Pairwise Teacher-Student Network for Semi-Supervised Hashing

Shifeng Zhang, Jianmin Li and Bo Zhang
Institute for Artificial Intelligence, State Key Lab of Intelligent Technology and Systems,
Beijing National Research Center for Information Science and Technology,
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
zhangsf15@mails.tsinghua.edu.cn, lijianmin@mail.tsinghua.edu.cn, dcszb@mail.tsinghua.edu.cn

Abstract

Hashing method maps similar high-dimensional data to binary hashcodes with smaller hamming distance, and it has received broad attention due to its low storage cost and fast retrieval speed. Pairwise similarity is easily obtained and widely used for retrieval, and is widely applied in most supervised hashing algorithms. As labeling all data pairs is difficult, semi-supervised hashing is proposed which aims at learning efficient codes with limited labeled pairs and abundant unlabeled ones. Existing methods build graphs to capture the structure of dataset, but they are not working well for complex data as the graph is built based on the data representations and determining the representations of complex data is difficult. In this paper, we propose a novel teacher-student semi-supervised hashing framework in which the student is trained with the pairwise information produced by the teacher network. The network follows the smoothness assumption in which the retrieval results are similar for neighborhood queries. Experiments on large-scale datasets show that the proposed method reaches impressive gain over the supervised baselines and is superior to state-of-the-art semi-supervised hashing methods.

1. Introduction

With the explosion of high-dimensional media data, hashing is widely applied for efficient approximate Nearest Neighbor(ANN) search [6] due to its short retrieval time and small storage space [6, 16, 22, 26]. Hashing aims at encoding high-dimensional data into compact hashcodes, so that similar data are mapped to hashcodes with similar hamming distance.

Data-dependent learning-to-hash methods aim at learning hash functions with the training data, and the learned codes is able to capture the data distributions. Learning-to-hash methods can be divided into three categories: unsupervised hashing [7, 17], supervised hashing [16, 14] and semi-supervised hashing [28, 27, 20]. Experiments convey that the codes learned by (semi-)supervised hashing methods can capture more semantic information than unsupervised ones. Recently, deep hashing methods have achieved great success [26, 29, 28, 2]. It aims at learning hashcodes and the deep networks simultaneously, thus the codes generated by deep networks contain much better semantic information.

For ANN search, pairwise similarities between data pairs play an important role in evaluating the quality of search. For generating efficient hashcodes, the pairwise similarity is adopted in (deep) supervised hashing problems such that similar data pairs should be mapped to codes with small hamming distance. Most hashing methods model the similarities with the pairwise losses. For ease of back-propagation, these methods simply generate data pairs within a mini-batch and achieve good results [15, 3, 4, 13].

Despite the success of supervised hashing, labeling all the database data (pairs) is almost intractable as the number of data is dramatically increasing. To utilize the abundant database data, deep semi-supervised hashing [28, 27] has been proposed in which the hash function is trained with the labeled data pairs and abundant unlabeled ones. The success of semi-supervised hashing lies in the smoothness assumption such that neighborhood data are likely to have the same outputs. These methods construct graphs for the unlabeled data to capture the neighborhood structure among the samples. However, the data and their representations may lie in high-dimensional nonlinear manifolds and may not contain enough semantic information with limited labeled data. As the graph is built based on data representations, the graph may not model the neighborhood structure of data precisely, which violates the smoothness assumption to some extent and affect the hashing performance.

Recently, perturbation-based teacher-student semi-supervised learning (SSL) algorithms have witnessed great success [11, 23]. These methods follow the smoothness assumption in which the learned classifiers produce consensus prediction of a noisy input, thus they can better capture the structure of unlabeled data [30] with the learned

representations [18]. However, the proposed teacher-student method can just deal with single data point, but not consider the pairwise relationship between samples. By carefully designing the teacher-student architecture and the loss for pairwise similarities, we may utilize the advantage of this architecture and obtain a novel semi-supervised hashing method.

In this paper, we propose a novel semi-supervised hashing algorithm called *Pairwise Teacher-Student Semi-Supervised Hashing*(PTS^{3}H) in which the codes are trained with pairwise similarities and abundant unlabeled data pairs. The proposed PTS^{3}H is a teacher-student network architecture where the student is trained with pairwise loss and unsupervised regularizers, and the teacher is the ensemble of the student to generate efficient pairwise representations. For modeling the pairwise information, we propose the general *consistent pairwise loss* to follow the smoothness assumptions [23, 11] such that similar pairs correspond to similar pairwise similarities. More specifically, we propose two types of losses to model the pairwise similarities with local and global pairwise information: *consistent similarity loss* for consistent pairwise similarities, and *quantized similarity loss* in which the quantized [8] similarities can be modeled by global information within data pairs. Experiment shows that the proposed PTS^{3}H achieves great improvement over the supervised baselines, and it is superior or comparable with the state-of-the-art semi-supervised hashing algorithms.

2. Background

Suppose we are given n data samples $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n \in \mathcal{X}$, and $\mathcal{X}$ is the training dataset. Denote $\mathcal{S}$ as a set such that $(i,j) \in \mathcal{S}$ implies $\mathbf{x}_i, \mathbf{x}_j$ have similarity information, and we denote $s_{ij} = 1$ if $(i,j) \in \mathcal{S}$, and $s_{ij} = 0$ otherwise. In practical applications, the similarity information of some data pairs is unknown, which are denoted as $\mathcal{U}$.

