Diffusion Hashing

Atsushi Tatsuma* and Masaki Aono*
* Toyohashi University of Technology, Aichi, Japan
E-mail: aono@tut.jp Tel: +81-532-44-6764

***Abstract*—With the worldwide spread of the broadband Internet, massive multimedia data including texts, images, and videos are increasing explosively and available for interactive applications over the Internet. At the same time, more and more attention has been paid to aiming at fast retrieval from massive multimedia databases. Hash-based Approximate Nearest Neighbor (ANN) search is a technology that achieves fast retrieval by regarding the hash key as a retrieval index, where the similarity of data is maintained and embedded in the neighborhood of the hash key. In other words, the closer the Hamming codes between hash keys, the more similar the data become. In general, short binary codes are preferred for storing hash keys and values. The difficulty is to define the similarity between data and reflect it in binary codes. In this paper, we propose Diffusion Hashing (DH) as a novel ANN search technique based on hashing with an anisotropic diffusion kernel. DH aims to transform the search index into as short binary codes as possible, preserving the similarity induced by random walk on the data manifold in higher dimensional space. From comparative experiments, we will demonstrate that DH outperforms previously known hash-based ANN search techniques including Locality Sensitive Hashing and Spectral Hashing.**

I. Introduction

We have observed massive multimedia data, such as documents, images, sounds, and videos almost ubiquitously available on the Internet. To take advantage of these massive data, attention has been paid to fast similarity search techniques in many research fields, including computer vision and text mining.

Generally speaking, the dimension of feature vectors representing documents and images could be extremely large, ranging from several hundred to several hundreds of thousands. If we construct databases of such media for information retrieval, a linear search will become impracticable if feature vectors are very high dimensional. Recently, to cope with this problem, approximate nearest neighbor search algorithms have been investigated.

Approximate Nearest Neighbor (ANN) search technologies can be roughly divided into two types: Tree structure based [1], [2], [18] and hash-based [5], [8], [22]. Tree structure based ANN search iteratively subdivides the feature space to produce tree data structures, aiming at fast retrieval to narrow down the range of search when retrieving data. The range of search is determined by a hypersphere having its radius defined by the distance between a query and the tree structure with a predetermined tolerance, subject to the curse of dimensionality when the dimension of data becomes very high. On the other hand, hash-based ANN search transforms high dimensional data into short binary codes to be used for hash keys, which results in fast retrieval. It is possible to suppress the curse of dimensionality by transforming the data so that the Hamming distance between binary codes is minimized. Transforming into short binary codes also makes it possible to compress the storage necessary for search index.

Locality Sensitive Hashing (LSH) [5], [8], [9] is one of the well-known algorithms for hash-based ANN search. LSH embeds the high dimensional vector data into lower dimensional space by means of random projection such that two arbitrary data with smaller distances in feature space tend to have similar binary codes to each other. Given n number of vector data $\{\mathbf{x}_i\}_{i=1}^n \in \mathbb{R}^d$ in the database, the hash keys are defined by hash function $\{h_i\}_{i=1}^k$ that produces k-bit binary codes. It should be noted that hash function h must meet the following characteristic [5]:

$$\Pr[h(\mathbf{x}_i) = h(\mathbf{x}_j)] = \text{sim}(\mathbf{x}_i, \mathbf{x}_j),$$

where $\text{sim}(\mathbf{x}_i, \mathbf{x}_j) \in [0, 1]$ is a similarity function between two vectors. The data similar to each other in the retrieved database have the possibility of collision into an identical hash key. Charikar et al considered the similarity defined by inner product $\text{sim}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$, and proposed a hash function as the product of two vectors on a d-dimensional random hyperplane, which conforms to standard normal distribution $\mathcal{N}(0, I)$ [5]. The hash function itself takes the value of either 0 or 1 as below:

$$h(\mathbf{x}) = \begin{cases} 1 & \mathbf{r}^T \mathbf{x} \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

For k number of random vectors $\mathbf{r} \in \mathbb{R}^d$, k number of hash functions are defined, corresponding to $\mathbf{r}$. Kulis et al extended the idea of LSH by allowing non-linear mapping $\Phi(\mathbf{x})$ for the similarity computation by inner product $\text{sim}(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j)$, and proposed Kernelized Locality Sensitive Hashing (KLSH) [11]. The basic idea of KLSH is to approximate random vectors $\mathbf{r}$, conforming to standard normal distribution $\mathcal{N}(0, I)$, from the subset of a given database. The datum $\Phi(\mathbf{x})$ is considered to be distributed in the database, having mean μ and covariance Σ. By defining vectors $\mathbf{z} = \frac{1}{t} \sum_{i \in S} \Phi(\mathbf{x}_i)$ consisting of the subset S with t number of data, it turns out that $\tilde{\mathbf{z}} = \sqrt{t}(\mathbf{z} - \mu)$ conforms to Gaussian distribution $\mathcal{N}(0, \Sigma)$. It is also shown that $\Sigma^{-1/2}\tilde{\mathbf{z}}$, obtained by whitening the above vectors, obeys the Gaussian distribution $\mathcal{N}(0, I)$. Replacing the random vector $\mathbf{r}$ by $\Sigma^{-1/2}\tilde{\mathbf{z}}$, hash functions in non-linear transformation mapping space is defined as:

$$h(\Phi(\mathbf{x})) = \begin{cases} 1 & \Phi(\mathbf{x})^T \Sigma^{-1/2} \tilde{\mathbf{z}} \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

