Supervised Discrete Hashing With Relaxation

Jie Gui, *Senior Member, IEEE*, Tongliang Liu, Zhenan Sun, *Member, IEEE*,
Dacheng Tao, *Fellow, IEEE*, and Tieniu Tan, *Fellow, IEEE*

***Abstract*— Data-dependent hashing has recently attracted attention due to being able to support efficient retrieval and storage of high-dimensional data, such as documents, images, and videos. In this paper, we propose a novel learning-based hashing method called "supervised discrete hashing with relaxation" (SDHR) based on "supervised discrete hashing" (SDH). SDH uses ordinary least squares regression and traditional zero-one matrix encoding of class label information as the regression target (code words), thus fixing the regression target. In SDHR, the regression target is instead optimized. The optimized regression target matrix satisfies a large margin constraint for correct classification of each example. Compared with SDH, which uses the traditional zero-one matrix, SDHR utilizes the learned regression target matrix and, therefore, more accurately measures the classification error of the regression model and is more flexible. As expected, SDHR generally outperforms SDH. Experimental results on two large-scale image data sets (CIFAR-10 and MNIST) and a large-scale and challenging face data set (FRGC) demonstrate the effectiveness and efficiency of SDHR.**

***Index Terms*— Data-dependent hashing, least squares regression, supervised discrete hashing (SDH), supervised discrete hashing with relaxation (SDHR).**

I. Introduction

IN LARGE-SCALE visual searching, data must be indexed and organized accurately and efficiently. Hashing has attracted the interest of researchers in machine learning, information retrieval, computer vision, and related communities and has shown promise for large-scale visual searching. This paper focuses on hashing algorithms that encode images, videos, documents, or other data types as a set of short binary codes while preserving the original example structure (e.g., similarities between data points). One advantage of these hashing methods is that pairwise distance calculations can be performed extremely efficiently in Hamming space, and as a result, methods based on pairwise comparisons can be performed more efficiently and applied to large data sets. Due to the flexibility of binary representations, hashing algorithms can be applied in many ways, for example, searching efficiently by exploring only examples falling into buckets close to the query according the Hamming distance or using the hash codes for other tasks, such as image classification, face recognition, and indexing.

Hashing methods can be broadly classified into two categories: data-independent and data-dependent methods. Data-independent algorithms do not require training data and randomly construct a set of hash functions without any training. Representative methods include locality-sensitive hashing (LSH) [1] and its variants [2] and the Min-Hash algorithms [3]. A disadvantage of the LSH family is that they usually need a long bit length ($\geq$1000) for satisfactory performance. This results in large storage costs, thus limiting their applications.

Recently, data-dependent or learning-based hashing methods have become popular, because learned compact hash codes can effectively and efficiently index and organize massive amounts of data. Instead of randomly generating hash functions as in LSH, data-dependent hashing methods aim to generate short hash codes (typically $\leq$200) using training data. There is abundant literature on different data-dependent hashing methods that can be classified into four main categories.

The first is unsupervised hashing, which does not utilize the label information of training examples. Representative algorithms include spectral hashing [4], principal component hashing [5], principal component analysis [6] iterative quantization (PCA-ITQ) [7], [8], anchor graph-based hashing (AGH) [9], scalable graph hashing with feature transformation [10], and inductive manifold hashing (IMH) [11] with t-distributed stochastic neighbor embedding (t-SNE) [12]. However, since unsupervised hashing does not consider the label information of input data, useful information critical to pattern classification may be lost. Thus, various supervised hashing and semisupervised hashing techniques have been proposed, since it is generally believed that label information produces a more discriminative recognition algorithm.

The second category is supervised and semisupervised hashing (SSH), which take full consideration of the class labels. Representative methods in this group include

Manuscript received July 21, 2016; revised October 7, 2016; accepted November 30, 2016. Date of publication December 29, 2016; date of current version February 15, 2018. This work was supported in part by the National Science Foundation of China under Grant 61572463 and Grant 61573360, in part by the grant of the Open Project Program of the National Laboratory of Pattern Recognition (NLPR) under Grant 201700027, in part by the CCF-Tencent Open Fund, and in part by the Australian Research Council under Project DP-140102164, Project FT-130101457, and Project LE-140100061. *(Corresponding author: Zhenan Sun.) (Jie Gui and Tongliang Liu contributed equally to this work.)*

J. Gui is with the Institute of Intelligent Machines, Chinese Academy of Sciences, Hefei 230031, China. He is also a Visiting Researcher with the Shenzhen Key Laboratory of Media Security, Shenzhen University, Shenzhen 518060, China. (e-mail: guijie@ustc.edu).

T. Liu is with the Centre for Artificial Intelligence and the Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, NSW 2007, Australia (e-mail: tliang.liu@gmail.com).

Z. Sun and T. Tan are with the Center for Research on Intelligent Perception and Computing, National Laboratory of Pattern Recognition, Institute of Automation, CAS Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: znsun@nlpr.ia.ac.cn; tnt@nlpr.ia.ac.cn).

D. Tao is with the School of Information Technologies, Faculty of Engineering and Information Technologies, The University of Sydney, Sydney, NSW 2006, Australia (e-mail: dacheng.tao@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2016.2636870

SSH [13], kernel-based supervised hashing (KSH) [14], fast supervised hashing using graph cuts and decision trees (FastHash) [15], [16], supervised discrete hashing (SDH) [17], and linear discriminant analysis-based [18] hashing [19]. In our view, ranking-based methods [20]–[23] (in which ranking labels such as triplets constitute the supervised information) also belong to the supervised hashing group.

The third category is multimodal hashing, which includes multisource hashing and cross-modal hashing. Multisource hashing [24], [25] assumes that all the views are provided for a query and aims to learn better codes than unimodal hashing by using all these views. In cross-modal hashing [26], the query represents one modality, while the output represents another modality. For example, given a text query, images are returned corresponding to the text. Therefore, both multisource hashing and cross-modal hashing use multimodal information [27]. However, they are used in different applications and cross-modal hashing may have wider application than multisource hashing in practice.

The fourth category is deep learning-based [28]–[30] hashing. To the best of our knowledge, semantic hashing [31] represents the first use of deep learning for hashing. This seminal work utilized the stacked-restricted Boltzmann machine to learn compact hash codes for visual searching. However, the model was complex and required pretraining, which is inefficient in practice. Both deep regularized similarity comparison hashing [32] and [33] use the deep convolutional neural network for hashing. Some other related methods can be found in [34]–[36].

In general, the discrete constraints imposed on the hash codes generated by the hash objective functions lead to NP-hard mixed-integer optimization problems. To simplify the optimization in the hash code learning process, most algorithms first solve a relaxed problem by discarding the discrete constraints and then perform a quantization step that turns real values into the approximate hash code by thresholding (or quantization). This relaxation strategy greatly simplifies the original discrete optimization. However, such an approximate solution is suboptimal, typically of low quality, and often produces a less effective hash code due to the accumulated quantization error, especially when learning long-length codes. Most existing hashing methods do not take discrete optimization into account. Shen *et al.* [17] proposed a novel SDH method that aimed to directly learn the binary hash codes without relaxations. To make full use of label information, this method was formulated as a least squares classification to regress each hash code to its corresponding label.

