On Aggregation of Unsupervised Deep Binary Descriptor With Weak Bits

Gengshen Wu, Zijia Lin, Guiguang Ding, *Member, IEEE*,
Qiang Ni, *Senior Member, IEEE*, and Jungong Han

***Abstract*— Despite the thrilling success achieved by existing binary descriptors, most of them are still in the mire of three limitations: 1) vulnerable to the geometric transformations; 2) incapable of preserving the manifold structure when learning binary codes; 3) NO guarantee to find the true match if multiple candidates happen to have the same Hamming distance to a given query. All these together make the binary descriptor less effective, given large-scale visual recognition tasks. In this paper, we propose a novel learning-based feature descriptor, namely Unsupervised Deep Binary Descriptor (UDBD), which learns transformation invariant binary descriptors via projecting the original data and their transformed sets into a joint binary space. Moreover, we involve a $\ell_{2,1}$-norm loss term in the binary embedding process to gain simultaneously the robustness against data noises and less probability of mistakenly flipping bits of the binary descriptor, on top of it, a graph constraint is used to preserve the original manifold structure in the binary space. Furthermore, a weak bit mechanism is adopted to find the real match from candidates sharing the same minimum Hamming distance, thus enhancing matching performance. Extensive experimental results on public datasets show the superiority of UDBD in terms of matching and retrieval accuracy over state-of-the-arts.**

***Index Terms*— Image hashing, feature matching, local binary descriptor, similarity retrieval, deep learning.**

I. Introduction

RECENTLY, the local binary descriptor has attracted wide attention in various visual applications, such as patch matching, object recognition, image retrieval and 3D reconstruction [1]–[6]. Benefiting from the characteristics of high compactness and efficient bitwise calculation, binary descriptor is a more favorable option in conducting matching and retrieval in *large-scale* database over the traditional floating-point descriptors (e.g., SIFT [1], FAST [7] and SURF [8]) [9], [10]. This paper focuses on applying binary descriptors in both patch matching and image retrieval, where patches can be obtained from the full image via keypoint detection technology in the former applications [11].

Similar to traditional feature descriptors, binary descriptor is supposed to represent data (image/patch) accurately in despite of geometric transformations (e.g., rotation, translation and scaling) [1], [12]. Earlier binary descriptors (e.g., BRIEF [13], BRISK [12], ORB [14] and FREAK [15]) are generally data-independent, which adopt various hand-crafted sampling patterns and perform a series of pairwised intensity comparisons afterwards [16]. However, such predefined sampling modes and intensity comparisons are extremely vulnerable to the distortions/transformations, thus yielding unstable performance when tackling large-scale visual recognition tasks [10], [16], [17]. Consequently, many efforts have been devoted to developing learning-based binary descriptors. Existing methods draw on the soul idea from hashing techniques (e.g., LSH [9], ITQ [18], CMFH [19]), where the data points are projected from their original feature space into the compact binary space and the similar points could be represented by the similar binary descriptors (low Hamming distance) [16], [20], [21]. Although the learning-based binary descriptors obtain great performance gains over the handcrafted ones, some drawbacks become bottlenecks that impede their further development in large-scale application scenarios.

Firstly, they pay intensive attention to novel discrete optimization strategies, while the nature of local feature descriptor, anti-geometric transformation, cannot be fully guaranteed [10], [12]. That is crucial to the success of binary descriptors in large-scale visual recognition tasks. More worriedly, most paradigms of learning binary codes for patches, especially unsupervised ones, fail to preserve the manifold structure during the discrete optimization, which makes the binary descriptor less effective in large-scale neighbor search tasks [2], [22]. Many supervised methods relieve this issue by incorporating the supervision information (e.g., labels or affinity matrix) into their learning objectives [23]. However, they are not preferred in real-world applications because of intensive labeling work.

Furthermore, traditional binary descriptors measure the similarity between database and query via exhaustive Hamming distance calculations in the testing phase. In practice, however, it is more likely to return many candidates with equal Hamming distances to one specific query [24]. To clarify the problem, we plot the Hamming distance distribution of a query to the database (with randomly-selected $1,000$ candidates)

Manuscript received March 18, 2020; revised July 23, 2020; accepted September 4, 2020. Date of publication September 25, 2020; date of current version October 1, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant U1936202, Grant 61925107, and Grant 61971004. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Lucio Marcenaro. *(Corresponding author: Jungong Han.)*

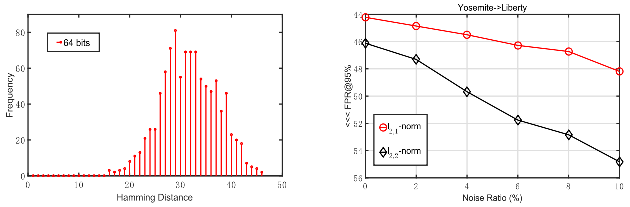
Gengshen Wu and Qiang Ni are with the School of Computing and Communication, Lancaster University, Lancaster LA1 4YW, U.K. (e-mail: gengshen.wu@lancaster.ac.uk; q.ni@lancaster.ac.uk).

Zijia Lin is with Alibaba Group, Beijing 100080, China (e-mail: jiulin.lzj@alibaba-inc.com).

Guiguang Ding is with the School of Software, Tsinghua University, Beijing 100084, China (e-mail: dinggg@tsinghua.edu.cn).

Jungong Han is with the Department of Computer Science, Aberystwyth University, Aberystwyth SY23 3DB, U.K. (e-mail: jungonghan77@gmail.com).

Digital Object Identifier 10.1109/TIP.2020.3025437



(a) Noise effects on matching (b) Distance distribution

Fig. 1. (a) An example of the Hamming distance distribution on Cifar-10 dataset at 64 bits, where 3 candidates are returned from the database with the same minimum Hamming distance of 16 to the query; (b) Noise effects on Brown dataset (train: *Yosemite* and test: *Liberty*) at 256 bits under $\ell_{2,1}$-norm and $\ell_{2,2}$-norm losses, where a sharper performance decline from $\ell_{2,2}$-norm against $\ell_{2,1}$-norm loss is observed at certain noise level.

on Cifar-10 dataset in Fig. 1(a). For instance, 3 candidates are returned from the database with the same minimum Hamming distance of 16 to the query at the code length of 64. That reduces the discriminative power of the binary descriptor dramatically. It is especially harmful to the matching performance, where *each query is expected to be matched with one exact candidate* (with the lowest Hamming distance) rather than a bunch of ambiguous options (with equal minimum Hamming distance).

In this paper, we propose a novel learning-based framework, termed **U**nsupervised **D**eep **B**inary **D**escriptor (**UDBD**), to overcome the above limitations in compact binary descriptor learning. Fig. 2 shows the flowchart of UDBD. Particularly, the original visual data, as well as their transformed counterparts, are projected into *common* Hamming subspace directly during the binary code learning. By doing so, transformation invariance could be conserved along with the binary embedding process, which is theoretically more advanced than the primitive approach [25] that simply minimizes the differences between the binary codes of original data and those transformed ones.

In the meantime, $\ell_{2,1}$-norm loss is employed together with the proposed binary embedding to improve the robustness of our binary descriptor against data noises/outliers for the patch-level recognition tasks [26], [27]. To make it clear, we plot the matching performance variations with increasing noise ratios using ITQ+ [26] on Brown dataset [11]: train: *Yosemite* and test: *Liberty*, in Fig. 1(b). As can be seen, there is a sharper performance decline from $\ell_{2,2}$-norm against $\ell_{2,1}$-norm loss function at certain noise levels. The main cause is that patches mainly contain micro-texture information, which are more prone to the noises/outliers, compared to natural images [6]. Without noticing it, previous methods directly adopt the squared ℓ_p-norm regularization to build their loss functions [26], which may exaggerate the adverse effects caused by severe noises/distortions, thus leading to worse results [26], [28]. That implies $\ell_{2,1}$-norm loss is more suitable for patch-level recognition.

Then an unsupervised graph constraint is formed and added into the loss function so as to preserve the original manifold structure of training data in the Hamming space [29], [30]. With an alternating optimization scheme, the binary code can be solved directly without relaxation, which avoids accumulating quantization errors that occurs in the two-step learning strategy [18], [31], [32]. By training a unified deep network with the guidance of the learned binary descriptors, the deep embedding function is able to generate robust binary codes for various visual tasks.

During the feature matching procedure, a weak bit scheme, where the Hamming distance is recalculated based on the reliability of each bit, is further applied to find the best match among the returned candidates with the same initial Hamming distance [33]. In summary, our work differs from the previous algorithms in the following three aspects:

- To the best of our knowledge, this is the first work that learns the transformation invariant binary descriptor via *embedding the original visual data and their transformed sets into a common Hamming space in an unsupervised manner*. Moreover, a graph constraint that preserves the manifold structure from the original feature space is employed in the unified binary representation learning, thus improving the code quality.
- Since patches mainly contain noise-sensitive local features, a $\ell_{2,1}$-norm loss is proposed to regularize the binary embedding. On one hand, ℓ_1-norm distance at the patch level provides the robustness against outlier samples. On the other hand, ℓ_2-norm measures the distance along space dimension, which spreads out the errors over each bit uniformly to lower the possibility that certain bits are mistakenly flipped after getting large errors. To this end, an alternating discrete optimization strategy is proposed to optimize the $\ell_{2,1}$-norm constrained objective function, where the binary code can be solved directly with no need for relaxation.
- As a means of distance re-measure, a weak bit scheme, which considers the reliability of each bit in a descriptor, is applied along with the proposed binary descriptor. It helps to find the best match if there are multiple candidates with the same distance to the query when comparing the Hamming distance of descriptors.

The rest of this paper is organized as follows. We discuss some related works in Section II. In Section III, the proposed method is elaborated along with the comprehensive analysis. Extensive experimental results are provided and analyzed in Section IV. Finally, the conclusion is given in Section V.

II. Related Works

In this section, we overview the related works from two aspects: handcrafted and learning-based feature descriptors.

A. Handcrafted Feature Descriptors

Most handcrafted local descriptors are real-valued in the early research stage. Two classical feature descriptors: SIFT [1] and SURF [8] are widely used in vision recognition tasks like image retrieval and feature matching. Particularly, the local gradient histograms are applied in SIFT to generate the scale-invariant descriptors. While the computation process of SIFT is accelerated dramatically by SURF, which takes advantage of the integral images in the calculations. However, the performance of both real-valued feature descriptors heavily relies on the high dimensionality (i.e., long descriptor length), which means the high storage requirement and computational complexities for feature matching by using those descriptors [13], [15].