It was shown that KLSH exhibited higher precision than LSH from the experiments using an image database [11].

Semantic Hashing produces binary codes using the network structure defined by multiple Restricted Boltzmann Machines (RBM), in a configuration with gradually decreasing unit numbers, stage by stage [17]. Torralba et al applied the Semantic Hashing to the content-based image retrieval (CBIR) and obtained higher precision than the one obtained by LSH [19]. Spectral Hashing produces binary codes such that the sum of Hamming distances between binary codes, weighted by Gaussian kernel in feature space, takes the minimum value [22]. Spectral Hashing achieved higher precision through experiments than the methods based on RBM and Boosting [19], assuming that vector data are distributed uniformly. Raginsky et al [15] proposed a method independent of data distribution, by using Random Fourier features [16]. Further examples include frameworks based on sequential projection learning [20] and semi-supervised hashing [21].

In this paper, we propose Diffusion Hashing (DH), a novel hash-based ANN search algorithm, which transforms vector data into binary codes. DH produces binary codes able to capture non-linear structure of vector data, based on the similarity relationship represented by transition probability given by random walk on the data manifold in a high dimensional feature space. DH minimizes the effect of biased distribution of vector data by means of anisotropic diffusion kernel [6], and determines the weights between data accordingly. Specifically, the weights are derived from a network structure for the entire database, by assuming the nodes as data objects, and the arcs as the similarity or the closeness between two data. Through experiments using document collections and image data benchmarks, we will demonstrate that DH outperforms previously known hash-based ANN search methods including Spectral Hashing (SH), Locality Sensitive Hashing (LSH), Kernelized Locality Sensitive Hashing (KLSH), and Shift Invariant Kernel Hashing (SIKH) especially when the number of bits in hash keys is small.

In the remainder of this paper, we will describe our proposed algorithm and the fundamental principles behind DH in Section 2. In Section 3, we will show the results of comparative experiments by taking four different data sets including documents collections and image data benchmarks. Finally, we will conclude our approach and discuss potential areas for future investigation in Section 4.

II. Binary codes for ANN search

In this section, we will first review and formulate a recent approach to binary code generation by hash functions for ANN search, based on manifold learning. Then, we will present our novel approach.

A. Manifold learning-based approach

It is vital to develop a method of producing as short binary codes as possible for high dimensional vector data in hash-based ANN search. Stated mathematically, given n number of d-dimensional vector data $X = \{\mathbf{x}_i\}_{i=1}^n \in \mathbb{R}^{d \times n}$, the objective is to find a hash function $\{h_i\}_{i=1}^k$ that transforms vector data, keeping the similarity relationship in high dimension, into k-dimensional binary codes $\{\mathbf{y}_i\}_{i=1}^n \in \mathbb{R}^{k \times n}$.

Spectral Hashing (SH) proposed by Weiss et al attempts to find effective binary codes for hash-based ANN search, by applying a balanced graph partitioning problem [22]. They considered the minimization problem to keep the similarity relationship in higher dimension defined by Gaussian kernel $W_{i,j} = \exp(-||\mathbf{x}_i - \mathbf{x}_j||^2/\sigma^2)$.
Objective function:

$$\sum_{i,j} ||\mathbf{y}_i - \mathbf{y}_j||^2 W_{i,j}$$

Three constraints:

$$\mathbf{y}_i = [y_1, y_2, \ldots, y_k] \in \{-1, 1\}^k$$

$$\sum_i \mathbf{y}_i = 0$$

$$\frac{1}{n} \sum_i \mathbf{y}_i \mathbf{y}_i^T = I$$

The first constraint states that each binary code takes the value either 1 or -1, the second constraint states that binary codes are uniformly sampled without deviation, while the third constraint demonstrates the independence between different binary codes. By relaxing the first constraint, Weiss et al simplified the minimization problem to a matter of solving the eigenvalue problem of Laplacian matrix $L = D - W$, defined by Gaussian kernel matrix W and diagonal matrix $D_{i,i} = \sum_j W_{i,j}$ [22]. Hash functions of SH are defined by thresholding eigenvectors $\{\mathbf{\Psi}_i\}_{i=1}^k \in \mathbb{R}^n$, corresponding to k-smallest eigenvalues, eliminating the ones taking 0 eigenvalues of Laplacian matrix L.

