Unsupervised Deep Learning of Compact Binary Descriptors

Kevin Lin, Jiwen Lu, Chu-Song Chen, Jie Zhou, and Ming-Ting Sun

Abstract—Binary descriptors have been widely used for efficient image matching and retrieval. However, most existing binary descriptors are designed with hand-craft sampling patterns or learned with label annotation provided by datasets. In this paper, we propose a new unsupervised deep learning approach, called DeepBit, to learn compact binary descriptor for efficient visual object matching. We enforce three criteria on binary descriptors which are learned at the top layer of the deep neural network: 1) minimal quantization loss, 2) evenly distributed codes and 3) transformation invariant bit. Then, we estimate the parameters of the network through the optimization of the proposed objectives with a back-propagation technique. Extensive experimental results on various visual recognition tasks demonstrate the effectiveness of the proposed approach. We further demonstrate our proposed approach can be realized on the simplified deep neural network, and enables efficient image matching and retrieval speed with very competitive accuracies.

Index Terms—Binary descriptors, unsupervised learning, deep learning, convolutional neural networks

1 Introduction

Feature descriptors have been widely used in numerous computer vision tasks [1], [2], [3], [4] such as object recognition, image classification and panorama stitching. A feature descriptor is desired to capture important and distinctive information in an image [1], [2] and also to be robust to various image transformations such as rotation and scaling [2], [5]. On the other hand, a highly efficient descriptor enables fast retrieval of images from a large corpus [6].

Over the past decade, high quality descriptors such as the rich features learned from the deep Convolutional Neural Networks (CNN) [7], [8], and the representative SIFT descriptor [2], have been widely explored. These descriptors demonstrate superior discriminability, and bridge the gap between low-level pixels and high-level semantic information [9]. However, they are high-dimensional real-valued descriptors, and usually require high computational cost to search for matched images. In order to reduce the computational complexity in matching, several lightweight binary descriptors have been proposed such as BRIEF [10], ORB [11], BRISK [12], and FREAK [13]. These binary descriptors are highly efficient for storing and matching. Given compact binary descriptors, one can rapidly measure the similarity of the images by computing the Hamming distance between binary descriptors via XOR bitwise operations. Since these early binary descriptors are computed by hand-crafted sampling patterns or simple intensity comparisons, they are usually unstable and sensitive to scales, rotations, and noises. Some previous works [14], [15], [16], [17] improved the binary descriptors by encoding the similarity information into binary codes with learning algorithms. These approaches formulate the problem as a learning-to-match optimization by encouraging similar patches to have similar binary descriptors. These methods learn similarity relationship within the image pairs by taking the advantage of supervised learning and pair-wised labels (e.g. matching and non-matching labels). With the supervised learning strategy, the learned binary descriptors are highly distinctive and partially invariant to lighting conditions and camera viewpoints. However, the success of these methods is mainly attributed to similarity labels. In other words, these methods are unfavorable in the case when training data do not have label annotations.

In this paper, we propose an unsupervised learning approach, dubbed DeepBit, for learning compact binary descriptors using a deep neural network. The basic idea behind our approach is that the binary descriptors are computed by applying a set of non-linear projection functions to the image and the projection functions can be learned without the label annotations provided by the dataset. On the basis of the idea, we enforce three important criteria on the top layer of the deep neural network, and optimize the parameters of the network with back-propagation. The binary descriptors are learned by minimizing the quantization error between real-valued deep features and binary descriptors. To make the binary descriptors more discriminative, we impose the designed constraint on the binary descriptors which encourages the binary descriptors to convey information as much as possible and to be partially invariant to geometric transformation. Fig. 1 shows the basic idea of our proposed approach. We employ our approach on different visual analysis tasks including patch matching, image and patch retrieval, as well as fine-grain object recognition.

- *K. Lin and M.-T. Sun are with the Department of Electrical Engineering, University of Washington, Seattle, USA. E-mail: kvlin@uw.edu; mts@uw.edu*
- *J. Lu and J. Zhou are with the Department of Automation, Tsinghua University, State Key Lab of Intelligent Technologies and Systems, Tsinghua National Laboratory for Information Science and Technology (TNList), Beijing, 100084, China. E-mail: lujiwen@tsinghua.edu.cn; jzhou@tsinghua.edu.cn*
- *C.-S. Chen is with the Institute of Information Science, Academia Sinica, Taipei, Taiwan. E-mail: song@iis.sinica.edu.tw*

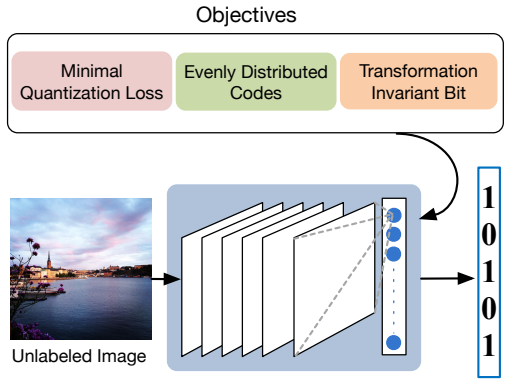


Fig. 1. The basic idea of our proposed method. We enforce three criteria, and optimize the parameters of the network with back-propagation. Our approach dose not require labels provided by the dataset, and is more practical to real-world applications in comparison to supervised binary descriptors. This figure shows the inference stage of our approach. Given an input image, our network encodes it as a binary feature descriptor.

Experimental results on several public benchmark datasets demonstrate the effectiveness of the proposed approach.

This paper is an extended version of our conference paper [18]. We make the following new contributions to the conference version. (1) A new objective function is proposed that encodes important characteristics of local descriptors into the binary descriptors, including rotation, translation, and scaling invariance. (2) We provided more in-depth discussion and analysis about our approach. We conducted extensive experiments on several pubic benchmark datasets including a large dataset with more than 1 million natural images. (3) We further demonstrate our approach can be realized on the simplified deep neural networks for efficient object matching.

2 Related Work

Binary Descriptors: Earlier works related to binary descriptors can be traced back to BRIEF [10], ORB [11], BRISK [12], and FREAK [13]. These binary descriptors are built upon hand-crafted sampling patterns, and a set of pairwise intensity comparisons. While these descriptors are efficient, their performance is limited because pairwise intensity comparison is sensitive to the scale and geometric transformation. To address these limitations, several supervised approaches [15], [16], [19], [20], [21], [22], [23] have been proposed to learn binary descriptors. D-BRIEF [19] encodes the desired similarity relationships and learns a project matrix to compute discriminative binary features. On the other hand, Local Difference Binary (LDB) [24] applies Adaboost to select optimal sampling pairs. Linear Discriminat Analysis (LDA) is also applied to learn binary descriptors [21]. Recently proposed BinBoost [20] learns a set of projection matrix using the boosting algorithm, and achieves state-of-the-art performance on patches matching. In addition, Supervised Discrete Hashing (SDH) [25] learns binary codes with the minimal classification loss of a linear classifier. Supervised Incremental Hashing (SIH) [26] jointly optimizes the classification error and the binary codes learning in a supervised manner. While these approaches have achieved impressive performance, their success is mainly attributed to pair-wise learning with similarity labels, and is unfavorable for the case when transferring the binary descriptor to a new task.

Unsupervised hashing algorithms learn compact binary descriptors whose distance is correlated to the similarity relationship of the original input data [27], [28], [29], [30]. Locality Sensitive Hashing (LSH) [27] applies random projections to map original data into a low-dimensional feature space, and then performs a binarization. Semantic hashing (SH) [29] builds a multi-layers Restricted Boltzmann Machines (RBM) to learn compact binary codes for text and documents. Spectral hashing (SpeH) [30] generates efficient binary codes by spectral graph partitioning. Iterative qauntization (ITQ) [28] uses iterative optimization strategy to find projections with minimal binarization loss. Even if these approaches have been proved effective, the binary codes are still not as accurate as the real-valued equivalents.

Deep Learning: Deep learning has drawn increasing attention in visual analysis since Krizhevsky*et al.* [7] demonstrated the outstanding performance of the deep CNN on the $1,000$ class image classification. Their success is attributed to training a deep CNN to learn rich mid-level image representations on millions of images. Recent studies [31], [32], [33], [34] show that training a Siamese deep network with contrastive loss based on pair-wised or triplet image pairs achieves superior performance to SIFT for the task of local patch matching or retrieval. Dosovitskiy *et al.* [35] showed that it is possible to learn discriminative real-valued descriptors with the surrogate patch labels. Paulin *et al.* [36] and Balntas *et al.* [37] showed that the success of the deep descriptors learning is mainly attributed to the convolution operation that extract local information of the patches. However, the descriptors learned by the above mentioned approaches are generally high-dimensional real valued descriptors, which require relatively high computational cost image matching. In contrast, the proposed binary descriptor not only reduces the computational complexity, but also enable efficient image and object matching.

Recent studies [38] have shown that deep learning is effective for binary codes learning. For example, CNNH [39], DNNH [40] and DSH [41] employed deep CNN to learn a set of hash functions. However, their methods require pair-wised similarity labels or triplets training data. VDSH [42] proposed to learn binary codes via layer-wise local updates, which potentially avoids the gradient vanishing problem. Instead of using the sign function, BDNN [43] directly generated binary codes with the designed constraints on the output layer. Huang *et al.* [44] proposed to jointly learn feature representations with unsupervised discriminative clustering and weakly-supervised hashing. Deep Hashing (DH) [45] builded a three-layer hierarchical neural network to learn the discriminative projection matrix, but their method does not take the advantage of deep transfer learning, thus makes the binary codes less effective. In contrast, the proposed DeepBit not only transfers the mid-level image representations pre-trained from ImageNet to the target domain, but also learns compact yet discriminative binary descriptor. We will show that our method achieves better or comparable performance than state-of-the-art descriptors on several public benchmark datasets.

