# Prototype-supervised Adversarial Network for Targeted Attack of Deep Hashing

Xunguang Wang[1,*], Zheng Zhang[1,2,*,†], Baoyuan Wu[3,4], Fumin Shen[5,6], Guangming Lu[1]
[1]Harbin Institute of Technology, Shenzhen, [2]Peng Cheng Laboratory
[3]School of Data Science, The Chinese University of Hong Kong, Shenzhen
[4]Secure Computing Lab of Big Data, Shenzhen Research Institute of Big Data
[5]University of Electronic Science and Technology of China, [6]Koala Uran Tech.
{xunguangwang, darrenzz219, wubaoyuan1987, fumin.shen}@gmail.com, luguangm@hit.edu.cn

## Abstract

*Due to its powerful capability of representation learning and high-efficiency computation, deep hashing has made significant progress in large-scale image retrieval. However, deep hashing networks are vulnerable to adversarial examples, which is a practical secure problem but seldom studied in hashing-based retrieval field. In this paper, we propose a novel prototype-supervised adversarial network (ProS-GAN), which formulates a flexible generative architecture for efficient and effective targeted hashing attack. To the best of our knowledge, this is the first generation-based method to attack deep hashing networks. Generally, our proposed framework consists of three parts, i.e., a PrototypeNet, a generator and a discriminator. Specifically, the designed PrototypeNet embeds the target label into the semantic representation and learns the prototype code as the category-level representative of the target label. Moreover, the semantic representation and the original image are jointly fed into the generator for flexible targeted attack. Particularly, the prototype code is adopted to supervise the generator to construct the targeted adversarial example by minimizing the Hamming distance between the hash code of the adversarial example and the prototype code. Furthermore, the generator is against the discriminator to simultaneously encourage the adversarial examples visually realistic and the semantic representation informative. Extensive experiments verify that the proposed framework can efficiently produce adversarial examples with better targeted attack performance and transferability over state-of-the-art targeted attack methods of deep hashing.*

## 1. Introduction

With the explosive growth of high-dimensional and large-scale multimedia data, approximate nearest neighbor (ANN) search [1] has attracted much attention in information retrieval due to its efficiency and effectiveness. As a solution of ANN, hashing [40] maps high-dimension data to compact binary codes meanwhile preserving the semantic similarities, yielding significant advantages in storage cost and retrieval speed. Benefiting from the strong representation ability of deep learning, deep hashing that employs deep neural networks (DNNs) to automatically extract features has achieved great success in learning to hash [42, 21, 47, 26, 28, 5], and also has been demonstrated its superior performance than the shallow hashing methods.

Notably, recent studies [38, 12, 20, 29, 6] have recognized that DNNs are usually vulnerable to adversarial examples, which are intentionally perturbed by adding imperceptible noises to original images but can fool the networks to make incorrect predictions. Deep hashing methods have achieved encouraging performance on many benchmarks, while, at the same time, they inevitably inherit the fundamental fragility of DNNs on handling adversarial examples [44, 2]. This imperceptible malicious attack poses a serious security threat to the deep hashing-based image retrieval. For example, when querying with an intentionally perturbed dog image, a hashing based retrieval system may return violent images. Accordingly, it is necessary to study the adversarial attacks on deep hashing models in order to recognize their flaws and help solve their security risks.

Currently, many works about adversarial examples have been studied in image classification, but *very few* researches focus on the security of deep hashing based retrieval. Different from the typical classification, hashing aims to learn the semantic similarity between images, and its final outputs are discrete binary codes instead of categories. Thus, these attack methods in image classification cannot be directly used or transferred to the deep hashing tasks. Existing adversarial attack methods of deep hashing only include an non-targeted attack method called HAG [44] and two targeted attack methods called P2P [2] and DHTA [2], respectively. Notwithstanding, they are verified to be ef-

*Equal contribution
†Corresponding author

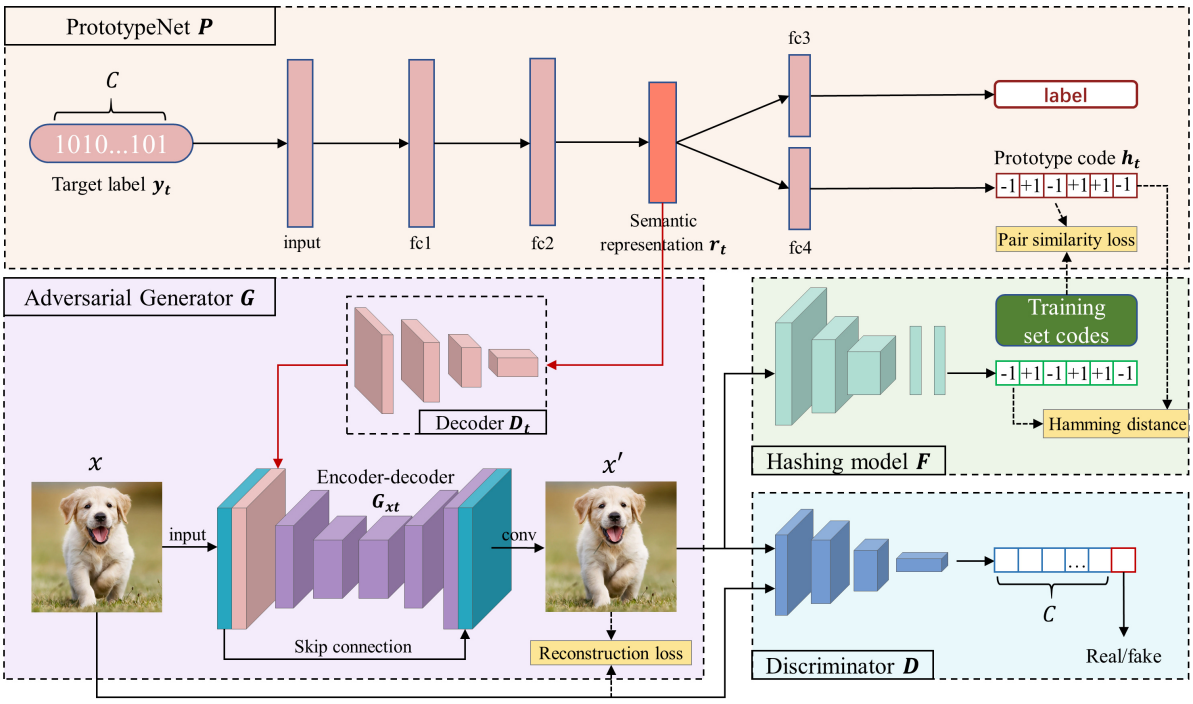



Figure 1. The framework of our Prototype-supervised Adversarial Network (ProS-GAN).

fective in attack, there are still some significant limitations hindering the current adversarial attacks in deep hashing. On one hand, these methods are inefficient because they are optimization-based methods that rely on a very time-consuming iterative gradient. For example, to make the attacked hashing model bias significantly, DHTA requires around 2000 iterations to optimize adversarial perturbations. On the other hand, these methods heuristically select a hash code as representative of the target label to guide the generation of the targeted adversarial example. However, this code can not represent the discriminative category-level semantics of the target label due to lack of preserving similarity with relevant labels and dissimilarity with irrelevant labels. Therefore, how to construct more representative semantic-preserving hash code of the target label becomes crucially important to achieve satisfactory performance in adversarial attack of deep hashing.