$$h_i(\mathbf{x}_j) = \begin{cases} 1 & y_i = \mathbf{\Psi}_i(j) \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

The first constraint is equivalent to the minimization problem of Laplacian Eigenmaps (LE) [3] classified into Manifold Learning. Manifold Learning is considered to be a non-linear dimensional reduction method by estimating a manifold structure in low dimensional space, embedded into high dimensional space [4]. SH can be regarded as thresholding vector data of reduced dimension by LE to produce binary codes. Belkin et al showed that the Laplacian matrix was to be an approximation to Laplace-Beltrami operator on the manifold, assuming that data distribution is uniform in their research on LE [3]. However, in real data, since it is often the case that uniform distribution cannot be assumed, and Laplacian matrix cannot be guaranteed to be an approximation of the Laplace-Beltrami operator, we might fail to estimate manifold structure. Since SH assumes uniform distribution analogously with LE, it is inferred that SH has a similar difficulty in maintaining the similarity relationship between vector data in high dimensional space, when projected into binary codes in lower dimensional space. On the other hand, the Diffusion Map proposed by Coifman et al [6] can cope with a similar minimization problem. It is also possible to estimate manifold structure by utilizing anisotropic diffusion kernel, which is shown to be an approximation to Laplace-Beltrami operator, even when data are not uniformly distributed.

B. Diffusion Hashing

We propose Diffusion Hashing (DH) to maintain the similarity relationship, defined by random walk in higher dimensional space, when projected by hash functions into binary codes in lower dimensional space. DH inherits salient features derived from the minimization problem for finding binary codes by SH. It also attempts to decrease the effect of data distribution in high dimensional space by taking advantage of the anisotropic diffusion kernel.

We will relax the first constraint defined in the previous section so that the hash function can take an arbitrary number. Consider a linear transformation given by transformation matrix $F \in \mathbb{R}^{d \times k}$, in order to make it easier to obtain binary codes for unknown data:

$$\mathbf{y}_i = F^T(\mathbf{x}_i - \bar{\mathbf{x}})$$

It should be noted that we subtract the average vector $\bar{\mathbf{x}}$ from the original vector $\mathbf{x}$ in order to satisfy the second constraint:

$$\bar{\mathbf{x}} = \frac{1}{n}\sum_i \mathbf{x}_i$$

In SH, Weiss et al employed the Gaussian kernel, which was subject to the bias of data distribution, and also subject to the difficulty in coping with non-linear behavior of data in higher dimension. We have adopted an anisotropic diffusion kernel [6] to overcome this problem:

$$K = Q^{-1}WQ^{-1}$$

$$W_{ij} = \exp\left(\frac{-||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma^2}\right)$$

$$Q_{ii} = \sum_j W_{ij}$$

The diagonal matrix Q_{ii} has a characteristic by which the element becomes smaller when the data distribution is sparse, whereas it becomes larger when the distribution is dense. By normalizing the data with the reciprocal of these elements in Q_{ii}, we expect to cope with a variety of data distributions. It is also possible to obtain the transition probability matrix from matrix K, if we normalize it such that the sum of each column becomes 1:

$$P = D^{-1}K$$

$$D_{ii} = \sum_j K_{ij}$$

P_{ij} represents the transition probability from point $\mathbf{x}_i$ to point $\mathbf{x}_j$. By modifying the original objective function to have its weight replaced by transition probability matrix P, it is possible to make binary codes reflect the properties induced by random walk in feature space. Let $\mathbf{y} = (y_1, y_2, \ldots, y_n)^T$ be such space.

$$\sum_{i,j}(y_i - y_j)^2 P_{ij}$$

The objective function with our choice of weights P_{ij} incurs a heavy penalty of high transition probability points $\mathbf{x}_i$ and $\mathbf{x}_j$ are mapped far apart. Therefore, minimizing it is an attempt to ensure that if $\mathbf{x}_i$ and $\mathbf{x}_j$ are high transition probability then y_i and y_j are close. Suppose $\mathbf{f}$ is a transformation vector, that is, $\mathbf{y}^T = \mathbf{f}^T X$, where the ith column vector of X is $\mathbf{x}_i$. Our proposed objective function can be expanded per below:

$$\begin{aligned}E =& \frac{1}{2}\sum_{i,j}(y_i - y_j)^2 P_{ij} \\ =& \frac{1}{2}\sum_{i,j}(\mathbf{f}^T\mathbf{x}_i - \mathbf{f}^T\mathbf{x}_j)^2 P_{ij} \\ =& \sum_i \mathbf{f}^T\mathbf{x}_i\mathbf{x}_i^T\mathbf{f} - \sum_i \mathbf{f}^T\mathbf{x}_i P_{ij}\mathbf{x}_j^T\mathbf{f} \\ =& \mathbf{f}^T X(I - P)X^T\mathbf{f}\end{aligned}$$