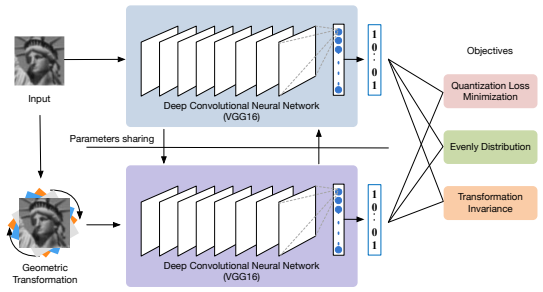


Fig. 2. An overview of the proposed unsupervised deep learning framework for computing binary descriptors. We create a new fully-connected layer on the top of the existing deep neural network, and employ the proposed objective functions to learn compact yet discriminative binary descriptor. The proposed approach takes two inputs from the original image and the geometric transformed one into two towers of our network, respectively. Both of the networks share and update the same parameters. Then, we learn binary descriptors through the optimization of the proposed objective functions. Note that the binary descriptor is computed by applying only one tower of our network in the test phase.

3 Proposed Approach

Fig. 2 shows the learning framework of our proposed method. We introduce an unsupervised deep learning approach, called DeepBit, to learn compact yet discriminative binary descriptors. Unlike previous works [14], [15], [19], [20] that optimize the projection matrix with hand-crafted features and label information from datasets, DeepBit learns a set of non-linear projection functions to compute compact binary descriptors in an unsupervised manner. We enforce three important criteria on the binary descriptors, and optimize the parameters of the proposed network with the stochastic gradient descent technique. In this section, we first give an overview of our approach, and then describe the proposed learning objectives in the following sections.

3.1 Overall Learning Objectives

The proposed DeepBit aims to learn K projection functions that map each image x_n into the binary vector $b_n = [b_{n1}, b_{n2}, \ldots, b_{nK}] \in \{0,1\}^{1\times K}$. $\mathcal{F}_k(\cdot)$ and W_k represent the k-th projection function and its associated parameter set. The projection function $\mathcal{F}_k(x_n; W_k)$ is a composition of a number of non-linear projections which can be written as:

$$\mathcal{F}_k(x_n; W_k) = f_{kL}(\cdots f_{k2}(f_{k1}(x_n; w_{k1}); w_{k2})\cdots; w_{kL}), \tag{1}$$

where L is the total number of layers in the deep neural network, and $W_k = [w_{k1}, w_{k2}, \ldots, w_{kL}]$. Note that w_{kl} represents the projection parameter learnt for the l-th layer of the deep neural network. Sigmoid activation function is applied on the last layer, thus $\mathcal{F}_k(\cdot) \in \mathbb{R}$ and $0 \leq \mathcal{F}_k(\cdot) \leq 1$. The k-th bit of the binary descriptor b_{nk} is computed by applying the projection function $\mathcal{F}_k(\cdot)$ to the input image x_n and binarizing the results:

$$b_{nk} = 0.5 \times (\mathrm{sign}(\mathcal{F}_k(x_n; W_k) - 0.5) + 1), \tag{2}$$

where $\mathrm{sign}(v) = 1$ if $v > 0$ and -1 otherwise.

Let $W = [W_1, W_2, \ldots, W_K]$, which consists of K parameters for computing the K-bit binary descriptor. To learn the K-bit binary descriptor, we define three important criteria to optimize W. First, the learned binary descriptor should preserve the information of the feature extracted from the last layer of the deep CNN. The quantization loss should be as small as possible after projection. Second, we encourage the binary descriptor to be evenly distributed, so that each bit of the binary descriptor will convey information as much as possible. The third is to make the descriptor invariant to geometry transformations.

To achieve these objectives, we formulate the following optimization problem to learn W:

$$\begin{aligned}\min_W E(W) &= \alpha E_1(W) + \beta E_2(W) + \gamma E_3(W) \\ &= \alpha \sum_{k=1}^{K}\sum_{n=1}^{N} ||b_{nk} - \mathcal{F}_k(x_n; W_k)||^2 \\ &+ \beta \sum_{k=1}^{K} ||\mu_k - 0.5||^2 \\ &+ \gamma \sum_{k=1}^{K}\sum_{n=1}^{N} \{(y)d + (1-y)(K-d)\},\end{aligned} \tag{3}$$

where N is the number of training data in a mini-batch. K is the bit length of the binary descriptor. μ_k is the mean of the k-th bit binary descriptor of the training data. d denotes the distance between the compared binary descriptors b_p and b_q. Note that b_p and b_q are computed from input image x_p and x_q. y denotes the relationship between the compared binary descriptors. $y = 1$ if $x_q = T(x_p)$, and $y = 0$ if $x_q \neq T(x_p)$. Note that $T(\cdot)$ computes the geometric transformation. Moreover, α, β, and γ are the hyper-parameters to balance different objectives.

To give a better understanding of the proposed objectives, we describe the physical meaning of (3) as below. First, E_1 minimizes the quantization loss between the binary descriptor and the input feature vector. Then, E_2 encourages the binary descriptor to be evenly distributed to increase the information capacity of the binary descriptor. Finally, E_3 tolerates the geometry transformations by minimizing the Hamming distance between the descriptors that describe the reference image and the transformed ones. We elaborate the details of each proposed objective as follows.

3.2 Quantization Loss Minimization

The proposed DeepBit seeks to learn the projections that map the input image into a binary string while preserving the discriminative information of the original input. We first rewrite (2) as follows:

$$b_{nk} - 0.5 = 0.5 \times \mathrm{sign}(\mathcal{F}_k(x_n; W_k) - 0.5). \tag{4}$$

The soul idea to keep the binary descriptors informative is to minimize the quantization loss between the binary descriptor and the rich image representation. We minimize the quantization loss of the k-bit binary descriptor as below:

$$\begin{aligned}\min_{W_k} E_1(W_k) &= \sum_{n=1}^{N} ||0.5 \times \mathrm{sign}(\mathcal{F}_k(x_n; W_k) - 0.5) \\ &- (\mathcal{F}_k(x_n; W_k) - 0.5)||^2 \\ &= \sum_{n=1}^{N} ||b_{nk} - 0.5 - \mathcal{F}_k(x_n; W_k) + 0.5||^2 \\ &= \sum_{n=1}^{N} ||b_{nk} - \mathcal{F}_k(x_n; W_k)||^2.\end{aligned} \tag{5}$$

The smaller the quantization loss is, the better the binary descriptor will preserve the information of the original data. Different from the previous work [28] that addresses this problem by iteratively updating W_k and b_{nk} with two alternating steps, we formulate this optimization problem as the neural networks training objective. The goal of the proposed deep neural network becomes learning W_k that minimizes the quantization loss between b_{nk} and $\mathcal{F}_k(x_n; W_k)$. To this end, we optimize $W = [W_1, W_2, \ldots, W_K]$ for the K-bit binary descriptor by minimizing the loss of the following objective function:

$$\min_W E_1(W) = \sum_{k=1}^{K} \sum_{n=1}^{N} ||b_{nk} - \mathcal{F}_k(x_n; W_k)||^2. \quad (6)$$

3.3 Learning Efficient Binary Descriptors

To increase the information capacity of the binary descriptors, we maximize the usage of each bin in the binary string. Considering the variance for each bin, the higher the entropy is, the more information the binary codes express. Accordingly, we enhance the binary descriptor by making each bit has 50% probability of being one or zero. In other words, there is no preference for each bit to be one or zero, and the resulting binary string will convey the information as much as possible. To achieve this goal, we minimize the loss computed by the forward pass of the network:

$$\min_W E_2(W) = \sum_{k=1}^{K} ||\mu_k - 0.5||^2, \quad (7)$$

where K is the bit length of the binary descriptor. For each bit, we compute mean μ_k using:

$$\mu_k = \frac{1}{N} \sum_{n=1}^{N} b_{nk}, \quad (8)$$

where N is the number of images in a mini-batch.

3.4 Learning Transformation Invariant Descriptors

Previous studies [1], [2], [4], [5] show that an effective local descriptor should not only be distinctive, but also invariant to geometry transformations. The local descriptors are desired to have three important properties including (1) rotation invariance, (2) translation invariance, and (3) scaling invariance. To learn binary descriptors that are invariant to these transformations, we address the problem by minimizing the difference between the binary descriptors which are computed from the reference image and the transformed one. The transformed image is computed by applying a set of rotation, translation and scaling functions. In order to make the binary descriptor more distinctive, we further enhance the binary descriptors by increasing the distance between the descriptors computed from arbitrary images. Inspired by Dosovitskiy *et al.* [35] that incorporate surrogate labels (or pseudo labels) for unsupervised learning, we formulate the optimization problem as the loss function below:

$$\min_W E_3(W) = \sum_{n=1}^{N} \{(y)d + (1-y)(K-d)\}, \quad (9)$$

where $d = ||b_p - b_q||$. b_p and b_q are the binary descriptors computed from training data x_p and x_q, respectively. y reveals the transformation relationship between x_p and x_q. $y = 1$ if $x_q = T(x_p)$, and $y = 0$ if $x_q \neq T(x_p)$. $T(\cdot)$ computes the geometric transformation with random combination of rotation, scaling, and translation functions. During learning, we encode the important characteristics of local descriptors into the top layer of the deep neural network, including rotation, translation, and scaling invariance. Then, the binary descriptors are learned by optimizing the network parameters through back-propagation. Since y in (9) indicates the geometric transformation *on* or *off*, the objective function does not take into account the label annotations provided by the datasets. Comparing to the objective function in our conference paper [18], the proposed loss function (9) is general for learning binary descriptors.