To overcome the above deficiencies and inspired by generation-based adversarial attacks [3, 43, 13] in classification, this paper proposes a prototype-supervised adversarial network (ProS-GAN) for efficient and effective targeted attack in deep hashing based retrieval. By feeding an original image and a target label into ProS-GAN, it can generate the targeted adversarial example, which would mislead the attacked hashing network to retrieve the images semantically related to the target label. Specifically, ProS-GAN is composed of three sub-networks: a prototype network (PrototypeNet), an adversarial generator network and a discriminator network, as shown in Figure 1. The designed PrototypeNet encodes the input target label into the semantic representation and learns the prototype code as the representative of the target label. Then, the adversarial generator incorporates the original image and the semantic representation into a self-reconstruction network for generating the adversarial example. Moreover, the prototype code is adopted to supervise the generator to construct the targeted adversarial example by minimizing the Hamming distance between the hash code of the adversarial example and the prototype code. In addition, the discriminator is used to distinguish real/fake images and categorize them into the original and target categories, respectively. The generator and discriminator are trained in an adversarial manner to encourage the generated images visually realistic and the semantic representation informative for further improving the targeted attack performance. In summary, the main contributions are outlined as follows:

- We propose a novel prototype-supervised adversarial network* (ProS-GAN) for flexible targeted hashing attack. *To the best of our knowledge, this is the very first work of formulating a generative architecture for arbitrary-target attack in deep hashing based retrieval.* Importantly, different from the existing attack methods, our work could efficiently and effectively generate preferable adversarial examples with robust transferability under one-forward pass.

*Code: https://github.com/xunguangwang/ProS-GAN

- Instead of heuristically selecting a hash code as representative of the target label, we leverage the invariant side of semantics to generate the flexible prototype code in our ProtypeNet as the expected mainstay of the target label from the optimization view.
- Extensive experiments validate the superior efficiency and transferability of the produced adversarial examples than state-of-the-art targeted attack methods in deep hashing based retrieval.

## 2. Related Work

### 2.1. Deep Hashing based Similarity Retrieval

Existing deep hashing methods can be roughly grouped into unsupervised deep hashing and supervised deep hashing. Unsupervised deep hashing methods learn deep features of samples by preserving the structure or metric consistence embedded in samples without using any semantic labels, which are usually achieved by unsupervised representation learning [34, 35, 10]. Although the unsupervised schemes are more general, their performance for retrieval is not satisfactory because of the semantic gap dilemma [37]. Supervised deep hashing methods use the class labels or pairwise similarities as the semantic supervision in the learning process, yielding promising results [42, 21, 47, 26, 28, 5, 17, 4, 45]. For example, the first deep hashing method [42] separates the whole hash learning into two steps: hash codes learning and data encoding. Recent works [21, 47, 26] validated the importance of jointly learning similarity-preserving hash codes and minimizing the quantization error in continuous-binary space transformation, and also showed the nonlinear deep hashing function learning could greatly improve the retrieval performance in an end-to-end training architecture. Thanks to the power of deep learning, researchers have extended the above methods to other complex tasks, *e.g.*, [16, 22, 46].

### 2.2. Adversarial Attacks

In image classification, an adversarial example is usually a carefully modified image, which is intentionally perturbed by adding visually imperceptible perturbations to the original image but can confuse the deep model to misclassify it. Since Szegedy *et al.* [38] discovered the properties of adversarial examples, various adversarial attack methods in image classification have been proposed to fool a trained DNN. According to the information of target model exposed to the adversary, adversarial attacks can be categorized as white-box attacks (*e.g.*, FGSM [12], I-FGSM [20], PGD [29] and C&W [6]) and black-box attacks (*e.g.* SBA [31] and ZOO [7]). For white-box attack, the adversary knows the whole network architecture and parameters so that it can design the adversarial perturbations by calculating the gradient of the loss *w.r.t.* inputs. As for black-box attack, only the input and the output are available to the adversary, thus it is more challenging and practical. However, these attack methods are optimization-based and they are quite slow for accessing the target model many times. Recently, generation-based attack methods (*e.g.*, [3, 43, 11, 30, 32, 13]) received much more attention due to their high-efficiency during test phase. Generation-based attack methods learn a generative model which transforms the input images into the adversarial samples. Once the generative model trained, it do not need to access the target model again and can generate adversarial examples with one-forward pass.

In addition to image classification, recent works on similarity retrieval [25, 39, 9, 44, 2, 23, 24] have also confirmed the vulnerability of DNNs to adversarial examples. Currently, there are only two works on attacking deep hashing models, *i.e.*, [44] for non-targeted attack and [2] for targeted attack. Specifically, HAG [44] is to make the hash code of the adversarial example as dissimilar as possible from that of the original example. Bai *et al*. [2] proposed the two targeted attack schemes in hashing based retrieval, dubbed *point-to-point* (P2P) and *deep hashing targeted attack* (DHTA). P2P randomly chooses a hash code of a sample with the target label to direct the adversarial example by maximizing their similarity of hash codes. DHTA transforms the targeted attack into a *point-to-set* optimization problem, which maximizes the similarity between the hash code of the adversarial example and the set of hash codes of samples with the target label. In detail, DHTA selects the *anchor code* [2] which has the smallest distance to the set as target code to guide the optimization of the adversarial example by minimizing the Hamming distance between the hash code of the adversarial example and the anchor code. Although making some progress in adversarial attacks of deep hashing, the existing works are generalized built on heuristic rules and optimization-based attack, which are inefficient and less effective. In this work, we, for the first time, design a neural network (PrototypeNet) to learn the prototype code of the target label for targeted attack and use a generative model to achieve the entire attack framework.

