GraphBit: Bitwise Interaction Mining via Deep Reinforcement Learning

Yueqi Duan[1,2,3], Ziwei Wang[1], Jiwen Lu[1,2,3,*], Xudong Lin[1], Jie Zhou[1,2,3]
[1]Department of Automation, Tsinghua University, China
[2]State Key Lab of Intelligent Technologies and Systems, China
[3]Beijing National Research Center for Information Science and Technology, China
duanyq14@mails.tsinghua.edu.cn; zw-wa14@mails.tsinghua.edu.cn; lujiwen@tsinghua.edu.cn;
linxd14@mails.tsinghua.edu.cn; jzhou@tsinghua.edu.cn

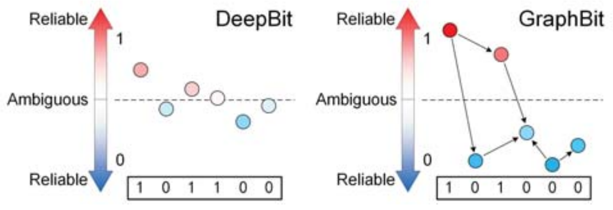
Abstract

In this paper, we propose a GraphBit method to learn deep binary descriptors in a directed acyclic graph unsupervisedly, representing bitwise interactions as edges between the nodes of bits. Conventional binary representation learning methods enforce each element to be binarized into zero or one. However, there are elements lying in the boundary which suffer from doubtful binarization as "ambiguous bits". Ambiguous bits fail to collect effective information for confident binarization, which are unreliable and sensitive to noise. We argue that there are implicit inner relationships between bits in binary descriptors, where the related bits can provide extra instruction as prior knowledge for ambiguity elimination. Specifically, we design a deep reinforcement learning model to learn the structure of the graph for bitwise interaction mining, reducing the uncertainty of binary codes by maximizing the mutual information with inputs and related bits, so that the ambiguous bits receive additional instruction from the graph for confident binarization. Due to the reliability of the proposed binary codes with bitwise interaction, we obtain an average improvement of 9.64%, 8.84% and 3.22% on the CIFAR-10, Brown and HPatches datasets respectively compared with the state-of-the-art unsupervised binary descriptors.

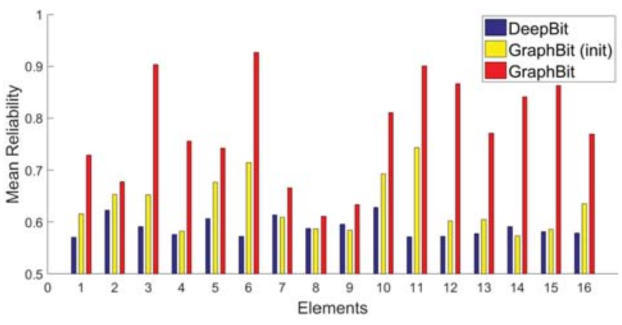
1. Introduction

Extracting effective descriptors is one of the most active issues in computer vision, which is widely applicable in numerous applications, such as face recognition [31, 42, 33], image classification [17, 27], object recognition [15, 30] and many others. Strong discriminative power and low computational cost are two essential properties for an effective descriptor. On one hand, it is crucial for a descriptor to be distinctive for image description and robust to various transformations. On the other hand, highly efficient descriptors present low memory cost and high computational speed, which are suitable for the scenarios of mobile devices with limited computational capabilities and real-time requirements. In recent years, a number of deep binary descriptors have been proposed due to their strong discriminative power and low computational cost [27, 12, 29, 20, 37].

* Corresponding author



(a) Comparison of the reliability



(b) Mean reliability of each bit on the CIFAR-10 dataset

Figure 1. Comparison of the reliability of DeepBit [27] and GraphBit. In Figure 1 (a), the position and color of the dots demonstrate the reliability of binary codes, and the arrows represent the directed bitwise interaction. Figure 1 (b) shows the mean reliability of each bit on CIFAR-10 [23] under the binary length of 16. We model the learned binary codes in binomial distributions and define the reliability of each bit as the confidence level of binarization according to (2). We observe that GraphBit (init) learns more confident binary codes than DeepBit, and GraphBit further improves the reliability with bitwise interactions. (Best viewed in color.)

Binary descriptors substitute real-valued elements with binary codes which are efficient for storage and matching, while deep learning obtains high quality representation by training numerous parameters with large amount of data.

For most existing deep binary descriptor learning approaches, binarization is an essential step to quantize each real-valued element into zero or one, which enhances the efficiency of the descriptors at the cost of quantization loss [12]. However, to the best of our knowledge, these methods directly perform binarization on the real-valued elements to obtain binary codes, which fail to consider their reliability. If a real-valued element lies in the boundary of the binarization for an input image, it would suffer from an unreliable result and an "ambiguous bit" is obtained. For example, the result of a sign function based binarization is doubtful for a real-valued element close to zero. Ambiguous bits fail to receive effective instruction from the corresponding inputs for confident binarization, which present little discriminative power and are sensitive to noise.

We argue that there are implicit relationships between bits for the learned binary codes, and the related bits can provide extra instruction to the ambiguous bits as prior knowledge. For example, it is ambiguous to decide whether a person is tall or short in 5 feet 9 inches. However, the answer becomes more certain if we consider an additional gender bit of female or an age bit of young child. In this paper, we propose a GraphBit method to eliminate the ambiguity through bitwise interaction mining, where we represent binary codes in a directed acyclic graph. The nodes of the graph are the elements in binary descriptors and the directed edges represent bitwise connections. As the learned deep binary descriptors usually fail to present clear physical meanings where the relationships between bits are implicit, we design a deep reinforcement learning model to decide the structure of the graph for bitwise interaction mining.