Denote b as the length of the hashcode to learn, the goal of the semi-supervised hash learning is to learn the hash function $H(\mathbf{x}) = [h_1(\mathbf{x}), ..., h_b(\mathbf{x})]^\top \in \{-1, 1\}^b$ with n data samples and the pairwise similarities. We denote $\mathbf{h}_i = H(\mathbf{x}_i), i = 1, 2, ..., n$ as the learned hashcode of $\mathbf{x}_i$.

2.1. Pairwise Loss for Supervised Hashing

Pairwise losses is widely used in (deep) supervised hashing algorithm [16, 14, 15, 3, 13, 27]. For the given training data and pairwise information, the basic formulation of pairwise loss is

$$\mathcal{L}_s = \frac{1}{|\mathcal{S}|} \sum_{(i,j)\in\mathcal{S}} l(u_{ij}, s_{ij}), \quad u_{ij} = \mathrm{sim}(\mathbf{h}_i, \mathbf{h}_j) \tag{1}$$

where $u_{ij} = \mathrm{sim}(\mathbf{h}_i, \mathbf{h}_j)$ are the similarity (or distance) between the codes $\mathbf{h}_i, \mathbf{h}_j$.

Different types of $l(u_{ij}, s_{ij})$ are discovered in different supervised hashing algorithms such that

- KSH loss: $l(u_{ij}, s_{ij}) = [b(2s_{ij} - 1) - u_{ij}]^2, u_{ij} = \mathbf{h}_i^\top \mathbf{h}_j$ in KSH [16] and FastH [14];
- DSH loss: $l(u_{ij}, s_{ij}) = -s_{ij}u_{ij} + (1 - s_{ij}) \max(0, 2b + u_{ij}), u_{ij} = -(\mathbf{h}_i - \mathbf{h}_j)^2$ in DSH [15];
- DPSH loss: $l(u_{ij}, s_{ij}) = -s_{ij}u_{ij} + \log(1 + e^{u_{ij}}), u_{ij} = \frac{1}{2}\mathbf{h}_i^\top \mathbf{h}_j$ in DPSH [13], DHN [3].

Optimizing $l(u_{ij}, s_{ij})$ is expected to learn hashcodes such that similar data pairs have codes with small hamming distance, and vice versa. It should be noticed that the supervised information is just pairwise information, which is widespread in the real world.

2.2. Semi-Supervised Hashing

Semi-supervised hashing focuses on learning hash function with limited labeled data pairs as well as abundant unlabeled pairs. The general form of loss to be optimized is

$$\mathcal{L} = \mathcal{L}_s + \omega \mathcal{R}_u \tag{2}$$

where $\mathcal{L}_s$ is Eq. (1), $\mathcal{R}_u$ ls the regularization term for unlabeled data. SPLH [24] adopts the bit-balanced constraint for regularization, but it does not consider the relationship between samples. Graph-based methods like SSDH [28] and BGDH [27] construct an affinity graph for unlabeled samples, and the regularization loss is constructed based on the graph, which intends to follow the smoothness assumptions. However, the graph is constructed by data representations where the semantic gap may be involved, violating the smoothness assumptions. Recently, deep generative models have achieved success in SSL, and DSH-GANs [20] proposes a GAN [21] based hashing method. The conditional GAN is trained with labeled and unlabeled data to generate labeled samples, which are used for training the hashing network. It achieves state-of-the-art in some datasets, but cannot be trained with the pairwise supervision.

2.3. Teacher-Student Network for Semi-Supervised Learning

Semi-supervised learning (SSL) aims at learning with limited labeled data and abundant unlabeled data. Most SSL methods lies in the smoothness assumption such that similar data correspond to the same label. Various approaches are discovered such as transductive approach [9, 25], graph-based approach [1, 30], but they are not working well in complex dataset as the underlying structure of data is hard to capture. Recently, perturbation-based semi-supervised learning approach has achieved great success, where the perturbed inputs correspond to the consensus prediction. These methods propose a dual role, i.e., the teacher and the

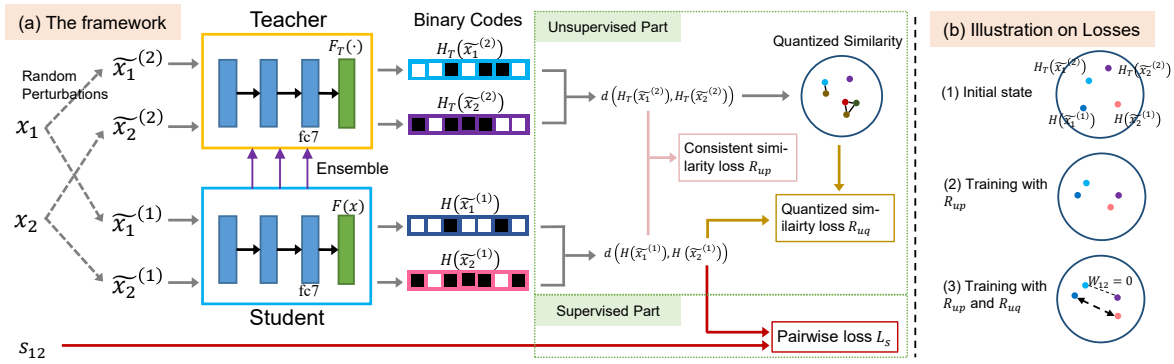

Figure 1. Overview of the PTS^{3}H algorithm. (a) The general framework of the PTS^{3}H algorithm. The input data is $\mathbf{x}_1, \mathbf{x}_2$ and the pairwise similarity is s_{12} if available. $\mathcal{L}_s$ is computed for labeled pairs and $\mathcal{R}_{uc}, \mathcal{R}_{up}, \mathcal{R}_{uq}$ are computed for unlabeled pairs. (b) Illustration on training with different unsupervised regularizations.

