Semisupervised Hashing via Kernel Hyperplane Learning for Scalable Image Search

Meina Kan, Dong Xu, *Senior Member, IEEE,* Shiguang Shan, *Member, IEEE,* and Xilin Chen, *Senior Member, IEEE*

***Abstract*—Hashing methods that aim to seek a compact binary code for each image are demonstrated to be efficient for scalable content-based image retrieval. In this paper, we propose a new hashing method called semisupervised kernel hyperplane learning (SKHL) for semantic image retrieval by modeling each hashing function as a nonlinear kernel hyperplane constructed from an unlabeled dataset. Moreover, a Fisher-like criterion is proposed to learn the optimal kernel hyperplanes and hashing functions, using only weakly labeled training samples with side information. To further integrate different types of features, we also incorporate multiple kernel learning (MKL) into the proposed SKHL (called SKHL-MKL), leading to better hashing functions. Comprehensive experiments on CIFAR-100 and NUS-WIDE datasets demonstrate the effectiveness of our SKHL and SKHL-MKL.**

***Index Terms*—Kernel hyperplane learning, multiple kernel learning (MKL), semisupervised hashing.**

I. Introduction

With the explosive growth of images and videos on the Internet, there is an increasing research interest in developing new methods for efficiently organizing and retrieving large-scale multimedia data (i.e., images and videos) based on their semantic contents [1]–[5]. Usually, searching the nearest neighbors for a query sample is $O(n)$ time complexity since all the n samples in the target database have to be scanned. This is nevertheless time consuming when n becomes very large. To address this problem, many approaches have been proposed for efficient image and video retrieval. Among them, the recently proposed hashing methods have achieved promising performances by generating a compact binary code. Various hashing methods have been proposed for large-scale image retrieval [2], [6], [7], which are also applicable for video retrieval because these techniques can be readily used for video key frames. The existing hashing methods can be roughly divided into three categories: unsupervised methods, supervised methods, and semisupervised methods. In the following, we briefly review the existing methods.

The unsupervised hashing methods employ only the unlabeled data for generating the binary code. Many methods belong to this category, such as locality sensitive hashing (LSH) [1], [2], [6], [8], spectral hashing (SH) [7], anchor graph hashing (AGH) [9], binary reconstructive embeddings (BRE) [10], iterative quantization (ITQ) [11], [12], and so on. In LSH, the hashing functions are randomly generated to preserve the locality of samples in the binary Hamming space [1], [2] or Euclidean space [8]. Later, LSH is extended to kernel locality sensitive hashing (KLSH) [13] and shift invariant kernel hashing (SKLSH) [14]. To further improve the performance, some data-dependent approaches were recently proposed, such as SH [7], AGH [9], BRE [10], and ITQ [11]. In SH [7], the Euclidean distance between the samples is embedded in the objective function by using Hamming distance. In [9], a graph-based hashing method is proposed to automatically discover the low-dimensional neighborhood structure on a manifold by utilizing an anchor graph to calculate the adjacency matrices. In [10], the hashing functions are developed by minimizing the reconstruction error between the original distance (e.g., Euclidean distance) and the Hamming distance. In [15], an angular quantization hashing method is proposed for nonnegative data with the angle between the vectors as the similarity measure. Gong *et al.* [11], [12] proposed an ITQ method to find a rotation for low dimensional data after dimension reduction. Most existing hashing techniques employ the hyperplane hashing functions. Heo *et al.* [16] proposed a new spherical hashing method by employing the hypersphere-based hashing functions.

Without using the label information, the hashing functions obtained by using the above unsupervised methods might not be distinctive enough for semantic image and video retrieval. Therefore, a few supervised methods, such as [17]–[19], are recently proposed for semantic image searches. In [19], a deep neural network together with Restricted Boltzman Machines (RBMs) is employed to learn the hashing functions for high dimensional data. The ITQ method can also work in the supervised scenario by using Canonical Correlation Analysis (CCA) [21] before conducting the rotation [11], [12]. Based on the latent structural SVM framework, a hinge loss-like hashing function is proposed to integrate the Hamming distance

Manuscript received December 18, 2012; revised May 24, 2013; accepted June 17, 2013. Date of publication August 6, 2013; date of current version April 2, 2014. This work was supported in part by the Singapore MoE Tier 2 Project (ARC42/13), and in part by the Natural Science Foundation of China under Contracts 61025010, 61222211, and 61272321. This paper was recommended by Associate Editor T. Zhang.

M. Kan, S. Shan, and X. Chen are with the Key Laboratory of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing 100190, China (e-mail: meina.kan@vipl.ict.ac.cn; shiguang.shan@vipl.ict.ac.cn; xilin.chen@vipl.ict.ac.cn).

D. Xu is with the School of Computer Engineering, Nanyang Technological University, Singapore 639798 (e-mail: dongxu@ntu.edu.sg).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2013.2276713

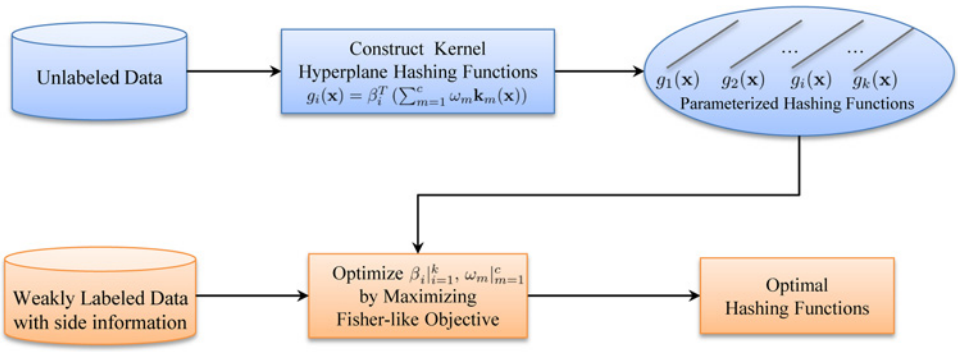

Fig. 1. Overview of the proposed semisupervised hashing method via kernel hyperplane learning (SKHL). In SKHL, each hashing function $h_i(\mathbf{x})$ (based on $g_i(\mathbf{x})$) is modeled as a nonlinear kernel hyperplane, which is constructed from an unlabeled dataset, and parameterized by $\boldsymbol{\beta}_i$. Based on these parameterized hashing functions, the hashing codes of the samples from the weakly labeled data set are used to formulate a Fisher-like objective function, which is maximized to learn the optimal hashing functions.

and binary quantization in [22], which directly minimizes a piecewise linear upper bound on the empirical loss of the binary code, to avoid the relaxation from binary solutions to continuous solutions as used in most existing hashing methods. In [23], a kernel-based supervised hashing method is proposed to minimize the Hamming distances from similar pairs and maximize the Hamming distances from dissimilar pairs. Instead of using the Hamming distance as in most existing hashing methods, in [23] the inner product between the codes is utilized according to the equivalence between the inner product and Hamming distance, and the sigmoid function is additionally employed to smooth the binary codes-based function. In [24], a Hamming distance metric learning framework is proposed to optimize a hashing function with a ranking loss measuring the relative similarity from the triplet of binary codes. However, collecting labeled data is time consuming and expensive. However, it is common that only limited labeled data is available for these supervised hashing methods, which may degrade the retrieval performances of these methods.

To make use of the huge amount of unlabeled data, Wang *et al.* [25] recently proposed a semisupervised hashing method, which simultaneously minimizes the empirical error on the labeled data and maximizes the variance and independence of hashing codes over the labeled and unlabeled data. The BRE method [10] can also be used in the semisupervised scenario by setting the target distance of similar pair to be large. Mu *et al.* [20] proposed a weakly supervised hashing method by exploiting the so-called side information, i.e., whether a pair of images are from the same class. In this method, an SVM-like approach is proposed to maximize the margin between the hashing-induced partitions, which is also applicable in unsupervised and supervised settings.

In this paper, we propose a semisupervised hashing method via kernel hyperplane learning (SKHL) for large-scale semantic image retrieval. In our method, we model each hashing function as a nonlinear kernel hyperplane learnt from an unlabeled dataset. The optimal hashing functions are obtained by maximizing the Fisher-like criterion on a weakly labeled dataset in which only side information is available. An overview of our approach is depicted in Fig. 1. To further improve SKHL, multiple types of features are integrated by using multiple kernel learning (MKL) to learn a better hashing function. The optimization problem is relaxed from the binary solutions to the continuous solutions such that we can develop an iterative approach by utilizing the sparse linear discriminant analysis (SLDA) [26] and semi-definite programming (SDP) [42]. Extensive experimental evaluations demonstrate the superiority of our method over existing hashing algorithms. The similar idea of using an SVM to model each prototype hyperplane is also employed in face recognition [43]. However, [43] is designed for linear feature extraction based on a single type of feature, while our work is designed for learning binary codes with multiple types of features nonlinearly.