More specifically, we perform a sigmoid function at the end of CNN for normalization, modelling the normalized elements as the possibilities of being quantized into one in a binomial distribution. The probabilistic model describes the reliability of binarization, and we formulate the relationships between bits as conditional probabilities. We simultaneously train the parameters of CNN and the structure of the graph in an unsupervised manner, maximizing the mutual information of each bit with the observed inputs and the related bits for ambiguity elimination. For the deep reinforcement learning based bitwise interaction mining, we define the action to *add* or *remove* a directed connection between two nodes, and the state is the current structure of the graph. In Figure 1, DeepBit ignores the reliability during the training procedure, GraphBit (init) only maximizes the mutual information without bitwise interaction mining, and GraphBit mines the relationships between bits through deep reinforcement learning. We observe that both mutual information maximization and bitwise interaction enhance the reliability of the binary codes. Extensive experimental results show that GraphBit outperforms most existing unsupervised binary descriptors due to its strong reliability.

2. Related Work

Binary Descriptors: Binary descriptors have attracted much attention in computer vision due to their efficiency for storage and matching, where early works can be traced back to binary robust independent elementary feature (BRIEF) [8], binary robust invariant scalable keypoint (BRISK) [25], oriented FAST and rotated BRIEF (ORB) [35] and fast retina keypoint (FREAK) [1]. BRIEF computed binary descriptors through the intensity different tests between pixels. BRISK obtained scale and rotation invariance by leveraging a circular sampling pattern. ORB improved BRIEF by applying scale pyramids and orientation operators. FREAK utilized retinal sampling grid for acceleration.

As hand-crafted binary descriptors are heuristics and usually require strong prior knowledge, a number of learning based approaches have been proposed and achieved outstanding performance [41, 46, 44, 14]. For example, Strecha *et al.* [41] proposed LDA-Hash by applying linear discriminant analysis (LDA) before binarization. Trzcinski *et al.* [46] presented D-BRIEF by learning discriminative projections through similarity relationships. They also learned hash functions with boosting to obtain BinBoost [44]. Fan *et al.* [14] proposed a receptive fields descriptor (RFD) by thresholding responses of two different receptive fields, rectangular pooling area and Gaussian pooling area.

More recently, several deep binary descriptor learning approaches have been proposed [13, 27, 12, 29, 20, 37], which achieve the state-of-the-art performance. For example, Lin *et al.* [27] proposed DeepBit by training a deep neural network with essential properties in an unsupervised manner. Duan *et al.* [12] presented a deep binary descriptor with multi-quantization (DBD-MQ) to minimize the quantization loss of binarization with a K-autoencoders network. Shen *et al.* [37] proposed textual-visual deep binaries (TVDB) to simultaneously encode the detailed semantics of images and sentences. However, these deep binary descriptors fail to exploit bitwise interactions, which suffer from unreliable ambiguous bits.

Deep Reinforcement Learning: Reinforcement learning aims to learn the policy of sequential actions for decision-making problems [43, 21, 28]. Due to the recent success in deep learning [24], deep reinforcement learning has aroused more and more attention by combining reinforcement learning with deep neural networks [32, 38]. Deep reinforcement learning algorithms have obtained very promising results [39, 32, 38], which can be mainly divided

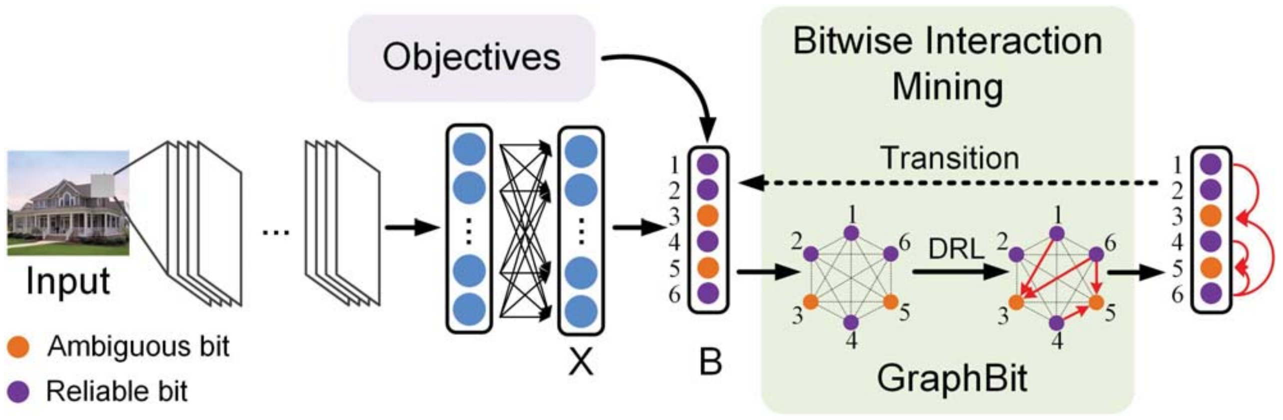


Figure 2. The flowchart of the proposed GraphBit. For each input image, we first learn a normalized feature with a pre-trained CNN by replacing the softmax layer with a fully connection layer followed by a sigmoid function. The normalized feature ranges from 0 to 1, representing the possibility of being binarized into one for clear description of reliability. Then, we simultaneously train the parameters of CNN and the structure of the graph for ambiguity elimination, mining bitwise relationships through deep reinforcement learning. We optimize the parameters with back-propagation in an unsupervised manner.

into two categories: deep Q-networks and policy gradient. For example, Mnih *et al.* [32] played ATARI games through deep Q-networks. Silver *et al.* [38] designed a program named *AlphaGo* with Monte-Carlo tree search based deep reinforcement learning, which defeated the world champion in the game of Go.

More recently, deep reinforcement learning approaches have been employed in many computer vision applications [7, 50, 9, 22, 26, 34]. For example, Kong *et al.* [22] proposed a collaborative multi-agent deep reinforcement learning algorithm to exploit the contextual information for joint object search. Liang *et al.* [26] presented a deep variation-structured reinforcement learning (VRL) approach to grasp global visual cues. However, to our best knowledge, no relevant deep reinforcement learning works have been focused on the fundamental problem of binary representation extraction, which is of significant importance in a variety of visual analysis tasks.

3. Proposed Approach

In this section, we first introduce the reliability of binary codes, and then present the objective function of GraphBit. Lastly, we detail the deep reinforcement learning model for bitwise interaction mining.