## 3. Generation-based Hashing Targeted Attack

We propose a novel prototype-supervised adversarial network (ProS-GAN) to efficiently generate adversarial examples for targeted attack of deep hashing-based image retrieval. Given a query image and a target label, targeted attack aims to learn an adversarial example for the query, whose nearest neighbors retrieved by the target hashing model from the database are semantically relevant to the target label. As shown in Figure 1, the overall framework is a generative adversarial network [11] and includes three sub-networks: a well-designed PrototypeNet $P$ for learning representative embedding of target labels, an adversarial

generator $G$ and a discriminator $D$ for generating adversarial examples. Specifically, $P$ embeds the target label into the semantic representation and outputs the predicted label and the corresponding prototype code which can be used to supervise the generation of adversarial examples. $G$ learns to transform the given query image into the targeted adversarial example. $D$ aims at distinguishing the generated image from the real one and categorizing them into the target and the original category, respectively. $G$ and $D$ are trained in an adversarial manner, which encourages $G$ to generate more realistic images and ensures the semantic representation informative.

### 3.1. Problem Formulation

Let $O = \{(x_i, y_i)\}_{i=1}^{N}$ denote a dataset containing $N$ instances labeled with $C$ classes, where $x_i$ indicates the original image for the $i$-th instance, and $y_i = [y_{i1}, ..., y_{iC}] \in \{0, 1\}^C$ corresponds to a multi-label vector. $y_{ij} = 1$ indicates that $x_i$ belongs to class $j$. Let $L = \{y_i\}_{i=1}^{M}$ denote all unique label dataset from $O$, where $M$ is the number of labels and $M \leq N$. We use similarity matrix $S$ to describe semantic similarities between each pair of data points. For any two instances $x_i$ and $x_j$, $S_{ij} = 1$ indicates they share as least one label, otherwise $S_{ij} = 0$. Similarly, we can use $S_{tj} = 1$ to indicate that a label $y_t$ and a instance $(x_j, y_j)$ have similar semantics.

Hashing aims to transform semantically similar data items into similar binary codes for efficient nearest neighbor search [40]. For a given deep hashing model $F(\cdot)$, the hash code of the sample $x_i$ is generated by

$$b_i = F(x_i) = \text{sign}(f_\theta(x_i)), \quad \text{s.t. } b_i \in \{-1, 1\}^K, \tag{1}$$

where $f(\cdot)$ is a DNN with parameters $\theta$ to approximate $F(\cdot)$, $\text{sign}(\cdot)$ is the sign function which binarizes the output of $f_\theta(\cdot)$ to $-1$ or 1, and $K$ is the hash code length. We use $B = (b_1\ b_2\ ...\ b_N)_{K\times N}$ to represent the hash code matrix for $O$. In general, $f(\cdot)$ is a convolutional neural network (CNN) [19, 36, 14], which consists of a convolutional feature extractor followed by fully-connected layers. In particular, deep hashing methods adopt $\tanh(\cdot)$ function to approximate the $\text{sign}(\cdot)$ function during training process.

In image retrieval, given a benign query image $x$ and a target label $y_t$, the goal of targeted attack is to generate corresponding adversarial example $x'$, which could cause the target model to retrieve the images semantically related to the target label. In addition, the adversarial perturbations (*i.e.*, $x' - x$) should be as small enough to be imperceptible to human eyes. In this paper, we aim to design a function $\Phi$ to achieve such objective, *i.e.*,

$$\begin{gathered}\Phi : (x, y_t) \rightarrow x', \\ \text{s.t. } \min \textstyle\sum_i d(F(x'), F(x_i^{(t)})) - \sum_j d(F(x'), F(x_j^{(n)})), \\ \|x - x'\|_p \leq \epsilon,\end{gathered} \tag{2}$$

where $d(\cdot, \cdot)$ is a distance measure, $\|\cdot\|_p$ ($p = 1, 2, \infty$) denotes $L_p$ norm, and $\epsilon$ is the maximum magnitude of adversarial perturbations. $x_i^{(t)}$ is a sample semantically relevant to the target label, and $x_j^{(n)}$ is a irrelevant sample. The minimized objective in Eqn. (2) ensures the hash code of the adversarial example $x'$ as close as possible to those of the semantically relevant samples, and simultaneously stays away from those of semantically irrelevant ones.

### 3.2. Prototype Generation

Unlike targeted attack in image classification, deep hashing models aim to generate semantic-preserving hash codes instead of categories, and thus labels can not directly used for guiding the generation of targeted adversarial samples. In hashing based retrieval, the most intuitive idea for targeted attack is that we can construct the most representative hash code of samples with the target label, and then use it to supervise the learning process of the adversarial example generation. As such, we construct a semantic encoding strategy, *i.e.*, PrototypeNet, to produce the prototype codes, which are used for representing the target labels. In PrototypeNet, the semantic representations are transformed into the corresponding prototype codes, and meanwhile could preserve the category knowledge of each target label.

Let $\theta_p$ denote the network parameters of the PrototypeNet $P$, and the objective function is defined as follows:

$$\begin{aligned}\min_{\theta_p} \mathcal{L}_{pro} &= \alpha_1 \mathcal{J}_1 + \alpha_2 \mathcal{J}_2 + \alpha_3 \mathcal{J}_3 \\ &= -\alpha_1 \sum_{i=1}^{M}\sum_{j=1}^{N} \left(S_{ij}\Omega_{ij} - \log\left(1 + e^{\Omega_{ij}}\right)\right) \\ &+ \alpha_2 \left\|H - B^{(p)}\right\|_F^2 + \alpha_3 \left\|\hat{Y} - Y\right\|_F^2, \\ &\text{s.t.} \quad B^{(p)} \in \{-1, 1\}^{K\times M},\end{aligned} \tag{3}$$

where $S$ is the semantic similarity matrix between the target labels and image instances from $O$, $\Omega_{ij} = \frac{1}{2}(H_{*i})^T(B_{*j})$, and $B$ is the hash code matrix for $O$. $H$ is the predicted hash codes for the targeted labels $Y$, and $\hat{Y}$ are the predicted labels. $B^{(p)}$ is the expected binary codes of $H$, *i.e.*, $B^{(p)} = \text{sign}(H)$. $\alpha_1$, $\alpha_2$, $\alpha_3$ are hyper-parameters. $\|\cdot\|_F$ denotes Frobenius norm.