3.5 Implementation Details

We implement our proposed approach using the open source Caffe [46]. The proposed approach includes two stages. First, we initialize our network with the parameters from the VGG16 [47], which is trained on the ImageNet dataset. Second, we replace the classification layer of the VGG16 with a new fully-connected layer following by the sigmoid activation function, and enforce the neurons in the new fully-connected layer to learn binary descriptor. The proposed method learns binary descriptors based on the Siamese-like network architecture, which consists of two towers of our network. During learning, we optimize W using the proposed objective function (see (3)). Note that the two towers of the network share and update the same parameter W. To this end, we use the stochastic gradient descent (SGD) method and back-propagation to train our network. The bit-length of our binary descriptor can be customized from dataset to dataset by setting the number of neurons in the last fully-connected layer. We generate a training list with $2M$ pairs of x_p and x_q, where M is the total number of images in the dataset. We shuffle the training list, and feed to the network batch by batch. Other settings are listed below. Mini-batch size is 32. We train our model with a learning rate 0.0001, a momentum 0.9 and a weight decay 0.0005. μ_k in E_2 is replaced with the mean of the minibatch during training. Images are normalized to 256×256 and then randomly cropped to 224×224 for VGG16 as the network input. For the parameters of the geometric transformation function, the rotation range is from -10 to 10, shifting range is from -32 to 32 pixels, and the scaling factor is from 0.8 to 1.2. Note that we empirically set these parameters for simulating human perspective with small variations of viewpoints, and the performance of the binary descriptors could be further boosted by using optimal parameter settings via hyperparameter selections. The source codes will be publicly available[1].

4 Experimental Results

We conduct experiments on several public datasets as well as a large-scale dataset with more than one million natural images. We provide extensive evaluations of the proposed

1. https://sites.google.com/site/kevinlin311tw/

TABLE 1
Statistics of datasets used in the experiments.

Dataset	Image type	Label	Training	Test
Brown	64×64 patches	Match or not	200,000	100,000
RomePatches	51×51 patches	Match or not	10,000	10,000
CIFAR-10	32×32 patches	10 class	50,000	10,000
Flower-17	Natural images	17 class	680	340
Flower-102	Natural images	102 class	1,020	6,149
ILSVRC2012	Natural images	1,000 class	$\sim$1.2 M	50,000
Oxford5k	Landscape images	N/A	N/A	55
Paris6k	Landscape images	N/A	N/A	55
Holidays	Natural images	N/A	N/A	500
UKB	Object images	2,550 class	N/A	10,200

binary descriptor, and demonstrate its performance on various visual recognition tasks. We start with introducing the datasets and then present the comparative evaluations with the state-of-the-art methods. Information of the datasets can be found in Table 1.

4.1 Datasets

Brown [48] is a standard dataset for evaluating local descriptors. It consists of three subsets, which are collected from the Photo Tourism reconstructions from three landmarks (Liberty, Notre Dame, and Yosemite). Each of them contains different views of a given landmark. For each landmark, there are more than $400,000$ gray-scale patches, resulting in a total of $1,200,000$ patches. Each subset is split into training and test sets, with $200,000$ training pairs ($100,000$ matched and $100,000$ non-matched pairs) and $100,000$ test pairs ($50,000$ matched, and $50,000$ non-matched pairs), respectively.

CIFAR-10 [49] is one of the most widely used datasets for evaluating classification and retrieval. It contains 10 object categories and each class consists of $6,000$ images, resulting in a total of $60,000$ images. The dataset is split into training and test sets, with $50,000$ and $10,000$ images, respectively. We employ the training set for network training, and use the test set as the queries for retrieval evaluation.

RomePatches [36] is another dataset for descriptor evaluation. Different from Brown dataset, RomePatches consists of a series of local patches with color information. The dataset is collected from the 3D reconstruction of several landmarks in Rome. The patches are obtained by projecting the 3D feature points back to the 2D original images. For both training and test sets, there are $1,000$ 3D feature points with 10 different views, resulting in a total of $10,000$ training patches and $10,000$ test patches.

ILSVRC2012 [50] is a standard benchmark for the ImageNet Large Scale Visual Recognition Challenge. It consists of $1,000$ object classes, and approximately 1.2 million training images, $50,000$ validation images and $100,000$ test images. Similar to the setting in CIFAR-10, we employ the training set to learn network parameters, and use the validation set as the queries in the evaluation.

Flower-17 [51] contains 17 categories and each class consists of 80 images, resulting in a total of $1,360$ images. The dataset is split into the training (40 images per class), validation (20 images per class), and test (20 images per class) sets.

Flower-102 [52] contains 102 categories and each class consists of $40 \sim 102$ images, resulting in a total of $8,189$ images. The dataset is split into the training (10 images per class), validation (10 images per class), and test (the rest $6,149$ images) sets.

Oxford [53] has $5,062$ images of 11 Oxford landmarks. Images are with different variations in viewpoints and scales, which pose practical challenges for image retrieval. In this dataset, 55 queries are used for the evaluation.

Paris [54] has $6,412$ images of Paris landmarks. Similar to Oxford, there are 55 queries for the retrieval evaluation.

INRIA Holidays [55] has $1,491$ images corresponding to 500 image groups. The images are with various rotations and scales, making the retrieval more challenging. The performance is evaluated on 500 queries.

UKB [56] contains $2,550$ object categories and each object has 4 images in different viewpoints, resulting in a total of $10,200$ object images.

4.2 Results on Brown Dataset

Comparison with unsupervised approaches: To evaluate the performance of local descriptors, we first compare our method with state-of-the-art unsupervised binary descriptors (DBD-MQ [57], BRIEF [10], BRISK [12], Boosted SSC [58]). Following the standard evaluation protocol [14], we conduct cross-validation on the three subsets, and compute the false positive rate at 95% recall rate (95%ERR). We train our model on one subset, and apply to the other two subsets. Thus, there are in a total of 6-round validations. The validations can be seen as the cross-scene evaluation since the three subsets are collected from different environments including natural scene, statue, and architecture, respectively. Fig. 3 shows the ROC curves for our proposed model and the compared methods, and Table 2 summarizes the 95% ERR for the Brown dataset. The overall performance of our model achieves 38.15% error rate when the recall rate is at 95%, which achieves lower error rates than the compared unsupervised binary descriptors over different training and testing configurations of the Brown dataset. In addition, we study the influence of the loss function designed for the transformation invariance objective. As shown in Table 2, our model learned with the transformation invariance objective achieves 38.15% error rate, which is better than 40.67% of our conference paper version [18]. Since the proposed new objective function is more general for descriptor learning, our new model performs more favorably against the previous one.

Comparison with supervised approaches: We also compare the proposed unsupervised binary descriptors with the state-of-the-art supervised binary descriptors (Binary L2-Net [59], BinBoost [14], RFD [15] and others). In Table 2, supervised approaches employ similarity supervision (matched and non-matched labels) to optimize the model training, so that their binary descriptors are more effective for estimating the visual similarity given the pair-wised input patches. Since our method is unsupervised, where the learning process does not take advantage of the training labels provided by the dataset, our method performs less favorably than the supervised approaches.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2833865, IEEE Transactions on Pattern Analysis and Machine Intelligence

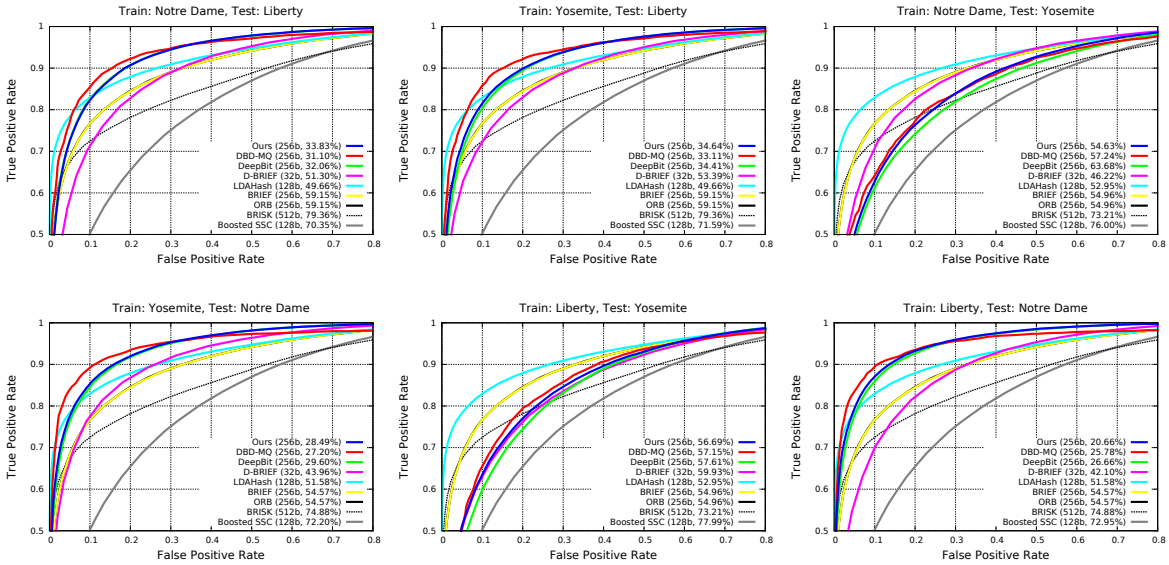


Fig. 3. ROC curves of the proposed DeepBit descriptors and the compared binary descriptors, across all the splits of training and testing configurations on the Brown datasets. In parentheses: the bit length of the binary descriptor *(b)*, and the 95% error rates.

TABLE 2
Comparison of the proposed binary descriptor to the state-of-the-art binary descriptors, in terms of 95% error rates (ERR) across all the splits of training and testing configurations. For reference, we also provide the results of real-valued descriptor SIFT [2]. The proposed method achieves better performance than the unsupervised binary descriptors in most cases, while remaining competitive to supervised approaches.