3.1. Reliability of Binary Codes

The reliability of binary codes is an essential property in binary descriptor learning, which determines the credibility of each bit. As aforementioned, ambiguous bits may lead to weak discriminative power and unrobustness, as they fail to collect effective information from the inputs. However, most existing approaches directly perform binarization on real-valued elements without considering the reliability, which suffer from ambiguous bits.

In this paper, we aim to learn reliable binary codes and we first define the confidence of binarization. As shown in Figure 2, we initialize the CNN with the pre-trained 16 layers VGG network [40], substituting the softmax layer with a fully connection layer followed by an activation function of sigmoid. Unlike most existing binary descriptors which directly utilize a sign function for binarization, we first perform a sigmoid function at the end of CNN to normalize each element in the range of 0 to 1 for better reliability estimation. The normalized elements are regarded as the possibilities of being quantized to one in a binomial distribution, and the strategy of binarization is shown as follows:

$$b_{kn} = \begin{cases} 1 & 0.5 \leqslant f(t_{kn}) \leqslant 1 \\ 0 & 0 \leqslant f(t_{kn}) < 0.5, \end{cases} \tag{1}$$

where t_{kn} represents the kth real-valued element of the nth input image without considering bitwise interaction, $f(t_{kn})$ is the normalized value and b_{kn} is the corresponding bit.

As it is unlikely to promise each element to be zero or one, we choose the binarization results with higher possibilities. Obviously, the binarization results are more convincing for the $f(t_{kn})$ close to 0 or 1, while it is ambiguous for the values near 0.5. Let $b_k \sim p(b_k)$ be the variable of the kth bit, following a binomial distribution of $p(b_k = 1|\mathbf{x}_n) = f(t_{kn})$. With the binarization strategy of (1), we obtain the reliability of binary codes as follows:

$$p(b_k = b_{kn}|\mathbf{x}_n) = |f(t_{kn}) - 0.5| + 0.5, \tag{2}$$

which ranges from 0.5 to 1. Our goal is to maximize the confidence level of binarization for reliable binary codes through bitwise interaction mining. In the following, we denote $p(b_k = b_{kn}|\mathbf{x}_n)$ as $p(b_{kn}|\mathbf{x}_n)$ for short.

3.2. Objective Function

Let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N]$ be the N input samples of the image set. The objective of GraphBit is to simultaneously learn deep binary descriptors $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_N]$ and the structure of graph $\Phi = \{(\mathbf{b}_t^T, \mathbf{b}_s^T)\}$ which represents the directed connections from the tth bit to the sth bit. In order to describe bitwise relationships, we denote the interaction between random variables X and Y by mutual information $I(X;Y)$, which describes the decrease of entropy of X when Y is tractable:

$$I(X;Y) = H(X) - H(X|Y), \tag{3}$$

where the entropy

$$\begin{aligned} H(X) &= -\mathbb{E}_{x\sim p(X)}[\log p(x)] && (4)\\ H(X|Y) &= -\mathbb{E}_{y\sim p(Y)}[\mathbb{E}_{x\sim p(X|Y)}[\log p(x|y)]]. && (5)\end{aligned}$$

Here, $H(X)$ and $H(X|Y)$ reveal the uncertainty of the variables. Large $I(X;Y)$ represents much reduction of uncertainty in X when Y is observed, and in the extreme case when X and Y are independent, $I(X;Y)$ is equal to zero.

Inspired by the above motivations, we formulate the following objective function to learn GraphBit:

$$\begin{aligned} \min J &= J_1 + \alpha J_2 + \beta J_3 \\ &= \sum_{k=1}^{K} || \sum_{n=1}^{N} (b_{kn} - 0.5)||^2 \\ &- \alpha \sum_{n=1}^{N} (\sum_{b_{rn}\notin \mathbf{b}_s^T} I(b_{rn}; \mathbf{x}_n) + \sum_{\Phi} I(b_{sn}; \mathbf{x}_n, b_{tn})) \\ &+ \beta \sum_{n=1}^{N} \sum_{\Phi} ||p(b_{sn}|\mathbf{x}_n) - p(b_{sn}|\mathbf{x}_n, b_{tn})||^2, \end{aligned} \tag{6}$$

where K is the length of the binary descriptors and $(\mathbf{b}_t^T, \mathbf{b}_s^T) \in \Phi$. α and β are two parameters to balance the weights of different terms. b_{sn} represents the binarization result under the instruction of b_{tn}, and we use an additional bitwise weight of w_{ts} to represent b_{sn} based on the normalization result of a weighted sum:

$$p(b_{sn}|\mathbf{x}_n) = |f(t_{sn}) - 0.5| + 0.5, \tag{7}$$

$$p(b_{sn}|\mathbf{x}_n, b_{tn}) = |f(t_{sn} + \sum_t w_{ts} t_{tn}) - 0.5| + 0.5. \tag{8}$$

We detail the physical meanings of the three terms in the objective function as follows:

1) J_1 is to make each bit in the learned GraphBit evenly distributed. If an element in the learned binary descriptors stay the same for all the samples, it would present no discriminative power. Instead, we encourage each bit equal to zeros for half of the samples and ones for the others to convey more information.

2) J_2 encourages the independent bits b_{rn} to obtain most information from the input samples by maximizing the mutual information, which reduces the uncertainty in b_{rn} with $\mathbf{x}_n$ observed according to (3). For the interacted bits b_{sn}, they simultaneously receive instruction from the corresponding inputs $\mathbf{x}_n$ and the related bits b_{tn}. This term ensures the reliability of the learned binary codes, where each bit chooses to be instructed by either only inputs or with additional related bits.

3) J_3 aims to prevent the interacted bits to become trivial. Under the guidance of J_2, those ambiguous bits that fail to collect effective information from the inputs may tend to receive extra directions from other more reliable bits. However, they may become redundant as a repeat of the related bits if suffering from too strong instructions. The goal of J_3 is to guarantee the independence of the interacted bits.

We apply variational information maximization to simplify J_2 in (6) with the upper bounding, which is then approximated with Monte Carlo simulation [5, 10].

