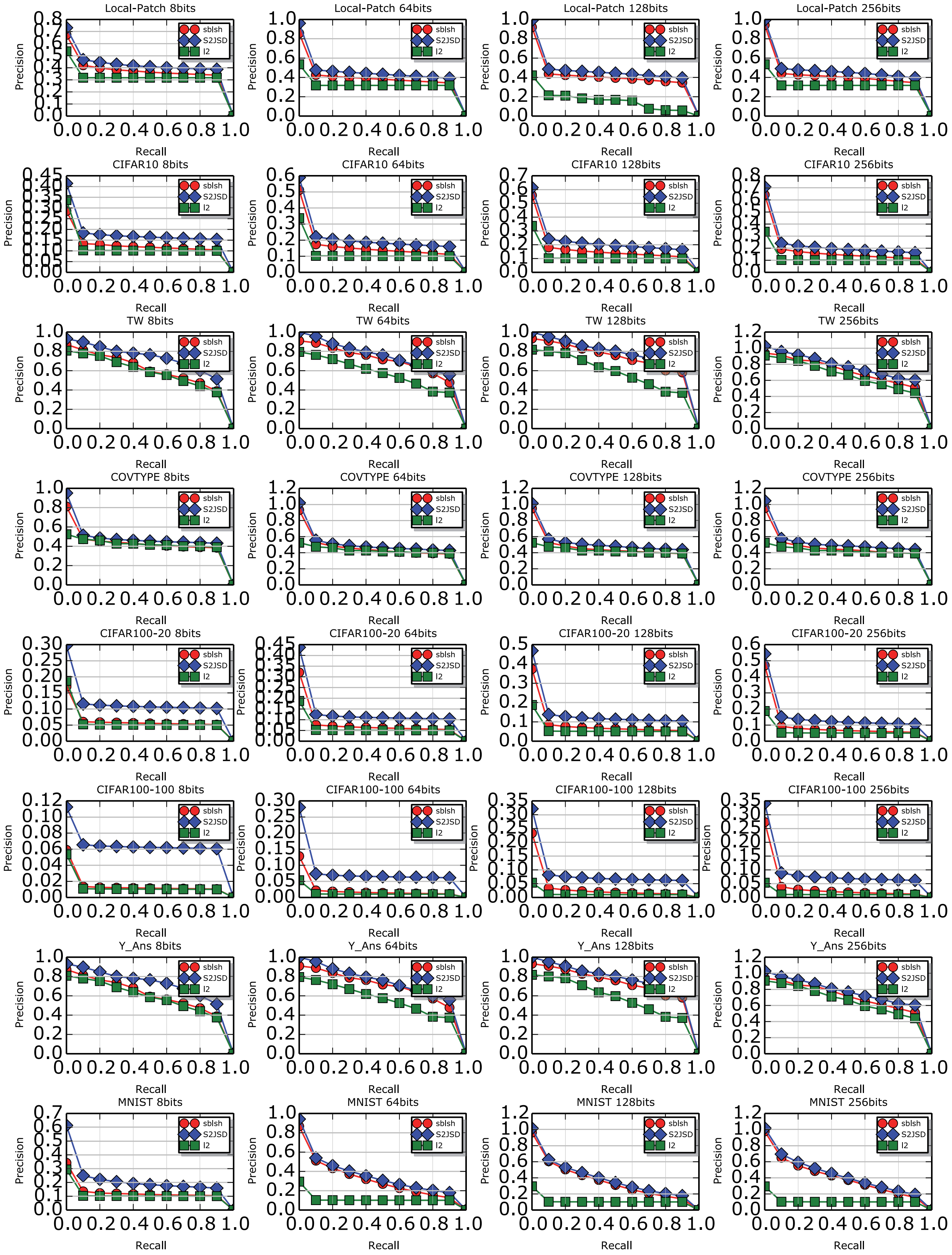

Figure 3: Precision-recall curve on eight data sets.