However, the ordinary least squares problem may not be optimal for classification. To further improve the performance, here we propose a novel method called "SDH with relaxation" (SDHR) to directly learn the regression targets from data. SDHR is essentially a single and compact learning method for multiclass classification. The regression targets learned by SDHR can guarantee that each example (data point) are correctly classified with a large margin. During learning, SDHR does not consider the absolute values in regression targets and only forces relative values to satisfy a large margin for correct classification. Therefore, SDHR is much more accurate and flexible than SDH. The optimization problem of solving the regression target for SDHR is convex, and we employ an efficient alternating procedure to find the regression target. Our experimental results show that SDHR generally performs better than SDH.

TABLE I
NOTATIONS

Notation	Description
X	the data matrix
x_i	the i-th data point
n	the training sample size: the number of the total training data points
B	the hash codes
b_i	the i-th row of B (the hash code for x_i)
l	the length of hash code
Y	the label matrix
c	the number of classes
y_i	the i-th row of the matrix Y
y_{ik}	the k-th element of y_i
W	the projection matrix for the hash code
$F(\cdot)$	a nonlinear embedding to approximate the hash code
m	the number of anchor points
$\phi(\cdot)$	an m-dimensional row vector obtained by the RBF kernel
P	the projection matrix for the nonlinear embedding
t	the translation (offset) vector used in SDHR
R	the regression target matrix (code words) used in SDHR
R_i	the i-th row of the matrix R
L_i	the label of x_i
e_n	an n-dimensional column vector with all elements equal to one
r	the Hamming radius

The remainder of this paper is organized as follows. Section II outlines SDH. In Section III, we describe our proposed method in detail. The experimental results are presented in Section IV, followed by conclusions in Section V.

II. Brief Review of Supervised Discrete Hashing

In this section, we introduce the related works SDH [17] by way of introduction. Suppose that we have n examples $X = \{x_i\}_{i=1}^n$ (Table I for a list of notations used in this paper). We aim to learn a set of hash codes $B = \{b_i\}_{i=1}^n \in \{-1, 1\}^{n\times l}$ to preserve their similarities in the original space, where the ith vector b_i is the l-bits hash codes for x_i. The corresponding class labels of all training examples are denoted as $Y = \{y_i\}_{i=1}^n \in R^{n\times c}$, where c is the number of classes and $y_{ik} = 1$ if x_i belongs to class k and 0 otherwise. The term y_{ik} is the kth element of y_i.

The objective function of SDH is defined as

$$\begin{aligned}\min_{B,F,W} \sum_{i=1}^{n} \|y_i - b_i W\|_2^2 + \lambda \|W\|_F^2 \\ + \upsilon \sum_{i=1}^{n} \|b_i - F(x_i)\|_2^2 \\ \text{s.t. } \forall i \quad b_i \in \{-1, 1\}^l. \end{aligned} \tag{1}$$

That is

$$\begin{aligned}&\min_{B,F,W} \|Y - BW\|_F^2 + \lambda \|W\|_F^2 + \upsilon \|B - F(X)\|_F^2 \\ &\text{s.t. } B \in \{-1, 1\}^{n\times l}\end{aligned} \tag{2}$$

where $\|\cdot\|_F$ is the Frobenius norm of a matrix. The first term of (1) is the ordinary least squares regression, which is used to regress each hash code to its corresponding class label. The term W is the projection matrix. The second term of (1) is for regularization. $F(\cdot)$ in the last term of (1) is a simple yet powerful nonlinear embedding to approximate the hash code

$$F(x) = \phi(x)P \tag{3}$$

where $\phi(x)$ is an m-dimensional row vector obtained by the RBF kernel: $\phi(x) = [\exp(\|x - a_1\|^2/\sigma), \ldots, \exp(\|x - a_m\|^2/\sigma)]$, $\{a_j\}_{i=1}^{m}$ are randomly selected m anchor points from the training examples, and σ is the Gaussian kernel parameter. The matrix $P \in R^{m\times l}$ projects $\phi(x)$ onto the low-dimensional space. Similar formulations as (3) are widely used in other methods, such as KSH [14] and binary reconstructive embedding (BRE) [37].

The need of a Gaussian kernel function is shown as follows: existing methods such as LSH do not apply for high-dimensional kernelized data when the underlying feature embedding for the kernel is unknown. Moreover, the using of kernel generalizes such methods as LSH to accommodate arbitrary kernel functions, which is possible to preserve the algorithm's sublinear time similarity search guarantees for many useful similarity functions [43].

The optimization of (2) involves three steps: the F-step solving P, the G-step solving W, and the B-step solving B:

F-step: By fixing B, the projection matrix P is easily computed

$$P = (\phi(X)^T\phi(X))^{-1}\phi(X)^T B. \tag{4}$$

G-Step: If B is fixed, it is easy to solve W, which has a closed-form solution

$$W = (B^T B + \lambda I)^{-1} B^T Y. \tag{5}$$

B-Step: By fixing other variables, B also has a closed-form solution. Please refer to [17] for details.

III. Our Proposed Method: Supervised Discrete Hashing With Relaxation

In this section, we introduce our proposed SDHR method in detail. The first term of SDH's objective function is to regress y_i on b_i. That is $\min \|y_i - b_i W\|_2^2$, and the optimal solution W^* is $y_i = b_i W^*$, which is a linear function that will go through the origin of the coordinate. It is more appropriate and flexible to have a translation (offset) vector t and employing the hypothesis $y_i = b_i W + t^T$, where t^T is the transpose of the column vector t. In SDH, y_i is fixed and $y_{ij} = 1$ if x_i belongs to the class j and 0 otherwise. To make SDH more discriminative and flexible, we use R_i instead of y_i as the regression target (code words) for the ith hash code b_i, which satisfies

$$\begin{aligned}&\min_{R_i} \|R_i - b_i W - t^T\|_F^2\\ &\text{s.t. } R_{i,L_i} - \max_{k\neq L_i} R_{ik} \geq 1\end{aligned} \tag{6}$$

where L_i is the label of x_i. The term R_i is optimized instead of being given. Our aim is to produce a large margin between the code words of the true label and all the other wrong labels that is larger than one to satisfy the large margin criterion $R_{i,L_i} - \max_{k\neq L_i} R_{ik} \geq 1$. The objective function of SDHR is therefore defined as

$$\begin{aligned}\min_{B,R,t,F,W} &\sum_{i=1}^{n} \|R_i - b_i W - t^T\|_2^2 + \lambda\|W\|_F^2\\ &+ \upsilon \sum_{i=1}^{n} \|b_i - F(x_i)\|_2^2\\ \text{s.t. } &\forall i \quad b_i \in \{-1, 1\}^l\\ &\forall i \quad R_{i,L_i} - \max_{k\neq L_i} R_{ik} \geq 1.\end{aligned} \tag{7}$$

That is

$$\begin{aligned}\min_{B,R,t,F,W} &\|R - BW - e_n t^T\|_F^2 + \lambda\|W\|_F^2\\ &+ \upsilon\|B - F(X)\|_F^2\\ \text{s.t. } &B \in \{-1, 1\}^{n\times l}\\ &\forall i \quad R_{i,L_i} - \max_{k\neq L_i} R_{ik} \geq 1\end{aligned} \tag{8}$$

where e_n is an n-dimensional column vector with all elements equal to one. Hashing is not specifically designed for classification. Nevertheless, the supervised information is still very helpful. We want the hash codes to be similar for examples from the same class and we want them to be different for examples from different classes. Therefore, we relax the fixed label values to relative ones to satisfy the large margin criterion, which is more flexible than using the fixed label values.