The first part of J_2 can be rewritten as follows:

$$\begin{aligned} I(b_{rn}; \mathbf{x}_n) &= H(b_{rn}) - H(b_{rn}|\mathbf{x}_n) \\ &= H(b_{rn}) + \mathbb{E}_{\mathbf{x}_n\sim\mathbf{X}}[\mathbb{E}_{b'_{rn}\sim p(b_{rn}|\mathbf{x}_n)} \\ &\quad [\log p(b'_{rn}|\mathbf{x}_n)]] \\ &= H(b_{rn}) + \mathbb{E}_{\mathbf{x}_n\sim\mathbf{X}}[D_{KL}(p(\cdot|\mathbf{x}_n)||q(\cdot|\mathbf{x}_n)) \\ &+ \mathbb{E}_{b'_{rn}\sim p(b_{rn}|\mathbf{x}_n)}[\log q(b'_{rn}|\mathbf{x}_n)]] \\ &\geq H(b_{rn}) + \mathbb{E}_{\mathbf{x}_n\sim\mathbf{X}}[\mathbb{E}_{b'_{rn}\sim p(b_{rn}|\mathbf{x}_n)} \\ &\quad [\log q(b'_{rn}|\mathbf{x}_n)]], \end{aligned} \tag{9}$$

where $q(\cdot)$ is the auxiliary distribution for the posterior distribution $p(\cdot)$. In this paper, we parametrize $q(b'_{rn}|\mathbf{x}_n)$ based on the reliability of binary codes defined in (2). When the distribution of $q(\cdot)$ approaches the distribution of $p(\cdot)$, the bound is tight because $D_{KL}(q(\cdot)||p(\cdot)) \to 0$. In our experiments, we specially set $q(b'_{rn}|\mathbf{x}_n) = |t_{rn} - 0.5|$ to obtain large magnitude by the log function for strict constraint of reliability. For simplicity, $H(b_{rn})$ is regarded to be constants because each bit should have a prior equal possibility to be zero or one.

Similarly, the second part in J_2 can be rewritten as follows:

$$\begin{aligned} I(b_{sn}; \mathbf{x}_n, b_{tn}) &\geq H(b_{sn}) + \mathbb{E}_{\mathbf{x}_n\sim\mathbf{X}}[\mathbb{E}_{b'_{sn}\sim p(b_{sn}|\mathbf{x}_n, b_{tn})} \\ &\quad [\log q(b'_{sn}|\mathbf{x}_n, b_{tn})]]. \end{aligned} \tag{10}$$

We apply the stochastic gradient decent with backpropagation to train the CNN model in an unsupervised manner.

3.3. Deep Reinforcement Learning for Bitwise Interaction Mining

Mining bitwise interaction for ambiguity elimination can be viewed as a Markov Decision Process (MDP), which is

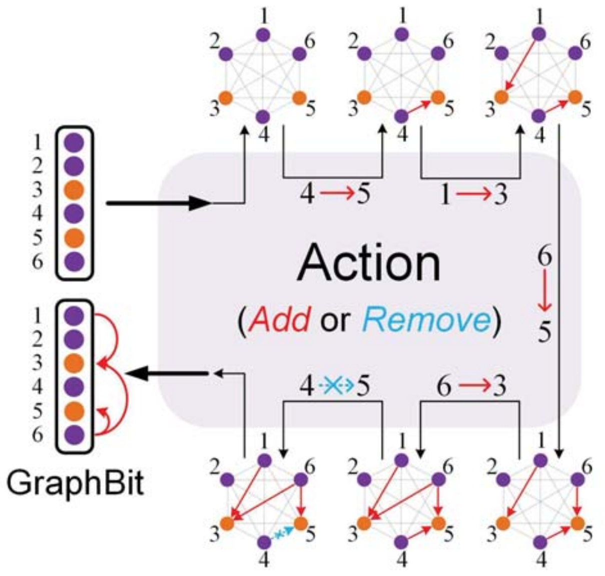


Figure 3. An explanation of deep reinforcement learning based bitwise interaction mining. In this example, we sequentially *add* directed connections of the 4th-5th bits, the 1st-3rd bits, the 6th-5th bits and the 6th-3rd bits, and then *remove* the connection of the 4th-5th bits. We repeat the process of bitwise interaction mining until finalizing the structure of graph.

formally defined as $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{T}(\mathcal{S}, \mathcal{A}), \mathcal{R}(\mathcal{S}, \mathcal{A})\}$.

At each step, the agent takes the action to *add* or *remove* a directional connection between bits based on the current structure of the graph, which iteratively explores the bitwise interaction to maximize the reward. The policy network recurrently *adds* and *removes* the edges until convergence or achieving the maximum step. At the end of the sequence, we retrain the parameters of CNN with the learned structure of graph under the guidance of the objective function.

States: The state space $\mathcal{S}$ represents the current structure of the graph, which can be defined as a binary matrix $W_s \in \{0,1\}^{K\times K}$. For the element $w_{ij}^s \in W_s$, it equals to one if there is a directed edge between the ith bit and the jth bit, and equals to zero otherwise. GraphBit under a zero matrix for W_s is equivalent to existing deep binary descriptors without bitwise interactions as a special case.

Action: Given the current graph W_s, the agent aims to select one action from all possible connections and disconnections. $\mathcal{A}$ is the set of actions divided into three categories: $\mathcal{A}_c \bigcup \mathcal{A}_r \bigcup \{stop\}$. The action to *add* a bitwise edge is denoted as $\mathcal{A}_c$, while $\mathcal{A}_r$ represents to *remove* bitwise connections. The action of *stop* is executed for convergence or the maximum time step. Figure 3 shows an example of transition of stages with the actions.

Transition Function: $\mathcal{T}(\mathcal{S}, \mathcal{A}) \rightarrow \mathcal{S}'$ is the transition function which shows the movement of the stage. $\mathcal{T}$ is constructed under the observation of states and performance of actions, which can be represented as a transition matrix $W_t \in \mathbb{R}^{K\times K}$. The element $w_{ij}^t \in [0,1]$ represents the probability of the connection from the ith bit to the jth bit. We select the actions based on the following rules:

1) *Add*: We connect the ith bit to the jth bit if $w_{ij}^t = \max\{W_t\}$ together with $w_{ij}^t \geqslant k_1$.