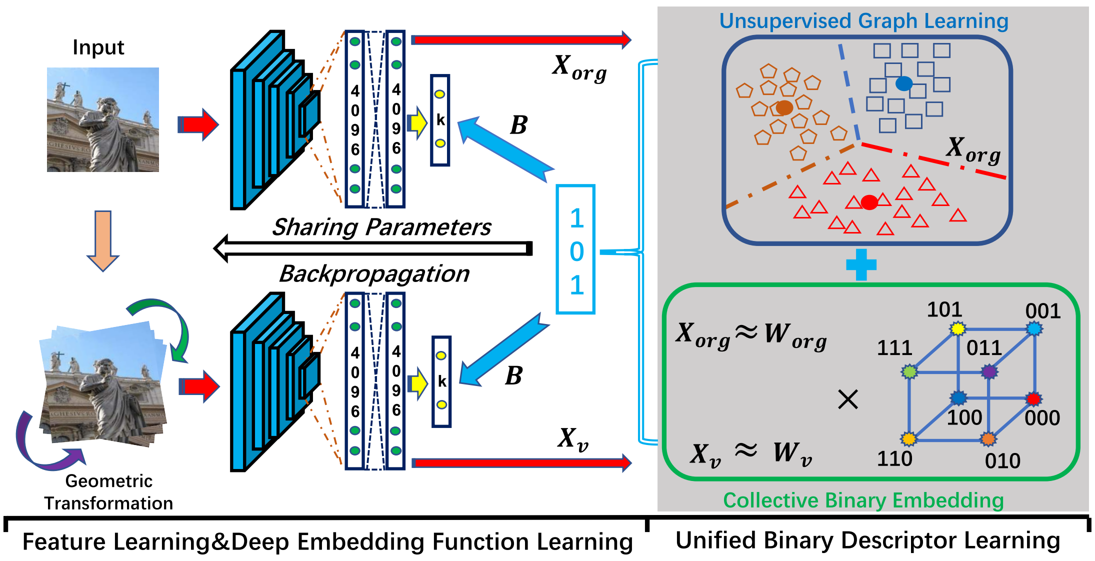

Input
Geometric Transformation
4096
4096
k
X_{org}
B
Sharing Parameters
Backpropagation
B
X_v
1
0
1
Unsupervised Graph Learning
X_{org}
$X_{org} \approx W_{org}$
×
$X_v \approx W_v$
101
001
111
011
100
000
110
010
Collective Binary Embedding
Feature Learning&Deep Embedding Function Learning
Unified Binary Descriptor Learning

Fig. 2. The proposed binary descriptor learning framework is made up of deep feature extraction, unified binary code learning and deep embedding function learning. The descriptor size is set to 3 as an example.

Consequently, many efforts have been devoted to developing *binary* local descriptors, such as BRIEF [13], ORB [14], BRISK [12], and FREAK [15]. These descriptors perform a set of pairwise intensity comparisons to generate compact binary codes. While the efficiency of these binary descriptors for the similarity search tasks has been improved significantly due to the XOR operations in Hamming space, their robustness is relatively worse than that of the real-valued local descriptors. The reason is that these binary descriptors are mainly built upon manually predefined sampling modes and intensity comparisons, which are sensitive to the geometric transformations and distortions on the original images. Therefore, they may not perform well, given complex visual tasks [10], [16], [17].

Recently, a novel binary RGB-D descriptor termed GEOBIT is presented in [34] for the textured depth map tracking, which is claimed to be invariant to the non-rigid transformation by integrating the appearance and the geometric information from RGB-D images into the code learning. Similarly, SRBD [35] proposes a new kernel-distance-based clustering method to select the stable superpixels from the templates and encodes the dominant gradient orientation of each superpixel as its rotation-invariant binary descriptor. Though promising, they still adopt handcrafted patterns like BRIEF [13] and ORB [14], which indicates their weak generalization ability.

B. Learning-Based Feature Descriptors

More recently, the learning-based feature descriptors, which involve a dedicated training process of encoding function on massive training data, are widely developed to boost the descriptor performance and gain better robustness. Without loss of generality, these learning-based feature descriptors can further be categorized into *supervised* and *unsupervised* approaches, which are differentiated based on whether the supervision information (e.g., labels, similarity matrix) is utilized during the training process.

1) Supervised Feature Descriptors: Earlier learning-based works learn the shallow projections to generate the local descriptors. For example, LDAHash [20] is proposed that uses linear projections combining linear discriminant analysis to generate binary descriptors. D-BRIEF [36] generates the descriptors by projecting the training data into a latent subspace. To deal with the nonlinear data structure, BinBoost [10] learns a set of nonlinear classifiers in encoding the data, which makes the learned binary codes more discriminative. Online learning is adopted in BOLD [37], which aims at selecting binary intensity tests to produce low intra-class and high inter-class distances in the code learning. However, these methods generally adopt simple binary intensity tests and some important cues of a patch cannot be captured in the to-be-learned descriptor.

With the development of deep learning techniques, more recent works apply CNN network and deep features in learning the compact feature descriptor. For example, in [38], a supervised hashing framework is proposed to generate bit-scalable hashing codes directly by training the network with triplet samples and an additional regularization term. A novel DHN architecture is designed in [39] to jointly learn good image representation along with compact hash code, where the quantization error is claimed to be well controlled. In [40], a deep supervised discrete hashing algorithm is proposed to learn the hash code within one stream framework, where the pairwise label and classification information are considered in the loss function. Dosovitskiy *et al.* [41] train a CNN network by optimizing the classification loss, where the output vectors before the classification layer are used as the patch descriptors. Instead of simply optimizing the classification loss, Siamese loss is introduced in the network training of DeepDesc [42], where the patch pairs as the network inputs are selected by applying an aggressive searching strategy. Subsequently, L2-NET [17] trains a Siamese network for pairwised patches and produces binary codes by directly quantizing the real-valued outputs, where different regularization terms are applied on the intermediate layer outputs to improve the code quality. In [43], Second

Order Similarity Regularization (SOSR) is incorporated into the proposed SOSNet as a regularization term to boost the matching performance. In [44], they train a deep network termed DN4, where a local descriptor is learned based on image-to-class measure. In [45], the context awareness is introduced to augment local feature descriptors by aggregating the cross-modality contextual information like visual context from high-level image representation and geometric context from 2D keypoint distribution. HardNet [46] proposes a triplet loss function that explores the hard examples by an effective mining strategy to mimic the matching procedure in a batch fashion, where at least one positive pair is guaranteed in building the triplet input. DOAP [47] is proposed to train the deep network via optimizing a new loss function termed Average Precision (AP) directly, which improves the ranking-based retrieval performance. On top of that, CDbin [16] is proposed to generate the binary descriptors via jointly optimizing four complementary loss functions in an end-to-end manner. In such cases, prior knowledge (e.g., labels) is required, which is usually impractical in real application scenarios.

2) Unsupervised Feature Descriptors: More works have been done recently to learn the binary descriptor in unsupervised manner. For instance, DH [48] optimizes the binary descriptor with independence and even distribution. In [49], the proposed unsupervised hashing framework unifies the quantization error minimization, likelihood and mutual information maximization to preserve the feature distribution for better code quality. In UDHPA [50], they produce compact binary codes by using pseudo-label as self-supervised information in the network training. In DistillHash [51], Bayesian learning framework is integrated into the hash code learning, where a distilled data set is investigated automatically and further utilized to learn the compact binary code. In [52], the proposed DVB adopts a conditional auto-encoding variational Bayesian networks to estimate the training data structure under the probabilistic inference process with hashing objectives, thus improving the code quality. More than just pairwised inputs in [17], [42], the triplet loss is incorporated in the objective function of [53] to further guarantee the code discriminativeness. In [54], they propose a binary mean shift (bMS) to find frequent and informative image patterns directly in binary space such that the computation and memory costs can be reduced dramatically. In [55], C-CBFD is proposed to generate binary codes under three complementary learning objectives: high variance for information preservation, low quantization errors and even-distribution at each bit. To overcome the limitations of two-step optimization, DBD-MQ [21] adopts a multi-quantization strategy that reduces the quantization errors within the K-AutoEncoders (KAEs) networks. GraphBit [2] integrates the reinforcement learning with binary code learning, where the uncertainty of binary codes is minimized by maximizing the mutual information between the real-valued inputs and the corresponding bits. Despite the great success achieved by those descriptors, the transformation-invariant nature of the local descriptor is not considered in the training process. Consequently, DeepBit [25] is proposed to learn compact binary descriptors via optimizing several loss functions in network training, one of which minimizes the Hamming distances of the binary codes from the original patch and their transformed versions in pairwised manner. Although it takes the transformation invariance into consideration to some extent, the objective to minimize the Euclidean distances of original data and their transformed sets in the binary space potentially deems they are not *identical*. With the powerful GAN [56], BinGAN [57] learns the compact binary descriptors from patches via optimizing two additional losses from distance matching and entropy regularizers. GAN has also been employed in [58] to facilitate image retrieval and compression.

Moreover, such learning-based binary descriptors have been widely developed in many other applications like palmprint and face recognition [59], [60]. For example, DDBC [59] learns a simple mapping function to project the convolution difference vectors to the neighboring directions of the templates. Subsequently, the one-stage learning strategy is utilized in SLBFLE [61], where the binary codes and the encoding codebook are jointly optimized for local face patches. Later on they extend these works as CA-LBFL [60] and RI-LBD [62], which learns the robust local binary descriptor to further improve the efficiency and accuracy in face recognition. In these applications, the learning-based binary descriptors act as the main contributing roles in improving the performance of the specific tasks.

III. Methodology

A. Framework Overview

Some mathematical symbols are defined to ease the following explanations on the framework. Assuming that the training set consists of n data samples (images/patches) and each one has m different transformation sets, where the transformed versions of each sample could be obtained by rotation, scaling and translation [25]. We denote the training set as $\mathcal{O} = \{o_i\}_{i=1}^{n}$, $o_i = \{x_{\upsilon}^{i}\}_{\upsilon=1}^{m}$, where υ denotes the index of the transformation set, $x_{\upsilon}^{i} \in \mathbb{R}^{p_{\upsilon}}$ is a feature vector and p_{υ} represents the dimensionality of x_{υ}^{i} in the set. For each transformation set, we denote the feature matrix as $\mathbf{X}_{\upsilon} = [x_{\upsilon}^{1}, x_{\upsilon}^{2}, \ldots, x_{\upsilon}^{n}] \in \mathbb{R}^{p_{\upsilon} \times n}$. Given the code length k, the goal of the proposed method is to learn the *unified* binary descriptor $\mathbf{B} \in \{-1, +1\}^{k \times n}$ for the training samples within all transformation sets. Particularly, each sample and its transformed versions should be encoded as the same binary code because the semantics of those samples keeps unchanged even after certain transformations.