Method	Dimension	Type	Supervised	Yosemite Notre Dame	Yosemite Liberty	Notre Dame Yosemite	Notre Dame Liberty	Liberty Notre Dame	Liberty Yosemite	Average 95% ERR
SIFT [2]	128	Float	N	28.09	36.27	29.15	36.27	28.09	29.15	31.17
MatchNet [32]	4096	Float	Y	5.67	10.77	8.39	6.9	3.87	10.88	7.74
DeepCompare [31]	256	Float	Y	2.11	7.20	4.10	4.85	1.90	5.00	4.19
BRISK [12]	512	Binary	N	74.88	79.36	73.21	79.36	74.88	73.21	75.81
BRIEF [10]	256	Binary	N	54.57	59.15	54.96	59.15	54.57	54.96	56.23
ORB [11]	256	Binary	N	54.57	59.15	54.96	59.15	54.57	54.96	56.23
DeepBit [18]	256	Binary	N	29.60	34.41	63.68	32.06	26.66	57.61	40.67
DBD-MQ [57]	256	Binary	N	27.20	33.11	57.24	31.10	25.78	57.15	38.59
Boosted SSC [58]	128	Binary	Y	72.20	71.59	76.00	70.35	72.95	77.99	73.51
LDAHash [21]	128	Binary	Y	51.58	49.66	52.95	49.66	51.58	52.95	51.40
D-BRIEF [19]	32	Binary	Y	43.96	53.39	46.22	51.30	43.10	47.29	47.54
BinBoost [14]	64	Binary	Y	14.54	21.67	18.96	20.49	16.90	22.88	19.24
RFD [15]	293-598	Binary	Y	11.68	19.40	14.50	19.35	13.23	16.99	15.86
Binary L2-Net [59]	256	Binary	Y	2.51	6.65	4.04	4.01	1.90	5.61	4.12
Ours	256	Binary	N	28.49	34.64	54.63	33.83	20.66	56.69	38.15

Visualization of patch matching: We further visualize the image matching results on the Brown dataset, and Fig. 5 shows the visualization results. As can be seen, our model successfully matches pairs of patches when they are visually similar, as shown in the first three columns of Fig. 5. The proposed method could also mismatch some patches as shown in the fourth column of Fig. 5. It is worth noting that the mismatched patches are still visually similar although they are from different scenes or locations. More specifically, the patches from Liberty and Notre Dame describe the local structure of the statue and architecture, where the visual similarity between different patches is usually weak. The proposed method achieves more favorable performance in these two datasets. However, the patches from Yosemite depict the surface of a mountain. Different local patches (such as snow and forest) could generate visually similar patterns, making them difficult to be distinguished. This could be the reason why our method, which tends to match patterns that are visually similar, performs less favorable than some methods for the Yosemite dataset.

Matching descriptors with binary operation: To better understand the advantage of binary descriptors, we show that our method can quickly perform the similarity matching via computing the Hamming distance or other binary operations for a given binary string. Following [14], we show the matching time of different descriptors in Fig. 4. We see that binary descriptors are more efficient than the real-valued

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2833865, IEEE Transactions on Pattern Analysis and Machine Intelligence

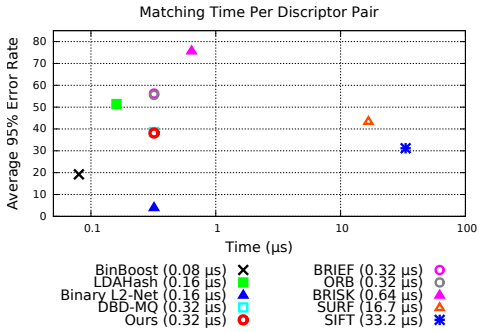


Fig. 4. Comparative evaluation of the matching time with different descriptors. The estimated computational cost were computed for 100K pairs from a test dataset on a desktop with the POPCOUNT instruction and averaged over 100 runs.

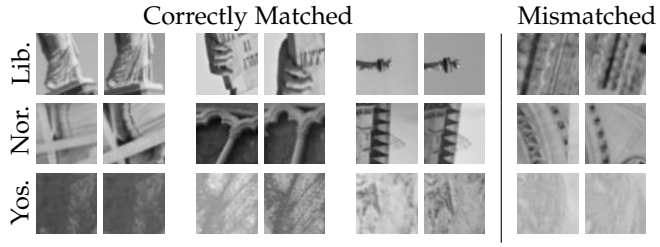


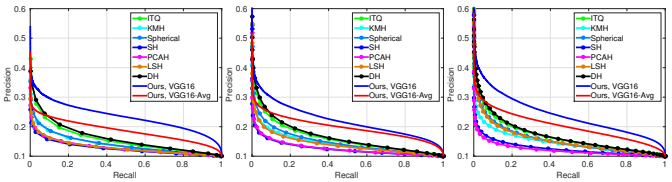
Fig. 5. Correctly matched patches and mismatched ones from the Brown dataset. Top row shows the patches from Liberty classified as matched pairs; the first three are correctly classified, but the fourth is mismatched, which describes different architectures. Middle row shows the image pairs from Notredame classified as the matched pairs; the fourth is mismatched although both of them share similar pattern. Bottom row shows the patches from Yosemite classified as matched pairs; the last one is mismatched, which are visually similar but belong to different locations.

descriptors (SIFT, SURF) in terms of the matching time. Among these binary descriptors, supervised descriptors (BinBoost, Binary L2-Net), which learn the similarity with supervision information, have lower error rates than these unsupervised binary descriptors. Our binary descriptor has comparable error rate and matching time in comparison to other unsupervised binary descriptors. We refer readers to Sec 4.9 for more discussions of the computational cost.

4.3 Results on CIFAR-10 Dataset

To evaluate the discriminability of the proposed binary descriptor, we further test our method on the task of image retrieval. We compare our method with several unsupervised hashing methods, including conventional hashing [27], [28], [30], and deep learning based approaches [43], [44], [45], [57] on the CIFAR-10 dataset. Following [45], we train the conventional approaches (such as ITQ) on the GIST descriptors. In addition, we also train PCA-ITQ and LSH on the ℓ_2-normalized feature extracted from the $fc7$ layer of the VGG16 network. These can be seen as the deep learning baselines.

Performance comparison: Following the settings in [45], Table 3 shows the retrieval results on CIFAR-10 in terms of the mean Average Precision (mAP) of all returned images of different methods. Fig. 6 shows the Precision/Recall curves of different unsupervised hashing methods with 16, 32, 64 bits, respectively. Our method achieves comparable or



(a) 16 bits (b) 32 bits (c) 64 bits

Fig. 6. Precision/Recall curves of different unsupervised hashing methods on the CIFAR-10 dataset with respect to 16, 32 and 64 bits, respectively.

TABLE 3
Performance comparison (mAP, %) of different unsupervised hashing algorithms on the CIFAR-10 dataset. This table shows the mean Average Precision (mAP) of all returned images with respect to different number of hash bits.

Method	16 bit	32 bit	64 bit
GIST + SpeH [30]	12.55	12.42	12.56
GIST + SH [29]	12.95	14.09	13.89
GIST + PCAH [60]	12.91	12.60	12.10
GIST + LSH [27]	12.55	13.76	15.07
GIST + PCA-ITQ [28]	15.67	16.20	16.64
VGG16 + LSH	10.67	10.57	10.03
VGG16 + PCA-ITQ	20.97	21.74	22.32
DH [45]	16.17	16.62	16.96
Huang *et al.* [44]	16.82	17.01	17.21
UH-BDNN [43]	17.83	18.52	-
Ours	**21.70**	**20.64**	**23.07**

better mAPs than these compared methods with respect to different bit lengths. This suggests that the proposed objective function encourages each bit of our binary descriptor to be activated, and makes the binary descriptors more efficient. Specifically, the VGG16 + PCA-ITQ approach computes PCA to obtain the feature embedding, and performs ITQ for better feature binarization. As discussed in the literature [28], [61], PCA is very effective to preserve semantic consistency for the small code sizes. Since our method learns the projection function without semantic-preserving supervision, our method performs less favorably than VGG16 + PCA-ITQ when the code size is small.

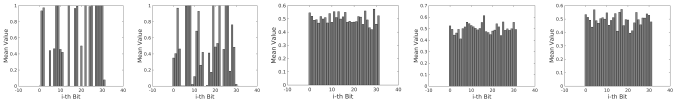
Following [18], [57], we report the mean Average Precision (mAP) of the top $1,000$ returned images of the recent state-of-the-art methods in Table 4. The results are consistent to previous finding. Our method achieves comparable or higher mAPs than the state-of-the-art deep learning approaches. Particularly, our method and DBD-MQ are initialized with the pre-trained parameters from ImageNet. The results show that network initialized with pre-trained parameters potentially avoids false local minima, and is useful to improve unsupervised learning of binary codes. Comparing to DBD-MQ which optimizes the binary descriptors with fine-grained multi-quantization, our proposed method is relatively robust against overfitting since our model produces binary codes with the standard sign function. It is worth noting that our method does not restrict the deep neural network architecture. We train our method on the VGG16-Avg, which is a simplified VGG16, and report the retrieval performance in Table 4. The retrieval performance is comparable to the counterparts. This suggests our pro-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2833865, IEEE Transactions on Pattern Analysis and Machine Intelligence

TABLE 4
Performance comparison (mAP, %) of different unsupervised hashing algorithms on the CIFAR-10 dataset. This table shows the mean Average Precision (mAP) of top $1,000$ returned images with respect to different number of hash bits.