The first term $\mathcal{J}_1$ in (3) is the negative log-likelihood of the pair-wise similarity in $S$. Given $S$, the probability of $S$ under the condition $B$ can be defined as follows:

$$p\left(S_{ij} \mid B\right) = \begin{cases} \sigma\left(\Omega_{ij}\right), & S_{ij} = 1 \\ 1 - \sigma\left(\Omega_{ij}\right), & S_{ij} = 0 \end{cases} \tag{4}$$

where $\sigma\left(\Omega_{ij}\right) = \frac{1}{1+e^{-\Omega_{ij}}}$. Notably, this pair-wise class encoding process can maximally capture the category information of the target label. Moreover, by using the above pair-wise similarity preservation loss in $\mathcal{J}_1$, the prototype codes can jointly maximize the compactness with the hash codes from semantically-relevant samples and separability

with those from semantically-irrelevant samples. Hence, by optimizing $\mathcal{J}_1$, the generated prototype codes can maintain the representative semantics of the target labels and the discriminative characteristics.

$\mathcal{J}_2$ is the quantization loss to minimize the approximation error between the prototype embedding $H$ and the expected binary codes $B^{(p)}$. $\mathcal{J}_3$ is the classification loss of the semantic representation $r_t$ to keep its category information.

### 3.3. Adversarial Generator $G$

Given a semantic representation $r_t$ from PrototypeNet and an original image $x$, we design an adversarial generator $G$ to learn the targeted adversarial example of $x$. Particularly, we integrate the decoded semantic representation and the original image into a well-designed encoder-decoder network with skip connection strategy. Generally, it is mainly composed of two parts: a semantic representation decoder $D_t$ and a image encoder-decoder $G_{xt}$. $D_t$ is used to upsample $r_t$ to $x_t$ with the same size of $x$. Then, $x$ and $x_t$ are concatenated into $G_{xt}$ to generate the adversarial example. Inspired by the skip connection in [33, 48], we concatenate the original image $x$ and the output of the last deconvolutional layer in $G_{xt}$, which can facilitate the reconstruction of the adversarial example $x'$ during training.

To guarantee the high-quality of the adversarial examples, we define the objective of the generator $G$ as follows:

$$\min_{\theta_g} \mathcal{L}_{gen} = \sum_{y_t \in L, (x,y) \in O} (\mathcal{J}_{ham} + \alpha \mathcal{J}_{re} + \beta \mathcal{J}_{adv}), \tag{5}$$

where $\alpha$, $\beta$ are the weighting factors, and $\theta_g$ is the parameters of $G$. $\mathcal{J}_{ham}$ is the Hamming distance loss, $\mathcal{J}_{re}$ is the reconstruction loss and $\mathcal{J}_{adv}$ is the adversarial loss.

$\mathcal{J}_{ham}$ is the Hamming distance loss that enforces the hash code of the adversarial example similar to the hash codes of samples relevant to the target label. Since we take the prototype code as representative of the target label, we can choose the prototype code as a target code to guide the generation of the adversarial example. As such, we can directly minimize the Hamming distance between the hash code of the adversarial example and the prototype code:

$$\mathcal{J}_{ham} = d_H(h_{x'}, h_t), \tag{6}$$

where $d_H(\cdot, \cdot)$ is the Hamming distance operator, $h_{x'}$ is the hash code of $x'$, and $h_t$ is the prototype code of the target label $y_t$. Due to $d_H(h_i, h_j) = \frac{1}{2}(K - h_i^T h_j)$, we can replace Hamming distance with inner product. Besides, we normalize the Hamming distance to range $[0, 2]$. Therefore, the final $\mathcal{J}_{ham}$ for instance $x$ is calculated as follows:

$$\begin{aligned} \mathcal{J}_{ham} &= -\frac{1}{K} h_t^T f(x') + 1 \\ &= -\frac{1}{K} h_t^T f(G(x, r_t)) + 1, \end{aligned} \tag{7}$$

where $h_{x'}$ is approximated by $f(x')$, and $r_t$ is the semantic representation of $y_t$ produced by PrototypeNet $P$.

$\mathcal{J}_{re}$ is the reconstruction loss that ensures the pixel difference between the adversarial example and the original image as small as possible, *i.e.*, the adversarial perturbations are small enough to be imperceptible. We simply adopt $L_2$-norm loss to measure the reconstruction error as follows:

$$\mathcal{J}_{re} = \|x - x'\|_2^2 = \|x - G(x, r_t)\|_2^2. \tag{8}$$

$\mathcal{J}_{adv}$ is the adversarial loss that ensures the semantic representation informative to enhance attack performance and encourages the generated adversarial example looks real. We re-formulate the objective label of $\mathcal{J}_{adv}$ as follows:

$$\widetilde{y_t} = [\underbrace{y_{t1}, y_{t2}, ..., y_{tC}}_{y_t}, 0], \tag{9}$$

where $y_t$ is in one-hot encoding, and the last node in $\widetilde{y_t}$ is for the fake sample. Hence, $\mathcal{J}_{adv}$ is defined as follows:

$$\mathcal{J}_{adv} = \|D(x') - \widetilde{y_t}\|_2^2. \tag{10}$$

### 3.4. Discriminator $D$

The designed discriminator is to distinguish the fake images (*i.e.*, the adversarial images) from the real images (*i.e.*, the benign images) and to categorize them into the target categories and the original categories, respectively. This strategy has two advantages: on one hand, the adversarial learning between the $G$ and $D$ encourages the generated adversarial examples look more realistic; on the other hand, the discriminator for category classification can ensure the representation $r_t$ semantically informative, which can enforce the category information of the representation to be embedded into the generated images and further improve the targeted attack performance. Specifically, the output of $D$ is a Sigmoid layer with $C + 1$ nodes in order to predict the category and to distinguish real/fake together, where the first $C$ nodes and the last one indicate the category and the falsity for the input image, respectively.

By inputting the real image $x$, the objective label for the discriminator is re-formulated as follows:

$$\widetilde{y} = [\overbrace{y_1, y_2, ..., y_C}^{y}, 0], \tag{11}$$

where $y$ is the label of the original image $x$. We set $(C + 1)$-th node as 0 for real samples. For the fake image, the objective label is re-formulated as follows:

$$\widetilde{y_t} = [\overbrace{y_{t1}, y_{t2}, ..., y_{tC}}^{y_t}, 1], \tag{12}$$

where $y_t$ is the target label, and we set $(C + 1)$-th node as 1 for fake samples. In summary, the objective loss function of $D$ is formulated as below:

$$\min_{\theta_d} \mathcal{L}_{dis} = \sum_{y_t \in L, (x,y) \in O} \frac{1}{2} \left( \|D(x) - \widetilde{y}\|_2^2 + \|D(x') - \widetilde{y_t}\|_2^2 \right), \tag{13}$$

where $\theta_d$ is the parameters of $D$.