To achieve this learning goal, the deep features of different input sources are first extracted from the fully-connected (*fc*) layers of a pre-trained VGG-16 network [63]. Then the features are fed into binary code learning that generates the uniformed binary descriptor for various transformation sets via exploring their common binary space. With shared network parameters Θ, a unified deep embedding function $\mathcal{H}$ is built through projecting all transformation sets into the learned binary code, which helps generate descriptors for a query. In the online stage, a weak bit scheme that excludes the contributions of unreliable bits in distance calculation is adopted to further improve the matching performance. The major mathematical symbols used in this chapter are summarized in Table I for the ease of explanation. Other symbols like $\mathbf{G}_{\upsilon}$

TABLE I

Mathematical Symbols and Descriptions

Symbol	Description	Symbol	Description
$\mathcal{O}$	training set	n	training sample number
x_v	feature vector	m	transformation set number
$\mathbf{X}_v$	feature matrix	$\mathbf{B}$	unified binary descriptor
$\mathbf{S}$	affinity matrix	v	transformation set index
$\mathbf{L}$	Laplacian matrix	$\mathcal{H}(.)$	deep embedding function
k	code length	$\mathbf{W}_v$	latent embedding matrix
α_v	weight factors	f	real-valued feature vector
β,γ	balance parameters	th	threshold
z	weak bit mark	Θ	network parameter

and $\tilde{\mathbf{X}_v}$ are applied as the auxiliary parameters in the equation deduction, which are omitted in this table.

B. Learning Unified Binary Descriptor

In this section, we analyze two involved sub-modules within the unified binary descriptor learning: collective binary embedding and unsupervised graph learning.

1) Collective Binary Embedding: The ideas behind this module can be explained from two aspects: 1) the original image patch and its transformed versions should be encoded with the same binary descriptor, which can be achieved via embedding all those sets into a *common* Hamming space; 2) the unified binary code is learned from different transformation sets, which encodes the nature of transformation invariance to the maximum. Particularly, we formulate the objective function of this part as below:

$$\min_{\mathbf{B},\mathbf{W}_v,\alpha_v} \sum_{v=1}^{m} (\alpha_v)^{\gamma} \, \|\mathbf{X}_v - \mathbf{W}_v \mathbf{B}\|_{2,1},$$

$$\text{s.t. } \mathbf{B} \in \{-1,+1\}^{k \times n}, \ \sum_{v=1}^{m} \alpha_v = 1, \ \alpha_v > 0. \qquad (1)$$

where $\mathbf{B} \in \{-1,+1\}^{k \times n}$, $\sum_{v=1}^{m} \alpha_v = 1$ and $\alpha_v > 0$. Here, $\mathbf{X}_v$ are the deep features extracted from the $fc7$ layer of the pretrained VGG-16 model, $\mathbf{W}_v \in \mathbb{R}^{p_v \times k}$ are the latent embedding matrices that connect the unified binary descriptor with the deep features. α_v are the weight factors that measure the contributions of different transformation sets in learning the binary descriptor. γ is the balance parameter. $\ell_{2,1}$-norm is defined as $\|Y\|_{2,1} = \sum_{i=1}^{n} \|y_i\|_2$ for a matrix $Y = [y_1, y_2, \ldots, y_n] \in \mathbb{R}^{p \times n}$ [28].

Generally speaking, this module is proposed to encode the transformation invariance maximally in the to-be-learned binary descriptor via applying affine-transformation and performing matrix factorization on every single patch. It is worth noting that this module differs data augmentation in traditional classification tasks. From the functionality perspective, data augmentation involves the process of creating new data points by manipulating the original data to increase the training data diversity, thus avoiding overfitting [63]. However, the overfitting issue is not our concern here and the transformed data is provided merely for the proposed invariance encoding. From the technical perspective, being identified as the same category label is the only optimization goal for the original image and its augmented ones in the classification. The same category label does not necessarily guarantee the same feature descriptor. In our method, Eq. (1) regularizes all transformation sets of each patch to be represented by a unified binary descriptor (i.e., feature). Therefore, our learning objective is more stringent and optimizing such complicated loss functions is much more challenging.

More importantly, the proposed embedding function is upgraded to make it compatible with the local binary descriptor learning [19], [31]. First, $\ell_{2,1}$-norm is introduced into the discrete optimization model to reduce the negative effects caused by severe noises/distortions. In contrast to the widely-used squared ℓ_2-norm that is prone to noises/outliers, $\ell_{2,1}$-norm is a more rational choice in the patch-level transformation invariant descriptor learning. On one hand, ℓ_1-norm distance at the patch level provides the robustness against outlier samples after random transformations in this case (see Fig. 1(b)). On the other hand, ℓ_2-norm distance enables the allocation of the errors to each bit uniformly across the space dimension. Doing so lowers the possibility that certain bits are mistakenly flipped after getting large errors [26]–[28], [64], [65]. Those flipped bits may dramatically disturb the subsequent Hamming distance measure. Moreover, we solve the unified binary representation $\mathbf{B}$ directly under the restrictions of $\ell_{2,1}$-norm in the proposed model, where the accumulated quantization errors from the two-step learning paradigms [19], [26], [29], [31], [66] are avoided, thus enhancing the robustness of the learned binary code.

2) Unsupervised Graph Learning: As discussed above, the essence of the binary descriptor learning can be described as a process of projecting the high-dimensional original features into the compact binary space properly [9], [18], [26]. During the projection, the neighbourhood relationship preservation plays an important role in *generating the similar binary descriptors for those data (images/patches) that belong to the same category*. In this work, an unsupervised Laplacian constraint is derived from the *original* data set and imposed on all the transformation sets during the optimization, which shares a similar idea in common dictionary learning [26], [32], [67]. The reason is that the relative positions of data points in the feature space will be shifted after geometric transformations and provide unreliable neighborhood structures within the transformation sets [3], [27], [68]. That will mislead the unified binary descriptor learning and thus adversely affect the code quality. The basic functionality of using such Laplacian term is to keep the consistency between the original and binary feature spaces during the code learning [29], [30], [69]. Let $\mathbf{B}_{*,j}$ and $\mathbf{B}_{*,l}$ denote the j-th and l-th columns of $\mathbf{B}$, the affinity matrix $\mathbf{S} \in \mathbb{R}^{n \times n}$ from the original patch set, the graph problem can be formulated below:

$$\min_{\mathbf{B}} \frac{1}{2} \sum\nolimits_{j=1}^{n} \sum\nolimits_{l=1}^{n} \|\mathbf{B}_{*,j} - \mathbf{B}_{*,l}\|_F^2 \mathbf{S}_{j,l} = \min_{\mathbf{B}} tr(\mathbf{B}\mathbf{L}\mathbf{B}^T), \qquad (2)$$

where $\mathbf{B} \in \{-1,+1\}^{k \times n}$. $\mathbf{L} \in \mathbb{R}^{n \times n}$ is the Laplacian constraint and computed as $\mathbf{L} = diag(\mathbf{S1}) - \mathbf{S}$. $diag(\mathbf{S1})$ represents the diagonal matrices with each diagonal element being calculated as the sum of values in the corresponding row of $\mathbf{S}$, where $\mathbf{S}$ is constructed via k-Nearest-Neighbour (kNN). Particularly, the anchor graph scheme [70] can be adopted to reduce the computational complexity following the previous works [69], [71].

Based on the discussions, the unified binary code for the training data can be learned via jointly optimizing the above learning objectives. By incorporating Eq. (2) into Eq. (1), the overall objective function of unified binary descriptor learning can be formulated as:

$$\min_{\mathbf{B},\mathbf{W}_v,\alpha_v} \sum_{v=1}^{m} (\alpha_v)^{\gamma} (\|\mathbf{X}_v - \mathbf{W}_v\mathbf{B}\|_{2,1} + \beta tr(\mathbf{BLB}^T)), \tag{3}$$

where $\sum_{v=1}^{m} \alpha_v = 1$, $\alpha_v > 0$. $\mathbf{W}_v \in \mathbb{R}^{p_v \times k}$, $\mathbf{L} \in \mathbb{R}^{n \times n}$ and $\mathbf{B} \in \{-1, +1\}^{k \times n}$. β is the balance parameter.

C. Optimization Algorithm

It is intractable to solve the objective function Eq. (3) directly because of the discrete-constrained conditions and the non-convex $\ell_{2,1}$-norm term, which refers to an NP-hard problem [26]. Consequently, an alternating optimization strategy is employed to tackle this issue, which is presented as the following steps.

1) $\mathbf{W}_v$ *Step:* For $\mathbf{W}_v$ with other parameters fixed, the objective function in Eq. (3) can be simplified as follow:

$$\begin{aligned} \psi_v &= \min_{\mathbf{W}_v} \|\mathbf{X}_v - \mathbf{W}_v\mathbf{B}\|_{2,1} \\ &= \min_{\mathbf{W}_v} \sum_{i=1}^{n} \|\mathbf{X}_v^i - \mathbf{W}_v\mathbf{B}^i\|_2, \end{aligned} \tag{4}$$

where $\mathbf{W}_v \in \mathbb{R}^{p_v \times k}$, $\mathbf{X}_v^i$ and $\mathbf{B}^i$ are the i-th columns of $\mathbf{X}_v$ and $\mathbf{B}$, respectively. Then we can calculate the gradient of ψ_v with respect to $\mathbf{W}_v$ as:

$$\begin{aligned} \frac{\partial \psi_v}{\partial \mathbf{W}_v} &= \sum_{i=1}^{n} \frac{\mathbf{W}_v\mathbf{B}^i(\mathbf{B}^i)^T - \mathbf{X}_v^i(\mathbf{B}^i)^T}{\|\mathbf{X}_v^i - \mathbf{W}_v\mathbf{B}^i\|_2} \\ &= (\mathbf{W}_v\mathbf{B} - \mathbf{X}_v)\mathbf{D}_v\mathbf{B}^T. \end{aligned} \tag{5}$$

Here, the diagonal matrix $\mathbf{D}_v$ are led into the problem and its i-th diagonal element is obtained as:

$$(\mathbf{D}_v)_{i,i} = \frac{1}{\|\mathbf{X}_v^i - \mathbf{W}_v\mathbf{B}^i\|_2}. \tag{6}$$

Although there is no closed-form solution for $\mathbf{W}_v$ in the above equation, the calculation of $(\mathbf{X}_v - \mathbf{W}_v\mathbf{B})$ can be leveraged to compute $\mathbf{D}_v$ and $\frac{\partial \psi_v}{\partial \mathbf{W}_v}$ directly with the minimal efforts. Then a gradient descent strategy can be employed to optimize the objective function [72].