The reminder of this paper is organized as follows. Section II presents our proposed SKHL in details. Section III evaluates the proposed method and existing methods on the CIFAR-100 [27] and NUS-WIDE [28] datasets. Finally, we give the conclusion and future works in Section IV.

II. Semisupervised Hashing via Kernel Hyperplane Learning (SKHL)

The overall framework of the proposed SKHL is shown in Fig. 1. In SKHL, each hashing function is modeled as a nonlinear kernel hyperplane, which is constructed from an unlabeled dataset (parameterized by $\boldsymbol{\beta}_i$). Then, based on these parameterized hashing functions, the hashing codes of the weakly labeled samples with side information are used to formulate a Fisher-like objective function. Finally, optimizing the objective can achieve the optimal hashing functions.

In this section, we first describe the formulation of the SKHL method and further present how to extend SKHL by exploiting multiple types of features through MKL. Finally, we give a detailed description of the optimization of our objective function.

A. Notations

The hashing function is a transformation to generate a compact code for quickly searching in large collection of data. Like most hashing methods mentioned earlier, each hashing

function $h_i(\mathbf{x})$ is designed to map a sample $\mathbf{x}$ into a binary code based on a function $f_i(\mathbf{x})$

$$h_i(\mathbf{x}) = \begin{cases} 1 & \text{if } f_i(\mathbf{x}) > \tau_i \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

where τ_i is a threshold and it will be discussed later. So, given k hashing functions $\{h_i(\mathbf{x})|i = 1, \cdots, k\}$, $\mathbf{x}$ can be encoded by k binary bits denoted as $\mathbf{h}(\mathbf{x})$

$$\mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x}), h_2(\mathbf{x}), \cdots, h_k(\mathbf{x})]^T \tag{2}$$

and the corresponding real-valued magnitude code is denoted as $\mathbf{f}(\mathbf{x})$

$$\mathbf{f}(\mathbf{x}) = \left[f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_k(\mathbf{x})\right]^T. \tag{3}$$

Let us denote an unlabeled dataset $\mathcal{X}^u$ with the data matrix represented as $\mathbf{X}^u = [\mathbf{x}_1^u, \mathbf{x}_2^u, \cdots, \mathbf{x}_N^u] \in R^{p\times N}$, where p is the feature dimension and N is the total number of samples in $\mathcal{X}^u$. A weakly labeled dataset with side information is also defined, which consists of two subsets $\mathcal{X}^0$ and $\mathcal{X}^1$: $\mathcal{X}^0 = \left\{\left(\mathbf{x}_1^0, \mathbf{z}_1^0\right), \left(\mathbf{x}_2^0, \mathbf{z}_2^0\right), \cdots, \left(\mathbf{x}_{M_0}^0, \mathbf{z}_{M_0}^0\right)\right\}$, where $(\mathbf{x}_i^0, \mathbf{z}_i^0)$ is a pair of images from different classes or not considered as nearest neighbors and M_0 is the total number of pairs in $\mathcal{X}^0$; $\mathcal{X}^1 = \left\{\left(\mathbf{x}_1^1, \mathbf{z}_1^1\right), \left(\mathbf{x}_2^1, \mathbf{z}_2^1\right), \cdots, \left(\mathbf{x}_{M_1}^1, \mathbf{z}_{M_1}^1\right)\right\}$, where $(\mathbf{x}_i^1, \mathbf{z}_i^1)$ represents a pair of images from the same class or considered as nearest neighbors and M_1 is the total number of pairs in $\mathcal{X}^1$. This side information is a type of weakly labeled information since only the information that whether a pair of images are from the same class is known, but the exact class label for each sample is unknown. We also denote all the training data as $\mathcal{X} = \mathcal{X}^u \cup \mathcal{X}^0 \cup \mathcal{X}^1$, which can also be represented as a data matrix $\mathbf{X} = \left[\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{\tilde{N}}\right] \in R^{p\times\tilde{N}}$ with $\tilde{N} = N + M_0 + M_1$ being the total number of training samples.

B. Semisupervised Hashing via Kernel Hyperplane Learning

The real world data from the large-scale image or video datasets, usually form a nonlinear distribution. Therefore, following the Representer theorem [29], we model each function $f_i(\mathbf{x})$ as a nonlinear kernel hyperplane, which can be constructed from the unlabeled dataset $\mathcal{X}^u$

$$\begin{aligned} f_i(\mathbf{x}) &= \sum\nolimits_{j=1}^{N} \alpha_{ij} y_{ij} < \boldsymbol{\phi}(\mathbf{x}_j^u), \boldsymbol{\phi}(\mathbf{x}) > \\ &= \sum\nolimits_{j=1}^{N} \alpha_{ij} y_{ij} k(\mathbf{x}_j^u, \mathbf{x}), \quad \mathbf{x}_j^u \in \mathcal{X}^u \end{aligned} \tag{4}$$

where $<\cdot,\cdot>$ represents the inner product and $k(\mathbf{x}, \mathbf{z})$ is the kernel function related to the transformation $\boldsymbol{\phi}$, i.e., $k(\mathbf{x}, \mathbf{z}) =< \boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{z}) >$, α_{ij} and y_{ij} are the combination coefficients and unknown class label of $\mathbf{x}_j^u$ for $f_i(\mathbf{x})$, respectively. Following the work in [23], the bias term τ_i in (1) is set as the mean of $\{f_i(\mathbf{x}), \mathbf{x} \in \mathcal{X}\}$. As $\alpha_{ij}y_{ij}$ is enough for determining $f_i(\mathbf{x})$, we merge them as one linear combination coefficient

$$\beta_{ij} = \alpha_{ij} y_{ij} \tag{5}$$

then (4) can be reformulated as

$$f_i(\mathbf{x}) = \sum\nolimits_{j=1}^{N} \beta_{ij} k(\mathbf{x}_j^u, \mathbf{x}) = \boldsymbol{\beta}_i^T \mathbf{k}(\mathbf{x}) \tag{6}$$

with

$$\mathbf{k}(\mathbf{x}) = [k(\mathbf{x}_1^u, \mathbf{x}), k(\mathbf{x}_2^u, \mathbf{x}), \cdots, k(\mathbf{x}_N^u, \mathbf{x})]^T \tag{7}$$

$$\boldsymbol{\beta}_i = [\beta_{i1}, \beta_{i2}, \cdots, \beta_{iN}]^T. \tag{8}$$

Using (6), $\mathbf{f}(\mathbf{x})$ in (3) can be reformulated as

$$\mathbf{f}(\mathbf{x}) = \left[\boldsymbol{\beta}_1^T \mathbf{k}(\mathbf{x}), \cdots, \boldsymbol{\beta}_k^T \mathbf{k}(\mathbf{x})\right]^T = \mathbf{B}^T \mathbf{k}(\mathbf{x}) \tag{9}$$

where $\mathbf{B} = [\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \cdots, \boldsymbol{\beta}_k] \in R^{N\times k}$ is a matrix containing all the linear combination coefficients.

As $f_i(\mathbf{x})$ is designed for semantic retrieval, the samples from the same class are expected to lie on the same side of a hyperplane, while the samples from different classes are expected to lie on different sides of the hyperplane. Therefore, each hyperplane is expected to lie on the margin between two classes, which means the kernel hyperplane is only determined by those support vectors near the margin of two classes. In other words, only a small fraction of β_{ij} corresponding to those $\mathbf{x}_j^u$ near the margin of two classes are nonzeros, while most β_{ij} corresponding to those $\mathbf{x}_j^u$ that are far away from the margin (i.e., not support vectors) are zero. Therefore, we can formulate a constraint on $\boldsymbol{\beta}_i$ by using the L_0-norm. Considering that it is difficult to solve the L_0-norm related problem, we employ the L_1-norm in this paper, namely

$$|\boldsymbol{\beta}_i|_1 \le t \tag{10}$$

where the parameter t controls the sparsity of $\boldsymbol{\beta}_i$. Although t for each $\boldsymbol{\beta}_i$ can be different, we use the same t for simplicity in this paper.