The problem in (8) is a mixed binary optimization problem. By fixing the other variables, solving a single variable is relatively easy. Based on this decomposition, an alternating optimization technique can be adopted to iteratively and efficiently solve this optimization problem. Each iteration alternatively solves B, R, W, t, and P. Therefore, SDHR's optimization has five steps.

t-Step: When all variables except t are fixed, (8) can be equivalently rewritten as

$$\begin{aligned}&\min_t \|e_n t^T - (R - BW)\|_2^2\\ &= \min_t tr\begin{pmatrix}(te_n^T - (R - BW)^T)\\ \times(e_n t^T - (R - BW))\end{pmatrix}.\end{aligned} \tag{9}$$

By setting the derivative of (9) with respect to t to zero, t can be solved with the closed-form solution

$$t = \frac{(R - BW)^T e_n}{n}. \tag{10}$$

B-Step: For solving B, (8) can be rewritten as

$$\begin{aligned}&\min_B \|R - BW - e_n t^T\|_F^2 + \upsilon\|B - F(X)\|_F^2\\ &= \min_B \|BW - (R - e_n t^T)\|_F^2 + \upsilon\|B - F(X)\|_F^2\\ &= \min_B \|BW\|_F^2\\ &\quad - 2tr(B(W(R - e_n t^T)^T + \upsilon F(X)^T))\\ &\quad + \upsilon tr(B^T B).\end{aligned}$$

Since $tr(B^T B)$ is a constant, we have

$$\min_B \|BW\|_F^2 - 2tr(BQ) \quad \text{s.t. } B \in \{-1, 1\}^{n\times l} \tag{11}$$

where $Q = ((R - e_n t^T)W^T + \upsilon F(X))^T$.

Thus, B can be solved using the discrete cyclic coordinate descent method, which is similar to solving B in SDH.

F-Step: The F-step of SDHR is the same as that of SDH

$$P = (\phi(X)^T \phi(X))^{-1} \phi(X)^T B. \tag{12}$$

G-Step: If all variables except W are fixed, we put (10) into (8) and get

$$\begin{aligned}\min_W & \left\| R - BW - \frac{e_n e_n^T (R - BW)}{n} \right\|_F^2 + \lambda \|W\|_F^2 \\ &= \min_W \left\| \left(I - \frac{e_n e_n^T}{n} \right) (R - BW) \right\|_F^2 + \lambda \|W\|_F^2. \end{aligned} \tag{13}$$

By setting the derivative of (13) with respect to W to zero, it is easy to solve W, which has a closed-form solution

$$W = \left(B^T \left(I - \frac{e_n e_n^T}{n} \right) B + \lambda I \right)^{-1} B^T \left(I - \frac{e_n e_n^T}{n} \right) R. \tag{14}$$

R-Step: When we fix B, t, P, and W, how to solve R in (8) can be transformed to

$$\begin{aligned}&\min_R \|R - (BW + e_n t^T)\|_F^2 \\ &\text{s.t. } \forall i \quad R_{i,L_i} - \max_{k \neq L_i} R_{ik} \geq 1.\end{aligned} \tag{15}$$

It is natural to solve the matrix R row by row, which has a general form of

$$\begin{aligned}&\min_r \sum_{q=1}^{c} (r_q - a_q)^2 \\ &\text{s.t. } r_j - \max_{k \neq j} r_k \geq 1\end{aligned} \tag{16}$$

where r and a are the ith row of R and $(BW + e_n t^T)$, respectively, and r_q and a_q are the qth element of r and a, respectively. We can use the Lagrangian multiplier method to solve (16). The Lagrangian function is defined as

$$\begin{aligned}L(r_q, \lambda_k) = &\sum_{q=1}^{c} (r_q - a_q)^2 \\ &+ \sum_{k=1, k \neq j}^{c} \lambda_k (1 + r_k - r_j)\end{aligned} \tag{17}$$

where λ_k is the Lagrangian multiplier and $\lambda_k \geq 0$. By setting the derivative of $L(r_q, \lambda_k)$ with respect to r_j and $r_k (k \neq j)$ to zero, we have

$$2(r_j - a_j) - \sum_{k=1, k \neq j}^{c} \lambda_k = 0 \tag{18}$$

$$2(r_k - a_k) + \lambda_k = 0, \quad k \neq j. \tag{19}$$

Furthermore, using the Kuhn–Tucker condition, we have

$$\lambda_k (1 + r_k - r_j) = 0, \quad k \neq j. \tag{20}$$

With (18)–(20), we have $(2c - 1)$ equations and $(2c - 1)$ variables. Thus, (16) can be solved in this way. Please note that (16) can be solved using other optimization methods. For example, a method called retargeting proposed in [38] can also be used to solve this optimization problem. In summary, our algorithm for solving SDHR is presented in Algorithm 1.

Algorithm 1 SDHR

Inputs: training data $\{x_i, y_i\}_{i=1}^n$; code length l; maximum iteration number t; parameter λ
Output: binary codes $\{b_i\}_{i=1}^n \in \{-1, 1\}^{n \times l}$
Randomly select m examples $\{a_j\}_{j=1}^m$ from the training examples and get the $\phi(x)$ via the Gaussian kernel function;
Initialize b_i as a $\{-1, 1\}^l$ vector randomly;
Initialize R as $R = \{R_{ij}\} \in R^{n \times c}$ where $R_{ij} = \begin{cases} 1, & if\ y_i = j \\ 0, & otherwise \end{cases}$;
Use (14) to initialize W;
Use (10) to initialize t;
Use (12) to initialize P;
repeat
  B-step Use (11) to solve B;
  R-step Use (18), (19) and (20) to solve R;
  G-step Use (14) to solve W;
  t-step Use (10) to solve t;
  F-step Use (12) to solve P;
until convergence

IV. Experimental Results

In this section, we investigate SDHR's performance by conducting experiments on a server with an Intel Xeon processor (2.80 GHz), 128-GB RAM, and configured with Microsoft Windows Server 2008 and MATLAB 2014b.