2) $\mathbf{B}$ *Step:* For $\mathbf{B}$ with other parameters fixed, the objective function (3) can be further rewritten as follow:

$$\min_{\mathbf{B}} \sum_{v=1}^{m} (\alpha_v)^{\gamma} (\|\mathbf{X}_v - \mathbf{W}_v\mathbf{B}\|_{2,1} + \beta tr(\mathbf{BLB}^T)), \tag{7}$$

where $\mathbf{B} \in \{-1, 1\}^{k \times n}$. Inspired by recent coordinate descent based methods [29], the objective loss can be minimized via optimizing all the bits in $\mathbf{B}$ sequentially. Here, we denote $b^T \in \{-1, 1\}^{1 \times n}$ as the i-th row of $\mathbf{B}$, and $\mathbf{B}'$ the matrix of $\mathbf{B}$ excluding b^T. Let $w_v \in \mathbb{R}^{p_v}$ be the i-th column of $\mathbf{W}_v$, $\mathbf{W}'_v$ be the matrix of $\mathbf{W}_v$ excluding w_v. Considering $\mathbf{W}_v\mathbf{B} = \mathbf{W}'_v\mathbf{B}' + w_v b^T$, $tr(\mathbf{BLB}^T) = tr(\mathbf{B}'\mathbf{L}\mathbf{B}'^T) + b^T\mathbf{L}b$ and $tr(\mathbf{B}'\mathbf{L}\mathbf{B}'^T)$ is *const*, (7) with respect to $b \in \{-1, 1\}^n$ can be formulated as:

$$\min_{b} \sum_{v=1}^{m} (\alpha_v)^{\gamma} (\|\mathbf{X}_v - \mathbf{W}'_v\mathbf{B}' - w_v b^T\|_{2,1} + \beta b^T\mathbf{L}b). \tag{8}$$

Let $\widetilde{\mathbf{X}}_v = \mathbf{X}_v - \mathbf{W}'_v\mathbf{B}'$, Eq. (8) is further simplified as:

$$\min_{b} \sum_{v=1}^{m} (\alpha_v)^{\gamma} (\|\widetilde{\mathbf{X}}_v - w_v b^T\|_{2,1} + \beta b^T\mathbf{L}b). \tag{9}$$

The above derivations transform the objective function into a similar form like Binary Quadratic Problem (BQP), but more complex. The closed-form solution of b cannot be obtained directly from Eq. (9). Following the previous works, it is still feasible to optimize the objective function via flipping each bit sequentially in b, where the bit would be flipped if the flipping operation decreases the objective function loss [29]. Fortunately, the initial value for b, denoted as b^0, can be set properly to minimize the first term in Eq. (9). Namely, the j-th bit in b^0 is calculated as:

$$b_j^0 = sign(\sum_{v=1}^{m} (\alpha_v)^{\gamma} (\|\widetilde{\mathbf{X}}_v^j + w_v\|_2 - \|\widetilde{\mathbf{X}}_v^j - w_v\|_2)), \tag{10}$$

where $b_j^0 \in \{-1, 1\}$ and $\widetilde{\mathbf{X}}_v^j \in \mathbb{R}^{p_v}$ is the j-th column of $\widetilde{\mathbf{X}}_v$. $Sign(x) = 1$ if $x \geq 0$ and otherwise -1. After getting b^0, we can flip each bit sequentially as in [29] to optimize the objective function.

3) α_v *Step:* For α_v with other parameters fixed and let $\mathbf{G}_v = \|\mathbf{X}_v - \mathbf{W}_v\mathbf{B}\|_{2,1} + \beta tr(\mathbf{BLB}^T)$, we can rewrite Eq. (3) as:

$$\min_{\alpha_v} \sum_{v=1}^{m} (\alpha_v)^{\gamma}\mathbf{G}_v, \ s.t. \sum_{v=1}^{m} \alpha_v = 1, \alpha_v > 0. \tag{11}$$

By introducing the Lagrange multiplier η, the above problem is then transformed to:

$$\min \mathcal{E}(\alpha_v, \eta) = \sum_{v=1}^{m} (\alpha_v)^{\gamma}\mathbf{G}_v - \eta(\sum_{v=1}^{m} (\alpha_v)^{\gamma} - 1), \tag{12}$$

where the partial derivatives with respect to α_v and η are calculated as:

$$\begin{cases} \dfrac{\partial \mathcal{E}_v}{\partial \alpha_v} = \gamma\,(\alpha_v)^{\gamma-1}\mathbf{G}_v - \eta, \\ \dfrac{\partial \mathcal{E}_v}{\partial \eta} = \sum_{v=1}^{m} \alpha_v - 1. \end{cases} \tag{13}$$

By setting those derivatives as 0, we have the optimal solution of α_v as:

$$\alpha_v = \frac{(\mathbf{G}_v)^{\frac{1}{1-\gamma}}}{\sum_{v=1}^{m} (\mathbf{G}_v)^{\frac{1}{1-\gamma}}}. \tag{14}$$

By repeating the above steps, the objective function converges to a local minimum after a few iterations (the iteration number $t \leq 10$ in the experiment), thus obtaining unified binary descriptors for the training data. The major difference against the previous discrete optimization strategies is that only the gradient descent is performed to make the overall objective function keep decreasing in the proposed method. There is no need to find the closed-form solution for each variable during each optimization iteration [29], [31], [73].

D. Generating Out-of-Sample Binary Descriptor

After learning the binary descriptors for training data, a unified deep embedding function $\mathcal{H}(\mathbf{X}_v; \Theta)$ is trained as the code generator for out-of-sample data. Particularly, the input data $\mathbf{X}_v$ from multiple sets ($v = 1, \ldots, m$) are sequentially fed

Algorithm 1 Unsupervised Deep Binary Descriptor

```
Input: Deep features $\mathbf{X}_v$ for different transformation sets,
    code length $k$, parameters $\beta$ and $\gamma$, Laplacian matrix $\mathbf{L}$,
    maximum epoch $T$. Randomly initialize binary code $\mathbf{B}$,
    latent embedding matrices $\mathbf{W}_v$ and deep parameters $\Theta$.
    Set average weights $\alpha_v$, $v = \{1, ..., m\}$.
Output: Deep hash function $\mathcal{H}(\mathbf{X}_v; \Theta)$;
Extract the feature matrices $\mathbf{X}_v$ from $fc7$ layers;
for $t = 1$ to $T$ do
    Update the latent embedding matrices $\mathbf{W}_v$ by Eq. (5)~(6);
    Update the unified hash code $\mathbf{B}$ by Eq. (8)~(10);
    Update the weight factors $\alpha_v$ by Eq. (14);
end for
Update the network parameters $\Theta$ by Eq. (15);
return $\mathcal{H}(\mathbf{X}_v; \Theta)$;
```

into the deep network and the Euclidean distances between feature vectors from the last output layer and their corresponding binary representations $\mathbf{B}$ are minimized, as shown in Fig. 2. By doing so, geometric transformation invariance could be preserved maximally during the deep embedding function learning. Moreover, the computational complexity can be reduced by updating the sharing weight Θ for the original data and its transformation sets simultaneously, instead of training different deep networks for them separately as in [25]. The objective function of this process is presented as:

$$\min_{\Theta} \sum_{v=1}^{m} \|\mathcal{H}(\mathbf{X}_v; \Theta) - \mathbf{B}\|_F^2, \ \text{s.t. } \mathbf{B} \in \{-1, +1\}^{k \times n}. \quad (15)$$

The optimization problem can be solved by fine-tuning the deep network with Stochastic Gradient Descent (SGD), where the shared weight Θ is iteratively optimized until convergence. Given a query instance $\mathbf{x}_q$, we can obtain its binary descriptor by simply calculating $sign(\mathcal{H}(\mathbf{x}_q; \Theta))$. The proposed algorithm is summarized in Algorithm 1.

E. Refined Matching via Weak Bit Selection

Once we have obtained binary descriptors for both query and gallery data, the matching can be done by comparing their Hamming distance. However, as the binary representation reduces the discriminative power of data, it is often that there are multiple candidates with the same minimum Hamming distance (even 0 in the worst-case scenarios) to a specific query (see Fig. 1(b)). It might be acceptable for applications like retrieval but is definitely problematic for local feature points matching, where one true match should be provided. In this case, a means to conduct the second distance measurement is required. Inspired by the advocate of unreliable bit in fingerprinting systems [33], [74]–[77], we found that the contribution/reliability of each bit within the binary codes differs. Hence, such information can be useful to refine the initial Hamming distance computation. Concretely, the unreliable bits (with values closed to 0) for each input $x \in \mathbb{R}^p$ are selected based on its real-valued vector $f \in \mathbb{R}^k$, which is extracted from the last output layer of the deep embedding network. With a certain threshold $th > 0$, the weak bit $z \in \{0, 1\}^k$ in its binary code $b \in \{-1, 1\}^k$ can be defined as:

$$z_k = \begin{cases} 1, & |f_k| < th; \\ 0, & |f_k| \geq th, \end{cases} \quad (16)$$

where the bits with values in the range of $(-th, th)$ are marked as weak bits (ie, $z_k = 1$). Here, the intuition is that the closer the real-valued feature gets to 0 the weaker it will be. This does make sense because the value closer to 0 is likely to be mistakenly flipped in the existence of noises, considering the fact that we use a *sign* function to convert a real value to a binary bit. In this second matching procedure, a sequence of binary digits of a query, formed by weakness indications at each bit location, will be compared against the counterpart digits of a candidate. As a result of doing this, the aggregated distance enables to find the best match, thus improving the matching performance.

IV. Experiment

In this section, we conduct extensive experiments on four public datasets to evaluate the matching and retrieval performance of the proposed binary descriptor.