tion: For all datasets, *S2JSD-LSH* performs better than $sblsh$ and l_2 under most settings, which shows that the proposed method is effective. For example, for $sblsh$ over CIFAR10 dataset, mAP at 128 bits is 0.1391, and mAP at 256 bits is 0.1413; Meanwhile, for *S2JSD-LSH*, mAP at 128 bits and 256 bits are 0.1453 and 0.1430 respectively. Also, for $sblsh$ over Y_Ans dataset, mAP at 8 bits is 0.8793, and mAP at 64 bits is 0.8846; Meanwhile, for *S2JSD-LSH*, mAP at 8 bits and 64 bits are 0.9031 and 0.9238 respectively. Obviously, the mAP values of *S2JSD-LSH* are better than the corresponding ones of *sblsh*. This shows that *S2JSD-LSH* is effective to capture the similarity information in probability-distribution-type data.

Many applications such as search engines only care about the correctness of the top-n results, and p@N is a common measure for it. The p@5 values for different methods with different code sizes on eight datasets are shown in Table 2, and we can obtain similar conclusions with mAP values in Table 1. The best p@5 among *sblsh*, l_2 and *S2JSD-LSH* under the same setting is shown in bold face. In Table 2, we have observed: For all datasets, *S2JSD-LSH* performs better than $sblsh$ and l_2 under most settings, which shows that the proposed hashing schema is effective. For example, for $sblsh$ over COVTYPE dataset, p@5 at 128 bits is 0.804, and p@5 at 256 bits is 0.858; Meanwhile, for *S2JSD-LSH*, p@5 at 128 bits and 256 bits are 0.848 and 0.889 respectively. Also, for $sblsh$ over TW dataset, p@5 at 128 bits is 0.796, and p@5 at 256 bits is 0.880; Meanwhile, for *S2JSD-LSH*, p@5 at 128 bits and 256 bits are 0.819 and 0.907 respectively. Obviously, the mAP values of *S2JSD-LSH* are better than the corresponding ones of $sblsh$. Moreover, p@10, p@15 and p@20 values have similar trends, which are omitted for space saving.

Note that the mAP and p@5 values for l_2 do not vary with different hash bits on each dataset. After checking the hash codes generated by l_2, we found nearly all hash codes are the same, which means the hashing method l_2 lacks the ability to distinguish probability-distribution-type data.

Figure 3 shows the precision-recall curves for code sizes ranging from 8 bits to 256 bits on eight datasets. Once again, we can easily find that proposed method *S2JSD-LSH* significantly outperforms other state-of-the-art methods.

Conclusion

The existing data-independent hashing methods mainly focus on vector-type and set-type data. In this paper, we investigate the practicability of hashing methods for probability distributions, and propose a novel S2JSD-distance based hashing schema by introducing a new approximation formula for S2JSD-distance. The experiments show that proposed algorithms are more effective than the state-of-the-art baselines.

Acknowledgments

This work was supported by 863 Program (2015AA015404), China National Science Foundation (61402036, 60973083, 61273363), Science and Technology Planning Project of Guangdong Province (2014A010103009, 2015A020217002), Guangzhou Science and Technology Planning Project (201604020179).

References

Andoni, A., and Indyk, P. 2006. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, 459–468. IEEE.

Andoni, A., and Razenshteyn, I. 2015. Optimal data-dependent hashing for approximate near neighbors. *arXiv preprint arXiv:1501.01062*.

Andoni, A.; Indyk, P.; Nguyen, H. L.; and Razenshteyn, I. 2014. Beyond locality-sensitive hashing. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, 1018–1028. SIAM.

Broder, A. Z.; Glassman, S. C.; Manasse, M. S.; and Zweig, G. 1997. Syntactic clustering of the web. *Computer Networks and ISDN Systems* 29(8):1157–1166.

Charikar, M. S. 2002. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, 380–388. ACM.

Chen, C.; Buntine, W.; Ding, N.; Xie, L.; and Du, L. 2015. Differential topic models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 37(2):230–242.

Datar, M.; Immorlica, N.; Indyk, P.; and Mirrokni, V. S. 2004. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry*, 253–262. ACM.

Endres, D. M., and Schindelin, J. E. 2003. A new metric for probability distributions. *IEEE Transactions on Information theory*.