student. The student is learned as before; the teacher generates the targets for training the student. Formally, considering the dataset $\mathcal{X}$ where part of data are labeled, we aim at optimizing the following loss function:

$$\mathcal{L}^{(c)} = \mathcal{L}_s^{(c)} + \omega \mathcal{R}_u^{(c)} \tag{3}$$

where c denotes classification, $\mathcal{L}_s^{(c)}$ is the supervised term such as the softmax loss, $\mathcal{R}_u^{(c)}$ is the unsupervised regularization such that

$$\mathcal{R}_u^{(c)} = \sum_{\mathbf{x} \in \mathcal{X}} d(f(\tilde{\mathbf{x}}^{(1)}), f_T(\tilde{\mathbf{x}}^{(2)})) \tag{4}$$

where $\tilde{\mathbf{x}}^{(1)}, \tilde{\mathbf{x}}^{(2)}$ are two random perturbations, $f(\cdot), f_T(\cdot)$ are the outputs of student and teacher network respectively, and $d(\cdot, \cdot)$ is the distance between two output. There are several ways to define the teacher f_T. TempEns [11] considers f_T as the exponentially moving average(EMA) of the student's output; Mean Teacher [23] ensembles the student to form the teacher network; VAT [19] regards the adversarial perturbations to form the teacher. These methods achieve state-of-the-art on SSL problems.

In spite of this, perturbation-based methods is just able to regularize the single data point, but do not consider neighborhood structure between samples. SNTG [18] constructs a graph by the teacher to capture the neighborhood structure, and experiments convey that introducing the graph achieves better performance. However, the graph in SNTG is built specifically for classification.

With the success of teacher-student network for semi-supervised learning, in this paper, we propose a novel teacher-student framework for semi-supervised hashing in which only small portion of pairwise similarity information is provided. Considering we perform the hamming distance learning, we propose a novel *consistent pairwise loss* in which the consistent distances for similar data pairs are reached so that it is able to follow the smoothness assumption where neighborhood queries achieve similar retrieval results. Experiments show its superiority over the state-of-the-art semi-supervised hashing algorithms.

3. Methodology

In this section, we propose the novel deep semi-supervised hashing called *Pairwise Teacher-Student Semi-Supervised Hashing(PTS^{3}H)*, in which the teacher-student network is adopted.

3.1. The Teacher-Student Framework

The proposed PTS^{3}H is a teacher-student architecture shown in Figure 1(a). The architecture of teacher network and the student are the same, in which the last layer is the fully-connected layer with b outputs (b is the hashcode length), and the rest layers can be the basic deep network like AlexNet, VGGNet, etc.

The update rule of the teacher-student network is similar as Mean Teacher [23]. The student is learned with labeled data pairs and guided by the teacher. Denote $\theta(t)$ and $\theta_T(t)$ as the parameters of the student and teacher network at training step t respectively, the teacher network is updated by EMA as follows:

$$\theta_T(t) = \alpha \theta_T(t-1) + (1-\alpha)\theta(t) \tag{5}$$

as the teacher is the average of the student, the teacher's output can be regarded as the mean of the student's.

Denote $F(\mathbf{x}), F_T(\mathbf{x}) \in \mathbb{R}^b$ as the output of the student and teacher networks respectively, the binary codes of data $\mathbf{x}$ can be easily obtained with the embedded teacher network $H_T(\mathbf{x}) = \text{sgn}(F_T(\mathbf{x}))$. Note that the $\mathbf{x}$ is not perturbed.

3.2. Loss Function

The general form of loss to be optimized is Eq. (2). For labeled data pairs, the training loss is the pairwise loss in Eq. (1). For unlabeled data, $\mathcal{R}_u$ should be defined such that the teacher network generates targets to guide the student network. As we focus on the pairwise similarities, learning the similarities of the embedded hamming space are quite important. For input pairs, the targets for the student should be the similarities of the codes generated by the teacher. We

therefore propose the general form of the *consistent pairwise loss* such that

$$\begin{aligned}\mathcal{R}_u &= \frac{1}{|\mathcal{X}|^2}\sum_{\mathbf{x}_1,\mathbf{x}_2\in\mathcal{X}} l_c(u_{12}, u_{T12}) \\ u_{12} &= \mathrm{sim}(H(\tilde{\mathbf{x}_1}^{(1)}), H(\tilde{\mathbf{x}_2}^{(1)})) \\ u_{T12} &= \mathrm{sim}(H_T(\tilde{\mathbf{x}_1}^{(2)}), H_T(\tilde{\mathbf{x}_2}^{(2)}))\end{aligned} \tag{6}$$

where $\tilde{\mathbf{x}_i}^{(1)}, \tilde{\mathbf{x}_i}^{(2)}, i = 1, 2$ are two random perturbations of $\mathbf{x}_i$, $l_c(u, u_T)$ is a certain type of loss and u, u_T denote the pairwise similarities of codes generated from the student and the teacher respectively. Eq. (6) is quite different from the original Mean Teacher [23] in which only the single data point is considered for training.

For Eq. (6), We propose two efficient losses named *consistent similarity loss* and *quantized similarity loss*.