For semantic image retrieval, a good hashing function is expected to be discriminative, i.e., if two samples $\mathbf{x}$ and $\mathbf{z}$ are from the same semantic class, the distance between $\mathbf{h}(\mathbf{x})$ and $\mathbf{h}(\mathbf{z})$ should be small; otherwise, the distance should be large. Therefore, to obtain semantic hashing functions $\mathbf{h}$, a Fisher-like criterion is exploited to measure the discriminative capability of the hashing functions on the weakly labeled dataset, i.e., $\mathcal{X}^1$ and $\mathcal{X}^0$

$$\begin{aligned} \mathbf{B}^* = \arg\max_{\mathbf{B}} & \frac{\sum_{i=1}^{M_0} \|\mathbf{h}(\mathbf{x}_i^0) - \mathbf{h}(\mathbf{z}_i^0)\|^2}{\sum_{i=1}^{M_1} \|\mathbf{h}(\mathbf{x}_i^1) - \mathbf{h}(\mathbf{z}_i^1)\|^2} \\ & s.t. \quad |\boldsymbol{\beta}_i|_1 \le t, i = 1, \cdots, k. \end{aligned} \tag{11}$$

Following the work in [25], the maximum variance criterion is employed to enforce the hashing bucket balance constraint. Let us denote the variance of the binary codes from all samples, i.e., $\{h_i(\mathbf{x}), \mathbf{x} \in \mathcal{X}\})$ as $\mathrm{var}(h_i(\mathbf{x}))$. Based on (11) and the maximum variance criterion, we then propose a new objective function as

$$\begin{aligned} \mathbf{B}^* &= \arg\max_{\mathbf{B}} G(\mathbf{B}) \\ &= \arg\max_{\mathbf{B}} \frac{\sum_{i=1}^{M_0} \|\mathbf{h}(\mathbf{x}_i^0) - \mathbf{h}(\mathbf{z}_i^0)\|^2 + \gamma\tilde{N} \sum_{i=1}^{k} \mathrm{var}(h_i(\mathbf{x}))}{\sum_{i=1}^{M_1} \|\mathbf{h}(\mathbf{x}_i^1) - \mathbf{h}(\mathbf{z}_i^1)\|^2} \\ & s.t. \quad |\boldsymbol{\beta}_i|_1 \le t, i = 1, \cdots, k \end{aligned} \tag{12}$$

where γ is a tradeoff parameter. This optimization problem is NP-hard, because the solutions are constrained to be binary values. According to [25], $\sum_{i=1}^{k} \mathrm{var}(h_i(\mathbf{x}))$, i.e., the variance of binary codes, can be relaxed to the variance of real value

codes as $\text{var}(\mathbf{f}(\mathbf{x})) = \frac{1}{\tilde{N}}\text{Tr}\left(\mathbf{B}^T\tilde{\mathbf{K}}^t\tilde{\mathbf{K}}^{tT}\mathbf{B}\right)$, with $\tilde{\mathbf{K}}^t \in R^{N\times\tilde{N}}$ defined as

$$\tilde{\mathbf{K}}^t = \left[\mathbf{k}(\mathbf{x}_1) - \bar{\mathbf{k}}^t, \mathbf{k}(\mathbf{x}_2) - \bar{\mathbf{k}}^t, \cdots, \mathbf{k}(\mathbf{x}_{\tilde{N}}) - \bar{\mathbf{k}}^t\right] \tag{13}$$

where

$$\bar{\mathbf{k}}^t = \frac{1}{\tilde{N}}\sum\nolimits_{i=1}^{\tilde{N}} \mathbf{k}(\mathbf{x}_i),\ \ \mathbf{x}_i \in \mathcal{X}. \tag{14}$$

As in the existing work [7], [25], we relax $G(\mathbf{B})$ as $G'(\mathbf{B})$ by allowing the solutions to be real values as

$$\begin{aligned} G'(\mathbf{B}) &= \frac{\sum_{i=1}^{M_0}\|\mathbf{f}(\mathbf{x}_i^0)-\mathbf{f}(\mathbf{z}_i^0)\|^2 + \gamma\tilde{N}\text{var}(\mathbf{f}(\mathbf{x}))}{\sum_{i=1}^{M_1}\|\mathbf{f}(\mathbf{x}_i^1)-\mathbf{f}(\mathbf{z}_i^1)\|^2} \\ &= \frac{\sum_{i=1}^{M_0}\|\mathbf{B}^T\mathbf{k}(\mathbf{x}_i^0)-\mathbf{B}^T\mathbf{k}(\mathbf{z}_i^0)\|^2 + \gamma\text{Tr}\left(\mathbf{B}^T\tilde{\mathbf{K}}^t\tilde{\mathbf{K}}^{tT}\mathbf{B}\right)}{\sum_{i=1}^{M_1}\|\mathbf{B}^T\mathbf{k}(\mathbf{x}_i^1)-\mathbf{B}^T\mathbf{k}(\mathbf{z}_i^1)\|^2} \\ &= \frac{\text{Tr}\left(\mathbf{B}^T\mathbf{K}^0\mathbf{K}^{0T}\mathbf{B} + \gamma\mathbf{B}^T\tilde{\mathbf{K}}^t\tilde{\mathbf{K}}^{tT}\mathbf{B}\right)}{\text{Tr}\left(\mathbf{B}^T\mathbf{K}^1\mathbf{K}^{1T}\mathbf{B}\right)} \\ &s.t.\ \ |\boldsymbol{\beta}_i|_1 \le t, i = 1, \cdots, k \end{aligned} \tag{15}$$

where $\mathbf{K}^0 \in R^{N\times M_0}$ and $\mathbf{K}^1 \in R^{N\times M_1}$ are, respectively, defined as

$$\mathbf{K}^0 = \left[\mathbf{k}(\mathbf{x}_1^0)-\mathbf{k}(\mathbf{z}_1^0), \mathbf{k}(\mathbf{x}_2^0)-\mathbf{k}(\mathbf{z}_2^0), \cdots, \mathbf{k}(\mathbf{x}_{M_0}^0)-\mathbf{k}(\mathbf{z}_{M_0}^0)\right] \tag{16}$$

$$\mathbf{K}^1 = \left[\mathbf{k}(\mathbf{x}_1^1)-\mathbf{k}(\mathbf{z}_1^1), \mathbf{k}(\mathbf{x}_2^1)-\mathbf{k}(\mathbf{z}_2^1), \cdots, \mathbf{k}(\mathbf{x}_{M_1}^1)-\mathbf{k}(\mathbf{z}_{M_1}^1)\right]. \tag{17}$$

With the relaxed objective function $G'(\mathbf{B})$, the optimal $\mathbf{B}^*$ can be obtained as

$$\begin{aligned} \mathbf{B}^* &= \arg\max_{\mathbf{B}} \frac{\text{Tr}\left(\mathbf{B}^T\left(\mathbf{K}^0\mathbf{K}^{0T} + \gamma\tilde{\mathbf{K}}^t\tilde{\mathbf{K}}^{tT}\right)\mathbf{B}\right)}{\text{Tr}\left(\mathbf{B}^T\mathbf{K}^1\mathbf{K}^{1T}\mathbf{B}\right)}. \\ &s.t.\ \ |\boldsymbol{\beta}_i|_1 \le t, i = 1, \cdots, k. \end{aligned} \tag{18}$$

According to [30], the objective function in (18) is in the form of trace ratio, so the closed form solution does not exist. We therefore reformulate it into a more tractable one in the form of ratio trace as

$$\begin{aligned} &\mathbf{B}^* = \\ &\arg\max_{\mathbf{B}} \text{Tr}\left(\left(\mathbf{B}^T\mathbf{K}^1\mathbf{K}^{1T}\mathbf{B}\right)^{-1}\left(\mathbf{B}^T\left(\mathbf{K}^0\mathbf{K}^{0T} + \gamma\tilde{\mathbf{K}}^t\tilde{\mathbf{K}}^{tT}\right)\mathbf{B}\right)\right) \\ &s.t.\ \ |\boldsymbol{\beta}_i|_1 \le t, i = 1, \cdots, k. \end{aligned} \tag{19}$$

After obtaining the optimal $\mathbf{B}^* = [\boldsymbol{\beta}_1^*, \boldsymbol{\beta}_2^*, \cdots, \boldsymbol{\beta}_k^*]$, the optimal hashing function $f_i^*(\mathbf{x})$, can thus be obtained as

$$f_i^*(\mathbf{x}) = \boldsymbol{\beta}_i^{*T}\mathbf{k}(\mathbf{x}). \tag{20}$$