A. Dataset Descriptions

Brown [11] is the most popular dataset in the evaluation of local feature descriptors, which contains three subsets: *Notre Dame*, *Yosemite*, and *Liberty* collected from the Photo Tourism reconstructions. In each subset, there are more than $400,000$ gray-scale patches with the size of 64×64. Those patches are split into training and test sets, which contains $200,000$ pairs ($100,000$ matched and non-matched pairs) and $100,000$ pairs ($50,000$ matched and non-matched pairs), respectively. Cifar-10 [78] consists of $60,000$ images with the size of 32×32 from 10 different categories, which are split into training and test sets with $50,000$ and $10,000$ images separately. The training set is employed for the code learning, and use the test set as the queries for retrieval evaluation. NUS-WIDE [79] is a multi-labeled dataset consists of 269,648 images within 81 concepts. After removing the invalid image urls, 21 most frequent topics are selected, which create a subset of 195,834 images. 100 images for each topic are randomly picked as the testing set. HPatches [80] consists of about 1 million patches extracted from 116 images using the combination of various interest point detectors, where the patches are collected from the 3D reconstructions of several landmarks in Rome. Each patch is annotated with its ground truth label and then post-processed after extraction with a fixed size of 65×65. We follow the default settings in [16] and test the performance on the full split within the dataset.

B. Implementation Details

The experiments are carried out on Linux Ubuntu Server with the configuration of Intel i7-5960X CPU@3.0GHz, 64GB RAM and NVIDIA GTX 1080 Ti GPU. Most source codes of the baselines are publically available online, which can be tuned via open source software (e.g., *Caffe* [82]) according to the papers. Specifically, the geometric transformations of the input patches are implemented by following the data augmentation in [25], where the rotation angles are within the

TABLE II

COMPARISON OF THE PROPOSED UDBD TO THE STATE-OF-THE-ART BINARY DESCRIPTORS IN TERMS OF FPR@95% ON BROWN DATASET. DIM, SP AND USP DENOTE DIMENSION, SUPERVISED AND UNSUPERVISED, RESPECTIVELY. † AND ‡ INDICATE THE TRAIN AND TESTING SUBSETS. THE RESULTS FROM SIFT AND SUPERVISED METHODS ARE PROVIDED AS REFERENCES. BOLD VALUES ARE THE BEST RESULTS IN UNSUPERVISED BINARY DESCRIPTORS

Method	**Dim**	**Type**	**Notre Dame**†	**Notre Dame**†	**Liberty**†	**Liberty**†	**Yosemite**†	**Yosemite**†	**Average**
			Liberty‡	**Yosemite**‡	**Notre Dame**‡	**Yosemite**‡	**Notre Dame**‡	**Liberty**‡	**FPR@95%**
SIFT [1]	128	USP	36.27	29.15	28.09	29.15	28.09	36.27	31.17
BinBoost [10]	64	SP	20.49	18.96	16.9	22.88	14.54	21.67	19.24
L2-Net [17]	128	SP	7.53	7.74	5.92	9.12	5.43	9.25	7.49
HardNet [46]	128	SP	2.22	2.28	0.57	2.13	0.96	2.35	1.9
CDbin [16]	128	SP	6.81	3.02	7.92	3.02	4.26	9.0	6.46
BRIEF [13]	256	USP	59.15	54.96	54.57	54.96	54.57	59.15	56.23
BRISK [12]	512	USP	79.36	73.21	74.88	73.21	74.88	79.36	75.82
ORB [14]	256	USP	56.26	54.13	48.03	54.13	48.03	56.26	52.81
DBD-MQ [21]	256	USP	31.1	57.24	25.78	57.15	27.2	33.11	38.59
BinGAN [57]	256	USP	25.76	**40.8**	27.84	**47.64**	16.88	26.08	30.83
DeepBit [25]	256	USP	33.83	54.63	20.66	56.69	28.49	34.64	38.15
GraphBit [2]	256	USP	24.24	50.54	16.75	49.11	21.09	27.23	31.49
UDBD	256	USP	**18.99**	52.6	**11.76**	52.17	**14.61**	**20.79**	**28.49**

range of $[-10, 10]$. Particularly, 5 different rotation angles: $[-10, -5, 0, 5, 10]$, are imposed on each input patch, which simulates the small viewpoint variations from human perspective [25]. Their deep features are extracted from the *fc7* layer (4096-d) of the pre-trained VGG-16 [63].

In the proposed method, γ and β are set as 5 and 10^{-3} during the discrete optimization, while the discrete optimization usually converges within 10 iterations. The number of data points is set to 10,000 in the code learning. In the network training phase, the VGG-16 model is used as the backbone with the output size of k and $tanh$ as the activation function in the last *fc* layer. The back-propagation is performed in the whole network. The basic learning rate as 0.0001, momentum as 0.9 and weight decay as 0.0005. The batch size is 32 and the maximum iteration is 30000. The threshold is set to 0.3 via cross-validation in the weak bit selection.

C. Comparisons With State-of-the-Arts

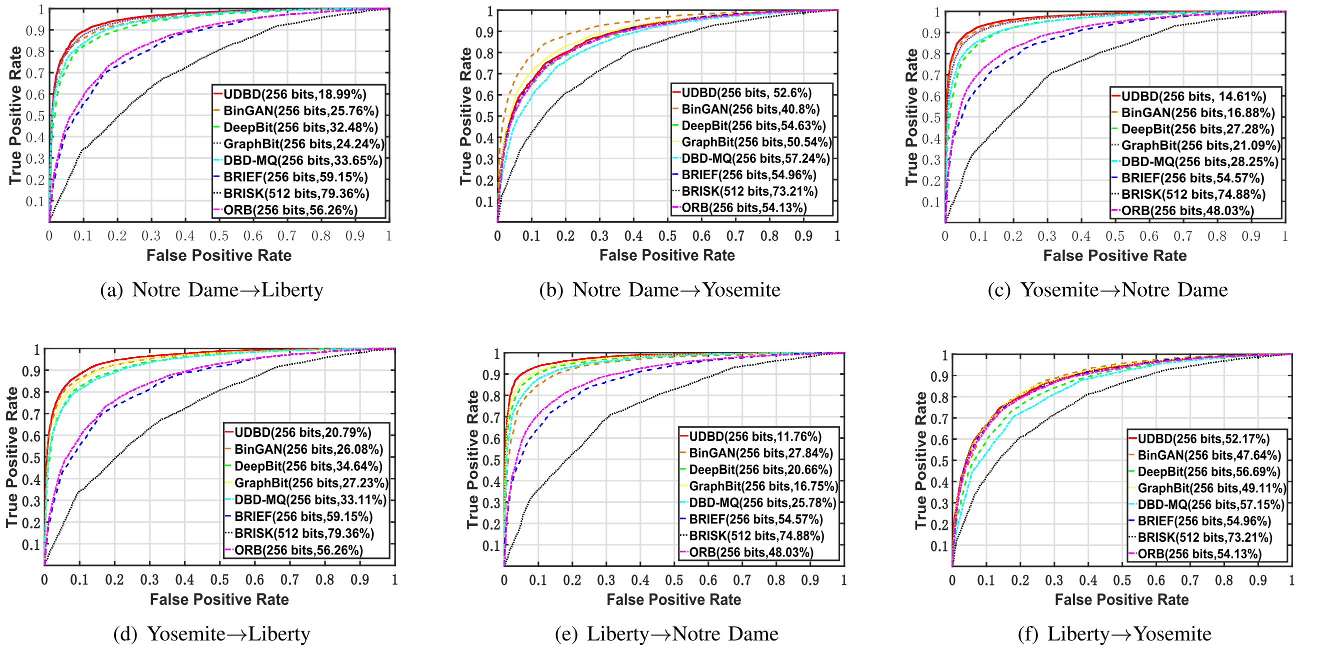
1) Results on Brown Dataset: On the Brown dataset, we conduct extensive comparisons on the patch matching performance between our approach and several state-of-the-art binary descriptors. These baselines are categorized into unsupervised (e.g., BRIEF [13], GraphBit [2] and DeepBit [25], etc.) and supervised approaches (e.g., BinBoost [10], L2-Net [17], HardNet [46] and CDbin [16]). The results from floating-pointed (SIFT [1])and supervised methods are provided as reference. Following [25] and [16], False Positive Rates at 95% (FPR@95%) from the cross-validations on three subsets are provided in Table II. *Lower FPR@95% indicates better performance.* As can be seen, the proposed method outperforms unsupervised approaches on most training and test configurations. Particularly, the error rates achieved by UDBD are 18.99%, 52.6%, 11.76%, 52.17%, 14.61% and 20.79% from bottom left to right. However, our method performs less favorable than BinGAN on *Yosemite*. The subset contains too many visually similar patches (e.g., snow and forest), which makes them difficult to be distinguished [25]. Nevertheless, UDBD still achieves the best average FPR@95% (28.49%) among unsupervised binary descriptors. Compared with the supervised methods, UDBD is highly competitive, where our method even has better result (11.76%) against BinBoost [10] (16.9%) on the setting of *Liberty* and *Notre Dame*.

Moreover, the ROC curves of those unsupervised descriptors on different subsets are plotted in Fig. 3 to further verify the above discussions. As shown in the figures, the ROC curves from UDBD rank at the top under most settings.

2) Results on Cifar-10 Dataset: Without loss of generality, on the Cifar-10 dataset, we first compare our method with several unsupervised binary descriptors regarding image retrieval performance, including the unsupervised binary descriptors and some classical hashing methods. The retrieval performance is evaluated under mean Average Precision (mAP) at top $1,000$ returned images, which is detailed in Table III at the code length of 16, 32 and 64. As observed from Table III, our method improves the mAP@1000 values by 2.19%, 1.52% over BinGAN on 16 and 32 bits, while 1.63% on 64 bits over GraphBit. Moreover, we provide the Precision-Recall curves on the Cifar-10 dataset at different code lengths in Fig. 4, where the results are consistent with the above discussions. Based on the results above, the proposed descriptor can outperform most existing local binary descriptors when tackling general image retrieval. However, when comparing with some specified image hashing methods like [49] and [50], most local binary descriptors fail to outperform the reported results in the papers. The main reason is that local binary descriptors have to address unique properties (e.g., anti-geometric transformation and noise) in the learning objective so as to encode more microtexture information in local patches for better matching performance, which might be suboptimal to deal with such general image retrieval. Nevertheless, the results exhibit the generality of our algorithm from the side.

Additionally, the matching results measured by Precision@Top 1 returned candidate from several state-of-the-arts are provided in Table IV, where the image is treated as a big patch. As can be seen, the proposed method obtains the highest values in terms of Precision at top 1, at least 4.74% higher than the most competitive local descriptors, which consolidates the contribution on improving the matching accuracy.