Consistent Similarity Loss It is expected that the learned codes should follow the smoothness assumption in that a noisy input query correspond to the consistent retrieval results. To what follows, the similarities of codes between the noisy data pairs should be consistent. As illustrated in Figure 1(b.2), if $\mathbf{x}_1, \mathbf{x}_2$ is quite similar and so as $\mathbf{x}_3, \mathbf{x}_4$, the difference between $\mathrm{sim}(H(\mathbf{x}_1), H(\mathbf{x}_3))$ and $\mathrm{sim}(H_T(\mathbf{x}_2), H_T(\mathbf{x}_4))$ should be small. Thus the *consistent similarity loss* is defined with

$$l_c(u, u_T) = (u - u_T)^2 \tag{7}$$

where $l_c(u, u_T)$ are the same as Eq. (6). We rename the $\mathcal{R}_u$ as $\mathcal{R}_{up}$ if Eq. (7) is introduced.

Quantized Similarity Loss The consistent similarity loss is only able to capture the locally structure of a certain data pair, ignoring the global structure between samples. Inspired by the quantization methods in which large amount of information can be compressed with quantization [8], we quantize the pairwise similarity produced by the ensembled teacher to guide the hash learning. As the quantization procedure is based on global unlabeled data pairs, it is expected that the quantized similarities contain global pairwise information, leading to better learned codes.

We denote $\mathbf{W} \in \{0, 1\}^{n\times n}$ as the quantized similarity matrix to be learned, where n is the number of training samples. Denote W_{ij} as the element at ith row and jth column, thus $W_{ij} = 1$ indicates $\mathbf{x}_i$ and $\mathbf{x}_j$ are *pseudo similar pair*, and 0 otherwise. Considering the teacher output $H_T(\mathbf{x})$ is the ensemble of embedded codes of $\mathbf{x}$, $H_T(\mathbf{x})$ can be regarded as the precise feature embedding of the data point $\mathbf{x}$. To what follows, we use the teacher output to determine the pseudo similar pairs. The similarity matrix is defined according to the distances of teacher output such that

$$W_{ij} = \begin{cases} 1 & u_{Tij} \geq thr \\ 0 & u_{Tij} < thr \end{cases} \tag{8}$$

where $u_{Tij} = \mathrm{sim}(H_T(\tilde{\mathbf{x}_i}^{(2)}), H_T(\tilde{\mathbf{x}_j}^{(2)}))$ is defined the same as Eq. (6), thr is the threshold, which is set according to the dataset. In practical applications, the distribution between labeled and unlabeled pairs are expected to be the same. We can set thr such that the ratio of pseudo similar pairs is the same as the ratio of similar pairs among labeled pairs, so that the distribution of similarities in unlabeled data are expected to the same as labeled ones.

Given the generated pseudo similarity pairs, we can simply train the student with Eq. (1) to capture the global structure of the hashcodes. We propose the *quantized similarity loss* by defining l_c such that:

$$l_c(u_{12}, u_{T12}) = l(u_{12}, W_{12}) \tag{9}$$

where $l(\cdot, \cdot)$ has the same form as Eq. (1). It should be noticed that Eq. (9) can be regarded as the ranking loss for the global data pairs in that similar representations produced by the teacher are more likely to be pseudo similar pairs. We rename $\mathcal{R}_u$ as $\mathcal{R}_{uq}$ if Eq. (9) is introduced.

Overall Training Loss The overall training loss is defined the same as Eq. (2), where $\mathcal{L}_s$ is defined in Eq. (1), and $\mathcal{R}_u$ can be regarded as the combination of *consistent similarity loss* and *quantized similarity loss* such that

$$\mathcal{R}_u = \mathcal{R}_{up} + \gamma\mathcal{R}_{uq} \tag{10}$$

As the teacher outputs in Eq. (10) lead to better representations and can model the pairwise information locally and globally, it is expected that the proposed loss can better meet the smoothness assumptions. Moreover, the hamming distances is accordant with the similarities on both labeled and unlabeled data.

Implementation and Relaxation Eq. (10) conveys that both the original and the perturbed samples should be fed into the network. For simplicity, we just regard the perturbed data as input, shown in Figure 1.

It is clear that directly optimizing Eq. (2) is intractable as the discrete constraints are involved. As used in most deep hashing algorithms [3, 28, 27], the simple and efficient way is removing the sgn function and adding the quantization loss. We reformulate the relaxed problem as follows

$$\min_F \mathcal{L} = \mathcal{L}_s^{(r)} + \omega\mathcal{R}_u^{(r)} + \eta\frac{1}{|\mathcal{X}|}\sum_{\mathbf{x}\in\mathcal{X}} \|\mathbf{h} - F(\tilde{\mathbf{x}}^{(1)})\|_1 \tag{11}$$

where $\mathbf{h} = \mathrm{sgn}(F(\tilde{\mathbf{x}}^{(1)}))$, $\mathcal{L}_s^{(r)}, \mathcal{R}_u^{(r)}$ is the relaxation of Eq. (1,10) respectively such that

$$\begin{aligned}\mathcal{L}_s^{(r)} &= \frac{1}{|\mathcal{S}|}\sum_{(i,j)\in\mathcal{S}} l(u_{ij}^r, s_{ij}) \\ \mathcal{R}_u^{(r)} &= \frac{1}{|\mathcal{X}|^2}\sum_{\mathbf{x}_1,\mathbf{x}_2\in\mathcal{X}} \left[(u_{12}^r - u_{T12}^r)^2 + \gamma l(u_{12}^r, W_{12})\right]\end{aligned} \tag{12}$$