C. Semisupervised Hashing via Kernel Hyperplane Learning With Multiple Kernel Learning

Most of the existing methods exploit a single type of feature. However, usually it is not discriminative enough for large-scale semantic image and video searches. Therefore, in this paper, we employ multiple types of features to obtain better hashing functions, i.e., each hashing function is obtained by binarizing the sum of the outputs from multiple hyperplanes corresponding to multiple types of features

$$h_i(\mathbf{x}) = \begin{cases} 1 & \text{if } g_i(\mathbf{x}) = \sum_{m=1}^{c}\omega_m f_{mi}(\mathbf{x}) > \tau_i \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

where c is the number of different types of features. The different types of features are actually extracted from the same image; thus, these features can be considered as different views of each sample. Therefore, the spaces from different views of features roughly have the same structure, which means that the optimal hashing hyperplanes from different views also roughly have the same structure, i.e., the same support vectors and the same coefficients. Therefore, we can use the same support vectors (with different feature representations) and the same coefficients to represent the optimal hashing hyperplane. As a result, $g_i(\mathbf{x})$ can be reformulated as

$$g_i(\mathbf{x}) = \boldsymbol{\beta}_i^T\left(\sum\nolimits_{m=1}^{c}\omega_m\mathbf{k}_m(\mathbf{x})\right) \tag{22}$$

where $\mathbf{k}_m(\mathbf{x})$ is computed from the m^{th} type of feature according to (7), and ω_m is the weight for the m^{th} type of feature. Thus, (19) can be reformulated as

$$\begin{aligned} \mathbf{B}^* = &\arg\max_{\mathbf{B}} \text{Tr}\left(\left(\mathbf{B}^T\mathbf{S}_w\mathbf{B}\right)^{-1}\left(\mathbf{B}^T\mathbf{S}_b\mathbf{B}\right)\right) \\ &s.t. \sum_{m=1}^{c}\omega_m = 1, \omega_m \ge 0, m = 1, \cdots, c \\ &|\boldsymbol{\beta}_i|_1 \le t, i = 1, \cdots, k \end{aligned} \tag{23}$$

with

$$\begin{aligned} \mathbf{S}_w &= \left(\sum\nolimits_{m=1}^{c}\omega_m\mathbf{K}_m^1\right)\left(\sum\nolimits_{m=1}^{c}\omega_m\mathbf{K}_m^1\right)^T \\ \mathbf{S}_b &= \left(\sum_{m=1}^{c}\omega_m\mathbf{K}_m^0\right)\left(\sum_{m=1}^{c}\omega_m\mathbf{K}_m^0\right)^T + \\ &\gamma\left(\sum_{m=1}^{c}\omega_m\mathbf{K}_m^t - \tilde{\mathbf{k}}^t\mathbf{1}_{1\times\tilde{N}}\right)\left(\sum_{m=1}^{c}\omega_m\mathbf{K}_m^t - \tilde{\mathbf{k}}^t\mathbf{1}_{1\times\tilde{N}}\right)^T. \end{aligned} \tag{24}$$

The kernel matrices $\mathbf{K}_m^0$ and $\mathbf{K}_m^1$ are calculated from the m^{th} type of feature according to (16) and (17). The kernel matrix $\mathbf{K}_m^t$ is also calculated from the m^{th} type of feature as $\mathbf{K}_m^t = \left[\mathbf{k}_m(\mathbf{x}_1), \mathbf{k}_m(\mathbf{x}_2), \cdots, \mathbf{k}_m(\mathbf{x}_{\tilde{N}})\right]$. The mean vector $\tilde{\mathbf{k}}^t$ is defined as $\tilde{\mathbf{k}}^t = \frac{1}{\tilde{N}}\sum_{i=1}^{\tilde{N}}\sum_{m=1}^{c}\omega_m\mathbf{k}_m(\mathbf{x}_i),\ \ \mathbf{x}_i \in \mathcal{X}$.

D. Optimization

It is difficult to solve the objective function in (23) directly since it is nonconvex. Following MKL-DR [31], we iteratively optimize $\mathbf{B}$ and $\boldsymbol{\omega}$ ($\boldsymbol{\omega} = [\omega_1, \omega_2, \cdots, \omega_c]^T$) as shown in the algorithm in Table I. The details of the procedure are given on the next page.

Step 1: given $\boldsymbol{\omega}$, optimize B. We notice that, when fixing $\boldsymbol{\omega}$, the objective function (23) is similar to that of the SLDA [26]. The main difference is that we enforce the coefficients corresponding to the support vectors (rather than the final projection coefficients as in SLDA) to be sparse. Following [26], we introduce an intermediate variable $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_k] \in R^{N\times k}$

TABLE I
ALGORITHM OF SEMISUPERVISED HASHING VIA KERNEL HYPERPLANE LEARNING WITH MKL

Semi-supervised Hashing via Kernel Hyperplane Learning with MKL	
Input:	Unlabeled dataset $\mathcal{X}^u$; the number of hashing functions k; the weakly labeled dataset consisting of $\mathcal{X}^0$ and $\mathcal{X}^1$.
Output:	Coefficients $\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \cdots, \boldsymbol{\beta}_k$ for k hashing functions and the weights ω_m for each type of feature.
Initial:	Initialize $\mathbf{A}$, $\mathbf{B}$ and $\boldsymbol{\omega}$ with all the entries as 1; Calculate $\mathbf{K}_m^0$, $\mathbf{K}_m^1$, $\mathbf{H}_b$ and $\mathbf{R}_w$ according to (16), (17), (26) and (27).
Repeat	
Step 1:	Given $\boldsymbol{\omega}$, optimize $\mathbf{B}$
	Repeat
	1) Given $\mathbf{A}$, solve k independent Lasso problems $\min_{\boldsymbol{\beta}_i} \|\tilde{\mathbf{y}}_i - \tilde{\mathbf{W}}\boldsymbol{\beta}_i\|^2, s.t. \|\boldsymbol{\beta}_i\|_1 \le t, i = 1, \cdots, k.$
	2) Given $\mathbf{B}$, $\mathbf{A}$ can be obtained by using SVD $\mathbf{R}_w^{-T}\left(\mathbf{H}_b^T\mathbf{H}_b\right)\mathbf{B} = \mathbf{U}\mathbf{D}\mathbf{V}^T, \mathbf{A} = \mathbf{U}_{N\times k}\mathbf{V}_{k\times k}^T$
	Until: the changes of $\mathbf{B}$ from two successive iterations is smaller than ε.
Step 2:	Given $\mathbf{B}$, optimize $\boldsymbol{\omega}$ by using SDP $\min_{\boldsymbol{\omega}} \boldsymbol{\omega}^T\mathbf{S}_w^{\mathbf{A}}\boldsymbol{\omega}, s.t. \boldsymbol{\omega}^T\mathbf{S}_b^{\mathbf{A}}\boldsymbol{\omega} = 1, \sum_{m=1}^{c}\omega_m = 1, \omega_m \ge 0.$
Until:	the changes of $\mathbf{B}$ and $\boldsymbol{\omega}$ from two successive iterations are smaller than ε.

and equivalently reformulate the objective function in (23) as a regression problem. (Please refer to [26] for more details.)

$$\begin{aligned}\min_{\mathbf{A},\mathbf{B}} \sum_{i=1}^{k} \left\{\|\mathbf{H}_b\mathbf{R}_w^{-1}\mathbf{a}_i - \mathbf{H}_b\boldsymbol{\beta}_i\|^2 + \lambda\boldsymbol{\beta}_i^T\mathbf{S}_w\boldsymbol{\beta}_i\right\} \\ s.t.\ |\boldsymbol{\beta}_i|_1 \le t, i = 1, \cdots, k, \quad \text{and } \mathbf{A}^T\mathbf{A} = \mathbf{I}_{k\times k}\end{aligned} \tag{25}$$

where $\mathbf{H}_b \in R^{(M_0+\tilde{N})\times N}$ and $\mathbf{R}_w \in R^{N\times N}$ are defined as

$$\mathbf{H}_b = \begin{bmatrix} \left(\sum_{m=1}^{c}\omega_m\mathbf{K}_m^0\right)^T \\ \sqrt{\gamma}\left(\sum_{m=1}^{c}\omega_m\mathbf{K}_m^t - \tilde{\mathbf{k}}^t\mathbf{1}_{1\times\tilde{N}}\right)^T \end{bmatrix} \tag{26}$$

$$\mathbf{S}_w = \mathbf{R}_w^T\mathbf{R}_w \tag{27}$$

where $\mathbf{R}_w$ is obtained by using the singular value decomposition (SVD) of $\mathbf{S}_w$. As suggested in [26], we employ the alternating optimization method to optimize $\mathbf{A}$ and $\mathbf{B}$.