Method	16 bit	32 bit	64 bit
VGG16 + LSH	10.55	11.39	10.80
VGG16 + PCA-ITQ	27.69	28.46	30.63
DeepBit [18]	19.43	24.86	27.73
DBD-MQ [57]	21.53	26.50	31.85
Ours	**26.36**	**27.92**	**34.05**

Method	16 bit	32 bit	64 bit
Ours, VGG16-Avg, before binarization	25.64	28.71	30.82
Ours, VGG16-Avg, after binarization	23.45	26.38	28.48
Ours, VGG16, before binarization	31.94	34.47	36.70
Ours, VGG16, after binarization	**26.36**	**27.92**	**34.05**

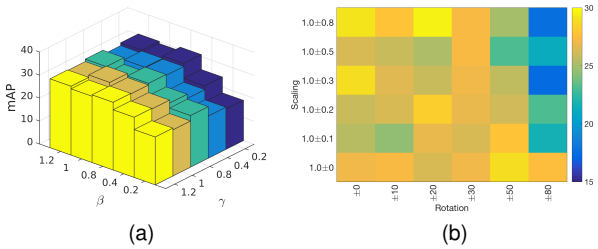


(a) $\beta = 0.2$ (b) $\beta = 0.4$ (c) $\beta = 0.8$ (d) $\beta = 1.0$ (e) $\beta = 1.2$

Fig. 7. Distribution of the binary codes when training with different weights of the objective function E_2. The generated binary codes are evenly distributed when the mean value for each bit is close to 0.5. The results show that our proposed objective function is effective for learning balanced binary codes.

posed method can be applied on the simplified network architectures for efficient computation. We refer readers to Sec 4.7.2 for more detailed configurations of the simplified networks. Moreover, we study how our model retain the discriminative ability of the real valued representations. In Table 4, we show the performance comparison of our binary codes and its real-valued representations before quantization. The loss of binarization is relatively small when training on the VGG16-Avg network. Particularly, the VGG16-Avg is a simplified network with less parameters compared to the original VGG16 network so that it is easier for our model training. When training with the VGG16 network, E_1 mitigates the quantization loss especially when the code length is longer.

Effectiveness of the learning objectives: We study the influence of individual terms of the proposed objective function (in Eq. (3)). The objective function consists of three terms that minimizing the binarization error, making codes evenly distributed, and learning transformation invariant bits. Since the quantization term E_1 in Eq. (3) is required to generate binary descriptors, we study how the other two objectives affect the performance. In particular, we fix α to 1.0, and iterate through all combinations of setting the parameter β and γ to 0.2, 0.4, 0.8, 1.0 and 1.2 on CIFAR-10 dataset. Fig. 8(a) shows the mAPs of the proposed method with different parameters. We see that E_2 is critical for learning effective binary codes since the retrieval performance is initially proportional to β, and converges to similar mAPs when β is higher than 0.8. We also see that our model has better mAPs given a higher γ, which indicates that E_3 is helpful to enhance the binary descriptors. Our model



(a) (b)

Fig. 8. Performance comparison (mAP, %) of DeepBit with different combinations of hyperparameters. The bit length is 32. (a) Different parameters of β and γ while α is set to 1, (b) Different scaling factors and rotation ranges.

achieves the highest mAP when $\{\alpha, \beta, \gamma\} = \{1.0, 1.0, 1.0\}$. Hence, combining all these terms together is beneficial to achieve better performance. In Fig. 8(b), we study the effect of different combinations of the rotation and scaling parameters. Particularly, we observe that our model achieves the highest mAP with ± 20 rotation degrees and the scaling factors from 0.2 to 1.8. It is worth noting that training with larger rotation ranges and larger scaling factors may introduce larger variation of the training images. This may reduce the visual similarity between the training images, and make the learning process of the proposed network more challenging. This is one of the key reasons that our model performs worse when training with large rotation ranges and large scaling factors. In Table 4, we compare our new model with the conference version of our work [18]. The proposed new model achieves more favorably mAPs than than the previous conference version, which means that our new model retrieves more relevant samples from the database. The results show that the transformation invariance objective is more general than the rotation invariant objective. Not only the learned feature space is discriminative to different images, but also the binary descriptors computed from the transformed images are enforced to be centralized.

Analysis of the generated binary codes: To further understand the learned binary feature space, we show the distribution of the generated binary codes on the test set of CIFAR-10 with different settings. We extract the binary codes where the bit length is 32, and Fig. 7 shows the mean value for each bit when training with different β. The results show that the binary codes are better activated when applying a larger β. On the other hand, it is worth noting that we train our model with mini-batches, and the parameter N in Eq. (8) is 32 in the experiments. Particularly, all the training images are used for updating the network parameters after a number of iterations, and the information in the training set are learned by the model. This indicates that E_2 is still effective to estimate the feature distribution for learning balance binary codes during training with mini-batches.

4.4 Results on RomePatches Dataset

RomePatches consists of a series of color patches for descriptor evaluation. For both training and test set, each of them has $1,000$ feature points that are viewed in 10 different orientations, resulting in $10,000$ local patches. Following the

TABLE 5
Performance comparison (mAP, %) of different binary descriptors on the RomePatches dataset. This table shows the mean Average Precision (mAP) of top $1,000$ returned patches.

Descriptor	Feature type	Bytes	mAP
CKN-grad [36]	Real-valued	1024	88.10
SIFT [2]	Real-valued	128	87.90
AlexNet-conv5 [7]	Real-valued	256	49.60
FREAK [13]	Binary	64	23.26
BRISK [12]	Binary	64	31.95
Ours	Binary	64	**53.04**
ORB [11]	Binary	32	43.83
Ours	Binary	32	**46.97**

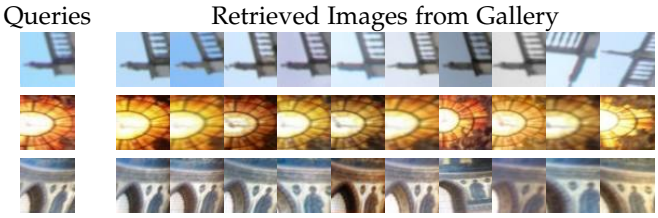


Fig. 9. Query patches and the corresponding top ranked 10 patches retrieved from RomePatches dataset. The code length of our binary descriptor is 512.

evaluation protocol in RomePatches dataset [36], we select $1,000$ feature points as the queries, and the $9,000$ remaining feature points as the targets. Then, we compute mAP for evaluation. As suggested in the CIFAR-10 experiments, we simply choose the hyper-parameters $\{\alpha, \beta, \gamma\} = \{1, 1, 1\}$ in Eq. (3) for evaluation.

We compare DeepBit with several local descriptors including real-valued descriptors (SIFT [2], CKN [36], and convolutional features [7]), and binary descriptors (ORB [11], FREAK [13], BRISK [12]). Following the evaluation protocol in [36], Table 5 shows the performance comparison. Real-valued descriptors demonstrate higher mAPs than the binary ones in most cases. Since convolutional feature (AlexNet-conv5) is only translation invariance, it may not be able to match relevant patches with different orientations. Comparing the binary descriptors when the bit length is 256 (32 Bytes), our method produces more favorably mAP than ORB. The result shows that our designed objective is helpful to enhance the binary descriptors. We find similar results when comparing with other 512 bits (64 Bytes) binary descriptors. DeepBit is capable of learning effective binary descriptors with different bit-lengths, which is consistent to the finding in the CIFAR-10 experiments.

4.5 Results on ILSVRC2012

ILSVRC2012 comprises more than 1.2 millions of images and $1,000$ object categories, which is more challenging then the above datasets compared. Different from the above experiments that the comparisons are made mostly with traditional learning approaches, we compare DeepBit with deep learning baselines in this experiment. Since our approach learns a set of non-linear projections that quantize the deep features to binary descriptors, we hence compare DeepBit with the combination of deep features + PCA-ITQ [28], which can be seen as the baseline in this experiment. PCA-ITQ [28] is one of the most representative quantization approaches for binary codes learning. Since the deep CNN models are pre-trained on ILSVRC2012, we directly extract the 4096 dimensional features from the last fully-connected layer of the networks, and apply PAC-ITQ for binary codes quantization. To compare with the baseline approach which does not take into account the geometric transformation, we train our binary descriptors by minimize the loss in Eq. (3) with the first two terms for fair comparison.

Fig. 10. Query images and their corresponding top 10 ranked images retrieved from ILSVRC2012 dataset. Upper two rows are the correct retrieval results. The bottom row shows our failure case. Images with red boarder are the false positives.

In the experiment, we randomly select $1,000$ images from the validation set as the queries, and use all the images in the training set (around 1.2 millions of images) to form a large-scale database. Then, we compute the mAP based on the top $1,000$ returned images. As can be seen in Table 6, DeepBit achieves higher mAPs than the compared methods. The results suggest our method is able to learn discriminative binary descriptors in a large corpus. In addition, DeepBit produces consistently better performance than the baselines according to the precision at top n samples. This suggests that our approach better centralizes the relevant images in the feature space. Comparing to the baseline method, our proposed approach which learns deep features and binary descriptors in a single framework can perform more favorably against the compared multi-stage approaches (deep features + PCA-ITQ). Moreover, we report the performance of our method when training on AlexNet in Table 6. AlexNet is another well-known deep neural network which consists of 5 convolutional layers following by two fully connected layers. The results are consistent to that of the VGG16. This indicates our approach can be realized with different deep network architectures.

Fig. 10 shows the retrieved top 10 images from the ILSVRC2012 dataset with our 512 bits binary descriptors. As can be seen, our approach is able to retrieve relevant images which are visually similar. Our method also centralizes binary descriptor of the relevant images when they are in different viewpoints. However, we notice some failure cases such as the bottom row in Fig. 10. The category of the query is Coucal and the returned images with red boarder are Black Grouse. It may be difficult for the binary descriptors to distinguish some object categories which have visually similar patterns.

4.6 Results on Instances Retrieval

To evaluate the robustness of the proposed method, we conduct experiments on the Oxford [53], Paris [54], INRIA Holidays [55], and UKB datasets [56] for instances retrieval applications, which include various scene types and the

TABLE 6
The mAP at top 1,000 returned images and precision at n samples of methods on the ILSVRC2012 validation set. The code size is 512.