The third constraint, shown before, is rephrased as follows:

$$\mathbf{y}^T\mathbf{y} = \mathbf{f}^T XX^T\mathbf{f} = 1$$

The minimization of the objective function is therefore given as below:

$$\underset{\substack{\mathbf{f} \\ \mathbf{f}^T XX^T\mathbf{f}=1}}{\operatorname{argmin}} \ \mathbf{f}^T X(I - P)X^T\mathbf{f}$$

By applying Lagrange's multiplier, we obtain

$$\mathcal{L}(\mathbf{f}) = \mathbf{f}^T X(I - P)X^T\mathbf{f} + \lambda(1 - \mathbf{f}^T XX^T\mathbf{f})$$

Taking a partial derivative and letting it be naught,

$$\frac{\partial\mathcal{L}}{\partial\mathbf{f}} = 2X(I - P)X^T\mathbf{f} - 2\lambda XX^T\mathbf{f} = 0$$

Finally, we obtain the following generalized eigenvalue problem:

$$X(I - P)X^T\mathbf{f} = \lambda XX^T\mathbf{f}$$

By taking $\lambda' = 1 - \lambda$, this is further reformulated as follows:

$$XPX^T\mathbf{f} = \lambda' XX^T\mathbf{f} \tag{1}$$

To find stable solutions to the generalized eigenvalue problem in (1), matrix XX^T must be non-singular. If the dimension of vector data is larger than the number of samples, matrix XX^T could be singular. If this happens, we employ singular value decomposition to project subspace whose rank is the same as the rank of the matrix:

$$X = U\Sigma V^T$$

$$\tilde{X} = U^T X = \Sigma V^T,$$

where Σ is a diagonal matrix of r by r, whose elements are singular values $s_1 \geq ... \geq s_r$, and U and V are orthogonal matrices of size $d \times r$ and $n \times r$, respectively:

$$\tilde{X} P \tilde{X}^T \tilde{\mathbf{f}} = \lambda \tilde{X} \tilde{X}^T \tilde{\mathbf{f}}$$

The transformation matrix is defined as follows:

$$\mathbf{f} = U \tilde{\mathbf{f}}$$

The transformation matrix F of DH is obtained by solving the generalized eigenvalue problem of equation (1). F can be uniquely expressed by the eigenvectors corresponding to eigenvalues in the following descending order: $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_k$.

$$F = [\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_k]$$

Here we define hash functions of DH using a k-number of vectors from transformation matrix F as below:

$$y_i = \mathbf{f}_i^T (\mathbf{x} - \bar{\mathbf{x}})$$

$$h_i(\mathbf{x}) = \begin{cases} 1 & y_i \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

C. Time Complexity Analysis

DH algorithm is summarized in Figure 1. In the training phase of DH, the time complexity is $O(n^2)$ by the calculation of the Gaussian kernel between all data points in step 2. However, in the test phase of DH to calculate the binary codes of new data, the time complexity is $O(dk)$ because it is only a multiplication of the transform matrix by a data vector in step 7. Therefore, when the new data such as a search query is given, the calculation time for the binary codes is the same as LSH.

D. Relation to Diffusion Maps

As with Diffusion Maps [6] in manifold learning, Diffusion Hashing (DH) defines the similarity between vectors in high dimensional space using anisotropic diffusion kernel, and derives the low dimensional expression of vectors by estimating the manifold structure. The generalized eigenvalue problem of DH represented by (1) is deduced to the ordinary eigenvalue problem of Diffusion Maps by assuming $\psi = \mathbf{y}^T = X^T \mathbf{f}$, where matrix X is a full rank matrix:

$$X P X^T \mathbf{f} = \lambda X X^T \mathbf{f}$$

$$P X^T \mathbf{f} = \lambda X^T \mathbf{f}$$

$$P \psi = \lambda \psi$$

From these equations, if the number of samples is large enough, compared to the dimension of vector data, and if vectors are linearly independent to each other, it can be shown that the embedding by the transformation matrix of DH is similar to the embedding by Diffusion Maps. Therefore, it is conjectured that both Diffusion Maps and DH could estimate the manifold structure under the assumption of non-uniform data distribution.

```
Input: $d$-dimensional training data $X = \{\mathbf{x}_i\}_{i=1}^n \in \mathbb{R}^{d \times n}$
Output: $k$ number of hash functions $\{h_i\}_{i=1}^k$
1. Compute the mean of x
     $\bar{\mathbf{x}} = \frac{1}{n} \sum_i \mathbf{x}_i$
2. Compute Gaussian kernel $W$ between each data
     $W_{ij} = \exp(-||\mathbf{x}_i - \mathbf{x}_j||^2 / 2\sigma^2)$
3. Normalize $W$ with the diagonal matrix $Q$ whose
     element is the sum of each column
     $K = Q^{-1} W Q^{-1},\ Q_{ii} = \sum_j W_{ij}$
4. Compute transition probability matrix $P$
     $P = D^{-1} K,\ D_{ii} = \sum_j K_{ij}$
5. Compute eigenvectors $\mathbf{f}_i$ from generalized eigenvalue
     problem
     $X P X^T \mathbf{f} = \lambda X X^T \mathbf{f}$
6. Compute transformation matrix $F$ from eigenvectors
     $F = [\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_k]$
7. When a query $\mathbf{z} \in \mathbb{R}^d$ is given, search similar data
     from binary codes using hash functions with
     pre-defined $F$ and $\bar{\mathbf{x}}$
     $h_i(\mathbf{z}) = \begin{cases} 1 & \mathbf{f}_i^T(\mathbf{z} - \bar{\mathbf{x}}) \geq 0 \\ 0 & \text{otherwise} \end{cases}$
```

Fig. 1. The Diffusion Hashing (DH) algorithm of fast retrieval with massive data sets

III. Experimental Results

To confirm the effectiveness of our proposed Diffusion Hashing (DH), we first conducted experiments using document benchmark data 20-newsgroups [12] and Reuters-21578. Secondly, we employed image databases MNIST Digits and CIFAR-10 [10], and conducted comparative experiments using our methods and previously known methods. For previously known methods, we chose Locality-Sensitive Hashing (LSH) [5], Kernelized Locality-Sensitive Hashing (KLSH) [11], Spectral Hashing (SH) [22], and Shift Invariant Kernel Hashing (SIKH) [15]. For evaluation measures, we have used precision within Hamming radius 2 when varying the number of bits, and precision recall curve using 16-bit codes. Search results are determined based on the Hamming distance between a binary code of a query and the code of each object in the database.

A. 20-newsgroups

20-newsgroup consists of 18,845 news group documents gathered from Usenet newsgroups [12]. Each document is classified into one of 20 distinct news groups. Among them 11,314 news are used for training, while the remaining 7,531 news are used for testing. For the training, we randomly selected 2,000 news from the training dataset. In our experiments, we

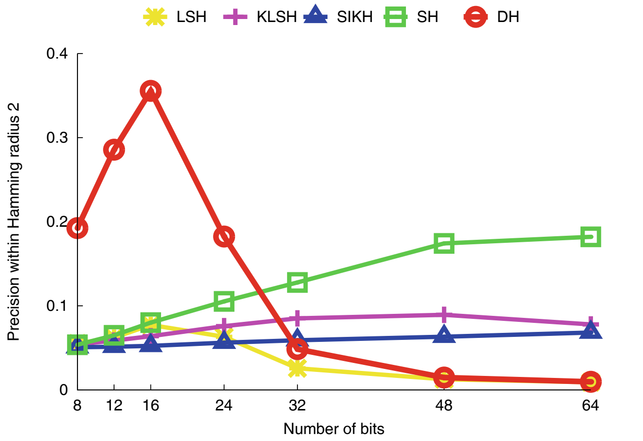

Fig. 2. Precision within Hamming radius 2 using hash lookup and the varying number of bits with 20-newsgroups data sets.

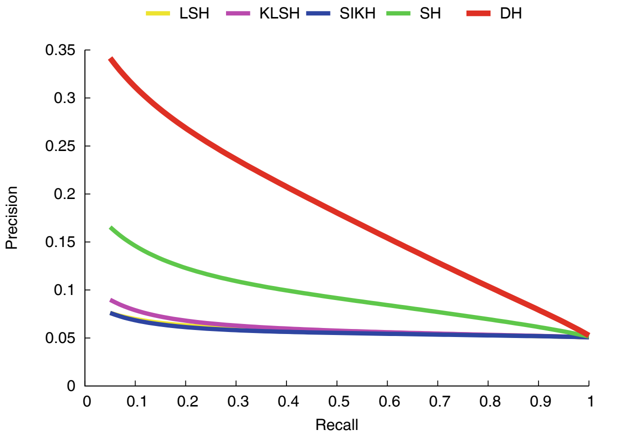

Fig. 3. Precision recall curve using 16-bit codes with 20-newsgroups data sets.

conducted word stemming [14] and stopword elimination[1] as pre-processing, followed by selecting 2,000 words from the largest document frequencies, and generated tf-idf weighted document vectors. The σ parameter to DH is set to be 80.0.

Figure 2 shows the precision using hash lookup within Hamming radius 2, by varying bit-numbers from 8 to 64. From 8 to 24 bits, DH exhibits the largest precision among the five methods. Figure 3 shows the precision recall curve using 16-bit codes. DH exhibits the largest precision over all the recall. From these two figures, it is shown that our proposed DH outperformed previously known major methods in terms of both precision and recall.