Given $\mathbf{A}$, we solve the following problem to obtain $\mathbf{B}$:

$$\begin{aligned}\mathbf{B}^* = \arg\min_{\mathbf{B}} \sum_{i=1}^{k} \left\{\|\mathbf{H}_b\mathbf{R}_w^{-1}\mathbf{a}_i - \mathbf{H}_b\boldsymbol{\beta}_i\|^2 + \lambda\boldsymbol{\beta}_i^T\mathbf{S}_w\boldsymbol{\beta}_i\right\} \\ s.t.\ \ |\boldsymbol{\beta}_i|_1 \le t, i = 1, \cdots, k.\end{aligned} \tag{28}$$

Observing that $\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \cdots, \boldsymbol{\beta}_k$ are independent in (28), we separately optimize each $\boldsymbol{\beta}_i$ by solving the following optimization problem:

$$\begin{aligned}\boldsymbol{\beta}_i^* &= \arg\min_{\boldsymbol{\beta}_i} \|\mathbf{y}_i - \mathbf{H}_b\boldsymbol{\beta}_i\|^2 + \lambda\boldsymbol{\beta}_i^T\mathbf{S}_w\boldsymbol{\beta}_i \\ &= \arg\min_{\boldsymbol{\beta}_i} \|\tilde{\mathbf{y}}_i - \tilde{\mathbf{W}}\boldsymbol{\beta}_i\|^2, \ s.t. \ |\boldsymbol{\beta}_i|_1 \le t\end{aligned} \tag{29}$$

where

$$\mathbf{y}_i = \mathbf{H}_b\mathbf{R}_w^{-1}\mathbf{a}_i, \tilde{\mathbf{y}}_i = \left[\mathbf{y}_i^T, \mathbf{0}_{1\times N}\right]^T, \tilde{\mathbf{W}} = \left[\mathbf{H}_b^T, \sqrt{\lambda}\mathbf{R}_w^T\right]^T. \tag{30}$$

In this paper, we use the least angle regression [32] to solve $\boldsymbol{\beta}_i$.

Given $\mathbf{B}$, we can ignore the constraint on $\boldsymbol{\beta}_i$ and directly solve $\mathbf{A}$ by minimizing the following optimization problem:

$$\begin{aligned}\mathbf{A}^* &= \arg\min_{\mathbf{A}} \sum_{i=1}^{k} \|\mathbf{H}_b\mathbf{R}_w^{-1}\mathbf{a}_i - \mathbf{H}_b\boldsymbol{\beta}_i\|^2 \\ &= \arg\min_{\mathbf{A}} \|\mathbf{H}_b\mathbf{R}_w^{-1}\mathbf{A} - \mathbf{H}_b\mathbf{B}\|^2, \ s.t. \ \mathbf{A}^T\mathbf{A} = \mathbf{I}_{k\times k}.\end{aligned} \tag{31}$$

The optimal $\mathbf{A}^*$ can be solved by using SVD, namely

$$\begin{aligned}\mathbf{R}_w^{-T}\left(\mathbf{H}_b^T\mathbf{H}_b\right)\mathbf{B} &= \mathbf{U}\mathbf{D}\mathbf{V}^T \\ \mathbf{A}^* &= \mathbf{U}_{N\times k}\mathbf{V}_{k\times k}^T\end{aligned} \tag{32}$$

where $\mathbf{U}_{N\times k}$ and $\mathbf{V}_{k\times k}$ are, respectively, the top k eigenvectors of $\mathbf{U}$ and $\mathbf{V}$ corresponding to the top k largest eigenvalues. Finally, we iteratively solve (28) and (31) until the absolute difference of $\mathbf{B}$ from two successive iterations is smaller than a predefined threshold.

Step 2: given B, optimize $\boldsymbol{\omega}$. When $\mathbf{B}$ is fixed, according to MKL-DR [31], the objective function in (23) becomes

$$\begin{aligned}&\min_{\boldsymbol{\omega}} \boldsymbol{\omega}^T\mathbf{S}_w^A\boldsymbol{\omega} \\ &s.t.\ \boldsymbol{\omega}^T\mathbf{S}_b^A\boldsymbol{\omega} = 1, \sum_{m=1}^{c}\omega_m = 1, \omega_m \ge 0\end{aligned} \tag{33}$$

where

$$\begin{aligned}\mathbf{S}_w^A &= \sum_{i=1}^{M_1}\left(\mathbf{K}(\mathbf{x}_i^1) - \mathbf{K}(\mathbf{z}_i^1)\right)^T\mathbf{B}\mathbf{B}^T\left(\mathbf{K}(\mathbf{x}_i^1) - \mathbf{K}(\mathbf{z}_i^1)\right) \\ \mathbf{S}_b^A &= \sum_{i=1}^{M_0}\left(\mathbf{K}(\mathbf{x}_i^0) - \mathbf{K}(\mathbf{z}_i^0)\right)^T\mathbf{B}\mathbf{B}^T\left(\mathbf{K}(\mathbf{x}_i^0) - \mathbf{K}(\mathbf{z}_i^0)\right) \\ &+\gamma\sum_{i=1}^{\tilde{N}}\left(\mathbf{K}(\mathbf{x}_i) - \tilde{\mathbf{k}}^t\mathbf{1}_{1\times c}\right)^T\mathbf{B}\mathbf{B}^T\left(\mathbf{K}(\mathbf{x}_i) - \tilde{\mathbf{k}}^t\mathbf{1}_{1\times c}\right)\end{aligned} \tag{34}$$

with

$$\mathbf{K}(\mathbf{x}) = [\mathbf{k}_1(\mathbf{x}), \mathbf{k}_2(\mathbf{x}), \cdots, \mathbf{k}_c(\mathbf{x})] \in R^{N\times c}. \tag{35}$$

In fact, the objective function in (33) is a nonconvex quadratically constrained quadratic programming (QCQP) problem, which is difficult to be solved. According to [31], this problem can be relaxed to a convex problem by adding an auxiliary variable Ω of the size $k \times k$. (Please refer to [31] for more details.)

$$\begin{aligned}&\min_{\boldsymbol{\omega},\Omega} \mathrm{Tr}\left(\mathbf{S}_w^{\mathbf{A}}\Omega\right) \\ &s.t.\ \mathrm{Tr}\left(\mathbf{S}_b^{\mathbf{A}}\Omega\right) = 1, \ \sum_{m=1}^{c}\omega_m = 1 \\ &\omega_m \ge 0, m = 1, 2, \cdots, c, \ \begin{bmatrix} 1 & \boldsymbol{\omega}^T \\ \boldsymbol{\omega} & \Omega \end{bmatrix} \succeq 0.\end{aligned} \tag{36}$$

This problem can be efficiently solved by the semidefinite programming (SDP) solver [42].

The above two steps are repeated alternatively until the absolute differences of $\mathbf{B}$ and $\boldsymbol{\omega}$ from two successive iterations

are both smaller than a predefined threshold ε ($\varepsilon = 0.001$ in this paper).

In the whole algorithm, the main cost lies in the Step 1, since k Lasso problems need to be solved in each iteration. Therefore, a smaller t leads to less time.

Note that if only a single type of feature is used as in (23), i.e., $c = 1$, we only need to conduct Step 1 for the optimization.

E. Discussion

The difference with WSH [20]. The work in [20] and our method both employ the kernel hyperplane as the hashing function, in which the hashing hyperplane is represented as a combination of support vectors according to the Representer theory. In [20], the random sampling method is employed to reduce the time complexity and it also exploits the typical SVM-based objective function. In contrast, in our work, we employ the Fisher-like criterion to choose the support vectors and simultaneously determine the corresponding combination coefficients.

The difference with MLH [22] and BRE [10]. The works in [22] and [10] measure the compatibility of Hamming distances between hashing codes and the binary labels (or the similarity) by using the hinge-like loss in [22] or L_2-norm loss in [10]. On the contrary, our method formulates the supervision information by using Fisher-like criterion. Moreover, our work can be easily extended to a multiclass scenario. However, it is unclear how to extend the works in [22] and [10] for multiclass problem except explicitly converting the multiclass classification problem into a set of binary classification problems.

The difference with SSH [25]. The work in [25] and our method maximize the distances between the dissimilar pairs, and meanwhile minimize the distances between the similar pairs, to learn the Hamming codes. However, the difference operation is used to compare the distances from the similar and dissimilar pairs in [25], while the Fisher-like criterion is adopted in our method. The experiments demonstrate that our method is more effective when compared with [25].

In summary, the benefits of our proposed method come from two aspects. 1) like most existing methods (e.g., WSH [20] and BRE [10]), our method represents each hashing function as a combination of the support vectors. However, most existing methods randomly select the support vectors and only optimize for the combination coefficients. In contrast, in our method the support vectors are automatically determined from the unlabeled data set and the combination coefficients are also learnt at the same time. 2) When compared with the existing semisupervised methods (e.g., BRE [10] and SSH [25]), we propose to use a Fisher-like objective function to utilize the weakly labeled data, which has already been proven to be effective for many applications such as face recognition.