Method	mAP (%)	prec. (%) at n samples					
		5	10	15	20	25	30
AlexNet + PCA-ITQ	31.21	43.01	41.56	40.65	39.96	39.37	38.88
VGG16 + PCA-ITQ	47.07	57.82	56.73	55.99	55.40	54.97	54.63
Ours, AlexNet	31.68	47.84	45.84	44.48	43.55	42.80	42.07
Ours, VGG16-Avg	46.28	70.78	67.16	63.90	60.43	56.62	52.86
Ours, VGG16	49.75	66.68	64.87	63.80	62.95	62.32	61.56

images are in different rotations, viewpoint and illumination changes. Since the object of interest may appear in different scales and viewpoints, these datasets present a challenging instance-level retrieval task. To apply our model on the instance-level retrieval, we follow the spatial search [8] approach, which divides the images into multiple local patches and measure the patch-level similarity. Specifically, the similarity between a query sub-patch and a gallery image is defined as the minimum among the Hamming distances of the query sub-patch and the gallery sub-patches. Then, the similarity between a query and a gallery image is defined as the average Hamming distance of every query sub-patches to the gallery image.

Following [62], we train our method on the Landmark dataset [62] for learning binary codes, and test our model on Paris, Oxford, and Holidays datasets. Table 7 shows the performance comparison with the state-of-the-art approaches. Among these compared approaches, ITQ [63] and Neural codes [62] compute binary codes for retrieval, and the rest approaches are based on real-valued features with 4096 [8], [64] or higher dimensions [65]. In Table 7, we show that deep learning approaches perform better than SIFT-based methods. Our method performs more favorably than the approaches based on other binary codes [62], [63] with the same code length. We also see that our method performs more favorably against CNN+aug+ss [8] in the Paris dataset. These results show that the proposed transformation invariant objective is effective for learning binary codes. Moreover, we conduct performance comparison on the UKB dataset [56]. Following the standard evaluation protocol in [56], Table 8 reports the precision at the top 4 returned images of different methods. The performance of the proposed method is comparable or slightly better than the counterparts, which indicates that the proposed method is relatively insensitive to the transformation variations of the input images.

4.7 Results on Fine-grained Recognition

Unlike previous binary descriptors that require matched/non-matched labels during training, the proposed DeepBit learns compact binary descriptors in an unsupervised manner; thus, DeepBit is flexible for various applications. In this section, we extend the evaluation to fine-grain recognition, which focuses on recognizing the subclasses of the same object category, e.g., recognizing Cowslip and Buttercup, which are both belong to flower category. Flower recognition is a classic visual analysis task, and it is challenging due to the variation of shapes, color distributions, and pose deformations. Besides,

TABLE 7
Performance comparison of instances retrieval performance (mAP, %) of different methods on Paris, Oxford, and Holidays datasets.

Method	Paris	Oxford	Holidays
SIFT-based methods			
BOW-200k-D [54]	46.0	36.4	54.0
IFV [65]	-	41.8	62.6
CNN-based methods			
ITQ, 256 bits [63], [66]	66.3	48.9	67.1
ITQ, 512 bits [63], [66]	66.8	50.8	73.9
Neural codes + PCA, 256 bits [62]	-	55.7	78.9
Neural codes + PCA, 512 bits [62]	-	55.7	78.9
CNN + aug + ss [8]	79.5	68.0	84.3
Ng *et al.* [64]	69.4	64.9	83.8
DF.FC1 + SL [9]	86.8	46.5	-
ReDSL.FC1 [9]	94.7	78.3	-
Ours+ss, 256 bits	82.5	60.3	81.8
Ours+ss, 512 bits	**82.9**	**62.7**	**82.7**

TABLE 8
Performance comparison (Precision, %) of the top 4 returned images of different methods on UKB dataset.

Method	Feature Type	Dimension	Precision
VGG16	Real-valued	4096	84.40
VGG16 + LSH	Binary	512	82.35
VGG16 + PCA-ITQ	Binary	512	82.25
Neural codes + PCA [62]	Binary	512	82.50
Ours	Binary	512	**82.55**

the computational cost and memory requirement become demanding while one wants to recognize the flowers in the wild using mobile devices. We show that the proposed binary descriptor performs more favorably against some basic real-valued descriptors such as HOG [67], and SIFT [2]. We further demonstrate that our approach can be realized on the compressed or simplified deep networks, and still maintains comparable recognition rate.

4.7.1 *Fine-grain recognition evaluation*

To get more insight about our binary descriptors, we train the multi-class SVM classifiers with different features such as Colour, SIFT, HOG and our binary descriptor. Then, we evaluate the quality of the feature representations according to the classification accuracy. In this experiment, we directly use our pre-trained DeepBit model presented in Sec.4.5 to compute the binary descriptors. Note that we use one descriptor per image for SVM training. Table 9 compares the classification accuracy of the 17 categories flowers using different descriptors proposed in [51], [52], including low-level (Colour, Shape, Texture), and high level (SIFT, and HOG) features. The results show that DeepBit is possible to achieve comparable or better fine-grain recognition rate than other existing features. We owe this to the fact that our approach optimizes the projection function, which maps the features extracted from the deep neural network to the binary string. In other words, our binary descriptor can be seen as the abstract of the rich feature representations computed from the deep neural network. Fig. 11a shows the confusion matrix results.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2833865, IEEE Transactions on Pattern Analysis and Machine Intelligence

TABLE 9
The categorization accuracy (mean%) for different features on the Flower-17 dataset.

Descriptors	Accuracy
Colour [51]	60.9
Shape [51]	70.2
Texture [51]	63.7
HOG [67]	58.5
HSV [52]	61.3
SIFT-Boundary [52]	59.4
SIFT-Internal [52]	70.6
DeepBit, 512 bits, AlexNet	**89.2**
DeepBit, 512 bits, VGG16	87.5

TABLE 10
The categorization accuracy (mean%) for different features on the Flower-102 dataset.

Descriptors	Accuracy
HSV [52]	43.0
SIFT-Internal [52]	55.1
SIFT-Boundary [52]	32.0
HOG [67]	49.6
BOW [68]	65.5
CNN+SVM [8]	74.7
DeepBit, 512 bits, AlexNet	**61.9**
DeepBit, 512 bits, VGG16	58.6

We also test our binary descriptor on a large flower dataset with 102 categories. Table 10 reports the accuracy of the SVM classifier when training with different feature descriptors, and Fig. 11b shows the confusion matrix of the 102 flower classes. Our binary descriptor (computed based on AlexNet) achieves 61.9% accuracy, which is comparable to many descriptors but worse than CNN-SVM [8] and BOW [68]. However, [8] and [68] employ high-dimensional real-valued features for classifier training. CNN-SVM [8] takes the original 4096 dimensional deep features computed from AlexNet. BOW [68] involves a series of pre-processing, including image segmentation to remove the background regions, and enhancing the descriptors by combining multiple features resulting in 4000 dimensional floating-valued descriptors. In contrast, our approach is more efficient because our descriptors are only of 512-bit binary codes.

4.7.2 Learning with simplified deep neural networks

Since the ultimate goal of binary descriptors is efficient matching, we investigate the possibility of learning binary codes with the simplified deep neural network. We adopt two more network architectures, SqueezeNet [69] and VGG16-Avg (of our own design [70]), for fine-grain flower recognition. Table 11 summarizes the required storage size and the number of parameters of different network designs. SqueezeNet [69] decreases the parameters of AlexNet by replacing 3×3 filters with 1×1 filters, and maintains the accuracy with large activation map by max pooling with stride of 2 in the deeper layers. VGG16-Avg [70] reduces the parameters in VGG16 by replacing the fully-connected layers with an average polling layer. Since the average polling layer preserves the spatial information from the previous convolutional layers, the reduced features are still effective for visual recognition.

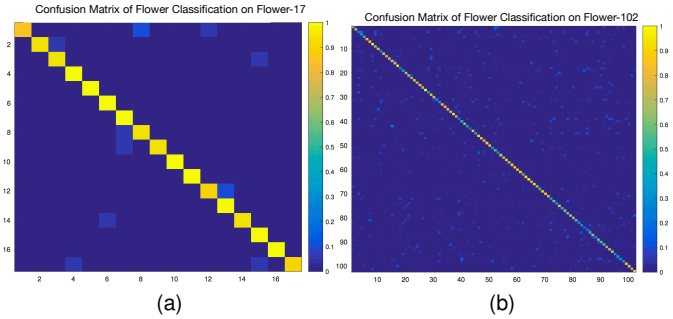


Fig. 11. Confusion matrix of flower classification using the proposed binary descriptor. (a) Flower-17 dataset. (b) Flower-102 dataset.

TABLE 11
Parameters and storage size of different network models implemented with CAFFE.

	DeepBit-512			
	AlexNet	VGG16	VGG16-Avg	SqueezeNet
# parameters	57 M	134 M	15 M	0.9 M
required storage	228 MB	537 MB	49 MB	6MB

To learn our binary descriptors, we replace the last softmax layer of the SqueezeNet with a fully-connected layer. On the other hand, we add a new fully-connected layer after the average polling layer in VGG16-Avg. Both fully-connected layers have 512 neurons, and their parameters are initialized with random Gaussian. Then, we enforce the proposed criteria on these neurons to learn binary descriptors, and optimized the network parameters though back-propagation. Table 12 shows the fine-grain recognition rates using different networks. Overall, DeepBit achieves better performance than SIFT in most cases when using different networks. This indicates it is feasible to realize DeepBit with the relatively cheaper networks, and maintains comparable performance. Note that DeepBit + SqueezeNet produces comparable accuracy to SIFT, but worse than that of DeepBit + AlexNet. Since the compression process usually requires strongly supervision to maintain accuracy, the results suggest our approach, which belongs to unsupervised learning, may slightly drop the performance during deep network compression. However, the recognition rate of our binary descriptor is still comparable to some floating-valued descriptors such as SIFT-Boundary. Moreover, our study shows that DeepBit + VGG16-Avg not only reduces the model size, but achieves more favorably accuracy against DeepBit + the others. The main idea of VGG16-Avg is reducing the model parameters by down-sampling the feature maps, and further decreasing the number of parameters in the following fully-connected layers. Since the average polling layer does not require parameter learning, it is easier for our unsupervised approach to learn binary descriptors comparing to other compression approaches such as SqueezeNet. Our study shows that the proposed approach can be realized on a small or light-weighted deep network, and enable efficient visual recognition.