Eshghi, K., and Rajaram, S. 2008. Locality sensitive hash functions based on concomitant rank order statistics. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 221–229. ACM.

Gorisse, D.; Cord, M.; and Precioso, F. 2012. Locality-sensitive hashing for chi2 distance. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 34(2):402–409.

Heo, J.-P.; Lee, Y.; He, J.; Chang, S.-F.; and Yoon, S.-E. 2015. Spherical hashing: binary code embedding with hyperspheres. *IEEE transactions on pattern analysis and machine intelligence* 37(11):2304–2316.

Indyk, P., and Motwani, R. 1998. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, 604–613. ACM.

Jain, P.; Vijayanarasimhan, S.; and Grauman, K. 2010. Hashing hyperplane queries to near points with applications to large-scale active learning. In *Advances in Neural Information Processing Systems*, 928–936.

Ji, J.; Li, J.; Yan, S.; Zhang, B.; and Tian, Q. 2012. Super-bit locality-sensitive hashing. In *Advances in Neural Information Processing Systems*, 108–116.

Ji, J.; Li, J.; Yan, S.; Tian, Q.; and Zhang, B. 2013. Min-max hash for jaccard similarity. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, 301–309. IEEE.

Karpathy, A., and Fei-Fei, L. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the CVPR*, 3128–3137.

Krstovski, K.; Smith, D. A.; Wallach, H. M.; and McGregor, A. 2013. Efficient nearest-neighbor search in the probability simplex. In *Proceedings of the 2013 Conference on the Theory of Information Retrieval*, 22. ACM.

Kulis, B., and Grauman, K. 2012. Kernelized locality-sensitive hashing. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 34(6):1092–1104.

Li, P.; Konig, A.; and Gui, W. 2010. b-bit minwise hashing for estimating three-way similarities. In *Advances in Neural Information Processing Systems*, 1387–1395.

Li, P.; Owen, A.; and Zhang, C.-H. 2012. One permutation hashing. In *Advances in Neural Information Processing Systems*, 3113–3121.

Liu, W.; Mu, C.; Kumar, S.; and Chang, S.-F. 2014a. Discrete graph hashing. In *Advances in Neural Information Processing Systems*, 3419–3427.

Liu, Y.; Cui, J.; Huang, Z.; Li, H.; and Shen, H. T. 2014b. Sk-lsh: an efficient index structure for approximate nearest neighbor search. *Proceedings of the VLDB Endowment* 7(9):745–756.

Mu, Y., and Yan, S. 2010. Non-metric locality-sensitive hashing. In *AAAI*.

ODonnell, R.; Wu, Y.; and Zhou, Y. 2014. Optimal lower bounds for locality-sensitive hashing (except when q is tiny). *ACM Transactions on Computation Theory (TOCT)* 6(1):5.

Ramage, D.; Hall, D.; Nallapati, R.; and Manning, C. 2009. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, 248–256. Association for Computational Linguistics.

Terasawa, K., and Tanaka, Y. 2007. Spherical lsh for approximate nearest neighbor search on unit hypersphere. In *Algorithms and Data Structures*. Springer. 27–38.

Wang, J.; Shen, H. T.; Song, J.; and Ji, J. 2014. Hashing for similarity search: A survey. *arXiv preprint arXiv:1408.2927*.

Wang, J.; Liu, W.; Kumar, S.; and Chang, S.-F. 2016. Learning to hash for indexing big data-a survey. *Proceedings of the IEEE* 104(1):34–57.

Zhang, T.; Qi, G.-J.; Tang, J.; and Wang, J. 2015. Sparse composite quantization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4548–4556.

Zhao, W.-L.; Jégou, H.; and Gravier, G. 2013. Sim-min-hash: An efficient matching technique for linking large image collections. In *Proceedings of the 21st ACM international conference on Multimedia*, 577–580. ACM.

Zhen, Y.; Gao, Y.; Yeung, D.-Y.; Zha, H.; and Li, X. 2016. Spectral multimodal hashing and its application to multimedia retrieval. *IEEE transactions on cybernetics* 46(1):27–38.