In contrast to the traditional semisupervised hashing methods, we do not consider the data distributions or geometric structure of the unlabeled data. However, the unlabeled data is indeed used to construct the hyperplanes, leading to promising results. Moreover, the criterion related to the hash bucket balance is also helpful to improve the results.

III. Experiments

In this section, we compare the proposed method with the existing methods on two large-scale image datasets, CIFAR-100 [27] and NUS-WIDE [28]. Note that our work is also applicable for video retrieval because our method can be readily used for video key frames. For all experiments, the semantic class label is employed as the ground truth. Four commonly used measurements are employed to evaluate the quantitative performances of all approaches: 1) the average precision of top 500 by only evaluating the top 500 returned images for each query [11], [33]; 2) the mean average precision (mAP) [11] from the recall-precision curve; 3) the precision within Hamming radius 2 and 4) when using the precision within Hamming radius 2, the query that does not have any returned samples within the hamming ball of radius 2 is treated as a failed query with the precision as zero, so the corresponding success rate is also reported according to [25].

A. Experimental Setting

CIFAR-100 dataset [27] is a labeled subset of the 80 million tiny images dataset [34]. It consists of 60 000 32×32 color images from 20 classes, with 3 000 images per class. Among them, 50 000 images are used as the training set and the remaining 10 000 images are used as the test set. On this dataset, we extracted four types of features: 320-D GIST [35]; 225-D color moment (CM) [36], [37]; 73-D edge direction histogram (EDH) [38]; and 128-D wavelet texture (WT) [39].

The NUS-WIDE dataset [28] is a web image dataset created for image annotation and retrieval. This dataset contains 2 69 648 images and the ground-truth annotations for 81 concepts. This dataset provides six types of low-level features including 64-D color histogram (CH), 144-D color correlogram (CORR), 73-D EDH, 128-D WT, 225-D block-wise CM extracted over 5×5 fixed grid partitions and 500-D bag of words (BoW) feature based on SIFT descriptions. In our experiment, BoW is excluded due to its high dimensionality. After removing the samples with multiple class labels, we have about 47 572 images for the training set and 31 645 images for the test set.

For both datasets, 10% of the randomly selected images from the test set are used as the query images and the full training set is used as the target searching dataset. 10 000 randomly selected pairs (with 5000 pairs from the same class and 5000 pairs from the different classes) from the training set are used as weakly labeled information for SSH [25], BRE [10], and our method. Moreover, only 3000 randomly selected samples are used for constructing the kernel hyperplanes in our method, while about 50 000 (*resp.* 47 572) images are employed for training in other existing methods and formulating the hash bucket balance constraint in our method on the CIFAR-100 (*resp.* NUS-WIDE) dataset.

For our method, the radial basis function (RBF), i.e., $k_m(\mathbf{x}, \mathbf{z}) = \exp\left(-\rho\|\mathbf{x} - \mathbf{z}\|^2\right)$, is employed for all types of features with $\rho = \frac{1}{\Gamma}$ where Γ is the mean of all related distances, as suggested in [40] and [41]. The parameters t and γ in (23) are empirically set to 0.1 and 0.00001,

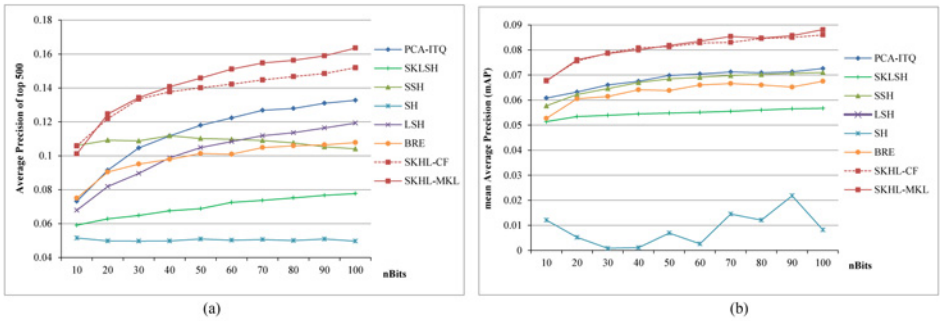

Fig. 2. Results of different methods on the CIFAR-100 dataset in terms of (a) average precision of top 500 and (b) mean average precision (mAP).

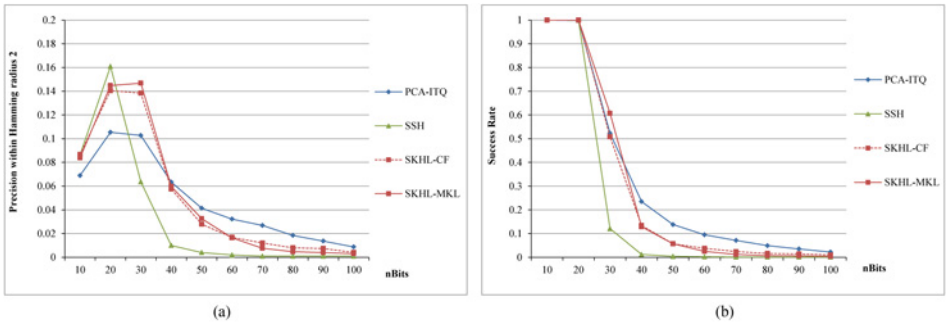

Fig. 3. Comparison of our methods SKHL-CF and SKHL-MKL with two most competitive methods PCA-ITQ and SSH on the CIFAR-100 dataset in terms of (a) precision within Hamming radius 2 and (b) success rate.

respectively. The proposed method using single type of feature is denoted as SKHL and it can converge within 3∼5 iterations. Our proposed method in Table I can converge to the local optimum within about 10 iterations. We also investigate our method without employing multiple kernel learning, namely, we concatenate all the features as a lengthy feature vector and directly use our SKHL on the concatenated feature in order to fairly compare our work with the exiting methods. We denote our method using MKL and without using MKL as SKHL-MKL and SKHL-CF, respectively.

In this paper, we compare our method with the existing methods including SSH without orthogonal relaxation [25], PCA-based iterative quantization (PCA-ITQ) [11], BRE [10], SKLSH [14], SH [7], and LSH [6], [2]. Among them, SSH, BRE and our method follow the semisupervised paradigm, SKLSH, SH, and LSH follow the unsupervised paradigm, and PCA-ITQ is the state-of-the-art method. We cannot compare our method with the CCA-ITQ [11] since full label information is unavailable.

B. Evaluation With Multiple Types of Features

We compare our SKHL using multiple types of features with the state-of-the-art methods including SSH [25], PCA-ITQ [11], BRE [10], SKLSH [14], SH [7], and LSH [6], [2]. In this evaluation, since these existing methods are designed with a single type of feature, to make a fair comparison all types of features are concatenated to form one lengthy feature vector.

Figs. 2 and 4 show the evaluation results of all methods on the CIFAR-100 dataset and the NUS-WIDE dataset in terms of average precision of top 500 and mAP, respectively. From Figs. 2 and 4, we observe that PCA-ITQ performs the best among all the existing methods since it considers the quantization error. Our method without using MKL (i.e., SKHL-CF) performs better than all the other methods in terms of the average precision of top 500 and mAP. Our method can further improve the performance after employing the MKL to fuse multiple types of features, i.e., SKHL-MKL is generally better than SKHL-CF and it also outperforms other existing methods in most cases. The weight obtained for each type of feature is shown in Fig. 6.

We also compare our methods with the two most competitive methods (i.e., SSH [25] and PCA-ITQ [11]) in terms of precision within Hamming radius 2 and success rate in Figs. 3 and 5. Again, our methods generally outperform the two existing methods on the NUS-WIDE dataset. On the CIFAR-100 dataset, our methods are generally better than SSH and it is comparable with PCA-ITQ. These results again demonstrate the effectiveness of our work. To visualize the returned semantic neighbors, we show the retrieval results for two example queries in Fig. 10, from which one can see our method can retrieve more relevant images than the existing methods.

C. Evaluation With a Single Type of Feature

The proposed method is also evaluated with single type of feature in terms of average precision of the top 500 returned neighbors. Again, we compare our method with the most competitive methods SSH [25] and PCA-ITQ [11].