We conduct experiments on two large-scale image data sets, CIFAR-10[1] and MNIST,[2] and a large-scale and challenging face data set FRGC. The proposed method is compared with popular hashing methods, including BRE [37], SSH [13], KSH [14], FastHash [15], [16], AGH [9], and IMH [11] with t-SNE [12]. For ITQ [7], [8], we use both its supervised version CCA-ITQ and unsupervised version PCA-ITQ. Canonical correlation analysis (CCA) is used as the preprocessing step for CCA-ITQ. We use the public MATLAB codes and the parameters suggested by the corresponding authors. Specifically, for SDH and SDHR, λ and υ are empirically set to 1 and 1e-5, respectively; we set the maximum iteration number t to 5. For AGH, IMH, SDH, and SDHR, 1000 randomly selected anchor points are used.

The experimental results are reported in terms of Hamming ranking [mean average precision (MAP)], hash lookup (precision, recall, and F-measure of Hamming radius 2), accuracy, training time, and test time. The Hamming radius r is set to be 2 as in [17] and [39]. The F-measure is defined as two × precision × recall/(precision + recall). The following evaluation metric is also utilized to measure the performance of the different algorithms: precision at N samples (precision@sample = N), which is the percentage of true neighbors among the top N retrieved instances. N is set to be 500 as in [39]. Note that a query is considered to be a false case if no example is returned when calculating precisions. Ground truths are defined by the label information of the data sets.

[1] https://www.cs.toronto.edu/~kriz/cifar.html
[2] http://yann.lecun.com/exdb/mnist/

TABLE II

PRECISION, RECALL, AND F-MEASURE OF HAMMING DISTANCE WITHIN RADIUS 2, MAP, ACCURACY, AND TIME ON CIFAR-10. THE RESULTS ARE REPORTED WHEN THE NUMBER OF HASHING BITS IS 16. FOR SSH, 5000 LABELED EXAMPLES ARE USED FOR SIMILARITY MATRIX CONSTRUCTION. THE TRAINING AND TEST TIMES ARE IN SECONDS

Method	precision@r=2	recall@r=2	F-measure@r=2	MAP	accuracy	training time	test time
SDHR	0.4612	0.2835	**0.3511**	0.4091	**0.638**	42.4	2.7e-6
SDH	0.4588	0.2805	0.3481	0.4123	0.622	43.9	2.2e-6
BRE	0.1745	0.2118	0.1913	0.1431	0.305	66.0	6.3e-6
KSH	0.4763	0.1167	0.1875	0.4030	0.549	1.3e3	7.8e-5
SSH	0.1674	**0.3908**	0.2344	0.1687	0.295	17.2	3.0e-6
CCA-ITQ	0.3819	0.1382	0.2030	0.3137	0.539	4.6	**1.5e-7**
FastHash	**0.5560**	0.2310	0.3264	**0.5253**	0.588	1.2e3	4.3e-4
PCA-ITQ	0.2389	0.0221	0.0404	0.1616	0.371	**3.3**	**1.5e-7**
AGH	0.2223	0.0553	0.0886	0.1562	0.34	6.9	8.5e-5
IMH	0.1926	0.1454	0.1657	0.1696	0.321	42.5	6.3e-5

Fig. 1. Sample images from CIFAR-10.

Fig. 2. Sample images on the MNIST database.

As a subset of the famous 80-M tiny image collection [40], CIFAR-10 contains 60 000 images from 10 classes with 6000 examples per class. Example images from this data set are shown in Fig. 1. Each image in this data set is represented as a 512-D GIST feature vector [41]. MNIST contains 70 000 784-D handwritten digit images from "0" to "9," each image being 28×28 pixels; some example images are shown in Fig. 2. Both MNIST and CIFAR-10 are split into a test set with 1000 examples and a training set containing all remaining examples. FRGC is described in the following.

A. Experiments on CIFAR-10

The experimental results on CIFAR-10 are shown in Table II. SDHR performs better than SDH in terms of precision, recall, F-measure, and accuracy. For example, the accuracy of SDHR is 1.03 times higher than that of SDH. Moreover, from the last two columns in Table II, we can see that the training time cost of SDHR is lower than that of SDH, which allows the method to be applied to the whole training data. In contrast, KSH and FastHash take about 20 min to train. More specifically, the training of SDHR is about 28-times and 31-times faster than FastHash and KSH, respectively, in this case. SSH, CCA-ITQ, PCA-ITQ, AGH, and IMH are also very efficient; however, their performance is generally worse than SDHR. The precision at 500 examples (precision@sample = 500), precision of Hamming radius 2, and accuracy versus the number of hashing bits are shown in Figs. 3–5, respectively. Due to space limitations, we only show some algorithms in the corresponding figure. With respect to precision of Hamming radius 2, SDHR outperforms the other methods when the number of hashing bits is larger than 32, and KSH performs the best when the number of hashing bits is 16. SDHR outperforms all other methods in terms of precision at 500 examples (precision@sample = 500) and accuracy, highlighting the effectiveness of SDHR.

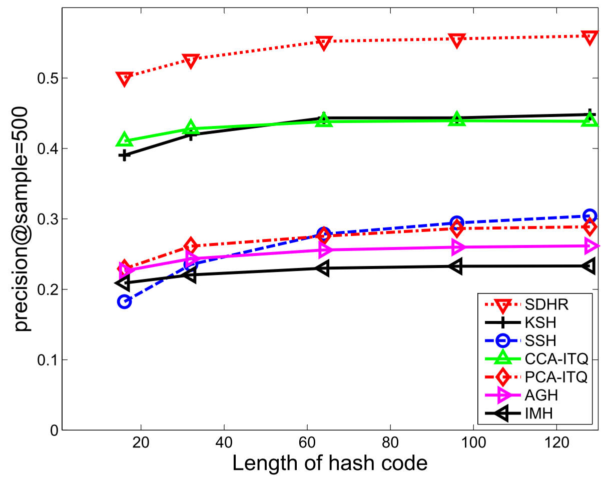


Fig. 3. Precision@sample = 500 versus the number of hashing bits (16, 32, 64, 96, and 128) on CIFAR-10.