For $\mathcal{L}_s^{(r)}$, we directly remove the sgn function to compute u_{ij}^r, and u_{ij}^r is defined the same as that in Eq. (1). For $\mathcal{R}_u^{(r)}$, we use $u_{12}^r = \mathrm{sim}(F(\tilde{\mathbf{x}_1}^{(1)}), F(\tilde{\mathbf{x}_2}^{(1)}))$, $u_{T12}^r = \mathrm{sim}(F_T(\tilde{\mathbf{x}_1}^{(2)}), F_T(\tilde{\mathbf{x}_2}^{(2)}))$, and $\mathrm{sim(s,t)} = -\|\frac{\mathbf{s}}{\|\mathbf{s}\|} - \frac{\mathbf{t}}{\|\mathbf{t}\|}\|^2$ where $\|\cdot\|$ is the L_2 normalization. The use of L_2 normalization is inspired by the original Mean Teacher where the consistent output is the normalized probabilities.

Algorithm 1 Mini-batch Training of PTS^{3}H

Require: Input data $\mathcal{X}$, pairwise labels $\mathcal{S}$, parameters $\omega(t), \eta, \gamma, \alpha$

for t in num-epochs **do**
  Determine the unsupervised weight $\omega = \omega(t)$
  for each mini-batch B **do**
    for $\mathbf{x}_i \in B$ **do**
      Sample two random perturbations $\tilde{\mathbf{x}_i}^{(1)}, \tilde{\mathbf{x}_i}^{(2)}$
    end for
    for $(\mathbf{x}_i, \mathbf{x}_j) \in B \times B$ **do**
      Compute W_{ij} by Eq. (8)
    end for
    Compute mini-batch version of $\mathcal{L}$ such that replacing $\mathcal{X}$ with B in Eq. (12)
    Update θ with optimizers, e.g. SGD
    Update θ_T with Eq. (5)
  end for
end for
return learned student and teacher networks

As a result, the consistent pairwise losses can capture both the local and global neighborhood structure. Moreover, semantic information can be embedded with supervised pairwise loss, and the real-valued space is able to be mapped into hamming space with the quantization loss.

3.3. Mini-batch Optimization

The training procedure is roughly the same as [23]. The teacher is updated by Eq. (5) each iteration, and the student is trained with back-propagation. We use the ramp-up procedure for both the learning rate and the regularization term $\omega = \omega(t)$ in the beginning of training. The training algorithm is summarized in Algorithm 1.

We mainly focus on training the student network. The student is trained by optimizing Eq. (11) with SGD. We randomly sample mini-batch to estimate the losses for each iteration, where we just compute the pairwise losses $\mathcal{L}_s^{(r)}, \mathcal{R}_u^{(r)}$ within each mini-batch B, and so as computing the pesudo similar pairs. The overall complexity of the loss just $O(a|B| + |B|^2)$ where a is the cost of network and $a > B$ if $|B|$ is small. It is clear that the network takes main computation time. To utilize both the labeled and unlabeled data, the ratio of number of labeled data pairs and unlabeled ones is constant in a mini-batch.

4. Experiments

In this section, we conduct various large-scale retrieval experiments to show the efficiency of the proposed PTS^{3}H method. We compare our PTS^{3}H method with recent state-of-the-art semi-supervised deep hashing methods on the retrieval performance. Some ablation studies and sensitivity of parameters are also discussed in this section.

4.1. Datasets and Evaluation Metrics

We run large-scale retrieval experiments on three image benchmarks: CIFAR-10[1], Nuswide[2] and ImageNet-100. CIFAR-10 consists of 60,000 32×32 color images from 10 object categories. ImageNet-100 is the subset of ImageNet dataset[3] with 100 randomly sampled classes. Nuswide dataset contains about 220K available images associating with 81 concept labels. Following [17], we only use the images associated with the 21 most frequent labels, where the total number of images is about 190K.

The experimental protocols are similar with [26]. In CIFAR-10 dataset, we randomly select 1,000 images (100 images per class) as query set, the rest 59,000 images as retrieval database where 5,000 images are selected as the training data. In Nuswide, we randomly select 2,100 images (100 images per class) as the query set and 10,500 images as the training set. In ImageNet-100, we use the same data split as HashNet [4] such that select 130 images per class for training, and regard all images in the selected classes from the validation set as the queries. The rest unlabeled data in the database are regarded as the unlabeled dataset.

As we just consider the pairwise similarity for training, the data pairs are constructed among the training data. For these dataset, similar images share at least one semantic label. The rest data pairs(pairs between unlabeled data and all the database) are regarded as the unlabeled pairs.

Our method is implemented with the PyTorch[4] framework. We adopt the pre-trained AlexNet [10] for deep hashing methods but replace the last fully-connected layer. The images are resized to 224×224 to train the network. For supervised pairwise loss in Eq. (1), we mainly use the DSH loss and DPSH loss and name them as **PTS^{3}H-DSH** and **PTS^{3}H-DPSH** respectively. SGD with momentum 0.9 is used for optimization, and the initial learning rate of the last layer is 10^{-2} which is ten times larger of the lower layers. The hyper-parameters ω, μ, α is different according to datasets, which are selected with the validation set. We first of all randomly select part of training data as validation set to determine the parameters. For CIFAR-10, we use $\{\omega = 0.8, \gamma = 0.5, \eta = 0.004\}$ with DSH loss and $\{\omega = 0.02, \gamma = 0.5, \eta = 0.01\}$ with DPSH loss; For Nuswide, we use $\{\omega = 0.8, \gamma = 0.1, \eta = 0.01\}$ with DSH loss and $\{\omega = 0.2, \gamma = 0.1, \eta = 0.01\}$ with DPSH loss. For ImageNet-100, we use $\{\omega = 0.5, \gamma = 0.1, \eta = 0.004\}$ with DSH loss and $\{\omega = 0.5, \gamma = 0.02, \eta = 0.004\}$ for DPSH loss. Following [23], we set $\alpha = 0.995$, and the ratio of number of unlabeled data pairs and labeled data