On the CIFAR-100 dataset, all methods are evaluated when using each of the four types of features including GIST, CM, EDH, and WT, and the results are shown in Fig. 7. On the NUS-WIDE dataset, all methods are evaluated when

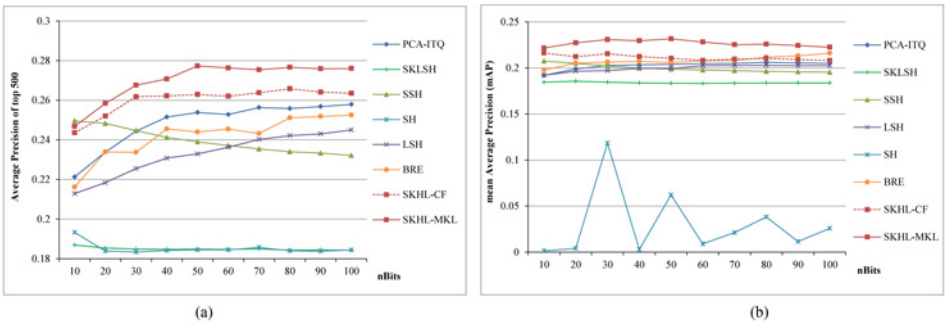

Fig. 4. Results of different methods on the NUS-WIDE dataset in terms of (a) average precision of top 500 and (b) mean average precision (mAP).

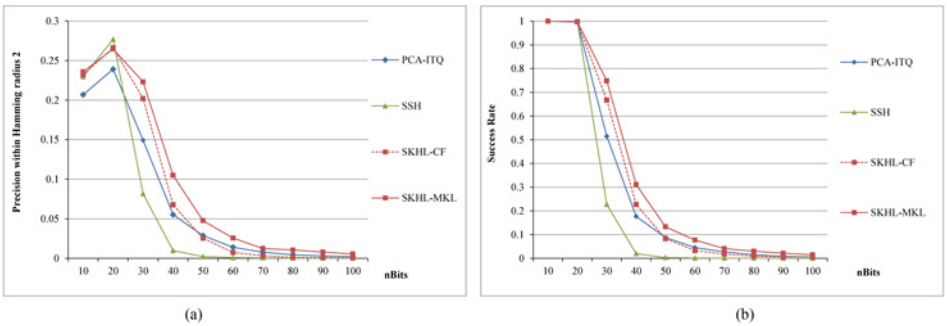

Fig. 5. Comparison of our methods SKHL-CF and SKHL-MKL with two most competitive methods PCA-ITQ and SSH on the NUS-WIDE dataset in terms of (a) precision within Hamming radius 2 and (b) success rate.

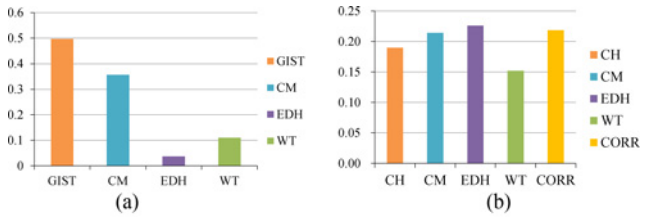

Fig. 6. Learnt weights for each type of feature from our SKHL-MKL on two datasets. (a) CIFAR-100 dataset. (b) NUS-WIDE dataset.

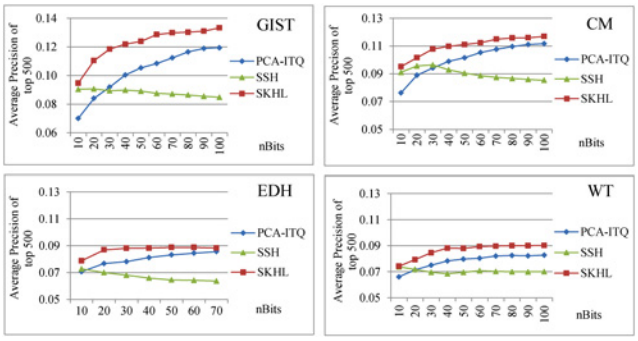

Fig. 7. Comparison between our method SKHL and two most competitive methods PCA-ITQ and SSH when using each type of feature (i.e., GIST, CM, EDH, or WT) on the CIFAR-100 dataset.

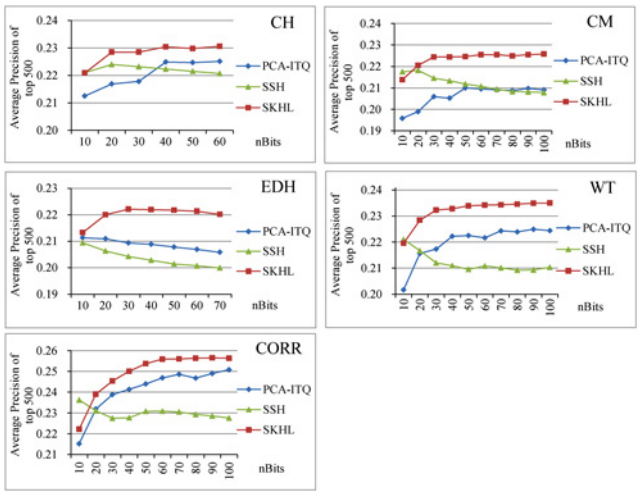

Fig. 8. Comparison between our method SKML and two most competitive methods PCA-ITQ and SSH when using each type of feature (i.e., CH, CM, EDH, WT, or CORR) on the NUS-WIDE dataset.

using each of the five types of features including CH, CM, EDH, WT, and CORR, and the results are shown in Fig. 8. According to both figures, SSH performs better when using a small number of bits, because most variance of the relaxed hashing functions lies on the top few directions. Our method performs better than both PCA-ITQ and SSH when using all types of features even though our method does not consider the quantization error as in PCA-ITQ. This demonstrates that our method can indeed obtain a set of more discriminant hashing functions.

There is a parameter t controlling the sparsity in the proposed SKHL. So we also evaluate the performance variations with different values of t in Fig. 9. When t is very small, SKHL-MKL does not perform well because only a limited number of support vectors are chosen. When t is set to be about 0.12, SKHL-MKL performs the best with less than 100 support vectors chosen for each hyperplane. When t

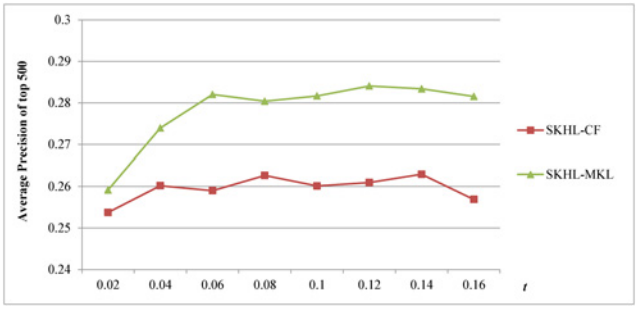

Fig. 9. Performance variations of our SKHL-CF and SKHL-MKL with different parameter t on the NUS-WIDE dataset.

Fig. 10. Retrieval results of two example queries. For each query, the image in the first column is the query image, while the images in four rows are respectively the top retrieved images by using our SKHL-MKL, PCA-ITQ, SSH, and LSH. Images highlighted in red rectangle are irrelevant images.

becomes larger, the performance of SKHL-MKL degrades. We will study how to automatically determine the optimal t in the future.

IV. Conclusion

In this paper, we propose a semisupervised Hashing approach via kernel hyperplane learning for scalable image search. In our method, each hashing function is modeled as a nonlinear kernel hyperplane constructed from an unlabeled dataset. Through maximizing a Fisher-like criterion on a weakly labeled dataset only with side information, we obtain a set of optimal kernel hyperplanes and hashing functions. Our method can also combine with MKL to fuse multiple types of features for generating better hashing codes. The experimental results on the CIFAR-100 and NUS-WIDE datasets demonstrate that our method achieves the state-of-the-art performance.

In this paper, each hashing function is updated independently in each iteration, and it is interesting to study how to update them jointly, which will be studied in the future. Moreover, the proposed method is also applicable for video retrieval by using the video key frames as the input. In the future, we will exploit the video structure or temporal information for more effective video retrieval.

V. Acknowledgments

The authors would like to thank the Associate Editor and the reviewers for their valuable comments and suggestions.