B. Experiments on MNIST

The experimental results on MNIST are presented in Table III. SDHR performs best in terms of precision, recall,

TABLE III

PRECISION, RECALL, AND F-MEASURE OF HAMMING DISTANCE WITHIN RADIUS 2, MAP, ACCURACY, TRAINING TIME, AND TEST TIME ON MNIST. THE RESULTS ARE REPORTED WHEN THE NUMBER OF HASHING BITS IS 64

Method	precision@r=2	recall@r=2	F-measure@r=2	MAP	accuracy	training time	test time
SDHR	**0.9330**	**0.7943**	**0.8581**	0.9417	0.967	186.2	4.9e-6
SDH	0.9269	0.7711	0.8419	0.9397	0.963	128	5.1e-6
BRE	0.3850	0.0011	0.0021	0.4211	0.839	24060.6	9.3e-5
KSH	0.6454	0.2539	0.3644	0.9103	0.927	1324.7	7.6e-5
SSH	0.6883	0.0738	0.1332	0.4787	0.734	260.2	5.7e-6
CCA-ITQ	0.7575	0.2196	0.3405	0.7978	0.894	16.6	**4.1e-7**
FastHash	0.8680	0.6735	0.7585	**0.9813**	**0.972**	4661.1	0.0012
PCA-ITQ	0.1680	9.3e-4	0.0018	0.4581	0.886	10.1	4.5e-7
AGH	0.8568	0.0131	0.0258	0.5984	0.899	**6.9**	6.4e-5
IMH	0.8258	0.0889	0.1606	0.6916	0.897	32.2	6.4e-5

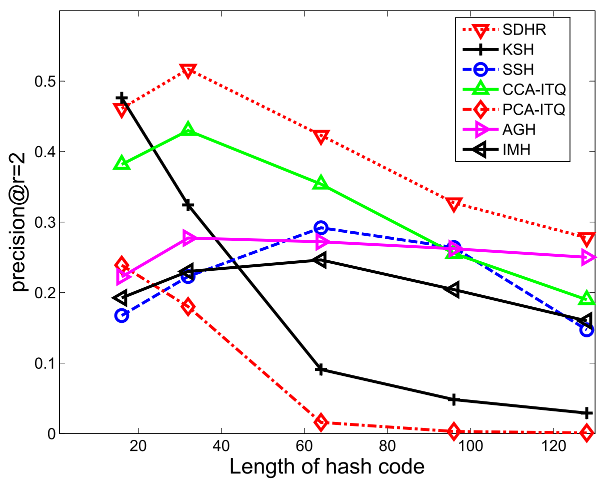


Fig. 4. Precision of Hamming radius 2 versus the number of hashing bits (16, 32, 64, 96, and 128) on CIFAR-10.

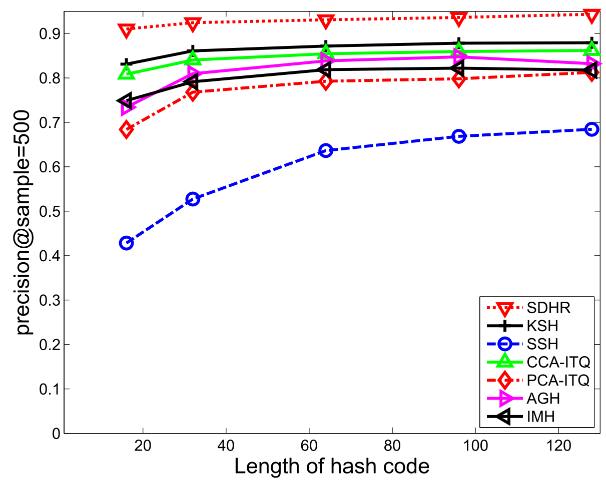


Fig. 6. Precision@sample = 500 versus the number of hashing bits (16, 32, 64, 96, and 128) on MNIST.

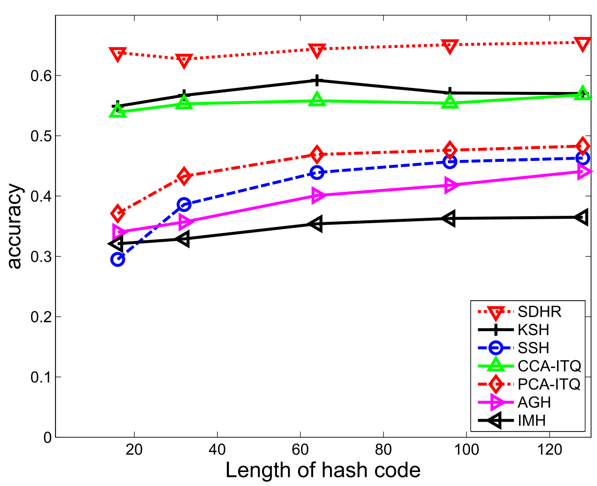


Fig. 5. Accuracy versus the number of hashing bits (16, 32, 64, 96, and 128) on CIFAR-10.

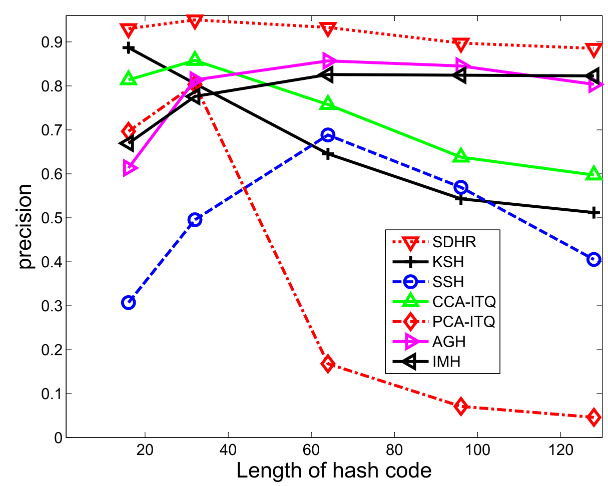


Fig. 7. Precision of Hamming radius 2 versus the number of hashing bits (16, 32, 64, 96, and 128) on MNIST.

and F-measure, while FastHash performs best in terms of MAP and accuracy. However, SDHR is much faster than FastHash: SDHR and FastHash take 186.2 and 4661.1 s, respectively. Thus, SDHR is about 25-times faster than FastHash in this case. Furthermore, the precision@sample = 500, precision of Hamming radius 2, recall of Hamming radius 2, F-measure of Hamming radius 2, MAP, and accuracy curves are shown in Figs. 6–11, respectively. Only some methods are shown due to space limitations. SDHR outperforms the other methods, and furthermore, SDHR particularly outperforms the other methods in terms of recall of Hamming radius 2 and F-measure of Hamming radius 2.

C. Experiments on FRGC

The FRGC version two face data set [42] is a large-scale and challenging benchmark face data set. FRGC experiment

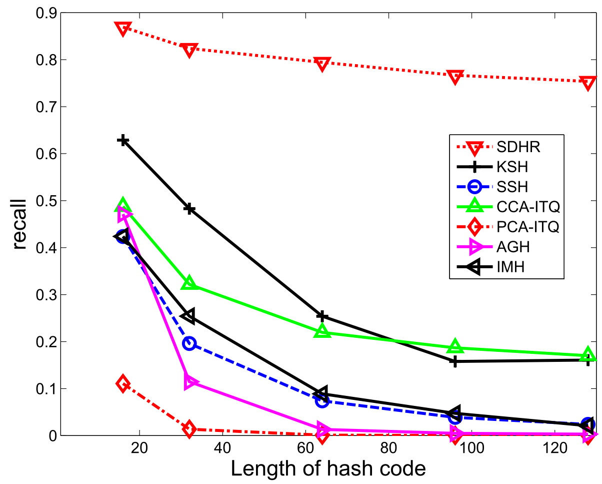


Fig. 8. Recall of Hamming radius 2 versus the number of hashing bits (16, 32, 64, 96, and 128) on MNIST.