Table 1. t-MAP (%) of targeted attack methods and MAP (%) of benign samples for different code lengths on three datasets.

| Method | Metric | FLICKR-25K | | | | NUS-WIDE | | | | MS-COCO | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 12 bits | 24 bits | 32 bits | 48 bits | 12 bits | 24 bits | 32 bits | 48 bits | 12 bits | 24 bits | 32 bits | 48 bits |
| Original | t-MAP | 63.58 | 63.49 | 63.49 | 63.59 | 55.51 | 55.57 | 55.67 | 55.86 | 42.33 | 42.60 | 42.67 | 42.90 |
| Noise | t-MAP | 63.37 | 63.40 | 63.45 | 63.55 | 54.23 | 54.59 | 55.61 | 55.83 | 42.33 | 42.60 | 42.67 | 42.90 |
| P2P | t-MAP | 82.55 | 83.79 | 84.65 | 84.44 | 70.33 | 71.44 | 71.98 | 73.26 | 56.77 | 59.11 | 59.87 | 59.72 |
| DHTA | t-MAP | 86.27 | 87.74 | 88.35 | 88.48 | 74.04 | 75.52 | 75.65 | 75.93 | 59.85 | 61.90 | 63.22 | 63.20 |
| ProS-GAN | t-MAP | **89.05** | **89.85** | **91.10** | **91.09** | **77.73** | **78.21** | **78.25** | **78.75** | **66.22** | **71.28** | **71.65** | **68.92** |
| Anchor code [2] | t-MAP | 86.40 | 87.98 | 88.68 | 88.97 | 75.15 | 77.41 | 78.40 | 78.12 | 60.35 | 63.14 | 64.41 | 64.65 |
| Prototype code | t-MAP | 91.33 | 92.20 | 92.91 | 93.12 | 79.04 | 80.71 | 81.34 | 81.50 | 67.38 | 71.30 | 71.93 | 69.98 |
| Original | MAP | 78.88 | 80.12 | 80.75 | 80.87 | 69.54 | 70.80 | 71.33 | 71.21 | 57.64 | 61.02 | 62.63 | 62.98 |

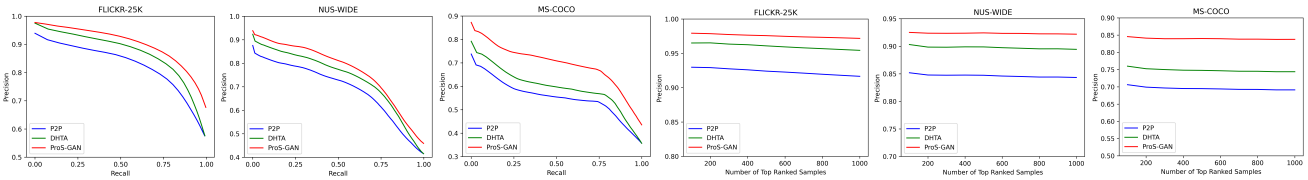



Figure 2. Precision-Recall and precision@topN curves on three datasets under 32 bits code length.

# 4. Experiments

## 4.1. Datasets

We evaluate our attack method on three popular multi-label datasets, *i.e.*, **FLICKR-25K** [15], **NUS-WIDE** [8] and **MS-COCO** [27]. ***FLICKR-25K*** contains $25,000$ images with $38$ classes. Follwing [41], we randomly sample $1,700$ images as queries, and the remaining as a database. Besides, we randomly select $5,000$ images from the database to train hashing models and our framework. ***NUS-WIDE*** consists of $269,648$ images in $81$ categories. We only select $195,834$ images comprising the $21$ most frequent concepts. Following [16], we take $2,100$ images as a query set, and the rest samples as a database. Moreover, we sample $10,500$ images from the database as a training set. ***MS-COCO*** contains $82,783$ training images and $40,504$ validation images, where each image is labeled with $80$ categories. We combine the training and validation sets, obtaining $122,218$ images. Following [5], we randomly sample $5,000$ images as queries, and the rest regarded as a database. $10,000$ images are randomly sampled from the database as training points.

## 4.2. Evaluation setup

For image hashing, we select the objective function of DPSH [26] as default method, which is one of the most representative deep hashing methods, to construct the target hashing model. Importantly, for any other popular deep hashing method, similar results could be achieved by our ProS-GAN. Specifically, VGG-11 [36] is adopted as the default backbone network. We replace the last fully connected layer of VGG-11 with the hashing layer, including a new fully-connected layer and the $\tanh$ activation.

For the network architecture, we built PrototypeNet with four-layer fully-connected networks ($y_t \rightarrow 4096 \rightarrow 512 \rightarrow r_t \rightarrow y_t, h_t$). We adopt a fully-connected layer and four deconvolutional layers for the Decoder $D_t$ to upsample the semantic representation $r_t$. We adapt the architecture for $G_{xt}$ from [49], and the discriminator contains five stride-2 convolutions and last layer with a $7 \times 7$ convolution.

After training ProS-GAN, we will use the PrototypeNet and the generator to attack the target hashing network. We set $\alpha_1$, $\alpha_2$ and $\alpha_3$ as $1$, $10^{-4}$ and $1$, respectively. The weighting factor $\alpha$ are set with $50$ for NUS-WIDE and MS-COCO, $100$ for FLICKR-25K, and $\beta$ is set as $1$. We train ProS-GAN using Adam [18] optimizer with initial learning rate $10^{-4}$. The training epochs are $100$ in batch size $24$. The ProS-GAN is implemented via PyTorch and is run on NVIDIA TITAN RTX GPUs. For the optimization procedure of ProS-GAN, please refer to the supplementary file.

Following [2], we adopt t-MAP (targeted mean average precision) to evaluate the targeted attack performance, instead of MAP (mean average precision). Since t-MAP uses the target labels as the test labels, the higher the t-MAP, the stronger the targeted attack. In image retrieval, we calculate t-MAP on all retrieved images from database. Besides, we also present the precision-recall curves (PR curves) and precision@topN curves. In detail, we randomly select a label as target label for each generation of adversarial examples. We compare the proposed framework with gradient-based methods, including P2P [2] and DHTA [2]. For fair comparison, the experimental settings of P2P and DHTA are same as [2], where the perturbation magnitude $\epsilon$ is set to $8/255$.