[1] http://www.cs.toronto.edu/~kriz/cifar.html
[2] http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm
[3] http://image-net.org
[4] http://pytorch.org/

Method	Net	CIFAR-10				Nuswide				ImageNet-100[4]			
		12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits	16 bits	32 bits	48 bits	64 bits
Semi-Supervised Hashing													
SSDH	VGG-F	0.801	0.813	0.812	0.814	0.773	0.779	0.778	0.778	-[1]	-	-	-
BGDH	VGG-F	**0.805**	0.824	0.826	0.833	**0.803**	0.818	0.822	0.828	-	-	-	-
PTS^{3}H-DSH	AlexNet	0.798 (+0.056)	**0.828** (+0.034)	**0.835** (+0.026)	**0.843** (+0.023)	0.752 (+0.012)	0.774 (+0.012)	0.783 (+0.019)	0.789 (+0.016)	**0.612** (+0.023)	**0.680** (+0.032)	**0.697** (+0.047)	**0.703** (+0.041)
PTS^{3}H-DPSH	AlexNet	0.789 (+0.038)	0.799 (+0.028)	0.801 (+0.025)	0.805 (+0.027)	**0.803** (+0.004)	**0.827** (+0.006)	**0.831** (+0.003)	**0.842** (+0.009)	0.397 (+0.018)	0.542 (+0.014)	0.618 (+0.027)	0.634 (+0.026)
Supervised Hashing Baselines													
DSH[2]	AlexNet	0.741	0.794	0.809	0.820	0.740	0.762	0.764	0.773	0.589	0.648	0.650	0.662
DPSH[2]	AlexNet	0.751	0.771	0.776	0.778	0.799	0.821	0.827	0.834	0.379	0.528	0.591	0.608
DSDH	VGG-F	0.740	0.786	0.801	0.820	0.776	0.808	0.820	0.829	-	-	-	-
DISH	AlexNet	0.758	0.784	0.799	0.791	0.787	0.810	0.810	0.813	-	-	-	-
HashNet[3]	AlexNet	0.686[3]	-	0.692	0.718	0.733[3]	-	0.755	0.762	0.502	0.622	0.661	0.682
DMDH[3]	AlexNet	0.704[3]	-	0.732	0.737	0.751[3]	-	0.781	0.789	0.513	0.612	0.673	0.692
MIHash	AlexNet	0.738	0.775	0.791	0.816	0.773	0.820	0.831	0.843	0.569	0.661	0.685	0.694

1: Results not available. 2: Our own implementation on the three datasets and most results are better than previously reported. 3: Results of HashNet, DMDH are referenced from DMDH [5]. 4: Results at 16 bits.

Table 1. Accuracy in terms of MAP for the semi-supervised and supervised hashing methods. The numbers in blankets are the relative gain compared with the baselines. Unless specified, the results are directly drawn from the original papers.

pairs within a minibatch is 15. The image perturbation strategy includes random resize, random cropping, random horizontal flipping, etc. The training is done on a server with two Intel(R) Xeon(R) E5-2683 v3@2.0GHz CPUs, 256GB RAM and a Geforce GTX TITAN Pascal with 12GB memory. We train 60 epochs for CIFAR-10, 20 epochs for Nuswide, and 240 epochs for ImageNet-100. We apply the teacher network to generate hashcodes.

Similar with [26, 4], for each retrieval dataset, we report the compared results in terms of *mean average precision*(MAP), precision at Hamming distance within 2, precision of top returned candidates. We calculate the MAP value within the top 5000 returned neighbors for NusWide and top 1000 for ImageNet-100, and report the MAP of all retrieved samples on CIFAR-10. Groundtruths are defined by whether two candidates are similar. We run each experiment for 5 times and get the average result.

4.2. Results

We compare our PTS^{3}H method with recent state-of-the-art deep hashing methods including SSDH [28], BGDH [27]. We do not take DSH-GANs [20] into consideration as it utilizes the label of each data point. Results on supervised hashing methods like DSH [15], DPSH [13], DSDH [12], DISH [29], DMDH [5] and MIHash [2] are also proposed for comparison. They follow similar settings, and the network used is either VGG-F or AlexNet, which share similar architectures. We report the result of supervised hashing in table 1 to show that DSH and DPSH are good supervised hashing algorithms, thus we regard the DSH and DPSH loss as the baselines of **PTS^{3}H-DSH** and **PTS^{3}H-DPSH** respectively. We also report the relative gains of the PTS^{3}H compared with the supervised baselines.