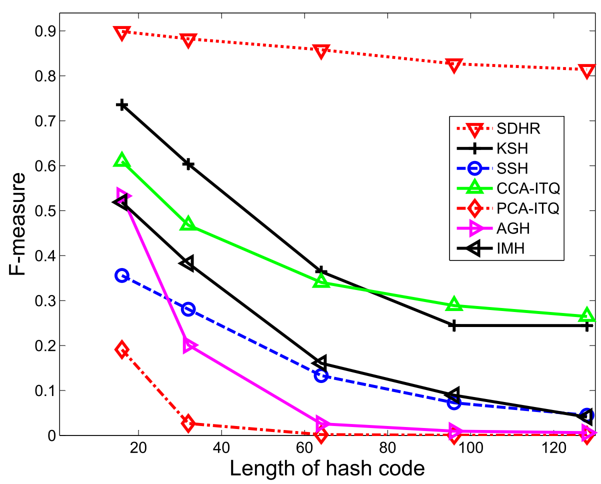


Fig. 9. F-measure of Hamming radius 2 versus the number of hashing bits (16, 32, 64, 96, and 128) on MNIST.

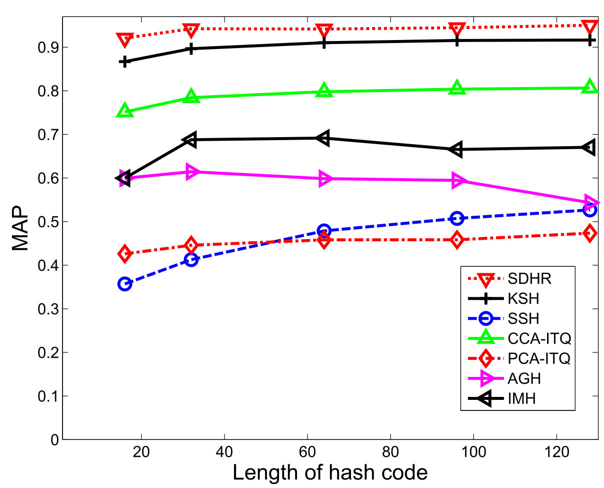


Fig. 10. MAP versus the number of hashing bits (16, 32, 64, 96, and 128) on MNIST.

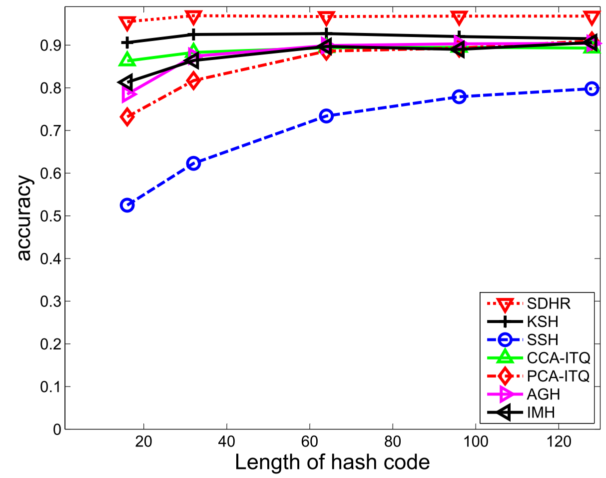


Fig. 11. Accuracy versus the number of hashing bits (16, 32, 64, 96, and 128) on MNIST.

Fig. 12. Cropped and resized examples of four randomly selected individuals in the FRGC face data set. Each row represents one individual.

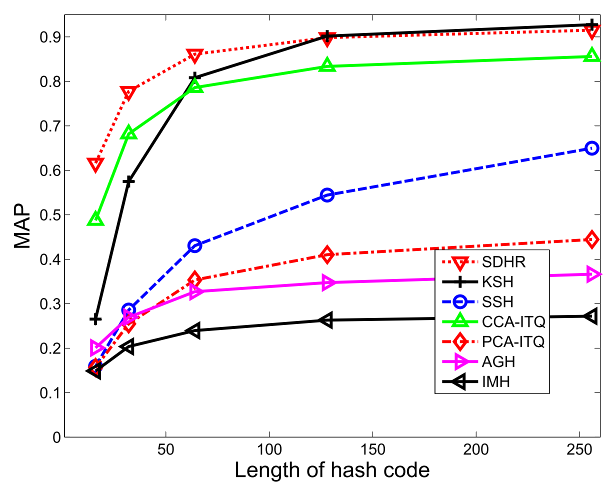


Fig. 13. MAP versus the number of hashing bits (16, 32, 64, 128, and 256) on FRGC.

4 contains 8014 face images from 466 persons in the query set. These uncontrolled images demonstrate variations in blur, expression, illumination, and time. In our experiment, we only select persons with over 10 images in the data set, resulting in 3160 images from 316 persons. Each image is cropped and resized to 32×32 pixels (256 gray levels per pixel) by fixing the positions of the eyes. For each individual, three images are randomly selected for testing and the remaining seven used for training. Fig. 12 shows example images from FRGC.

The experimental results on FRGC are presented in Table IV. SDHR outperforms the other methods in terms of precision, recall, F-measure, and MAP, while CCA-ITQ outperforms the other methods in terms of accuracy. Fig. 13 shows MAP versus the number of hashing bits on

TABLE IV

PRECISION, RECALL, AND F-MEASURE OF HAMMING DISTANCE WITHIN RADIUS 2, MAP, ACCURACY, AND TIME ON FRGC. THE RESULTS ARE REPORTED WHEN THE NUMBER OF HASHING BITS IS 16

Method	precision@r=2	recall@r=2	F-measure@r=2	MAP	accuracy	training time	test time
SDHR	**0.4893**	**0.5893**	**0.5346**	**0.6161**	0.563	5.9	1.4e-6
SDH	0.4851	0.5833	0.5297	0.6138	0.565	1.6	1.7e-6
BRE	0.0352	0.3798	0.0645	0.0851	0.145	117.7	1.3e-5
KSH	0.2697	0.2400	0.2540	0.2654	0.377	333.1	1.1e-4
SSH	0.1286	0.2402	0.1675	0.1593	0.263	5.8	5.7e-6
CCA-ITQ	0.4426	0.4776	0.4594	0.4874	**0.59**	1.3	1.1e-7
FastHash	0.1646	0.0895	0.1160	0.1199	0.283	62.0	6.2e-4
PCA-ITQ	0.1359	0.2307	0.1710	0.1566	0.275	**0.2**	**9.7e-8**
AGH	0.0688	0.4107	0.1179	0.2009	0.237	2.3	1.2e-4
IMH	0.0492	0.4460	0.0886	0.1489	0.198	35.0	8.9e-5