References

[1] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proc. ACM Symp. Theory Comput.*, 1998, pp. 604–613.
[2] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *Proc. Int. Conf. Very Large Data Bases*, 1999, pp. 518–529.
[3] S. Lee and C. D. Yoo, "Robust video fingerprinting for content-based video identification," *IEEE Trans Circuits Syst. Video Technol.*, vol. 18, no. 7, pp. 983–988, Jul. 2008.
[4] S.-S. Cheung and A. Zakhor, "Efficient video similarity measurement with video signature," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 1, pp. 59–74, Jan. 2003.
[5] S. Lee, C. D. Yoo, and T. Kalker, "Robust video fingerprinting based on symmetric pairwise boosting," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 9, pp. 1379–1388, Sep. 2009.
[6] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," *Commun. ACM*, vol. 51, no. 1, pp. 117–122, 2008.
[7] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Proc. Adv. Neural Inform. Process. Syst.*, 2008, pp. 1753–1760.
[8] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proc. Symp. Comput. Geometry*, 2004, pp. 253–262.
[9] W. Liu, J. Wang, S. Kumar, and S.-F. Chang, "Hashing with graphs," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 1–8.
[10] B. Kulis and T. Darrell, "Learning to hash with binary reconstructive embeddings," in *Proc. Adv. Neural Inform. Process. Syst.*, 2009, pp. 1042–1050.
[11] Y. Gong and S. Lazebnik, "Iterative quantization: A procrustean approach to learning binary codes," in *Proc. Conf. Comput. Vision Pattern Recogn.*, 2011, pp. 817–824.
[12] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PP, no. 99, p. 1, Sep. 2012.
[13] B. Kulis and K. Grauman, "Kernelized locality-sensitive hashing for scalable image search," in *Proc. Int. Conf. Comput. Vision*, 2009, pp. 2130–2137.
[14] M. Raginsky and S. Lazebnik, "Locality-sensitive binary codes from shift-invariant kernels," in *Proc. Int. Conf. Comput. Vision*, 2009, pp. 1509–1517.
[15] Y. Gong, S. Kumar, V. Verma, and S. Lazebnik, "Angular quantization-based binary codes for fast similarity search," in *Proc. Adv. Neural Inform. Process. Syst.*, 2012, pp. 1205–1213.
[16] J.-P. Heo, Y. Lee, J. He, S.-F. Chang, and S.-E. Yoon, "Spherical hashing," in *Proc. Conf. Comput. Vision Pattern Recogn.*, 2012, pp. 2957–2964.
[17] P. Jain, B. Kulis, and K. Grauman, "Fast image search for learnt metrics," in *Proc. Conf. Comput. Vision Pattern Recogn.*, 2008, pp. 1–8.
[18] G. Shakhnarovich, P. Viola, and T. Darrell, "Fast pose estimation with parameter sensitive hashing," in *Proc. Int. Conf. Comput. Vision*, vol. 2. 2003, pp. 750–757.
[19] A. Torralba, R. Fergus, and Y. Weiss, "Small codes and large image databases for recognition," in *Proc. Conf. Comput. Vision Pattern Recogn.*, 2008, pp. 1–8.
[20] Y. Mu, J. Shen, and S. Yan, "Weakly-supervised hashing in kernel space," in *Proc. Conf. Comput. Vision Pattern Recogn.*, 2010, pp. 3344–3351.
[21] H. Hotelling, "Relations between two sets of variable," *Biometrika*, vol. 28, pp. 312–377, Dec. 1936.
[22] M. Norouzi and D. J. Fleet, "Minimal loss hashing for compact binary codes," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 353–360.
[23] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang, "Supervised hashing with kernels," in *Proc. Conf. Comput. Vision Pattern Recogn.*, 2012, pp. 2074–2081.
[24] M. Norouzi, D. Fleet, and R. Salakhutdinov, "Hamming distance metric learning," in *Proc. Adv. Neural Inform. Process. Syst.*, 2012, pp. 1070–1078.
[25] J. Wang, S. Kumar, and S.-F. Chang, "Semi-supervised hashing for scalable image retrieval," in *Proc. Conf. Comput. Vision Pattern Recogn*, 2010, pp. 3424–3431.
[26] Z. Qiao, L. Zhou, and J. Z. Huang, "Sparse linear discriminant analysis with applications to high dimensional low sample size data," *Int. J. Appl. Math.*, vol. 39, no. 1, pp. 48–60, 2007.

[27] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.Sc. thesis, Comput. Sci. Dept., Univ. of Toronto, Toronto, ON, Canada, 2009.
[28] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y.-T. Zheng, "Nus-wide: A real-world web image database from National University of Singapore," in *Proc. ACM Int. Conf. Image Video Retrieval*, 2009, p. 48.
[29] B. Scholkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem," in *Proc. Annu. Conf. Comput. Learn. Theory*, 2001, pp. 416–426.
[30] H. Wang, S. Yan, D. Xu, X. Tang, and T. Huang, "Trace ratio vs. ratio trace for dimensionality reduction," in *Proc. Conf. Comput. Vision Pattern Recogn.*, 2007, pp. 1–8.
[31] Y.-Y. Lin, T.-L. Liu, and C.-S. Fuh, "Multiple kernel learning for dimensionality reduction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 6, pp. 1147–1160, Jun. 2011.
[32] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *Ann. Stat.*, vol. 39, no. 4, pp. 407–499, 2004.
[33] J. Wang and S. fu Chang, "Sequential projection learning for hashing with compact codes," in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 1127–1134.
[34] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large data set for nonparametric object and scene recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 11, pp. 1958–1970, Nov. 2008.
[35] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelop," *Int. J. Comput. Vision*, vol. 42, no. 3, pp. 145–175, 2001.
[36] A. Yanagawa, W. Hsu, and S.-F. Chang, "Brief descriptions of visual features for baseline trecvid concept detectors," Electr. Eng. Dept., Univ. Columbia, New York, NY, USA, Tech. Rep. #219-2006-5, Jul. 2006.
[37] L. Chen, D. Xu, I. Tsang, and J. Luo, "Tag-based web photo retrieval improved by batch mode re-tagging," in *Proc. Conf. Comput. Vision Pattern Recogn.*, 2010, pp. 3440–3446.
[38] D. K. Park, Y. S. Jeon, and Y. S. Jeon, "Effcient use of local edge histogram descriptor," in *Proc. ACM Workshops Multimedia*, 2000, pp. 51–54.
[39] B. S. Manjunath and W.-Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 8, pp. 837–842, Aug. 1996.
[40] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *Proc. Conf. Comput. Vision Pattern Recogn.*, 2008, pp. 1–8.
[41] L. Duan, D. Xu, I. W. Tsang, and J. Luo, "Visual event recognition in videos by learning from web data," in *Proc. Conf. Comput. Vision Pattern Recogn.*, 2010, pp. 1959–1966.
[42] L. Vandenberghe and S. Boyd, "Semidefinite programming," *SIAM review*, vol. 38, no. 1, pp. 49–95, 1996.
[43] M. Kan, D. Xu, S. Shan, W. Li, and X. Chen, "Learning prototype hyperplanes for face recognition in the wild," *IEEE Trans. Image Process.*, vol. 22, no. 8, pp. 3310–3316, Aug. 2013.

Meina Kan received the B.S. degree from Shandong University, Shandong, China, and the Ph.D. degree from Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China.

Her research interests include pattern recognition and computer vision, especially face recognition.

Dong Xu (SM'13) received the B.Eng. and Ph.D. degrees from the Electronic Engineering and Information Science Department, University of Science and Technology of China, Hefei, Anhui, China, in 2001 and 2005, respectively.

He is currently an Associate Professor with Nanyang Technological University, Singapore. While working toward the Ph.D. degree, he was with Microsoft Research Asia, Beijing, China, and Chinese University of Hong Kong, New Territories, Hong Kong. He also was with Columbia University, New York, NY, USA, for one year as a Post-Doctoral Research Scientist. His research interests include computer vision, pattern recognition, statistical learning, and multimedia content analysis.

Shiguang Shan (M'11) received the M.S. degree in computer science from Harbin Institute of Technology, Harbin, China, in 1999 and the Ph.D. degree in computer science from Institute of Computing Technology, Chinese Academy of Sciences (CAS), Beijing, China, in 2004.

Since 2002, he has been with the Institute of Computing Technology, CAS, where he has been a Professor since 2010. He is also the Executive Director of the Key Laboratory of Intelligent Information Processing, CAS. His research interests include image analysis, pattern recognition, and computer vision, focusing especially on face recognition-related research topics.

Dr. Shan received China's State Scientific and Technological Progress Award in 2005 for his work on face recognition technologies.

Xilin Chen (SM'09) received the B.S., M.S., and Ph.D. degrees in computer science from Harbin Institute of Technology (HIT), Harbin, China, in 1988, 1991, and 1994, respectively.

He was a Professor with HIT from 1999 to 2005, and he was a Visiting Scholar with Carnegie Mellon University, Pittsburgh, PA, USA, from 2001 to 2004. He has been a Professor with Institute of Computing Technology, Chinese Academy of Sciences, Beijing, since August 2004. His research interests include image processing, pattern recognition, computer vision, and multimodal interface.

Dr. Chen has received several awards, including China's State Scientific and Technological Progress Award in 2000, 2003, 2005, and 2012 for his research.