In 20-newsgroups, every news is classified into one of 20 classes. By closer examination, we could also put them into rough major groups such as IT technologies and sports. This observation leads to our guess that in high dimensional document vector space similar topical documents are distributed densely together, while isolated documents are distributed sparsely, to constitute complex structures. Since DH takes care of such a biased distribution, we consider it can handle complex document vector space appropriately as expected.

B. Reuters-21578

Reuters-21578.[2] has 21,578 articles, some of which have topic labels. In our experiments, we utilized datasets produced by ModApte-split. Eliminating documents classified into multiple classes, we have 59 classes after classification. Among these news documents, we randomly chose 2,000 documents from the training dataset. Pre-processing and generation of document vectors are the same as in the previous experiment. The σ parameter to DH is set to be 10.0.

Figure 4 shows the precision using hash lookup within Hamming radius 2, by varying bit-numbers from 8 to 64. From 8 to 16 bits, DH exhibits the highest precision among the five methods. Figure 5 shows the precision recall curve using 16-bit codes. DH exhibits the greatest precision over all the recall. This experiment demonstrates that, on average. DH outperforms the other methods in terms of both precision and recall.

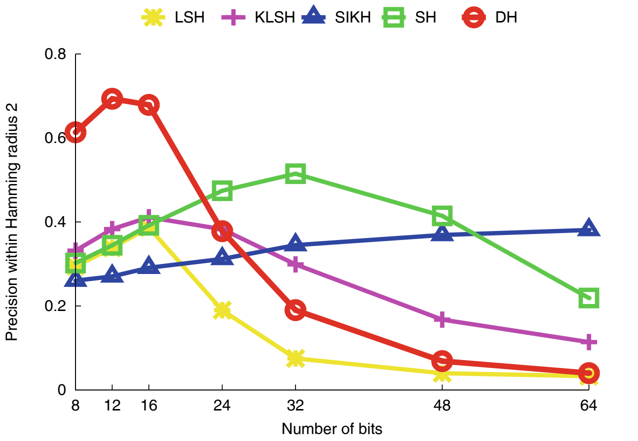

Fig. 4. Precision within Hamming radius 2 using hash lookup and the varying number of bits with Reuters-21578 ModeApte data sets.

As we pointed out, our data from Reuters-21578 were classified into 59 classes, where each class has distinctive news. We conjectured, however, that these 59 classes had some inter-dependence among them, which made it difficult to uniquely determine a class for an arbitrary news data. DH has a salient feature of taking good care of biased distribution, leading to higher precision, at the sacrifice of enumerating all the related data. This behavior might be the cause of lower recall compared to Spectral Hashing, when there is inherent inter-dependence among classes, such as 59 classes from Reuters-21578.

C. MNIST Digits

MNIST-Digit (MNIST)[3] includes 70,000 handwritten digit images from digit “0" to digit “9". 60,000 out of 70,000 images are dedicated to training, while the remaining 10,000 images

[1] http://jmlr.csail.mit.edu/papers/volume5/lewis04a/a11-smart-stop-list/english.stop

[2] http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html

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

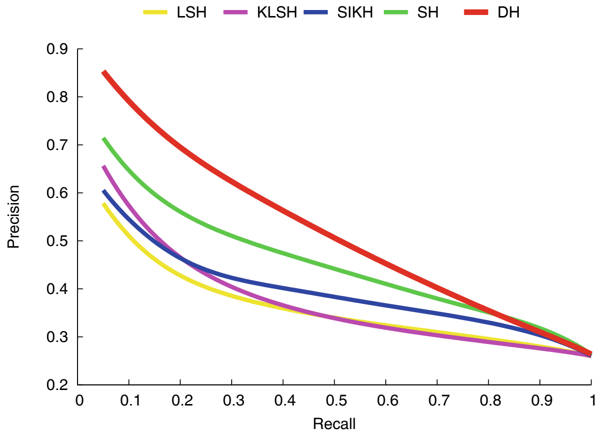

Fig. 5. Precision recall curve using 16-bit codes with Reuters-21578 ModeApte data sets.

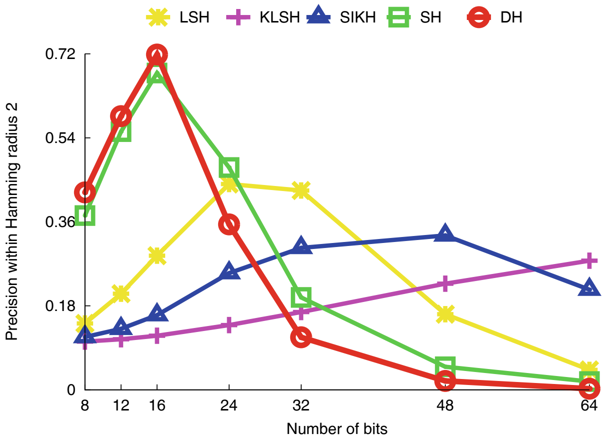

Fig. 6. Precision within Hamming radius 2 using hash lookup and the varying number of bits with MNIST data sets.