## 4.3. Results

**Targeted Attack Performance:** We provide the targeted attack performance of different methods, as shown in Table 1, using t-MAP criteria for comparison. The *Noise*

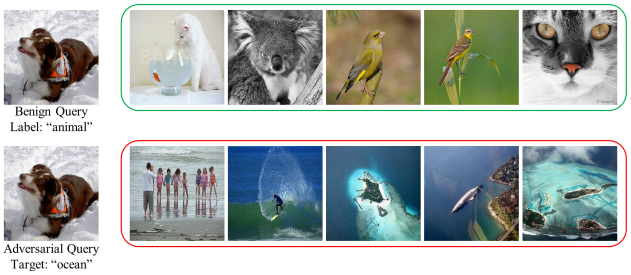



Figure 3. An example to retrieve top-5 similarity samples on NUS-WIDE with the benign query and its adversarial query.

in Table 1 is to query with noisy samples which are benign images with additive noises sampled from the uniform distribution $U(-\epsilon, +\epsilon)$. The t-MAP values of *Noise* are almost the same as the values of querying with benign samples (called *Original*) on FLICKR-25K, NUS-WIDE and MS-COCO datasets, which indicates the images with random noises can not bias the predictions of deep hashing models. In contrast, all the t-MAP values of DHTA and ProS-GAN are higher than the t-MAP values of 'Original', which verifies the effectiveness of adversarial attacks. Moreover, all the t-MAP values of our ProS-GAN are better than all the previous methods including P2P and DHTA. For example, compared with the state-of-the-art DHTA, we achieve absolute boosts of more than 2% in t-MAP for various number of bits on both FLICKR-25K and NUS-WIDE. On MS-COCO, our method outperforms DHTA over 5% in all cases. Especially, for 24 bits, the t-MAP of ProS-GAN is higher than DHTA by 9.38% on MS-COCO. As shown in Table 1, our superior performance benefits from the superiority of the prototype code over the anchor code, which indicates the prototype code is a more representative code of the target label. Furthermore, the targeted retrieval performance on three datasets in terms of the PR and precision@topN curves are shown in Figure 2 for comprehensive comparison. The curves of ProS-GAN are always above all other curves of previous methods, which also shows our performance does surpass all other methods. An example of the retrieval results with a benign image and its adversarial example generated by our method is displayed in Figure 3.

**Perceptibility & Efficiency:** In addition to attack performance, the *perceptibility* is also an important criteria to evaluate the quality of adversarial examples. Following [38], the perceptibility is calculated by $\sqrt{\frac{1}{Z}\|x' - x\|_2^2}$, where $Z$ is the pixel number, and pixel values are all normalized in the range $[0, 1]$. The higher the perceptibility, the worse visual quality of adversarial examples.

To make comprehensive comparison between efficiency and perceptibility of adversarial examples generated by various methods, we record t-MAP, perceptibility and generating time for 32-bits length on three datasets, which are summarized in Table 2. It is observed that ProS-GAN has

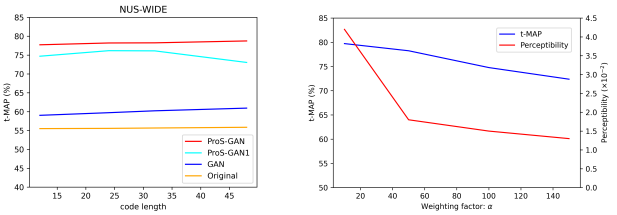
Table 2. t-MAP (%), perceptibility ($\times 10^{-2}$) between benign samples and adversarial samples (per image) and generating time (second per image) on attacking hashing models with 32 bits length. The hyper-parameter settings of gradient-based attacks: for FGSM [12], $\epsilon = 8/255$; for I-FGSM [20], $\epsilon = 8/255$ and step size $\alpha = 1/255$; for DHTA, the settings follows [2].

| Method | Iteration | FLICKR-25K | | | NUS-WIDE | | | MS-COCO | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | t-MAP | Per. | Time | t-MAP | Per. | Time | t-MAP | Per. | Time |
| DHTA + FGSM | 1 | 80.18 | 3.06 | 0.013 | 66.78 | 3.09 | 0.020 | 51.44 | 3.10 | 0.017 |
| DHTA + I-FGSM | 100 | 88.65 | 2.46 | 0.270 | 77.26 | 2.57 | 0.280 | 64.50 | 2.56 | 0.277 |
| DHTA | 2000 | 88.35 | **0.84** | 9.957 | 75.65 | **0.80** | 5.601 | 63.22 | **0.63** | 5.685 |
| ProS-GAN | 1 | **91.10** | 2.23 | **0.006** | **78.25** | 1.87 | **0.005** | **71.65** | 1.86 | **0.005** |

the highest targeted attack performance, the second-best visual quality and the fastest generation speed for all datasets. Specifically, DHTA with FGSM has fast speed to generate adversarial examples, but it has lower attack performance yet higher perceptibility. On FLICKR-25K, ProS-GAN outperforms DHTA about $1660\times$ of generation speed. Although ProS-GAN performs a little worse than DHTA [2] on visual quality, DHTA needs multiple gradient descent to optimize adversarial perturbations and can not attack hashing models in real time. In summary, ProS-GAN not only outperform all the previous methods in attack performance and speed, but also can produce adversarial examples with high visual quality.

### 4.4. Ablation studies

**Effect of the PrototypeNet:** In order to explore the influence of the PrototypeNet for targeted attack performance, we remove the Hamming distance loss from the proposed structure denoted as *GAN*. As shown in Figure 4(a), ProS-GAN outperforms GAN by a large margin. Thus, the prototype code produced by PrototypeNet determines the performance of the targeted attack.




(a) (b)

Figure 4. (a) t-MAP (%) for various code length on NUS-WIDE with different ablation architecture. (b) t-MAP (%) and perceptibility ($\times 10^{-2}$) for 32 bits code length on NUS-WIDE.

**Effect of the discriminator:** In addition to making the generated adversarial examples look more realistic, we argue that the discriminator plays an important role in category classification and enforce the semantic representations informative, which can boost the attack performance. In order to verify this point, we remove the classification module

Table 3. Transfer t-MAP (%) for the NUS-WIDE dataset. H-AlexNet, H-VGG11 and H-ResNet18 denote 12 bits DPSH models based on AlexNet [19], VGG11 [36] and ResNet18 [14], respectively, and "*" denotes their 32 bits variants.