2) *Remove*: We disconnect the original edge between the ith bit and the jth bit if $w_{ij}^t \leqslant k_2$.

3) *Stop*: We terminate the current epoch of bitwise interaction mining for convergence or achieving the maximum time step.

In this paper, we gradually increase the parameters k_1 and k_2 during the iterations, where the maximum values for k_1 and k_2 are 0.8 and 0.2, respectively.

Reward Function: We define the reward function $\mathcal{R}(\mathcal{S}, \mathcal{A})$ in round t as following:

$$r(s_t, a_t) = J(s_t) - J(s_{t+1}) \tag{11}$$

where $r(s_t, a_t) \in \mathcal{R}(\mathcal{S}, \mathcal{A})$ is the returning reward for the action a_t in the state s_t, and $J(s_t)$ is the loss of the objective function in the state s_t. We consider the bitwise interaction in high quality if it leads to lower loss in the unsupervised learning, which simultaneously enhances the discriminative power and the reliability of the learned GraphBit.

We employ a CNN network with the deconvolution layer in the end as our policy network. More specifically, the policy network has three convolutional layers, followed by two fully connected layers and two deconvolutional layers. We apply ReLU as the activation function in the middle layers and sigmoid for the last layer. We also perform dropout to prevent from overfitting. The input of the policy network is the state matrix W_s, and the output of the network predicts the probability of actions with W_t.

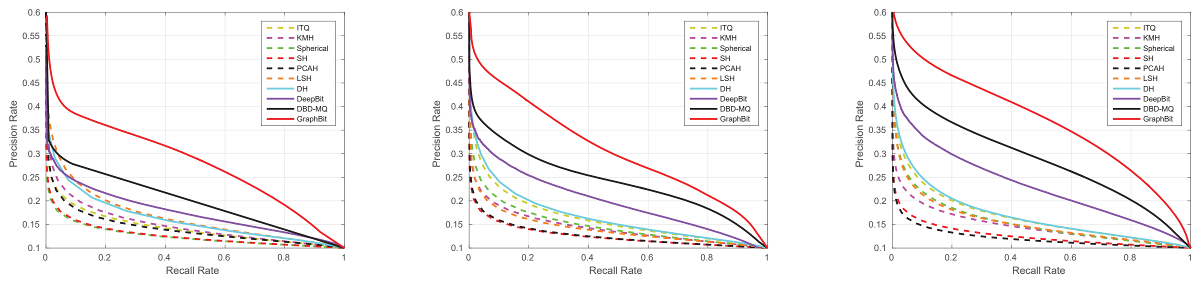
We utilize the REINFORCE algorithm [49] to update parameters in the policy network in response to the rewards from the environment. The objective is to maximize the expected reward over the entire GraphBit learning process:

$$\max_\theta Z(\theta) = \mathbb{E}_\pi[\sum_{t=1}^{T} \gamma r_t(s_t, a_t)], \tag{12}$$

where θ represents the parameters of the policy network, π is the selected policy and γ is the discount factor. We set γ as 0.9 throughout the experiments.

Following the REINFORCE algorithm, we obtain the gradient of (12) as follows:

$$\begin{aligned} \nabla_\theta Z &= \nabla_\theta[\sum_{q_t, a_t} \pi(a_t|s_t) r_t] \\ &= \sum_{q_t, a_t} r_t \pi(a_t|s_t) \nabla_\theta \log \pi(a_t|s_t) \\ &= \mathbb{E}_\pi[r_t \nabla_\theta \log \pi(a_t|s_t)]. \end{aligned} \tag{13}$$



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

Figure 4. Comparison of Precision/Recall curves on the CIFAR-10 dataset under varying binary lengths (a) 16 bits, (b) 32 bits and (c) 64 bits with the state-of-the-art unsupervised binary descriptors.

We apply the sampled sequences $(s_1, a_1; ...; s_{T_k}, a_{T_k})$ for the approximation of (13), and obtain the gradient of policies by sampling for M times:

$$\nabla_\theta Z \approx \frac{1}{M}\sum_{k=1}^{M} R_k \sum_{t=0}^{T_k} \nabla_\theta \log \pi(a_t|s_t), \tag{14}$$

where T_k is the number of steps in the sampled sequences and R_k represents the reward achieved in the kth sequence.

4. Experiments

In this section, we evaluated our GraphBit on three datasets for patch retrieval, image matching and patch verification tasks to demonstrate the effectiveness of the proposed GraphBit, where the CIFAR-10 [23], Brown [6] and HPatches [4] datasets were employed, respectively. Besides GraphBit, we also tested the performance of our approach without bitwise interaction mining by training with only J_1 and the first part of J_2 as GraphBit (init).Following [27], we resized each input image into 256 $\times$ 256 at first, and then cropped it to 224 $\times$ 224 for background removal. We empirically set the parameters α and β as 0.2 and 0.4 to balance the weights, respectively.

4.1. Results on CIFAR-10

The CIFAR-10 dataset [23] consists of 10 categories and each class contains 6,000 images. We applied 50,000 images as the training set and the other 10,000 as the test set. We followed the standard evaluation protocol [23] to evaluate the mean average precision (mAP) under different binary length of 16 bits, 32 bits and 64 bits.

Comparisons with the State-of-the-Art Unsupervised Binary Descriptors: We compared our GraphBit with several unsupervised binary descriptors on the CIFAR-10 dataset. Among the listed approaches, deep hashing (DH), DeepBit and DBD-MQ are the state-of-the-art unsupervised deep binary descriptors. Table 1 illustrates the comparison with mean average precision (mAP) and Figure 4 shows

Table 1. Comparison of mean average precision (mAP) (%) of top 1,000 returned images with the state-of-the-art unsupervised binary descriptors on CIFAR-10.

Method	16 bits	32 bits	64 bits
KMH [18]	13.59	13.93	14.46
SphH [19]	13.98	14.58	15.38
SpeH [48]	12.55	12.42	12.56
SH [36]	12.95	14.09	13.89
PCAH [47]	12.91	12.60	12.10
LSH [2]	12.55	13.76	15.07
PCA-ITQ [16]	15.67	16.20	16.64
DH [13]	16.17	16.62	16.96
DeepBit [27]	19.43	24.86	27.73
DBD-MQ [12]	21.53	26.50	31.85
GraphBit (init)	27.95	32.77	36.16
GraphBit	**32.15**	**36.74**	**39.90**

Table 2. Comparison of mean average precision (mAP) (%) of top 1,000 returned images under different learning strategies of GraphBit on CIFAR-10.