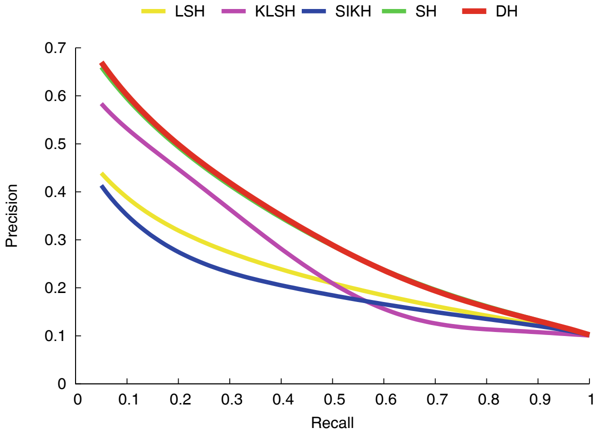

Fig. 7. Precision recall curve using 16-bit codes with MNIST data sets.

are prepared for testing. We randomly chose 2,000 data from the training data set. Since each digit image is a gray-scale image and has a pixel size of 28×28, we can assume that the image consists of 784 vector data. The σ parameter to DH is set to be 15.0.

Figure 6 shows the precision using hash lookup within Hamming radius 2, by varying bit-numbers from 8 to 64. From 8 to 16 bits, DH exhibits the largest precision among the five methods. Figure 7 shows the precision recall curve using 16-bit codes. DH is slightly superior to SH in precision over all the recall. In the feature vector comprised of pixel values, the search precision of DH is equal to SH.

It should be noted that the hand-written digit images have apparent visual similarity between digit-3 and digit-8 images, for instance. These data tend to stick closer together feature space, while visually different images tend to become more separated from each other, distributing with a combination of sparser and denser regions in feature space. Since DH naturally makes it possible to cope with biased distribution in feature space by means of transition probability to generate binary hash codes, we consider that DH generally has captured latent clusters more precisely.

D. CIFAR-10

CIFAR-10[4] consists of 60,000 color images of 32×32 pixel resolution with 10 class labels including airplanes, automobiles, dogs, and horses. Among them, 50,000 are data for training, while the remaining 10,000 are data for testing. We randomly chose 2,000 data for training. In our experiments, we extracted GIST feature vectors [13] having $3 \times 4 \times 4 \times 6 \times 4 = 1,152$ dimensions, and converted them to binary codes, where each RGB pixel is pre-processed by 4 scales in 6 directions within a 4×4 region. The parameter σ of DH was set to be 3.0.

Figure 8 shows the precision using hash lookup within Hamming radius 2, by varying bit-numbers from 8 to 64. From 8 to 16 bits, DH exhibits the largest precision among the five methods. Figure 9 shows the precision recall curve using 16-bit codes. DH exhibits the largest precision throughout the recall. From these two figures, it is shown that our proposed DH outperformed previously known major methods in terms of both precision and recall.

CIFAR-10 roughly consists of two types of images: images of vehicles such as airplanes and automobiles, and images of living animals such as dogs and deer. We conjecture that in feature space, dense and sparse regions are intermixed without clearly discriminating the border between the two types of images, i.e., vehicles and living animals, partly because specific colors are frequently used in both types of images as a whole, and partly because the background is often indistinguishable in both types. Since DH can estimate non-linear structures in feature vector space, capable of representing biased distribution, we find that these complex inter-dependences can be well captured by DH.

IV. Conclusion

We have proposed Diffusion Hashing (DH), as a novel algorithm for hash-based ANN search. DH has successfully

[4] http://www.cs.utoronto.ca/ kriz/cifar.html

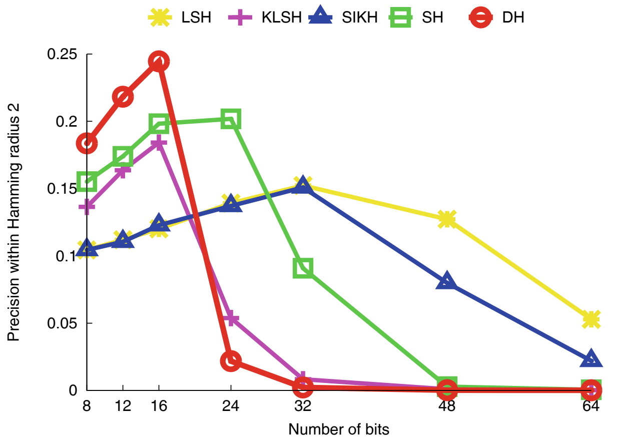

Fig. 8. Precision within Hamming radius 2 using hash lookup and the varying number of bits with CIFAR-10 data sets.