4.8 Applications to Panoramic Image Stitching

Panoramic image stitching consists of the keypoint detection, the descriptor matching and the homography estima-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2833865, IEEE Transactions on Pattern Analysis and Machine Intelligence

TABLE 12
Recognition accuracy (mean%) of our method with different deep neural networks on Flower-17 and Flower-102 datasets. We also report the recognition accuracy of SIFT for reference.

Descriptors	Flower-17	Flower-102
SIFT-Boundary [52]	59.4	32.0
SIFT-Internal [52]	70.6	55.1
DeepBit, 512 bits, SqueezeNet	76.8	33.7
DeepBit, 512 bits, AlexNet	89.2	61.9
DeepBit, 512 bits, VGG16	87.5	58.6
DeepBit, 512 bits, VGG16-Avg	**90.7**	**62.8**

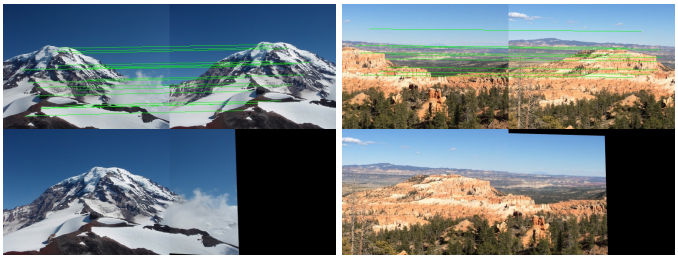


Fig. 12. Panoramic image stitching results using our binary descriptors. The top row shows the matched keypoints, and the bottom row shows the stitching results. The results show that our approach can produce stitched images in good quality.

tion for image wrapping. Particularly, the invariant descriptors are highly demanding for a successful stitching. Hence, we also study the effectiveness of our binary descriptors for this application. Following [4], [11], we perform image stitching with four main steps. First, we employ FAST keypoint detector to find the keypoints in the images. Second, we extract binary descriptors using our model presented in Sec.4.5 and perform brute-force matching to measure the correspondence between two images. After that, RANSAC algorithm is applied to estimate the homography matrix based on the matched binary descriptors. Finally, we stitch the images using the wrapping transformation with the estimated homography matrix. Fig. 12 shows that our method can be applied to panoramic image stitching and produce stitched results in good quality. We see that the results may have negligible illumination changes between two images since the photo were taken in different time. The results further verify the robustness of our binary descriptors. This is because our method is general for learning binary descriptors for various applications.

4.9 Computational Cost

Encoding time: We randomly select $1,000$ test images from CIFAR-10 dataset, and compute the average computational time. We report the encoding time of our approach on CPU and GPU, and the feature extraction time for both conventional binary descriptors and hashing methods. The experiments are carried out on a machine with an Intel Xeon 3.70 GHz CPU of 8 cores, and an NVIDIA Titan-X with CUDA-7.0. Fig. 13 shows the encoding time (in milliseconds) of different methods. Overall, considering only the encoding time, the conventional methods (such as ITQ) are much faster than deep learning approaches. However, when taking feature extraction time into consideration (such as SIFT+ITQ), the computational time of the conventional

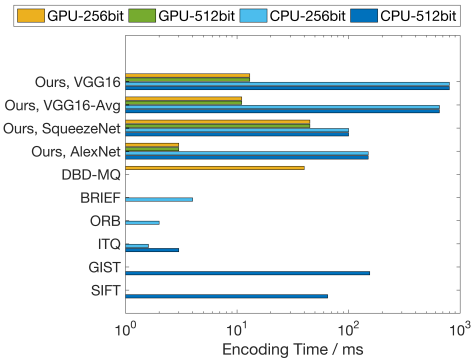


Fig. 13. Computational cost for encoding an input image on CIFAR-10.

methods is comparable or slower than some of our models. In addition, as discussed in [41], conventional methods may require additional processes (such as BOW or multiple features) to reach comparable retrieval performance. On one hand, since the major computational time of our method lies in the forward-pass of the network layers, "Ours, AlexNet" (7 layers) is faster than the very deep "Ours, VGG16" network (16 layers). In the near future, our method could be faster when employing the compressed deep networks. However, this is outside the major scope of this paper. On the other hand, it is worth noting that the computational time of our model is independent of the code lengths. When using the same network backbone, the encoding time of our 256-bit and 512-bit models remain the same. This is an advantage comparing to conventional methods such as ITQ.

Time complexity for image retrieval: The pipeline of image retrieval can be divided into (1) offline extracting binary codes of the images, and (2) online querying using the binary codes. Given the database consisting of n images, the offline process computes a single forward-pass for each image in the database, which takes time $O(n)$. Given a novel query, we firstly extract the binary codes of the query, and then perform linear search that computes the hamming distance between the query and each item in the database resulting in the time complexity $O(n)$. Though the complexity is linear to the number of images in the database, computing the hamming distance via *XOR* operation is very fast. It is worth noting that we primarily focus on learning discriminative binary codes for improving the retrieval performance, and the time complexity could be further reduced to sub-linear by using advanced implementations such as multiple hash tables or exploring the Hamming ball volume around the query [71], [72].

5 Conclusions

In this paper, we have presented an unsupervised deep learning framework to learn compact binary descriptors. We employ three criteria to learn the binary codes and estimate the parameters of the deep neural network to obtain binary descriptors. Our approach does not require label annotations provided by the dataset, and is more practical to real-world applications compared to supervised binary descriptors. Experiments on several public benchmark databases demonstrate that our method achieves better performance

than the state-of-the-art feature descriptors in most cases. How to simultaneously learn binary descriptor and its optimal code size seems to be an interesting future work.

ACKNOWLEDGEMENT

We thank the anonymous reviewers for their insightful comments. We thank NVIDIA for the donation of the GPU used for this research. This work was supported in part by the Ministry of Science and Technology of Taiwan under Contract MOST 105-2218-E-001-006. Jiwen Lu and Jie Zhou was supported by the National Key Research and Development Program of China under Grant 2016YFB1001001, the National Natural Science Foundation of China under Grants 61672306, 61572271, 61527808, 61373074 and 61373090, the National 1000 Young Talents Plan Program, the National Basic Research Program of China under Grant 2014CB349304, the Ministry of Education of China under Grant 20120002110033, and the Tsinghua University Initiative Scientific Research Program.