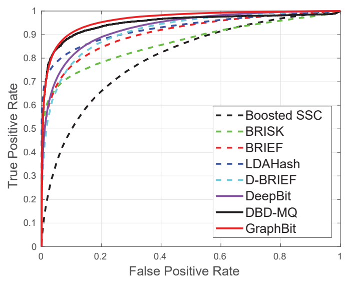
Method	Independent Bits	Interacted Bits
GraphBit (J_1)	21.45	26.20
GraphBit ($J_1 + J_2$)	27.95	30.43
GraphBit ($J_1 + J_2 + J_3$)	-	32.15

the Precision/Recall curves. DBD-MQ achieves the state-of-the-art performance, while GraphBit improves the mAP by 10.62%(=32.15%-21.53%), 10.24%(=36.74%-26.50%), 8.05%(=39.90%-31.85%) in the settings of 16-bit, 32-bit and 64-bit, respectively.

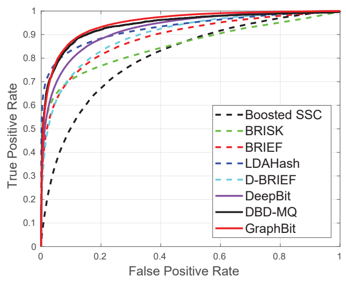
We observe that GraphBit obtains a 9.64% improvement on average on CIFAR-10 compared with the unsupervised binary descriptors, while GraphBit (init) also improves the mAP by 5.67% even without bitwise interaction mining. With the comparison of the reliabilities shown in Figure 1 at the beginning of the paper, we see that more reliable binary codes achieve better results. For GraphBit (init), the advantages mainly come from the second term J_2 of the objective function, which reduces the uncertainty of bina-

Table 3. Comparison of 95% error rates (ERR) on the Brown dataset with the state-of-the-art binary descriptors. Unsupervised binary descriptor include BRISK, BRIEF, DeepBit and DBD-MQ, and supervised binary descriptors include LDAHash, D-BRIEF, BinBoost and RFD. The real-valued feature SIFT is provided for reference.

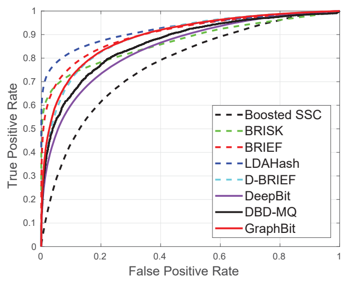
Train Test	Yosemite Notre Dame	Yosemite Liberty	Notre Dame Yosemite	Notre Dame Liberty	Liberty Notre Dame	Liberty Yosemite	Average ERR
SIFT [30] (128 bytes)	28.09	36.27	29.15	36.27	28.09	29.15	31.17
BRISK [25] (64 bytes)	74.88	79.36	73.21	79.36	74.88	73.21	75.81
BRIEF [8] (32 bytes)	54.57	59.15	54.96	59.15	54.57	54.96	56.23
DeepBit [27] (32 bytes)	29.60	34.41	63.68	32.06	26.66	57.61	40.67
DBD-MQ [12] (32 bytes)	27.20	33.11	57.24	31.10	25.78	57.15	38.59
LDAHash [41] (16 bytes)	51.58	49.66	52.95	49.66	51.58	52.95	51.40
D-BRIEF [46] (4 bytes)	43.96	53.39	46.22	51.30	43.10	47.29	47.54
BinBoost [44] (8 bytes)	14.54	21.67	18.96	20.49	16.90	22.88	19.24
RFD [14] (50-70 bytes)	11.68	19.40	14.50	19.35	13.23	16.99	15.86
GraphBit (init) (32 bytes)	21.18	28.33	51.62	25.00	18.32	52.58	32.83
GraphBit (32 bytes)	**17.78**	**24.72**	**49.94**	**21.18**	**15.25**	**49.64**	**29.75**



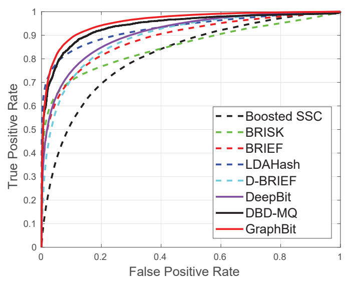
(a) Yosemite-Notre Dame



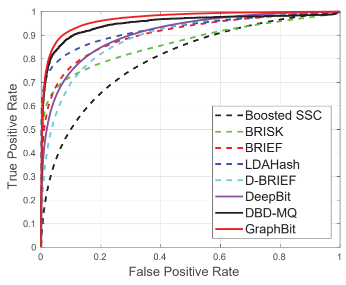
(b) Yosemite-Liberty



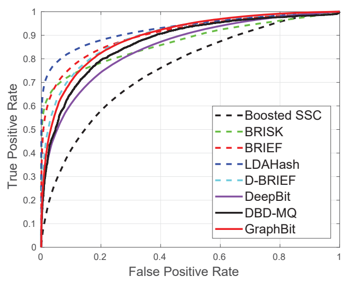
(c) Notre Dame-Yosemite



(d) Notre Dame-Liberty



(e) Liberty-Notre Dame



(f) Liberty-Yosemite

Figure 5. Comparison of ROC curves on the Brown dataset with several binary descriptors.

ry codes by mutual information maximization. As reliable bits receive more effective information from the inputs for confident quantization, they present stronger discriminative power and robustness. For GraphBit, each bit is also allowed to be instructed by other related bits to further enhance the reliability and obtains better performance. The numbers of bitwise connections are 22, 198 and 880 for 16 bits, 32 bits and 64 bits, respectively.