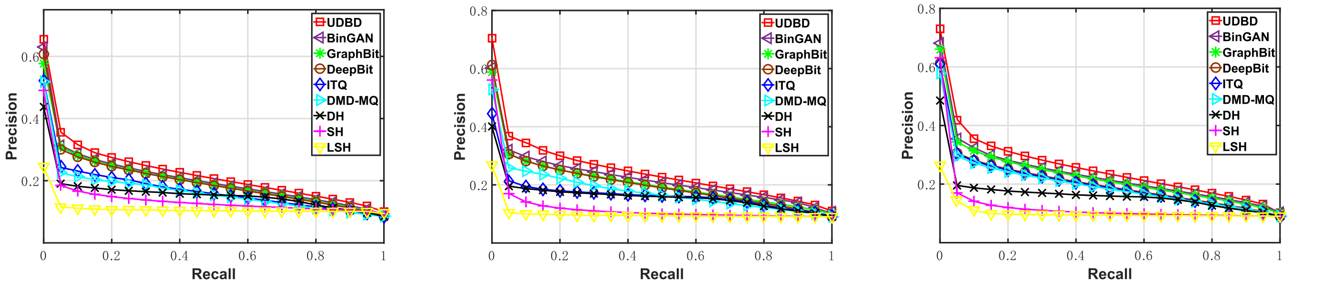
3) Results on NUS-WIDE Dataset: Then we briefly investigate the general image retrieval performance on NUS-WIDE



(a) Notre Dame→Liberty (b) Notre Dame→Yosemite (c) Yosemite→Notre Dame

(d) Yosemite→Liberty (e) Liberty→Notre Dame (f) Liberty→Yosemite

Fig. 3. ROC curves under different settings on Brown dataset when using various unsupervised binary descriptors.



(a) Cifar-10 at 16 bits (b) Cifar-10 at 32 bits (c) Cifar-10 at 64 bits

Fig. 4. Precision-Recall curves of the proposed method and the baselines on Cifar-10 dataset at 16, 32 and 64 bits.

TABLE III

MAP (%) OF TOP 1,000 RETURNED IMAGES AT DIFFERENT CODE LENGTH FROM VARIOUS UNSUPERVISED METHODS ON CIFAR-10 DATASET. BOLD VALUES ARE THE BEST RESULTS

Method	**mAP@1000 (%)**		
	16 bits	**32 bits**	**64 bits**
LSH [9]	10.31	11.39	13.74
ITQ [18]	24.85	27.32	30.84
SH [81]	16.25	19.64	20.91
DH [48]	22.43	23.21	25.84
DistillHash [51]	**34.56**	35.35	37.91
DBD-MQ [21]	21.53	26.5	31.85
BinGAN [57]	30.05	34.65	36.77
DeepBit [25]	26.36	27.92	34.05
GraphBit [2]	27.79	33.45	37.97
UDBD	32.24	**36.17**	**39.6**

dataset when using several unsupervised methods. Following previous works, the retrieval performance is evaluated under mAP at all returned images (mAP@all), which is reported in Table V at the code length of 16, 32 and 64. As can be seen, the proposed method can achieve higher results than these baselines at 32 and 64, where the gaps are 1.7% and 0.3% over DVB [52], respectively. The mAP value at 16 bits drops to 51.8%, which is still competitive against most baselines.

TABLE IV

PRECISION AT TOP 1 ON CIFAR-10 DATASET WHEN USING DEEPBIT, BINGAN, GRAPHBIT AND UDBD AT DIFFERENT BIT SIZES

Method	**Precision@Top 1 (%)**		
	16 bits	**32 bits**	**64 bits**
DeepBit [25]	24.38	32.51	39.74
BinGAN [57]	33.72	41.48	44.31
GraphBit [2]	32.12	41.39	46.79
UDBD	**38.46**	**46.63**	**52.06**

4) Results on HPatches Dataset: Finally, we report mAP values from the three visual tasks: matching, retrieval, and verification, on the HPatches dataset to provide broader insights on the binary descriptor performance. Specifically, the matching is conducted by comparing patch sets between a reference image and a target one and the retrieval aims at finding similar patches for each query. The verification is to classify whether two patches are matched or not [16], [80]. Following the evaluation protocols suggested in [16], [80], the results of the full split from HPatches are summarized in Table VI. We compared UDBD with several unsupervised binary descriptors and provided the results of SIFT [1], BinBoost [10], L2-Net [17] and CDbin [16] for references. Table VI shows that UDBD outperforms the most competitive

TABLE V

MAP (%) OF ALL RETURNED IMAGES AT DIFFERENT CODE LENGTH FROM VARIOUS UNSUPERVISED METHODS ON NUS-WIDE DATASET. BOLD VALUES ARE THE BEST RESULTS

Method	**mAP @all(%)**		
	16 bits	**32 bits**	**64 bits**
LSH [9]	23.9	26.6	26.6
SH [81]	34.6	35.8	36.5
ITQ [18]	51.2	52.6	53.8
DH [48]	40.4	46.7	42.7
DeepBit [25]	45.2	46.3	49.6
GraphBit [2]	51.8	55.2	58.1
Zhang, et al. [49]	**57.4**	56.2	53.8
DVB [52]	55.4	56.2	59.5
UDBD	52.3	**57.9**	**59.8**

TABLE VI

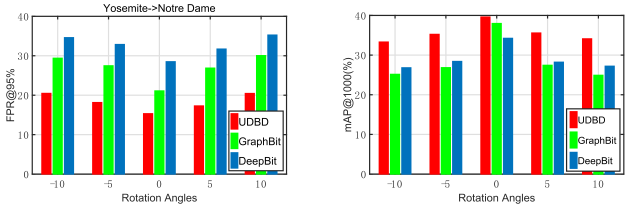
COMPARISON OF THE PROPOSED UDBD TO THE STATE-OF-THE-ART DESCRIPTORS IN TERMS OF MAP (%) ON HPATCHES DATASET. DIM, SP AND USP DENOTE DIMENSION, SUPERVISED AND UNSUPERVISED. MATCH. RETRI. AND VERI. DENOTE MATCHING, RETRIEVAL AND VERIFICATION. THE REAL-VALUED DESCRIPTOR (SIFT) AND THE SUPERVISED METHODS ARE PROVIDED AS REFERENCES. BOLD VALUES ARE THE BEST RESULTS IN UNSUPERVISED BINARY DESCRIPTORS

Method	**Dim**	**Type**	**Match.**	**Retri.**	**Veri.**
SIFT [1]	128	USP	25.47	31.98	65.12
BinBoost [10]	64	SP	14.77	22.45	66.67
L2-Net [17]	128	SP	30.89	41.29	70.58
CDbin [16]	128	SP	39.76	46.19	82.68
BRIEF [13]	256	USP	10.5	16.03	58.07
ORB [14]	256	USP	15.32	18.85	60.15
DBD-MQ [21]	256	USP	13.45	23.56	63.43
DeepBit [25]	256	USP	13.05	20.61	61.27
GraphBit [2]	256	USP	14.22	25.19	65.19
UDBD	256	USP	**17.27**	**28.88**	**69.77**

TABLE VII

ABLATION STUDY ON BROWN (FPR@95%): *Liberty (LIB)→Notre Dame (ND)* AND *Yosemite (YOS)→Liberty (LIB)*, HPATCHES: *Matching* (MAP) AND CIFAR-10 AT 64 BITS (MAP@1000) WHEN $\gamma = 0$ (I.E., UDBD$_{\gamma=0}$), $\beta = 0$ (I.E., UDBD$_{\beta=0}$) AND $\gamma = \beta = 0$ (I.E., UDBD$_{\gamma=\beta=0}$). BOLD VALUES SHOW THE BEST RESULTS

Method	**Brown** [11]		**Cifar-10** [78]	**HPatches** [80]
	Lib→ND	**Yos→Lib**	**64 bits**	**Matching**
UDBD$_{\gamma=0}$	14.18	23.62	35.33	14.24
UDBD$_{\beta=0}$	12.92	22.36	34.08	14.95
UDBD$_{\gamma=\beta=0}$	14.43	24.17	33.99	13.96
UDBD	**11.76**	**20.79**	**39.6**	**17.27**



(a) Brown at 256 bits (b) Cifar-10 at 64 bits

Fig. 5. Performances variations: FPR95% on matching and mAP@1000 on retrieval, under different rotation angles on test instances from GraphBit, DeepBit and UDBD on Brown and Cifar-10.

TABLE VIII

PERFORMANCE VARIATIONS (%) ON BROWN (FPR@95%): *Notre Dame (ND)→Liberty (LIB)* AND *Liberty→Notre Dame*, HPATCHES: *Matching* (MAP) AT 256 BITS, AND CIFAR-10 AT 32 BITS (MAP@1000) WHEN USING $\ell_{2,1}$-NORM AND $\ell_{2,2}$-NORM LOSS TERMS. BOLD VALUES SHOW THE BEST RESULTS

Loss	**Brown** [11]		**Cifar-10** [78]	**HPatches** [80]
	ND→Lib	**Lib→ND**	**32 bits**	**Matching**
$\ell_{2,2}$	22.51	15.81	34.24	14.81
$\ell_{2,1}$	**18.99**	**11.76**	**36.17**	**17.27**

GraphBit by 3.05%, 3.69% and 4.58% on matching, retrieval, and verification, respectively, which indicates the superiority of UDBD in generating effective binary descriptors for various visual tasks.

D. Further Analysis

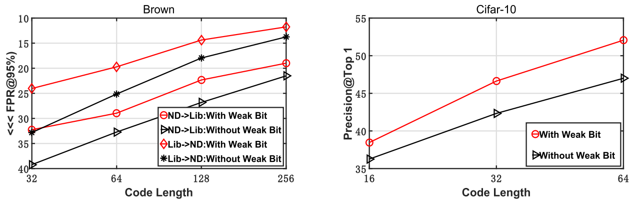
In this section, further insights are provided to address some key features in our proposed method.

1) Ablation Study: Firstly, the comprehensive analysis of the contribution of those involved components: view weighting scheme and Laplacian constraint, during the code learning, is provided in Table VII. Particularly, three different settings: $\gamma = 0$ (i.e., UDBD$_{\gamma=0}$: NO view weighting scheme), $\beta = 0$ (i.e., UDBD$_{\beta=0}$: NO graph loss term) and $\gamma = \beta = 0$ (i.e., UDBD$_{\gamma=\beta=0}$), are investigated on various datasets. For instance, mAP@1,000 result at 64 bits on Cifar-10 when $\beta = 0$ would decrease dramatically to 34.08%. These values with $\gamma = 0$ and $\gamma = \beta = 0$ are 35.33% and 33.99%, which are far below than the original result (39.6%) achieved by UDBD in Table III. That indicates the importance and necessity of the involved graph loss term and view weighting scheme in the proposed framework.