REFERENCES

[1] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE TPAMI*, vol. 27, no. 10, pp. 1615–1630, 2005.
[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, no. 2, pp. 91–110, 2004.
[3] K. Grauman and T. Darrell, "The pyramid match kernel: Discriminative classification with sets of image features," in *Proc. ICCV*, 2005.
[4] M. Brown and D. G. Lowe, "Automatic panoramic image stitching using invariant features," *IJCV*, vol. 74, no. 1, pp. 59–73, 2007.
[5] K. Mikolajczyk and C. Schmid, "Indexing based on scale invariant interest points," in *Proc. ICCV*, 2001.
[6] A. W. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," *IEEE TPAMI*, vol. 22, no. 12, pp. 1349–1380, 2000.
[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012.
[8] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *Proc. CVPRW*, 2014.
[9] J. Wan, D. Wang, S. C. H. Hoi, P. Wu, J. Zhu, Y. Zhang, and J. Li, "Deep learning for content-based image retrieval: A comprehensive study," in *Proc. ACM MM*, 2014.
[10] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *Proc. ECCV*, 2010.
[11] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: an efficient alternative to sift or surf," in *Proc. ICCV*, 2011.
[12] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *Proc. ICCV*, 2011.
[13] A. Alahi, R. Ortiz, and P. Vandergheynst, "Freak: Fast retina keypoint," in *Proc. CVPR*, 2012.
[14] T. Trzcinski, M. Christoudias, and V. Lepetit, "Learning image descriptors with boosting," *IEEE TPAMI*, vol. 37, no. 3, pp. 597–610, 2015.
[15] B. Fan, Q. Kong, T. Trzcinski, Z. Wang, C. Pan, and P. Fua, "Receptive fields selection for binary feature description," *IEEE TIP*, vol. 23, no. 6, pp. 2583–2595, 2014.
[16] S. Zhang, Q. Tian, Q. Huang, W. Gao, and Y. Rui, "Usb: ultrashort binary descriptor for fast visual matching and retrieval," *IEEE TIP*, vol. 23, no. 8, pp. 3671–3683, 2014.
[17] L. Zheng, S. Wang, and Q. Tian, "Coupled binary embedding for large-scale image retrieval," *IEEE TIP*, vol. 23, no. 8, pp. 3368–3380, 2014.
[18] K. Lin, J. Lu, C.-S. Chen, and J. Zhou, "Learning compact binary descriptors with unsupervised deep neural networks," in *Proc. CVPR*, 2016, pp. 1183–1192.
[19] T. Trzcinski and V. Lepetit, "Efficient discriminative projections for compact binary descriptors," in *Proc. ECCV*, 2012.
[20] T. Trzcinski, M. Christoudias, P. Fua, and V. Lepetit, "Boosting binary keypoint descriptors," in *Proc. CVPR*, 2013.
[21] C. Strecha, A. M. Bronstein, M. M. Bronstein, and P. Fua, "Lda-hash: Improved matching with smaller descriptors," *IEEE TPAMI*, vol. 34, no. 1, pp. 66–78, 2012.
[22] V. Balntas, L. Tang, and K. Mikolajczyk, "Bold-binary online learned descriptor for efficient image matching," in *Proc. CVPR*, 2015.
[23] X. Xu, F. Shen, Y. Yang, H. T. Shen, and X. Li, "Learning discriminative binary codes for large-scale cross-modal retrieval," *IEEE TIP*, pp. 2494–2507, 2017.
[24] X. Yang and K.-T. Cheng, "Local difference binary for ultrafast and distinctive feature description," *IEEE TPAMI*, vol. 36, no. 1, pp. 188–194, 2014.
[25] F. Shen, C. Shen, W. Liu, and H. Tao Shen, "Supervised discrete hashing," in *Proc. CVPR*, 2015.
[26] B. Ozdemir, M. Najibi, and L. S. Davis, "Supervised incremental hashing," *arXiv preprint arXiv:1604.07342*, 2016.
[27] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," in *Proc. FOCS*, 2006.
[28] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," *IEEE TPAMI*, vol. 35, no. 12, pp. 2916–2929, 2013.
[29] R. Salakhutdinov and G. E. Hinton, "Semantic hashing," *IJAR*, vol. 50, no. 7, pp. 969–978, 2009.
[30] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Proc. NIPS*, 2008.
[31] S. Zagoruyko and N. Komodakis, "Learning to compare image patches via convolutional neural networks," in *Proc. CVPR*, 2015, pp. 4353–4361.
[32] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg, "Matchnet: Unifying feature and metric learning for patch-based matching," in *Proc. CVPR*, 2015, pp. 3279–3286.
[33] J. Zbontar and Y. LeCun, "Stereo matching by training a convolutional neural network to compare image patches," *JMLR*, vol. 17, no. 1-32, p. 2, 2016.
[34] B. Kumar, G. Carneiro, I. Reid *et al.*, "Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions," in *Proc. CVPR*, 2016, pp. 5385–5394.
[35] A. Dosovitskiy, P. Fischer, J. T. Springenberg, M. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with exemplar convolutional neural networks," *IEEE TPAMI*, vol. 38, no. 9, pp. 1734–1747, 2016.
[36] M. Paulin, M. Douze, Z. Harchaoui, J. Mairal, F. Perronin, and C. Schmid, "Local convolutional features with unsupervised training for image retrieval," in *Proc. ICCV*, 2015.
[37] V. Balntas, E. Johns, L. Tang, and K. Mikolajczyk, "Pn-net: conjoined triple deep network for learning local image descriptors," *arXiv preprint arXiv:1601.05030*, 2016.
[38] J. Wang, T. Zhang, N. Sebe, H. T. Shen *et al.*, "A survey on learning to hash," *IEEE TPAMI*, 2017.
[39] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan, "Supervised hashing for image retreieval via image representation learning," in *Proc. AAAI*, 2014.
[40] H. Lai, Y. Pan, Y. Liu, and S. Yan, "Simultaneous feature learning and hash coding with deep neural networks," in *Proc. CVPR*, 2015.
[41] H. Liu, R. Wang, S. Shan, and X. Chen, "Deep supervised hashing for fast image retrieval," in *Proc. CVPR*, 2016.
[42] Z. Zhang, Y. Chen, and V. Saligrama, "Efficient training of very deep neural networks for supervised hashing," in *Proc. CVPR*, 2016.
[43] T.-T. Do, A.-D. Doan, and N.-M. Cheung, "Learning to hash with binary deep neural network," in *Proc. ECCV*, 2016.
[44] C. Huang, C. Change Loy, and X. Tang, "Unsupervised learning of discriminative attributes and visual representations," in *Proc. CVPR*, 2016.
[45] V. E. Liong, J. Lu, G. Wang, P. Moulin, and J. Zhou, "Deep hashing for compact binary codes learning," in *Proc. CVPR*, 2015.
[46] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. ACM MM*, 2014.
[47] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. ICLR*, 2015.
[48] M. Brown, G. Hua, and S. Winder, "Discriminative learning of local image descriptors," *IEEE TPAMI*, vol. 33, no. 1, pp. 43–57, 2011.

[49] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009.
[50] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *IJCV*, vol. 115, no. 3, pp. 211–252, 2015.
[51] M.-E. Nilsback and A. Zisserman, "A visual vocabulary for flower classification," in *Proc. CVPR*, 2006.
[52] ——, "Automated flower classification over a large number of classes," in *Proc. ICVGIP*, 2008.
[53] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *Proc. CVPR*, 2007.
[54] ——, "Lost in quantization: Improving particular object retrieval in large scale image databases," in *Proc. CVPR*, 2008.
[55] H. Jegou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," *Proc. ECCV*, 2008.
[56] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Proc. CVPR*, 2006.
[57] Y. Duan, J. Lu, Z. Wang, J. Feng, and J. Zhou, "Learning deep binary descriptor with multi-quantization," in *Proc. CVPR*, 2017.
[58] G. Shakhnarovich, "Learning task-specific similarity," Ph.D. dissertation, Massachusetts Institute of Technology, 2005.
[59] B. F. Y. Tian, "L2-net: Deep learning of discriminative patch descriptor in euclidean space," in *Proc. CVPR*, 2017.
[60] J. Wang, S. Kumar, and S. Chang, "Semi-supervised hashing for scalable image retrieval," in *Proc. CVPR*, 2010.
[61] T.-H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, and Y. Ma, "Pcanet: A simple deep learning baseline for image classification?" *arXiv preprint arXiv:1404.3606*, 2014.
[62] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, "Neural codes for image retrieval," in *Proc. ECCV*, 2014, pp. 584–599.
[63] Y. Gong and S. Lazebnik, "Iterative quantization: A procrustean approach to learning binary codes," in *Proc. CVPR*, 2011.
[64] J. Yue-Hei Ng, F. Yang, and L. S. Davis, "Exploiting local features from deep networks for image retrieval," in *Proc. CVPRW*, 2015.
[65] H. Jegou, F. Perronnin, M. Douze, J. Sánchez, P. Perez, and C. Schmid, "Aggregating local image descriptors into compact codes," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1704–1716, 2012.
[66] K. Reddy Mopuri and R. Venkatesh Babu, "Object level deep feature pooling for compact image representation," in *Proc. CVPRW*, 2015.
[67] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. CVPR*, 2005.
[68] B. Fernando, E. Fromont, and T. Tuytelaars, "Mining mid-level features for image classification," *IJCV*, vol. 108, no. 3, pp. 186–203, 2014.
[69] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.
[70] H.-F. Yang, K. Lin, and C.-S. Chen, "Supervised learning of semantics-preserving hash via deep convolutional neural networks," *IEEE TPAMI*, 2017.
[71] K. Grauman and R. Fergus, "Learning binary hash codes for large-scale image search," in *Machine learning for computer vision*. Springer, 2013, pp. 49–87.
[72] J. Wang, W. Liu, S. Kumar, and S.-F. Chang, "Learning to hash for indexing big dataa survey," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 34–57, 2016.

Kevin Lin received the M.S. degree from the Graduate Institute of Networking and Multimedia, National Taiwan University, in 2014. He is currently pursuing the Ph.D. degree with Dept. Electrical Engineering, University of Washington, Seattle, WA, USA. His research interests include computer vision, machine learning, and natural language processing.

Jiwen Lu (S'10-M'11-SM'15) received the B.Eng. degree in mechanical engineering and the M.Eng. degree in electrical engineering from the Xi'an University of Technology, Xi'an, China, and the Ph.D. degree in electrical engineering from the Nanyang Technological University, Singapore, in 2003, 2006, and 2012, respectively. He is currently an Associate Professor with the Department of Automation, Tsinghua University, Beijing, China. His research interests include computer vision, pattern recognition, and machine learning. He is an Associate Editor of *Pattern Recognition*, *Pattern Recognition Letters*, and *Journal of Visual Communication and Image Representation*.

Chu-Song Chen is a Research Fellow with the Institute of Information Science, and the Research Center for IT Innovation, Academia Sinica, Taiwan. He is an Adjunct Professor with the Graduate Institute of Networking and Multimedia, National Taiwan University. His research interests include computer vision, image processing, pattern recognition, and multimedia. He served as an Area Chair of ACCV '09 and ACCV'10, the Program Chair of IMV'12 and IMV'13, the Tutorial Chair of ACCV'14, the General Chair of IMEV'14, and the Workshop Chair of ACCV'16. He is on the Editorial Board of the *Machine Vision and Applications* journal.

Jie Zhou (M'01-SM'04) received the B.S. and M.S. degrees from the Department of Mathematics, Nankai University, Tianjin, China, in 1990 and 1992, respectively, and the Ph.D. degree from the Institute of Pattern Recognition and Artificial Intelligence, Huazhong University of Science and Technology, Wuhan, China, in 1995. He is a Full Professor with the Department of Automation, Tsinghua University. His current research interests include computer vision, pattern recognition, and image processing. He is an Associate Editor of the *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Ming-Ting Sun (S'79-M'81-SM'89-F'96) received the B.S. degree in electrical engineering from National Taiwan University and the Ph.D. degree in electrical engineering from the University of California at Los Angeles, Los Angeles, CA, USA. He was the Director of Video Signal Processing Research, Bellcore. He has been a Chaired/Visiting Professor at several universities. He joined the University of Washington in 1996, where he is currently a Professor. His main research interest is video and multimedia signal processing. He served as a General Co-Chair of the ICME 2016, an Honorary Chair of the VCIP 2015, a General Co-Chair of Visual Communications and Image Processing 2000. He is currently an Editor-in-Chief of the *Journal of Visual Communication and Image Representation*.