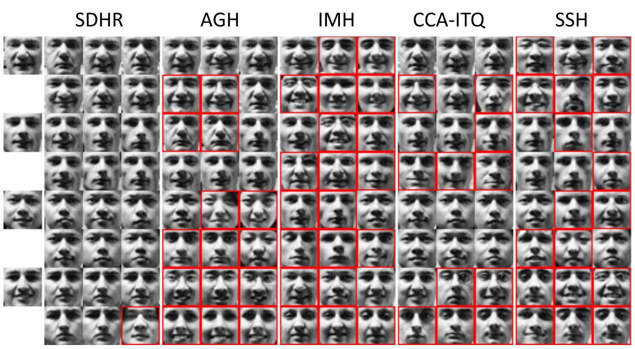


Fig. 14. Top six retrieved images of four queries returned by various hashing algorithms on FRGC. The query image is in the first column. From left to right: retrieved images by SDHR, AGH, IMH, CCA-ITQ, and SSH when 16-b binary codes are used for searching. False positive returns are marked with red borders.

this data set. Due to space limitations, Fig. 13 only shows representative methods. This figure illustrates two main points: first, SDHR outperforms other methods when the number of hashing bits is less than or equals 64, while KSH performs best when the number of hashing bits is larger than or equal to 128; second, the MAP of all methods increases as the number of hashing bits increases. This might be because, as the number of hashing bits increases, more information can be encoded in the hash code. Therefore, the hash code represents the face image in a more informative and discriminative way. Furthermore, Fig. 14 shows some example query images and the retrieved neighbors on FRGC when 16 b are used to learn the hash codes. We can see that SDHR shows better searching performance, because higher semantic relevance is obtained in the top retrieved examples.

V. CONCLUSION

In this paper, we propose a novel learning-based hashing method called "SDHR" based on "SDH." SDHR uses learned code words rather than the traditional fixed code words used in SDH to encode class label information. As expected, the SDHR's performance is better than that of SDH. Real-world image classification and face recognition experiments highlight the advantages of the proposed method.

Although hashing methods can reduce computational costs, the costs remain very large for large-scale image retrieval. Selecting or learning representative examples to represent each class must be further studied.

We used the pixel feature on the MNIST and FRGC data sets. It is possible that advanced visual features rather than the original pixel intensity values will further improve performance. Designing the most appropriate feature for specific hashing algorithms is also a thought-provoking direction for future studies.

REFERENCES

[1] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *Proc. VLDB*, 1999, pp. 518–529.

[2] M. Raginsky and S. Lazebnik, "Locality-sensitive binary codes from shift-invariant kernels," in *Proc. Neural Inf. Process. Syst.*, 2009, pp. 1509–1517.

[3] O. Chum and J. Matas, "Large-scale discovery of spatially related images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 2, pp. 371–377, Feb. 2010.

[4] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Proc. Neural Inf. Process. Syst.*, 2009, pp. 1753–1760.

[5] Y. Matsushita and T. Wada, "Principal component hashing: An accelerated approximate nearest neighbor search," in *Advances in Image and Video Technology*, Tokyo, Japan: Springer, 2009, pp. 374–385.

[6] Z. Hu, G. Pan, Y. Wang, and Z. Wu, "Sparse principal component analysis via rotation and truncation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 4, pp. 875–890, Apr. 2016.

[7] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 12, pp. 2916–2929, Dec. 2013.

[8] Y. Gong and S. Lazebnik, "Iterative quantization: A procrustean approach to learning binary codes," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2011, pp. 817–824.

[9] W. Liu, J. Wang, S. Kumar, and S.-F. Chang, "Hashing with graphs," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 1–8.

[10] Q.-Y. Jiang and W.-J. Li, "Scalable graph hashing with feature transformation," in *Proc. Int. Joint Conf. Artif. Intell.*, 2015, pp. 2248–2254.

[11] F. Shen, C. Shen, Q. Shi, A. Van Den Hengel, and Z. Tang, "Inductive hashing on manifolds," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 1562–1569.

[12] L. van der Maaten and G. Hinton, "Visualizing data using T-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

[13] J. Wang, S. Kumar, and S.-F. Chang, "Semi-supervised hashing for large-scale search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 12, pp. 2393–2406, Dec. 2012.

[14] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang, "Supervised hashing with kernels," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 2074–2081.

[15] G. Lin, C. Shen, and A. V. D. Hengel, "Supervised hashing using graph cuts and boosted decision trees," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 11, pp. 2317–2331, Nov. 2015.

[16] G. Lin, C. Shen, Q. Shi, A. van den Hengel, and D. Suter, "Fast supervised hashing with decision trees for high-dimensional data," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 1963–1970.

[17] F. Shen, C. Shen, W. Liu, and H. Tao Shen, "Supervised discrete hashing," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 37–45.

[18] X. Zhang, D. Chu, and R. C. Tan, "Sparse uncorrelated linear discriminant analysis for undersampled problems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 7, pp. 1469–1485, Jul. 2016.

[19] C. Strecha, A. M. Bronstein, M. M. Bronstein, and P. Fua, "LDAHash: Improved matching with smaller descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 1, pp. 66–78, Jan. 2012.

[20] M. Norouzi, D. J. Fleet, and R. R. Salakhutdinov, "Hamming distance metric learning," in *Proc. Neural Inf. Process. Syst.*, 2012, pp. 1061–1069.

[21] J. Wang, J. Wang, N. Yu, and S. Li, "Order preserving hashing for approximate nearest neighbor search," in *Proc. ACM Int. Conf. Multimedia*, 2013, pp. 133–142.

[22] J. Wang, W. Liu, A. X. Sun, and Y.-G. Jiang, "Learning hash codes with listwise supervision," in *Proc. Int. Conf. Comput. Vis.*, 2013, pp. 3032–3039.

[23] Q. Wang, Z. Zhang, and L. Si, "Ranking preserving hashing for fast similarity search," in *Proc. Int. Joint Conf. Artif. Intell.*, 2015, pp. 3911–3917.

[24] J. Song, Y. Yang, Z. Huang, H. T. Shen, and R. Hong, "Multiple feature hashing for real-time large scale near-duplicate video retrieval," in *Proc. ACM Int. Conf. Multimedia*, 2011, pp. 423–432.

[25] D. Zhang, F. Wang, and L. Si, "Composite hashing with multiple information sources," in *Proc. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2011, pp. 225–234.

[26] S. Kumar and R. Udupa, "Learning hash functions for cross-view similarity search," in *Proc. Int. Joint Conf. Artif. Intell.*, 2011, pp. 1360–1365.

[27] G. Chao and S. Sun, "Alternative multiview maximum entropy discrimination," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 7, pp. 1445–1456, Jul. 2016.

[28] M. Gong, J. Zhao, J. Liu, Q. Miao, and L. Jiao, "Change detection in synthetic aperture radar images based on deep neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 1, pp. 125–138, Jan. 2016.

[29] C. Xia, F. Qi, and G. Shi, "Bottom–up visual saliency estimation with deep autoencoder-based sparse reconstruction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 6, pp. 1227–1240, Jun. 2016.