2) Transformation Invariance: Then we investigate the performance variations: FPR95% on matching and mAP@1000 on retrieval, under certain affine transformation imposing on test images, where rotation is given as an example as plotted in Figure 5. The variations are calculated at 64 bits from GraphBit, DeepBit, and UDBD. Large rotation angles usually reduce visual similarity on the images, thus yielding worse performance [25]. However, UDBD still outperforms the others significantly at all angle ranges. Particularly, mAP@1000 for UDBD is 34.1% when rotating 10 degrees, which is much higher than those achieved by DeepBit (27.26%) and GraphBit (23.51%). That indicates the proposed local binary descriptor is more robust to the rotation because of the collective binary embedding and unified network training. More complicated transformations such as scaling, translation and occlusion will be considered in future work.

3) Loss Term: Moreover, we report the performance variations when using different loss terms (i.e., $\ell_{2,1}$-norm *vs* $\ell_{2,2}$-norm) in the code learning process, as shown in Table VIII. $\ell_{2,2}$-norm is selected as baseline because of its wide usage and high competitiveness. For example, on *Liberty* (*Lib*)→*Notre Dame* (*ND*), FPR@95% is 15.81% under $\ell_{2,2}$-norm, which is 4.05% lower than 11.76% when applying $\ell_{2,1}$-norm. Generally, $\ell_{2,1}$-norm loss yields better results compared to the widely used $\ell_{2,2}$-norm, which are consistent with the previous discussions on $\ell_{p,q}$-norm based similarity search and other regularizers even obtain worse performance [26].

4) Weak Bit Study: Then the impact of weak bit scheme on the system performance is investigated in Fig. 6 and Table IX,



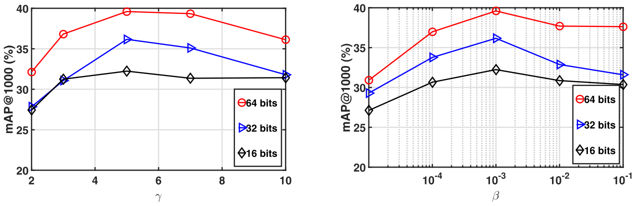
(a) Brown (b) Cifar-10

Fig. 6. Performance variations (%) at varying code lengths with/without applying weak bit scheme. (a) FPR@95% on Brown: *Notre Dame* (*ND*)→*Liberty* (*Lib*) and *Liberty*→*Notre Dame*; (b) Precision@Top 1 on Cifar-10.

TABLE IX

MAP VARIATIONS (%) ON HPATCHES WITH/WITHOUT APPLYING WEAK BIT SCHEME (UDBD‡/UDBD†). BOLD VALUES SHOW THE BEST RESULTS

Method	Matching	Retrieval	Verification
UDBD†	15.83	28.75	68.47
UDBD‡	**17.27**	**28.88**	**69.77**



(a) γ (b) β

Fig. 7. Parameter sensitivity analysis of γ and β at various bit sizes on Cifar-10 dataset.

under three measurements as FPR@95%, Precision@Top 1 and mAP on different datasets. As can be seen, noticeable performance gains have been achieved with the weak bit scheme on Brown and Cifar-10 datasets, especially when using shorter codes. For instance, Precision@Top 1 is 46.63% (with weak bit) and 42.33% (without weak bit) on Cifar-10 using 32 bits. While on HPatches, slight improvements also have been achieved by the proposed method when tackling three tasks (1.44%, 0.13% and 1.3%) at 256 bits with the weak bit scheme separately. The results show that the weak bit scheme plays a vital role in improving the matching performance, which further verifies the claimed contribution.

5) Parameter Analysis: Finally, more experiments are conducted on the Cifar-10 as examples in the retrieval performance analysis with varying hyperparameters (γ and β), as shown in Fig. 7. γ and β are varied in wide ranges from $\{2, 3, 5, 7, 10\}$ and $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$, where the best performance is given around the setting of 5 and 10^{-3}. It is worth noting that the performance degrades heavily when small β is being set, which inevitably weakens the impact of the graph constraint learning, thus yielding worse code quality.

V. CONCLUSION

In this paper, a novel learning-based unsupervised binary descriptor termed UDBD was proposed to facilitate large-scale visual recognition. Particularly, the binary descriptor is learned via exploiting the common binary space between the original and transformed data sets. With $\ell_{2,1}$-norm loss as a regularization term, the learned descriptor is highly robust to potential outliers. An unsupervised graph constraint is further employed to preserve the original manifold structure in the code learning, thus improving the code quality dramatically. Then the discrete and $\ell_{2,1}$-norm constrained objective function is solved directly without relaxation following an alternating discrete optimization strategy. Additionally, a weak bit scheme is used to address the ambiguous matching issue and further boost the matching performance of the proposed binary descriptor in the online search stage. Experiments on several public datasets show that UDBD outperforms state-of-the-arts significantly. In future work, we will, on one hand, apply the proposed binary descriptor to video retrieval [83], cross-modal retrieval [84] and matching [85]; and on the other hand, stabilize our networks via more suited Gradient Descent algorithm, e.g. CoGD [86].

REFERENCES

[1] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[2] Y. Duan, Z. Wang, J. Lu, X. Lin, and J. Zhou, "GraphBit: Bitwise interaction mining via deep reinforcement learning," in *Proc. CVPR*, Jun. 2018, pp. 8270–8279.

[3] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[4] H. Ren and Z.-N. Li, "Object detection using boosted local binaries," *Pattern Recognit.*, vol. 60, pp. 793–801, Dec. 2016.

[5] A. Moeini, K. Faez, H. Sadeghi, and H. Moeini, "2D facial expression recognition via 3D reconstruction and feature fusion," *J. Vis. Commun. Image Represent.*, vol. 35, pp. 1–14, Feb. 2016.

[6] R. Szeliski, *Computer Vision: Algorithms and Applications*. Berlin, Germany: Springer, 2010.

[7] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Proc. ECCV*. Berlin, Germany: Springer, 2006, pp. 430–443.

[8] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Proc. ECCV*. Berlin, Germany: Springer, 2006, pp. 404–417.

[9] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," in *Proc. 47th Annu. IEEE Symp. Found. Comput. Sci.*, Jun. 2006, pp. 459–468.

[10] T. Trzcinski, M. Christoudias, P. Fua, and V. Lepetit, "Boosting binary keypoint descriptors," in *Proc. CVPR*, Jun. 2013, pp. 2874–2881.

[11] M. Brown, G. Hua, and S. Winder, "Discriminative learning of local image descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 43–57, Jan. 2011.

[12] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *Proc. ICCV*, Nov. 2011, pp. 2548–2555.

[13] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *Proc. ECCV*. Springer, 2010, pp. 778–792.

[14] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. IICCV*, Nov. 2011, pp. 2564–2571.

[15] A. Alahi, R. Ortiz, and P. Vandergheynst, "FREAK: Fast retina keypoint," in *Proc. CVPR*, Jun. 2012, pp. 510–517.

[16] J. Ye, S. Zhang, T. Huang, and Y. Rui, "CDbin: Compact discriminative binary descriptor learned with efficient neural network," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 3, pp. 862–874, Mar. 2020.

[17] Y. Tian, B. Fan, and F. Wu, "L2-net: Deep learning of discriminative patch descriptor in Euclidean space," in *Proc. CVPR*, Jul. 2017, pp. 661–669.

[18] 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.

[19] G. Ding, Y. Guo, J. Zhou, and Y. Gao, "Large-scale cross-modality search via collective matrix factorization hashing," *IEEE Trans. Image Process.*, vol. 25, no. 11, pp. 5427–5440, Nov. 2016.

[20] 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.

[21] Y. Duan, J. Lu, Z. Wang, J. Feng, and J. Zhou, "Learning deep binary descriptor with multi-quantization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 8, pp. 1924–1938, Aug. 2019.

[22] F. Shen, C. Shen, Q. Shi, A. van den Hengel, Z. Tang, and H. T. Shen, "Hashing on nonlinear manifolds," *IEEE Trans. Image Process.*, vol. 24, no. 6, pp. 1839–1851, Jun. 2015.

[23] Z. Lin, G. Ding, J. Han, and J. Wang, "Cross-view retrieval via probability-based semantics-preserving hashing," *IEEE Trans. Cybern.*, vol. 47, no. 12, pp. 4342–4355, Dec. 2017.

[24] X. Liu, J. He, B. Lang, and S.-F. Chang, "Hash bit selection: A unified solution for selection problems in hashing," in *Proc. CVPR*, Jun. 2013, pp. 1570–1577.

[25] K. Lin, J. Lu, C.-S. Chen, J. Zhou, and M.-T. Sun, "Unsupervised deep learning of compact binary descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 6, pp. 1501–1514, Jun. 2019.

[26] Y. Guo, G. Ding, and J. Han, "Robust quantization for general similarity search," *IEEE Trans. Image Process.*, vol. 27, no. 2, pp. 949–963, Feb. 2018.

[27] W. Jiang, F. Nie, and H. Huang, "Robust dictionary learning with capped ℓ_1-norm," in *Proc. IJCAI*, 2015, pp. 3590–3596.

[28] J. Liu, S. Ji, and J. Ye, "Multi-task feature learning via efficient $\ell_{2,1}$-norm minimization," in *Proc. IJCAI*. Monterey, CA, USA: AUAI Press, 2009, pp. 339–348.

[29] F. Shen, C. Shen, W. Liu, and H. T. Shen, "Supervised discrete hashing," in *Proc. CVPR*, Jun. 2015, pp. 37–45.

[30] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Proc. NIPS*, 2009, pp. 1753–1760.

[31] J. Tang, K. Wang, and L. Shao, "Supervised matrix factorization hashing for cross-modal retrieval," *IEEE Trans. Image Process.*, vol. 25, no. 7, pp. 3157–3166, Jul. 2016.

[32] Y. Cao, M. Long, J. Wang, and S. Liu, "Collective deep quantization for efficient cross-modal retrieval," in *Proc. AAAI*, 2017, p. 5.

[33] J. Haitsma and T. Kalker, "A highly robust audio fingerprinting system," in *Proc. Ismir*, 2002, pp. 107–115.

[34] E. R. Nascimento, G. Potje, R. Martins, F. Chamone, M. Campos, and R. Bajcsy, "GEOBIT: A geodesic-based binary descriptor invariant to non-rigid deformations for RGB-D images," in *Proc. ICCV*, Oct. 2019, pp. 10004–10012.

[35] H. Yang, C. Huang, F. Wang, K. Song, and Z. Yin, "Robust semantic template matching using a superpixel region binary descriptor," *IEEE Trans. Image Process.*, vol. 28, no. 6, pp. 3061–3074, Jun. 2019.