Retrieval results of different methods are shown in Table 1 and Figure 2. We re-implement the DSH and DPSH algorithms, and most results are better than previously reported. Note that the settings of Imagenet-100 are the same as that in [4]. Under the same setting, the proposed PTS^{3}H algorithm performs much better than the baselines by about 1-5 % on MAP and precision at Hamming distance within 2 value, which conveys that the proposed semi-supervised setting is able to capture more semantic information. Moreover, our semi-supervised algorithm achieves much better retrieval performance at most bits if proper supervised baselines are selected (DSH for CIFAR-10,ImageNet-100 and

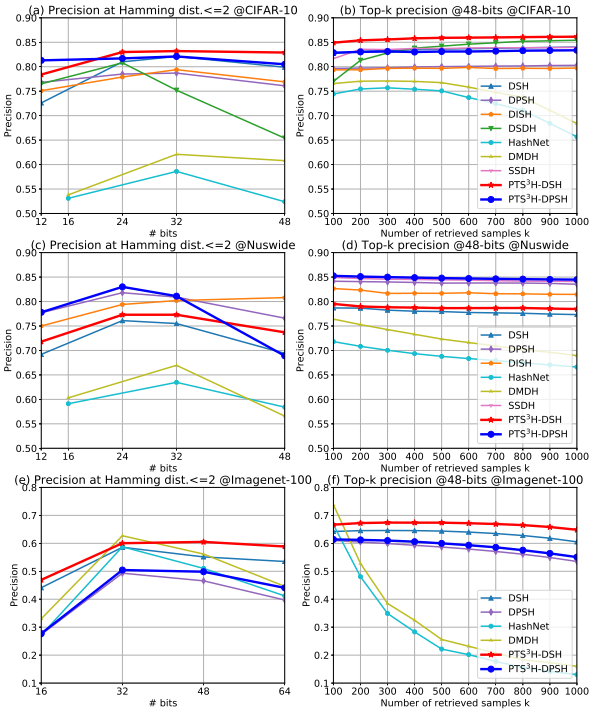

Figure 2. Precision at Hamming distance within 2 value and top-k precision curve of different deep hashing methods. AlexNet/VGG-F is used for pre-training in these algorithms.

Method	Dataset	MAP		Precision	
		32 bits	48 bits	32 bits	48 bits
PTS^{3}H-P	CIFAR-10	0.829	0.838	0.829	0.827
PTS^{3}H-Q		0.817	0.826	0.821	0.814
PTS^{3}H		**0.835**	**0.843**	0.832	0.829
PTS^{3}H-P	Nuswide	0.777	0.787	0.763	0.727
PTS^{3}H-Q		0.772	0.777	0.759	0.710
PTS^{3}H		0.782	**0.789**	0.770	0.737

Table 2. Results of the variants of the proposed PTS^{3}H algorithm on CIFAR-10 and Nuswide dataset. PTS^{3}H and PTS^{3}H-S are both proposed method but the codes are generated by the teacher and the student respectively. AlexNet is used for pre-training. Precision denotes the precision at Hamming distance within 2 value.

DPSH for Nuswide), showing the effectiveness of the proposed teacher-student architecture.

It should be noticed that the classification performance of VGG-F is slightly better than AlexNet, thus the hashing performance is expected not to decrease and may even be better if replacing AlexNet with VGG-F. Moreover, the proposed baselines are widely used but not the state-of-the-art, thus it is expected to achieve better results if adopting the state-of-the-art supervised hashing methods [5].

4.3. Ablation Study

In order to verify the effectiveness of our PTS^{3}H method, several variants are also considered. First we set $\gamma = 0$ to show the effectiveness of the $\mathcal{R}_{up}$, named PTS^{3}H-P. Then we remove $\mathcal{R}_{up}$ to show the effectiveness of $\mathcal{R}_{uq}$, denote PTS^{3}H-Q. The hyper-parameters of the variants are determined with the validation set. Retrieval results are shown in Table 2. The *consistent similarity loss* reaches about 70% performance gain as it produces consistent simialrities for smooth data pairs. The *quantized similarity loss* also achieves better performance as they model the pairwise similarities for perturbed inputs with global information. It should be noticed that there are little performance gain on MAP with the *quantized similarity loss* for Nuswide dataset, as the distribution of similar pairs underlying the dataset is a little complicated. Better results may achieved if better similarity construction strategy is involved.

4.4. Sensitivity to Parameters

In this section, the influence on different setting of the proposed PTS^{3}H is evaluated. The code length is 48 and we use DSH loss for evaluation. We do not report the influence on η as it has been discussed in the original papers [15, 13].

Influence of ω Figure 3(a)(b) shows the performance on different values of ω. It can be seen clearly that setting a certain ω achieves better hashing performance. It means that a proper consistent weight ω can arrive at better semi-supervised training.

Influence of γ Figure 3(c)(d) shows the performance on different values of γ. It should be noticed that a proper ω is set for different γ. There are some improvement for a proper

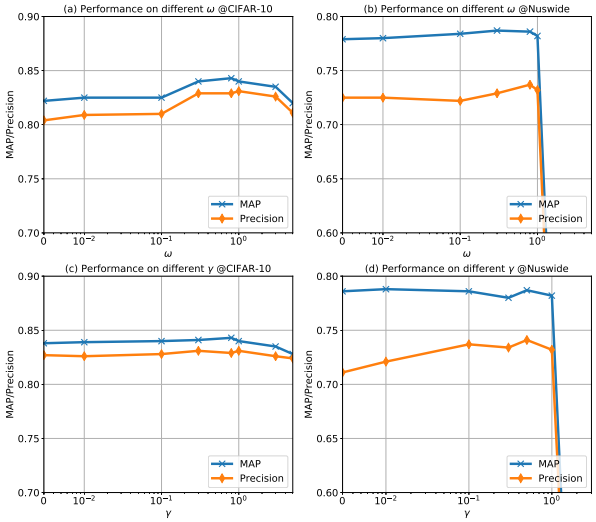

Figure 3. Comparative results of different hyper-parameters on CIFAR-10 and Nuswide dataset. The code length is 48. We use the DSH loss for training.