[30] A. Dundar, J. Jin, B. Martini, and E. Culurciello, "Embedded streaming deep neural networks accelerator with applications," *IEEE Trans. Neural Netw. Learn. Syst.*, to appear.

[31] R. Salakhutdinov and G. Hinton, "Semantic hashing," *Int. J. Approx. Reasoning*, vol. 50, no. 7, pp. 969–978, Jul. 2009.

[32] R. Zhang, L. Lin, R. Zhang, W. Zuo, and L. Zhang, "Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 4766–4779, Dec. 2015.

[33] K. Lin, H.-F. Yang, J.-H. Hsiao, and C.-S. Chen, "Deep learning of binary hash codes for fast image retrieval," in *Proc. Conf. Comput. Vis. Pattern Recognit. Workshops*, 2015, pp. 27–35.

[34] H. Lai, Y. Pan, Y. Liu, and S. Yan, "Simultaneous feature learning and hash coding with deep neural networks," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3270–3278.

[35] W.-J. Li, S. Wang, and W.-C. Kang. (Nov. 2015). "Feature learning based deep supervised hashing with pairwise labels." [Online]. Available: https://arxiv.org/abs/1511.03855

[36] H. Liu, R. Wang, S. Shan, and X. Chen, "Deep supervised hashing for fast image retrieval," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2064–2072.

[37] B. Kulis and T. Darrell, "Learning to hash with binary reconstructive embeddings," in *Proc. Neural Inf. Process. Syst.*, 2009, pp. 1042–1050.

[38] X. Zhang, L. Wang, S. Xiang, and C. Liu, "Retargeted least squares regression algorithm," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 9, pp. 2206–2213, Sep. 2015.

[39] V. Erin Liong, J. Lu, G. Wang, P. Moulin, and J. Zhou, "Deep hashing for compact binary codes learning," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 2475–2483.

[40] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large data set for nonparametric object and scene recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 11, pp. 1958–1970, Nov. 2008.

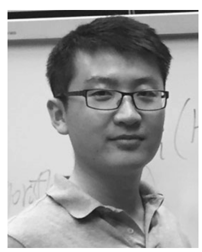
[41] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *Int. J. Comput. Vis.*, vol. 42, no. 3, pp. 145–175, 2001.

[42] P. J. Phillips *et al.*, "Overview of the face recognition grand challenge," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2005, pp. 947–954.

[43] B. Kulis and K. Grauman, "Kernelized locality-sensitive hashing for scalable image search," in *Proc. Int. Conf. Comput. Vis.*, 2009, pp. 2130–2137.

Jie Gui (SM'16) received the B.S. degree in computer science from Hohai University, Nanjing, China, in 2004, the M.S. degree in computer applied technology from the Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei, China, in 2007, and the Ph.D. degree in pattern recognition and intelligent systems from the University of Science and Technology of China, Hefei, in 2010.

He is currently an Associate Professor with the Hefei Institute of Intelligent Machines, Chinese Academy of Sciences. He is also a Visiting Researcher with the Shenzhen Key Laboratory of Media Security, Shenzhen University, Shenzhen, China. He has been a Post-Doctoral Fellow with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China. His current research interests include machine learning, pattern recognition, data mining, and image processing.



Tongliang Liu received the B.E. degree in electronic engineering and information science from the University of Science and Technology of China, Hefei, China, in 2012, and the Ph.D. degree from the University of Technology Sydney, Sydney, Australia, in 2016.

He is currently a Lecturer with the Faculty of Engineering and Information Technology, University of Technology Sydney. He has authored or co-authored over 10 research papers, including the IEEE TPAMI, T-NNLS, T-IP, NECO, ICML, KDD, IJCAI, and AAAI. His current research interests include statistical learning theory, computer vision, and optimization.

Dr. Liu received the Best Paper Award in the IEEE International Conference on Information Science and Technology 2014.

Zhenan Sun (M'07) received the B.S. degree in industrial automation from the Dalian University of Technology, Dalian, China, in 1999, the M.S. degree in system engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2002, and the Ph.D. degree in pattern recognition and intelligent systems from the Chinese Academy of Sciences (CASIA), Beijing, China, in 2006.

Since 2006, he has been a Faculty Member with the National Laboratory of Pattern Recognition, Institute of Automation, CASIA, where he is currently a Professor. He has authored/co-authored over 100 technical papers. His current research interests include biometrics, pattern recognition, and computer vision.

Dr. Sun is a member of the IEEE Computer Society and the IEEE Signal Processing Society. He is an Associate Editor of the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY and the IEEE BIOMETRICS COMPENDIUM.

Dacheng Tao (F'15) is currently a Professor with the School of Information Technologies, Faculty of Engineering and Information Technologies, The University of Sydney, Sydney, NSW, Australia. He mainly applies statistics and mathematics to data analytics problems. He has authored one monograph and over 200 publications at prestigious journals and prominent conferences, such as the IEEE T-PAMI, T-NNLS, T-IP, JMLR, IJCV, NIPS, ICML, CVPR, ICCV, ECCV, AISTATS, ICDM; and the ACM SIGKDD, with several best paper awards, such as the best theory/algorithm paper runner up award in the IEEE ICDM'07, the best student paper award in the IEEE ICDM'13, and the 2014 ICDM 10-year highest-impact paper award. His current research interests include spread across computer vision, data science, image processing, machine learning, and video surveillance.

Dr. Tao is a fellow of the OSA, IAPR, and SPIE. He received the 2015 Australian Scopus-Eureka Prize, the 2015 ACS Gold Disruptor Award, and the 2015 UTS Vice-Chancellors Medal for Exceptional Research.

Tieniu Tan (F'03) received the B.S. degree in electronic engineering from Xi'an Jiaotong University, Xi'an, China, in 1984, and the M.S. and Ph.D. degrees in electronic engineering from the Imperial College London, London, U.K., in 1986 and 1989, respectively.

In 1989, he joined the Department of Computer Science, University of Reading, Reading, U.K., where he was a Research Fellow, a Senior Research Fellow, and a Lecturer. In 1998, he joined the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences (CAS), Beijing, China, as a Full Professor. He was the Director General of the Institute of Automation, CAS, from 2000 to 2007 and the Director of NLPR from 1998 to 2013. He is currently the Director of the Center for Research on Intelligent Perception and Computing, Institute of Automation, and the Deputy President of CAS. He has authored over 500 research papers in refereed international journals and conferences in the areas of image processing, computer vision, and pattern recognition. He has authored or edited 11 books. He holds over 70 patents. His current research interests include biometrics, image and video understanding, and information forensics and security.

Dr Tan is a member (Academician) of the Chinese Academy of Sciences, a fellow of The World Academy of Sciences for the advancement of sciences in developing countries (TWAS), an International Fellow of the U.K. Royal Academy of Engineering, and a fellow of the IAPR (the International Association of Pattern Recognition). He is an Editor-in-Chief of the *International Journal of Automation and Computing*. He has given invited talks and keynotes at many universities and international conferences, and has received numerous national and international awards and recognitions.