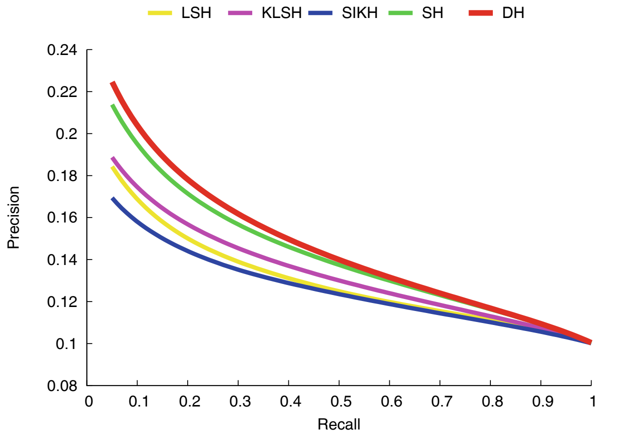

Fig. 9. Precision recall curve using 16-bit codes with CIFAR-10 data sets.

proved to capture non-linear structure of vector data by dimensionality reduction, represented by shorter binary codes, based on random walk in feature space. From our comparative experiments using a couple of document collections and image data sets, we have shown that DH outperformed previous methods including Spectral Hashing and Locality Sensitive Hashing, in terms of precision, especially when the number of bits for representing hash keys is small. DH also has a salient feature of being capable of representing complex high dimensional data with much smaller bits than conventional methods, which served as compact search indices.

DH has an internal parameter σ, which is subject to the distribution of data, and must be adaptively determined. It is an open problem to automatically adjust this parameter. It is also important to seek applications other than multimedia retrieval As a method for computing recommendation score for LSH has been proposed [7], information recommendation might be another application of DH to investigate further.

ACKNOWLEDGMENTS

This work was partially supported by the Ministry of Education, Culture, Sports, Science and Technology, for JSPS Fellows. The research was conducted by the Strategic Information and Communication R&D Promotion Programme (SCOPE 112306001) of Ministry of Internal Affairs and Communications (MIC) Japan.

REFERENCES

[1] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. ACM*, 45:891–923, November 1998.

[2] J. S. Beis and D. G. Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *Proc. IEEE Conf. Comp. Vision Patt. Recog*, pages 1000–1006, 1997.

[3] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *J. Neural Computation*, 15:1373–1396, 2002.

[4] L. Cayton. Algorithm for manifold learning. Technical report, Computer Science and Engineering Department, University of California at San Diego, 2008.

[5] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proc. of 34th STOC*, pages 380–388. ACM, 2002.

[6] R. R. Coifman and S. Lafon. Diffusion maps. *J. Applied and Computational Harmonic Analysis*, 21:5–30, July 2006.

[7] A. S. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *Proc. of the 16th international conference on World Wide Web*, WWW '07, pages 271–280, New York, NY, USA, 2007. ACM.

[8] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proc. of the 25th International Conference on Very Large Data Bases*, VLDB '99, pages 518–529, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[9] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proc. of the 30th Annual ACM Symposium on Theory of Computing*, STOC '98, pages 604–613, New York, NY, USA, 1998. ACM.

[10] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Computer Science Department, University of Toronto, 2009.

[11] B. Kulis and K. Grauman. Kernelized locality-sensitive hashing for scalable image search. In *Proc. of IEEE International Conference on Computer Vision (ICCV)*, 2009.

[12] K. Lang. Newsweeder: Learning to filter netnews. In *Proc. of the 12th International Machine Learning Conference (ML95*, 1995.

[13] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *J. International Journal of Computer Vision*, 42:145–175, 2001.

[14] M. F. Porter. *An algorithm for suffix stripping*, pages 313–316. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997.

[15] M. Raginsky and S. Lazebnik. Locality-sensitive binary codes from shift-invariant kernels. In *Proc. of Neural Information Processing Systems*, 2009.

[16] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Proc. of Neural Information Processing Systems*, 2007.

[17] R. Salakhutdinov and G. Hinton. Semantic hashing. *J. Approx. Reasoning*, 50:969–978, July 2009.

[18] C. Silpa-Anan and R. Hartley. Optimised kd-trees for fast image descriptor matching. In *Proc. of Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[19] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *Proc. of Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[20] J. Wang and S. fu Chang. Sequential projection learning for hashing with compact codes. In *Proc. of International Conference on Machine Learning*, pages 1127–1134, 2010.

[21] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for scalable image retrieval. In *Proc. of Computer Vision and Pattern Recognition*, pages 3424–3431, 2010.

[22] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Proc. of Neural Information Processing Systems*, 2008.