| Method | Attacked model | H-AlexNet | H-AlexNet* | H-VGG11 | H-VGG11* | H-ResNet18 | H-ResNet18* |
|---|---|---|---|---|---|---|---|
| DHTA | H-AlexNet | **71.11** | 70.39 | 56.25 | 57.09 | 53.64 | 52.89 |
| | H-AlexNet* | 68.27 | **71.86** | 55.98 | 56.98 | 53.36 | 52.54 |
| | H-VGG11 | 54.94 | 55.11 | **74.04** | 74.94 | 54.78 | 54.48 |
| | H-VGG11* | 54.82 | 55.24 | 73.32 | **75.65** | 54.32 | 54.16 |
| | H-ResNet18 | 54.11 | 54.56 | 54.69 | 55.51 | **67.55** | 66.38 |
| | H-ResNet18* | 54.03 | 54.46 | 54.31 | 55.41 | 65.34 | **70.08** |
| Ours | H-AlexNet | **75.13** | 74.96 | 63.70 | 64.14 | 58.51 | 58.43 |
| | H-AlexNet* | 73.81 | **78.03** | 64.35 | 65.94 | 62.13 | 63.12 |
| | H-VGG11 | 60.69 | 60.75 | **77.73** | 76.05 | 62.20 | 61.99 |
| | H-VGG11* | 60.61 | 61.84 | 76.42 | **78.25** | 63.34 | 63.52 |
| | H-ResNet18 | 59.06 | 59.35 | 60.59 | 60.50 | **70.79** | 69.69 |
| | H-ResNet18* | 59.25 | 58.89 | 59.36 | 59.76 | 66.92 | **75.21** |
| | Original | 54.09 | 54.45 | 55.51 | 55.67 | 53.54 | 53.28 |

of the discriminator denoted as *ProS-GAN1*, and the result is shown in Figure 4(a). The curve of ProS-GAN is above ProS-GAN1 in all different hash bits, which shows that the discriminator can indeed further improve the attack performance due to less interference to target information.

**Visual quality *vs*. targeted retrieval precision:** The weighting factor $\alpha$ controls the reconstruction quality of generated adversarial examples. To explore the impact of different $\alpha$ on visual quality and attack performance of adversarial examples, we make comparison results with 32 bits on NUS-WIDE, as shown in Figure 4(b). When $\alpha$ increases, the visual quality gradually increases with the decreasing of perceptibility values, but the attack performance gradually drops. Thus, $\alpha$ can control the balance between the imperceptible quality and attack performance of adversarial perturbations.

### 4.5. Transferability

Transferability refers to the capability of adversarial examples generated from one model to successfully attack another model, which is a way to achieve black-box attacks. To evaluate the transferability of our attack method, we carry on the transferable experiments for hashing models with different backbone or different hash bits, which is summarized in Table 3. We observe that the adversarial perturbations generated from one hash bit can achieve much similar t-MAP to another hash bit based on the same architecture of hashing model. Besides, our method equips with good transferability from one DNN to another DNN while DHTA fails to transfer cross networks. For example, when we adopt adversarial examples generated by ProS-GAN with H-AlexNet* to attack H-VGG11, an 8.84% targeted performance increases for the *Original* t-MAP, but the result of DHTA only changes by 0.47%.

### 4.6. Universality on different hashing methods

We argue that our proposed scheme is applicable to most existing popular deep hashing methods. To evaluate this point, we compare with targeted attacks (P2P and DHTA) on other hashing methods, including DPH [44], and HashNet [5]. The results are reported in Table 4. As shown in the table, even if tested on different deep hashing models, our targeted attack method is still effective and much better than the state-of-the-art DHTA in all cases, which further demonstrates the effectiveness of the proposed targeted hashing attack method. For example, the t-MAP value of our ProS-GAN is higher than DHTA by 3.12% on the HashNet with 32 bits code length.

Table 4. t-MAP (%) of targeted attack methods and MAP (%) of benign samples for different hashing models on NUS-WIDE.

| Method | Metric | DPH | | | | HashNet | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 12 bits | 24 bits | 32 bits | 48 bits | 12 bits | 24 bits | 32 bits | 48 bits |
| Original | t-MAP | 54.31 | 54.56 | 54.58 | 54.59 | 54.42 | 54.99 | 55.40 | 55.31 |
| Noise | t-MAP | 53.12 | 53.40 | 53.42 | 53.45 | 53.51 | 53.99 | 54.09 | 54.10 |
| P2P | t-MAP | 69.66 | 70.79 | 71.00 | 71.38 | 65.16 | 69.28 | 71.72 | 73.17 |
| DHTA | t-MAP | 72.75 | 73.86 | 74.29 | 74.07 | 66.23 | 71.25 | 73.83 | 76.29 |
| ProS-GAN | t-MAP | **75.57** | **76.87** | **78.62** | **77.62** | **71.29** | **75.06** | **76.95** | **78.48** |
| Anchor code | t-MAP | 74.25 | 75.72 | 75.94 | 75.97 | 70.41 | 75.68 | 77.92 | 79.56 |
| Prototype code | t-MAP | 78.01 | 79.83 | 79.85 | 78.79 | 73.99 | 78.58 | 80.67 | 81.67 |
| Original | MAP | 70.09 | 71.27 | 71.23 | 71.50 | 66.40 | 70.69 | 72.62 | 73.78 |

## 5. Conclusion

In this paper, we proposed a prototype-supervised adversarial network (ProS-GAN) for flexible targeted hashing attack, including a PrototypeNet, a generator and a discriminator. Specifically, we defined a category-level PrototypeNet to generate the semantic representation and to learn the prototype code as the representative of the target label for supervising the adversarial example generation. Moreover, the designed generator incorporated the decoded semantic representation into the original image to construct the adversarial example. Benefiting from the adversarial learning between the generator and the discriminator, the adversarial example could keep the visually realistic property and hold stronger attack performance. Extensive experiments showed that our ProS-GAN could achieve efficient and superior attack performance with higher transferability than the state-of-the-art targeted attack methods of deep hashing.

## 6. Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Grants Nos. 62002085, 62076213), the Guangdong Basic and Applied Basic Research Foundation (Grants Nos. 2019A1515110475, 2019B1515120055), and also supported by the university development fund of the Chinese University of Hong Kong, Shenzhen under grant No. 01001810, and the special project fund of Shenzhen Research Institute of Big Data under grant No. T00120210003.