Influences of Different Learning Strategies: In order to investigate the contributions of bitwise interactions and different terms in GraphBit, we evaluated the performance on CIFAR-10 under different training strategies with binary length of 16. We preserved J_1 in all the settings to ensure the variation of each bit. As aforementioned, GraphBit (init) is the situation of training independent bits with J_1 and J_2. Table 2 shows that bitwise interaction mining improves the experimental results with different training terms by exploring implicit relationships between bits. Experimental results also demonstrate the effectiveness of J_2 and J_3. While J_2 aims to enhance the reliability of binary codes through mutual information maximizing, J_3 prevents from trivial bits with over-strong bitwise instructions. Both J_2 and J_3 improve the experimental results, and the best performance can be achieved when all the terms are used together with bitwise interactions.

Computational Time: Our hardware equips with a 2.8-GHz CPU and a 32G RAM, and we utilized a GTX 1080 Ti GPU for acceleration. We evaluated the total time of extracting one probe feature and retrieving from 50,000 gallery features, where a 32-bit GraphBit took 0.016s to obtain retrieval result. HOG [11] and SIFT [30] required 0.030s and 0.054s, respectively. GraphBit substitutes the Hamming distance for the Euclidean distance and presents higher matching speed. As for the storage cost, a 32-bit GraphBit only required 4 bytes for each image, while 9 bytes were needed for HOG and 128 bytes for SIFT.

4.2. Results on Brown

We evaluated our GraphBit on the Brown dataset [6] for image patch matching. There are three subsets on the Brown dataset, which include Liberty, Notre Dame and Yosemite. Each subset contains 400,000 to 600,000 images for training and 100,000 pairs for test. Among test pairs, half of them are matched positive pairs and the others are mismatched negative pair. We followed the settings in [45] by evaluating the performance of GraphBit on all six training and test combinations, including Yosemite-Notre Dame, Yosemite-Liberty, Notre Dame-Yosemite, Notre Dame-Liberty, Liberty-Notre Dame and Liberty-Yosemite. We set the length of binary descriptor as 256 bits.

Table 3 illustrates the 95% error rates of GraphBit and the state-of-the-art binary descriptors on the Brown dataset and Figure 5 shows ROC curves. The compared approaches consist of unsupervised binary descriptors BRISK [25], BRIEF [8], DeepBit [27] and DBD-MQ [12], and supervised binary descriptors LDAHash [41], D-BRIEF [46], BinBoost [44] and RFD [14]. We also provide the performance of the real-valued SIFT [30] as an important reference. DBD-MQ achieves outstanding performance compared with other unsupervised binary descriptors by learning data-dependent binarization. However, DBD-MQ fails to consider the reliability of the learned binary codes, which suffers from ambiguous bits lying in the boundary of multi-quantization. GraphBit learns reliable binary codes through bitwise interaction mining, achieving an average improvement of 8.84% in the Brown dataset. Moreover, GraphBit obtains a lower average 95% error rate compared to the widely-used real-valued SIFT with a much smaller storage cost. As an unsupervised method, GraphBit obtains better average performance than the supervised LDAHash and D-BRIEF, which shows its applicability to the scenarios where label information is difficult to collect.

4.3. Results on HPatches

The HPatches dataset [4] is a recent benchmark to evaluate local descriptors, which provides three baseline visual analysis tasks including patch verification, image matching and patch retrieval. HPatches consists of 116 sequences in total, splitting into 57 with photometric changes and 59 with significant geometric deformations.

Table 4. Comparison of mean average precision (mAP) (%) with unsupervised binary codes and other baseline methods under various tasks on HPatches.

Method	Verification	Matching	Retrieval
BinBoost [44] (32 bytes)	66.67	14.77	22.45
SIFT [30] (128 bytes)	65.12	25.47	31.98
RSIFT [3] (128 bytes)	58.53	27.22	33.56
BRIEF [8] (32 bytes)	58.07	10.50	16.03
ORB [35] (32 bytes)	60.15	**15.32**	18.85
DeepBit [27] (32 bytes)	61.27	13.05	20.61
GraphBit (init) (32 bytes)	62.32	13.42	21.45
GraphBit (32 bytes)	**65.19**	14.22	**25.19**

We followed the standard evaluation protocol [4] to report the performance of mean average precision (mAP) on the three visual analysis tasks. We compared GraphBit with unsupervised binary descriptors including BRIEF [8], ORB [35] and DeepBit [27], and provided the results of BinBoost [44], SIFT [30] and RSIFT [3] for reference. Table 4 shows that GraphBit outperforms DeepBit by 3.92%, 1.17% and 4.58% on each tested visual analysis task respectively, which demonstrates the importance of the reliability of the binary codes. Moreover, GraphBit obtains comparable results to the supervised binary codes BinBoost without using any label information, which shows the effectiveness of the proposed binary descriptor.

5. Conclusion

In this paper, we have proposed an unsupervised deep binary descriptor learning method called GraphBit for image patch representation. Our GraphBit models binary codes in binomial distributions and maximizes the mutual information to reduce the uncertainty with the observed inputs and the related bits. Moreover, GraphBit mines the bitwise interaction through deep reinforcement learning to further enhance the reliability of the ambiguous bits. Extensive experimental results on the CIFAR-10, Brown and HPatches datasets have been presented to demonstrate the effectiveness of the proposed method.

Acknowledgement

This work was supported in part in part by the National Key Research and Development Program of China under Grant 2016YFB1001001, by the National Natural Science Foundation of China under Grant 61672306, Grant U1713214, Grant 61572271, and Grant 61527808, in part by the National 1000 Young Talents Plan Program, in part by the National Basic Research Program of China under Grant 2014CB349304, in part by the Shenzhen Fundamental Research Fund (Subject Arrangement) under Grant JCYJ20170412170602564.