[36] T. Trzcinski and V. Lepetit, "Efficient discriminative projections for compact binary descriptors," in *Proc. ECCV*. Berlin, Germany: Springer, 2012, pp. 228–242.

[37] V. Balntas, L. Tang, and K. Mikolajczyk, "BOLD–Binary online learned descriptor for efficient image matching," in *Proc. CVPR*, Jun. 2015, pp. 2367–2375.

[38] 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.

[39] H. Zhu, M. Long, J. Wang, and Y. Cao, "Deep hashing network for efficient similarity retrieval," in *Proc. AAAI*, 2016, pp. 2415–2421.

[40] Q. Li, Z. Sun, R. He, and T. Tan, "Deep supervised discrete hashing," in *Proc. NIPS*, 2017, pp. 2482–2491.

[41] A. Dosovitskiy, P. Fischer, J. T. Springenberg, M. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with exemplar convolutional neural networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 9, pp. 1734–1747, Sep. 2016.

[42] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer, "Discriminative learning of deep convolutional feature point descriptors," in *Proc. ICCV*, Dec. 2015, pp. 118–126.

[43] Y. Tian, X. Yu, B. Fan, F. Wu, H. Heijnen, and V. Balntas, "SOSNet: Second order similarity regularization for local descriptor learning," in *Proc. CVPR*, Jun. 2019, pp. 11016–11025.

[44] W. Li, L. Wang, J. Xu, J. Huo, Y. Gao, and J. Luo, "Revisiting local descriptor based image-to-class measure for few-shot learning," in *Proc. CVPR*, Jun. 2019, pp. 7260–7268.

[45] Z. Luo *et al.*, "ContextDesc: Local descriptor augmentation with cross-modality context," in *Proc. CVPR*, Jun. 2019, pp. 2527–2536.

[46] A. Mishchuk, D. Mishkin, F. Radenovic, and J. Matas, "Working hard to know your neighbor's margins: Local descriptor learning loss," in *Proc. NIPS*, 2017, pp. 4826–4837.

[47] K. He, Y. Lu, and S. Sclaroff, "Local descriptors optimized for average precision," in *Proc. CVPR*, Jun. 2018, pp. 596–605.

[48] V. E. Liong, J. Lu, G. Wang, P. Moulin, and J. Zhou, "Deep hashing for compact binary codes learning," in *Proc. CVPR*, Jun. 2015, pp. 2475–2483.

[49] H. Zhang, L. Liu, Y. Long, and L. Shao, "Unsupervised deep hashing with pseudo labels for scalable image retrieval," *IEEE Trans. Image Process.*, vol. 27, no. 4, pp. 1626–1638, Apr. 2018.

[50] Y. Zhu, Y. Li, and S. Wang, "Unsupervised deep hashing with adaptive feature learning for image retrieval," *IEEE Signal Process. Lett.*, vol. 26, no. 3, pp. 395–399, Mar. 2019.

[51] E. Yang, T. Liu, C. Deng, W. Liu, and D. Tao, "DistillHash: Unsupervised deep hashing by distilling data pairs," in *Proc. CVPR*, Jun. 2019, pp. 2946–2955.

[52] Y. Shen, L. Liu, and L. Shao, "Unsupervised binary representation learning with deep variational networks," *Int. J. Comput. Vis.*, vol. 127, nos. 11–12, pp. 1614–1628, Dec. 2019.

[53] X. Yu *et al.*, "Unsupervised extraction of local image descriptors via relative distance ranking loss," in *Proc. ICCV Workshops*, Oct. 2019, pp. 2893–2902.

[54] W. Zhang, X. Cao, R. Wang, Y. Guo, and Z. Chen, "Binarized mode seeking for scalable visual pattern discovery," in *Proc. CVPR*, Jul. 2017, pp. 3864–3872.

[55] J. Lu, V. E. Liong, X. Zhou, and J. Zhou, "Learning compact binary face descriptor for face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 10, pp. 2041–2056, Oct. 2015.

[56] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. NIPS*, 2014, pp. 2672–2680.

[57] M. Zieba, P. Semberecki, T. El-Gaaly, and T. Trzcinski, "BinGAN: Learning compact binary descriptors with a regularized GAN," in *Proc. NIPS*, 2018, pp. 3608–3618.

[58] J. Song, T. He, L. Gao, X. Xu, A. Hanjalic, and H. T. Shen, "Unified binary generative adversarial network for image retrieval and compression," *Int. J. Comput. Vis.*, vol. 128, no. 8, pp. 2243–2264, Feb. 2020.

[59] L. Fei, B. Zhang, Y. Xu, Z. Guo, J. Wen, and W. Jia, "Learning discriminant direction binary palmprint descriptor," *IEEE Trans. Image Process.*, vol. 28, no. 8, pp. 3808–3820, Aug. 2019.

[60] Y. Duan, J. Lu, J. Feng, and J. Zhou, "Context-aware local binary feature learning for face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 5, pp. 1139–1153, May 2018.

[61] J. Lu, V. E. Liong, and J. Zhou, "Simultaneous local binary feature learning and encoding for homogeneous and heterogeneous face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 8, pp. 1979–1993, Aug. 2018.

[62] Y. Duan, J. Lu, J. Feng, and J. Zhou, "Learning rotation-invariant local binary descriptor," *IEEE Trans. Image Process.*, vol. 26, no. 8, pp. 3636–3651, Aug. 2017.

[63] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556

[64] M. Qian and C. Zhai, "Robust unsupervised feature selection," in *Proc. IJCAI*, 2013, pp. 1621–1627.

[65] Y. Yang, H. T. Shen, Z. Ma, Z. Huang, and X. Zhou, "$\ell_{2,1}$-norm regularized discriminative feature selection for unsupervised," in *Proc. IJCAI*, 2011, pp. 1589–1594.

[66] W. Kong and W.-J. Li, "Isotropic hashing," in *Proc. NIPS*, 2012, pp. 1646–1654.

[67] Z. Zhang, L. Liu, F. Shen, H. T. Shen, and L. Shao, "Binary multi-view clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 7, pp. 1774–1782, Jul. 2019.

[68] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *Proc. CVPR*, Jun. 2016, pp. 2574–2582.

[69] W. Liu, C. Mu, S. Kumar, and S.-F. Chang, "Discrete graph hashing," in *Proc. NIPS*, 2014, pp. 3419–3427.

[70] W. Liu, J. Wang, S. Kumar, and S.-F. Chang, "Hashing with graphs," in *Proc. ICML*, Jun. 2011, pp. 1–8.

[71] F. Shen, Y. Xu, L. Liu, Y. Yang, Z. Huang, and H. T. Shen, "Unsupervised deep hashing with similarity-adaptive and discrete optimization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 3034–3044, Dec. 2018.

[72] F. Cakir and S. Sclaroff, "Adaptive hashing for fast similarity search," in *Proc. ICCV*, Dec. 2015, pp. 1044–1052.

[73] D. Wang, Q. Wang, and X. Gao, "Robust and flexible discrete hashing for cross-modal similarity search," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 10, pp. 2703–2715, Oct. 2018.

[74] Q. Lv, W. Josephson, Z. Wang, M. Charikar, and K. Li, "Multi-probe lsh: Efficient indexing for high-dimensional similarity search," in *Proc. VLDB*. Vienna, Austria: VLDB Endowment, 2007, pp. 950–961.

[75] S. Baluja and M. Covell, "Beyond 'near duplicates': Learning hash codes for efficient similar-image retrieval," in *Proc. 20th Int. Conf. Pattern Recognit.*, Aug. 2010, pp. 543–547.

[76] H. Shu, W. Jiang, and R. Yu, "Study on weak bit in vote count and its application in k-nearest neighbors algorithm," in *Proc. ICIEA*, Jun. 2015, pp. 119–122.

[77] R. Shinde, A. Goel, P. Gupta, and D. Dutta, "Similarity search and locality sensitive hashing using ternary content addressable memories," in *Proc. ACM SIGMOD*, 2010, pp. 375–386.

[78] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep. TR-2009, 2009.

[79] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng, "NUS-WIDE: A real-world Web image database from national University of Singapore," in *Proc. CIVR*, 2009, pp. 1–9.

[80] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk, "HPatches: A benchmark and evaluation of handcrafted and learned local descriptors," in *Proc. CVPR*, Jul. 2017, pp. 5173–5182.

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

[82] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. ACM Multimedia*, 2014, pp. 675–678.

[83] G. Wu *et al.*, "Unsupervised deep video hashing via balanced code for large-scale video retrieval," *IEEE Trans. Image Process.*, vol. 28, no. 4, pp. 1993–2007, Apr. 2019.

[84] G. Wu, J. Han, Z. Lin, G. Ding, B. Zhang, and Q. Ni, "Joint image-text hashing for fast large-scale cross-media retrieval using self-supervised deep learning," *IEEE Trans. Ind. Electron.*, vol. 66, no. 12, pp. 9868–9877, Dec. 2019.

[85] J. Han, E. J. Pauwels, and P. de Zeeuw, "Visible and infrared image registration in man-made environments employing hybrid visual features," *Pattern Recognit. Lett.*, vol. 34, no. 1, pp. 42–51, Jan. 2013.

[86] L. Zhuo *et al.*, "Cogradient descent for bilinear optimization," in *Proc. CVPR*, Jun. 2020, pp. 7956–7964.

Gengshen Wu is currently pursuing the Ph.D. degree with the School of Computing and Communications, Lancaster University, Lancaster, U.K. His research interests include large-scale multimedia retrieval and computer vision.

Zijia Lin is currently a Lead Researcher with Alibaba Group, China. His research interests include natural language processing, large-scale multimedia mining, and information retrieval.

Guiguang Ding (Member, IEEE) is currently an Associate Professor with the School of Software, Tsinghua University, China. His current research interests include the areas of multimedia information retrieval, computer vision, and machine learning.

Qiang Ni (Senior Member, IEEE) is currently a Professor with the School of Computing and Communications, and with the Data Science Institute, Lancaster University, Lancaster, U.K. His main research interests include the area of future generation communications and networking, including 5G and 6G, SDN, cloud networks, energy harvesting, the IoTs, cyber physical systems, AI, machine learning, big data analytics, urban surveillance systems, and smart city.

Jungong Han is currently a Chair Professor with the Department of Computer Science, Aberystwyth University, U.K. He also holds an honorary professorship with the University of Warwick, U.K. His research interests include computer vision, artificial intelligence, and machine learning.