## References


[1] Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *2006 47th annual IEEE Symposium on Foundations of Computer Science*, pages 459–468. IEEE, 2006. 1

[2] Jiawang Bai, Bin Chen, Yiming Li, Dongxian Wu, Weiwei Guo, Shu-tao Xia, and En-hui Yang. Targeted attack for deep hashing based retrieval. In *European Conference on Computer Vision*, pages 618–634. Springer, 2020. 1, 3, 6, 7

[3] Shumeet Baluja and Ian Fischer. Adversarial transformation networks: Learning to generate adversarial examples. *arXiv preprint arXiv:1703.09387*, 2017. 2, 3

[4] Yue Cao, Mingsheng Long, Bin Liu, and Jianmin Wang. Deep cauchy hashing for hamming space retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1229–1237, 2018. 3

[5] Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Philip S Yu. Hashnet: Deep learning to hash by continuation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5608–5617, 2017. 1, 3, 6, 8

[6] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy*, pages 39–57. IEEE, 2017. 1, 3

[7] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 15–26, 2017. 3

[8] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yantao Zheng. Nus-wide: a real-world web image database from national university of singapore. In *Proceedings of the ACM International Conference on Image and Video Retrieval*, pages 1–9, 2009. 6

[9] Yan Feng, Bin Chen, Tao Dai, and Shutao Xia. Adversarial attack on deep product quantization network for image retrieval. *Proceedings of the AAAI conference on Artificial Intelligence*, 2020. 3

[10] Kamran Ghasedi Dizaji, Feng Zheng, Najmeh Sadoughi, Yanhua Yang, Cheng Deng, and Heng Huang. Unsupervised deep generative adversarial hashing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3664–3673, 2018. 3

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. 3

[12] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *International Conference on Learning Representations*, 2015. 1, 3, 7

[13] Jiangfan Han, Xiaoyi Dong, Ruimao Zhang, Dongdong Chen, Weiming Zhang, Nenghai Yu, Ping Luo, and Xiaogang Wang. Once a man: Towards multi-target attack via learning multi-target adversarial network once. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5158–5167, 2019. 2, 3

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 4, 8

[15] Mark J Huiskes and Michael S Lew. The mir flickr retrieval evaluation. In *Proceedings of the ACM International Conference on Image and Video Retrieval*, pages 39–43, 2008. 6

[16] Qing-Yuan Jiang and Wu-Jun Li. Deep cross-modal hashing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3232–3240, 2017. 3, 6

[17] Qing-Yuan Jiang and Wu-Jun Li. Asymmetric deep supervised hashing. *Proceedings of the AAAI conference on artificial intelligence*, 2018. 3

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015. 6

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012. 4, 8

[20] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *International Conference on Learning Representations*, 2017. 1, 3, 7

[21] Hanjiang Lai, Yan Pan, Ye Liu, and Shuicheng Yan. Simultaneous feature learning and hash coding with deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3270–3278, 2015. 1, 3

[22] Chao Li, Cheng Deng, Ning Li, Wei Liu, Xinbo Gao, and Dacheng Tao. Self-supervised adversarial hashing networks for cross-modal retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4242–4251, 2018. 3

[23] Chao Li, Shangqian Gao, Cheng Deng, De Xie, and Wei Liu. Cross-modal learning with adversarial samples. In *Advances in Neural Information Processing Systems*, volume 32, 2019. 3

[24] Chao Li, Haoteng Tang, Cheng Deng, Liang Zhan, and Wei Liu. Vulnerability vs. reliability: Disentangled adversarial examples for cross-modal learning. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 421–429, 2020. 3

[25] Jie Li, Rongrong Ji, Hong Liu, Xiaopeng Hong, Yue Gao, and Qi Tian. Universal perturbation attack against image retrieval. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4899–4908, 2019. 3

[26] Wu-Jun Li, Sheng Wang, and Wang-Cheng Kang. Feature learning based deep supervised hashing with pairwise labels. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1711–1717, 2016. 1, 3, 6

[27] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proceedings of the European Conference on Computer Vision*, pages 740–755. Springer, 2014. 6

[28] Haomiao Liu, Ruiping Wang, Shiguang Shan, and Xilin Chen. Deep supervised hashing for fast image retrieval. In

*Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2064–2072, 2016. 1, 3

[29] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *International Conference on Learning Representations*, 2017. 1, 3

[30] Konda Reddy Mopuri, Utkarsh Ojha, Utsav Garg, and R Venkatesh Babu. Nag: Network for adversary generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 742–751, 2018. 3

[31] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 506–519, 2017. 3

[32] Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie. Generative adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4422–4431, 2018. 3

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015. 5

[34] Ruslan Salakhutdinov and Geoffrey Hinton. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978, 2009. 3

[35] Fumin Shen, Yan Xu, Li Liu, Yang Yang, Zi Huang, and Heng Tao Shen. Unsupervised deep hashing with similarity-adaptive and discrete optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):3034–3044, 2018. 3

[36] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations*, 2015. 4, 6, 8

[37] Arnold WM Smeulders, Marcel Worring, Simone Santini, Amarnath Gupta, and Ramesh Jain. Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, 2000. 3

[38] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *International Conference on Learning Representations*, 2014. 1, 3, 7

[39] Giorgos Tolias, Filip Radenovic, and Ondrej Chum. Targeted mismatch adversarial attack: Query with a flower to retrieve the tower. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5037–5046, 2019. 3

[40] Jingdong Wang, Ting Zhang, Nicu Sebe, Heng Tao Shen, et al. A survey on learning to hash. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):769–790, 2017. 1, 4

[41] Zijian Wang, Zheng Zhang, Yadan Luo, Zi Huang, and Heng Tao Shen. Deep collaborative discrete hashing with semantic-invariant structure construction. *IEEE Transactions on Multimedia*, 2020. 6

[42] Rongkai Xia, Yan Pan, Hanjiang Lai, Cong Liu, and Shuicheng Yan. Supervised hashing for image retrieval via image representation learning. In *Proceedings of the AAAI conference on Artificial Intelligence*, volume 1, page 2, 2014. 1, 3

[43] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks. *Proceedings of the International Joint Conference on Artificial Intelligence*, 2018. 2, 3

[44] Erkun Yang, Tongliang Liu, Cheng Deng, and Dacheng Tao. Adversarial examples for hamming space search. *IEEE Transactions on Cybernetics*, 2018. 1, 3, 8

[45] Zheng Zhang, Luyao Liu, Yadan Luo, Zi Huang, Fumin Shen, Heng Tao Shen, and Guangming Lu. Inductive structure consistent hashing via flexible semantic calibration. *IEEE Transactions on Neural Networks and Learning Systems*, 2020. 3

[46] Zheng Zhang, Li Liu, Fumin Shen, Heng Tao Shen, and Ling Shao. Binary multi-view clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(7):1774–1782, 2018. 3

[47] Han Zhu, Mingsheng Long, Jianmin Wang, and Yue Cao. Deep hashing network for efficient similarity retrieval. In *Proceedings of the AAAI conference on Artificial Intelligence*, 2016. 1, 3

[48] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. Hidden: Hiding data with deep networks. In *Proceedings of the European Conference on Computer Vision*, pages 657–672, 2018. 5

[49] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2223–2232, 2017. 6