References

[1] A. Alahi, R. Ortiz, and P. Vandergheynst. FREAK: Fast retina keypoint. In *CVPR*, pages 510–517, 2012. 2

[2] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *FOCS*, pages 459–468, 2006. 6

[3] R. Arandjelovic and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, pages 2911–2918, 2012. 8

[4] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk. H-Patches: A benchmark and evaluation of handcrafted and learned local descriptors. In *CVPR*, pages 5173–5182, 2017. 6, 8

[5] D. Barber and F. V. Agakov. The IM algorithm: A variational approach to information maximization. In *NIPS*, pages 201–208, 2003. 4

[6] M. Brown, G. Hua, and S. Winder. Discriminative learning of local image descriptors. *TPAMI*, 33(1):43–57, 2011. 6, 8

[7] J. C. Caicedo and S. Lazebnik. Active object localization with deep reinforcement learning. In *ICCV*, pages 2488–2496, 2015. 3

[8] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. BRIEF: Binary robust independent elementary features. In *ECCV*, pages 778–792, 2010. 2, 7, 8

[9] Q. Cao, L. Lin, Y. Shi, X. Liang, and G. Li. Attention-aware face hallucination via deep reinforcement learning. In *CVPR*, pages 690–698, 2017. 3

[10] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. In *NIPS*, pages 2172–2180, 2016. 4

[11] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages 886–893, 2005. 8

[12] Y. Duan, J. Lu, Z. Wang, J. Feng, and J. Zhou. Learning deep binary descriptor with multi-quantization. In *CVPR*, pages 1183–1192, 2017. 1, 2, 6, 7, 8

[13] V. Erin Liong, J. Lu, G. Wang, P. Moulin, and J. Zhou. Deep hashing for compact binary codes learning. In *CVPR*, pages 2475–2483, 2015. 2, 6

[14] B. Fan, Q. Kong, T. Trzcinski, Z. Wang, C. Pan, and P. Fua. Receptive fields selection for binary feature description. *TIP*, 23(6):2583–2595, 2014. 2, 7, 8

[15] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, pages 264–271, 2003. 1

[16] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *TPAMI*, 35(12):2916–2929, 2013. 6

[17] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *ICCV*, pages 1458–1465, 2005. 1

[18] K. He, F. Wen, and J. Sun. K-means hashing: An affinity-preserving quantization method for learning binary compact codes. In *CVPR*, pages 2938–2945, 2013. 6

[19] J.-P. Heo, Y. Lee, J. He, S.-F. Chang, and S.-E. Yoon. Spherical hashing. In *CVPR*, pages 2957–2964, 2012. 6

[20] H. Jain, J. Zepeda, P. Perez, and R. Gribonval. SUBIC: A supervised, structured binary code for image search. In *ICCV*, pages 833–842, 2017. 1, 2

[21] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *JAIR*, 4:237–285, 1996. 2

[22] X. Kong, B. Xin, Y. Wang, and G. Hua. Collaborative deep reinforcement learning for joint object search. In *CVPR*, pages 1695–1704, 2017. 3

[23] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master Thesis*, 2009. 1, 6

[24] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015. 2

[25] S. Leutenegger, M. Chli, and R. Siegwart. BRISK: Binary robust invariant scalable keypoints. In *ICCV*, pages 2548–2555, 2011. 2, 7, 8

[26] X. Liang, L. Lee, and E. P. Xing. Deep variation-structured reinforcement learning for visual relationship and attribute detection. In *CVPR*, pages 848–857, 2017. 3

[27] K. Lin, J. Lu, C.-S. Chen, and J. Zhou. Learning compact binary descriptors with unsupervised deep neural networks. In *CVPR*, pages 1183–1192, 2016. 1, 2, 6, 7, 8

[28] M. L. Littman. Reinforcement learning improves behaviour from evaluative feedback. *Nature*, 521(7553):445, 2015. 2

[29] H. Liu, R. Wang, S. Shan, and X. Chen. Learning multifunctional binary codes for both category and attribute oriented retrieval tasks. In *CVPR*, pages 3901–3910, 2017. 1, 2

[30] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 1, 7, 8

[31] J. Lu, V. E. Liong, X. Zhou, and J. Zhou. Learning compact binary face descriptor for face recognition. *TPAMI*, 37(10):2041–2056, 2015. 1

[32] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. 2, 3

[33] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *BMVC*, pages 1–12, 2015. 1

[34] Y. Rao, J. Lu, and J. Zhou. Attention-aware deep reinforcement learning for video face recognition. In *ICCV*, pages 3931–3940, 2017. 3

[35] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An efficient alternative to sift or surf. In *ICCV*, pages 2564–2571, 2011. 2, 8

[36] R. Salakhutdinov and G. Hinton. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978, 2009. 6

[37] Y. Shen, L. Liu, L. Shao, and J. Song. Deep Binaries: Encoding semantic-rich cues for efficient textual-visual cross retrieval. In *ICCV*, pages 4097–4106, 2017. 1, 2

[38] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016. 2, 3

[39] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller. Deterministic policy gradient algorithms. In *ICML*, pages 387–395, 2014. 2

[40] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, pages 1–14. 3

[41] C. Strecha, A. Bronstein, M. Bronstein, and P. Fua. LDA-Hash: Improved matching with smaller descriptors. *TPAMI*, 34(1):66–78, 2012. 2, 7, 8

[42] Y. Sun, X. Wang, and X. Tang. Deep learning face representation from predicting 10,000 classes. In *CVPR*, pages 1891–1898, 2014. 1

[43] R. S. Sutton and A. G. Barto. *Introduction to reinforcement learning*, volume 135. MIT Press, 1998. 2

[44] T. Trzcinski, M. Christoudias, P. Fua, and V. Lepetit. Boosting binary keypoint descriptors. In *CVPR*, pages 2874–2881, 2013. 2, 7, 8

[45] T. Trzcinski, M. Christoudias, and V. Lepetit. Learning image descriptors with boosting. *TPAMI*, 37(3):597–610, 2015. 8

[46] T. Trzcinski and V. Lepetit. Efficient discriminative projections for compact binary descriptors. In *ECCV*, pages 228–242, 2012. 2, 7, 8

[47] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for scalable image retrieval. In *CVPR*, pages 3424–3431, 2010. 6

[48] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *NIPS*, pages 1753–1760, 2009. 6

[49] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, pages 229–256, 1992. 5

[50] S. Yun, J. Choi, Y. Yoo, K. Yun, and J. Young Choi. Action-decision networks for visual tracking with deep reinforcement learning. In *CVPR*, pages 2711–2720, 2017. 3