γ, especially the precision at Hamming distance within 2 value on Nuswide dataset. Similar as ω, a proper γ should be set for better performance.

5. Conclusion and Future Work

In this paper, we propose a novel semi-supervised hashing algorithm named PTS^{3}H in which the pairwise supervision and abundant unlabeled data are provided. The proposed PTS^{3}H is a teacher-student network architecture which is carefully designed for labeled and unlabeled pairs. We propose the general *consistent pairwise loss* in which the pairwise information generated by the teacher network guides the training of the student. There are two types of losses: *consistent similarity loss* models the locally pairwise information, and *quantized similarity loss* models the information globally by quantizing the similarities between samples. This procedure aims at generating similar retrieval results for neighborhood queries. Experiment shows that the proposed PTS^{3}H achieves great improvement over the baselines, and it is superior or comparable with the state-of-the-art semi-supervised hashing algorithms.

It should be noticed that we use the popular pairwise loss baselines and achieve good hashing results. As the proposed PTS^{3}H algorithm is a general framework for semi-supervised hashing, it is expected to arrive at better retrieval performance by incorporating the state-of-the-art supervised hashing algorithm with pairwise supervisions.

Acknowledgement

This work is supported by the National Key Research and Development Program of China (under Grant 2017YFA0700904), and the National Natural Science

Foundation of China (Nos. 61620106010 and 61571261).

References

[1] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research*, 7(Nov):2399–2434, 2006. 2

[2] F. Cakir, K. He, S. A. Bargal, and S. Sclaroff. Hashing with mutual information. *arXiv preprint arXiv:1803.00974*, 2018. 1, 6

[3] Y. Cao, M. Long, J. Wang, H. Zhu, and Q. Wen. Deep quantization network for efficient image retrieval. In *AAAI*, pages 3457–3463, 2016. 1, 2, 4

[4] Z. Cao, M. Long, J. Wang, and S. Y. Philip. Hashnet: Deep learning to hash by continuation. In *ICCV*, pages 5609–5618, 2017. 1, 5, 6

[5] Z. Chena, X. Yuana, J. Lua, Q. Tiand, and J. Zhoua. Deep hashing via discrepancy minimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6838–6847, 2018. 6, 7

[6] A. Gionis, P. Indyk, R. Motwani, et al. Similarity search in high dimensions via hashing. In *VLDB*, volume 99, pages 518–529, 1999. 1

[7] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(12):2916–2929, 2013. 1

[8] H. Jegou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):117–128, 2011. 2, 4

[9] A. Krause, P. Perona, and R. G. Gomes. Discriminative clustering by regularized information maximization. In *Advances in neural information processing systems*, pages 775–783, 2010. 2

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 5

[11] S. Laine and T. Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016. 1, 2, 3

[12] Q. Li, Z. Sun, R. He, and T. Tan. Deep supervised discrete hashing. *arXiv preprint arXiv:1705.10999*, 2017. 6

[13] W.-J. Li, S. Wang, and W.-C. Kang. Feature learning based deep supervised hashing with pairwise labels. In *IJCAI*, pages 1711–1717, 2016. 1, 2, 6, 7

[14] G. Lin, C. Shen, Q. Shi, A. van den Hengel, and D. Suter. Fast supervised hashing with decision trees for high-dimensional data. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1971–1978. IEEE, 2014. 1, 2

[15] H. Liu, R. Wang, S. Shan, and X. 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, 2, 6, 7

[16] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang. Supervised hashing with kernels. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2074–2081. IEEE, 2012. 1, 2

[17] W. Liu, J. Wang, S. Kumar, and S.-F. Chang. Hashing with graphs. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 1–8, 2011. 1, 5

[18] Y. Luo, J. Zhu, M. Li, Y. Ren, and B. Zhang. Smooth neighbors on teacher graphs for semi-supervised learning. *arXiv preprint arXiv:1711.00258*, 2017. 2, 3

[19] T. Miyato, S.-i. Maeda, S. Ishii, and M. Koyama. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 2018. 3

[20] Z. Qiu, Y. Pan, T. Yao, and T. Mei. Deep semantic hashing with generative adversarial networks. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 225–234. ACM, 2017. 1, 2, 6

[21] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016. 2

[22] F. Shen, C. Shen, W. Liu, and H. Tao Shen. Supervised discrete hashing. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 1

[23] A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*, pages 1195–1204, 2017. 1, 2, 3, 4, 5

[24] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for large-scale search. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(12):2393–2406, 2012. 2

[25] J. Weston, F. Ratle, H. Mobahi, and R. Collobert. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*, pages 639–655. Springer, 2012. 2

[26] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan. Supervised hashing for image retrieval via image representation learning. In *Proceedings of the AAAI Conference on Artificial Intellignece*, pages 2156–2162, 2014. 1, 5, 6

[27] X. Yan, L. Zhang, and W.-J. Li. Semi-supervised deep hashing with a bipartite graph. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 3238–3244. AAAI Press, 2017. 1, 2, 4, 6

[28] J. Zhang and Y. Peng. Ssdh: semi-supervised deep hashing for large scale image retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 2017. 1, 2, 4, 6

[29] S. Zhang, J. Li, M. Jiang, and B. Zhang. Scalable discrete supervised multimedia hash learning with clustering. *IEEE Transactions on Circuits and Systems for Video Technology*, 2017. 1, 6

[30] X. Zhu, Z. Ghahramani, and J. D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pages 912–919, 2003. 1, 2