Semi-supervised Multi-view Discrete Hashing for Fast Image Search

Chenghao Zhang and Wei-Shi Zheng

***Abstract*—Hashing is an important method for fast neighbor search on large scale dataset in Hamming space. While most research on hash models are focusing on single-view data, recently the multi-view approaches with a majority of unsupervised multi-view hash models have been considered. Despite of existence of millions of unlabeled data samples, it is believed that labeling a handful of data will remarkably improve the searching performance. In this work, we propose a semi-supervised multi-view hash model. Besides incorporating a portion of label information into the model, the proposed multi-view model differs from existing multi-view hash models in three-fold: 1) a composite discrete hash learning modeling that is able to minimize the loss jointly on multi-view features when using relaxation on learning hashing codes, 2) exploring statistically uncorrelated multi-view features for generating hash codes, and 3) a composite locality preserving modeling for locally compact coding. Extensive experiments have been conducted to show the effectiveness of the proposed semi-supervised multi-view hash model as compared to related multi-view hash models and semi-supervised hash models.**

Keywords: Hash function learning, multi-view modeling, fast search, semi-supervised methods

I. Introduction

Recently, with the explosion of information, fast nearest neighbor search on huge dataset is becoming increasingly important for many tasks, for example, object recognition [30], linear classifier training [16], matching [3], retrieval [15], [29] and video segmentation [21]. For this purpose, hash models that learn binary embedding of data are attractive for achieving fast similarity search in Hamming space.

Among the developed hash models, most of them are either supervised hash models (e.g., binary reconstructive embedding (BRE) [14], minimal loss hashing (MLH) [23], kernel-based supervised hashing (KSH) [19], supervised discrete hashing (SDH) [27]), or unsupervised hash models (e.g., spectral hashing (SH) [35], k-means hashing (KMH) [7], anchor graph hashing (AGH) [20], complementary projection hashing (CPH) [9], and inductive hashing on manifolds (IHM) [28]). Labeling quite a lot of data samples for learning large-scale hashing is costly, while only learning hash function on vast unlabeled data cannot explore discriminant information in order to distinguish samples of different classes in Hamming space.Therefore, learning hash function in a semi-supervised way is a promising solution. Some methods have been proposed, including semi-supervised hashing (SSH) [33], semi-supervised bootstrap hashing (BT-SPLH) [36] and semi-supervised constraints preserving hashing (SCPH) [32].

However, most existing hash models are single-view methods, which only consider one type of feature descriptor for learning hash functions, so that they cannot process multi-view data effectively. Nowadays, similarity search on multi-view data is important. In practice, to make a more comprehensive description, objects/images are always represented via several different kinds of features, and each of them has its own characteristics. It is desirable to incorporate these heterogenous feature descriptors into hash function learning, leading to the multi-view hashing approach. Multi-view anchor graph hashing (MVAGH) [11] finds non-linear integrated binary codes which are determined by a subset of eigenvectors. Sequential spectral learning to hash with multiple representations (SU-MVSH) [12] finds a hash function that is sequentially determined by solving successive maximization of local data variance subject to decorrelation constraints. Multi-view alignment hashing (MAH) is an unsupervised method, which is based on nonnegative matrix factorization and finds a compact representation that can uncover hidden semantics. Composite hashing with multiple information sources (CHMIS) [38] integrates information from several different sources into binary hashing codes by adjusting the weights on each individual source for maximizing the coding performance. Deep hashing with multiple representations (DMVH) [10] utilizes deep neural network for performing multi-view hashing. Deep multimodal hashing with orthogonal regularization (DMHOR) [31] is also based on deep learning and imposes orthogonal regularizer between hash codes of different views/modalities. Recently, learning bridging mapping for cross-modal hashing(LBMCH) aims to learn heterogeneous Hamming spaces for different modalities, but LBMCH connects them only for cross-modal search [34].

A limitation of existing multi-view hash models is that most of them are unsupervised methods. While unsupervised hash models could be unsuitable for semantic fast search and labeling on large-scale image dataset is costly, how to learn hash functions on large-scale unlabeled image dataset provided

This work was supported partially by NSFC (No.61522115, 61472456, 61573387, 61661130157, 61628212), Guangdong Natural Science Funds for Distinguished Young Scholar under Grant S2013050014265, the GuangDong Program (No.2015B010105005), the Guangdong Science and Technology Planning Project (No.2016A010102012,2014B010118003), and Guangdong Program for Support of Top-notch Young Professionals (No.2014T Q01X779). The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Dimitrios Tzovaras. (Corresponding author: Wei-Shi Zheng)

Chenghao Zhang is with School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China; and is also with the Collaborative Innovation Center of High Performance Computing, National University of Defense Technology, Changsha 410073, China. Email: chenghaz@andrew.cmu.edu

Wei-Shi Zheng is with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China; and is also with the Key Laboratory of Machine Intelligence and Advanced Computing (Sun Yat-sen University), Ministry of Education, China. E-mail: wszheng@ieee.org.

with only a handful of labeled data samples still remains largely unsolved. In this work, we aim to develop a multi-view method for semi-supervised hashing. In principle, we present a constrained composite discrete hashing learning on multi-view data under semi-supervised setting. The novelties include:

- The proposed semi-supervised composite discrete multi-view hash modeling is able to minimize the loss jointly when using relaxation on learning hashing codes and meanwhile makes these hash codes more discriminant by quantifying the regression loss on class label over a portion of labeled samples.
- A statistically uncorrelated constraint between features of different views is introduced in order to make different views complementary to each other during the joint discrete learning on multiple feature channels, so that multi-view features collaborate more effectively.
- A composite multi-view locality preserving penalty is modeled to make hash codes compact in Hamming space.

Our experiments will show the effectiveness of the proposed approach as compared to the related semi-supervised hash models and existing unsupervised multi-view hash models.

II. Related Work on Multi-view Hashing

In this section, we will briefly review several multi-view hashing methods [38] [12] [11] [18] [31].

A. Composite Hashing with Multiple Information Sources(CHMIS) [38]

Let $\mathbf{x}_j^{(i)}$ be the i^{th} view of the j^{th} sample, $j = 1, \cdots, n$, where n is the number of samples. Composite Hashing learns binary code on weighted features from different source in a latent subspace. Its objective function is

$$\begin{aligned} \min_{\mathbf{B},\{\mathbf{W}_i\}_{i=1}^K,\mathbf{f}} \quad & T(\mathbf{B},\{\mathbf{W}_i\}_{i=1}^K,\mathbf{f}) \\ s.t. \quad & \mathbf{B}\mathbf{B}^T = \mathbf{I}, \mathbf{f}^T\mathbf{1} = \mathbf{1}, \mathbf{B}\mathbf{1} = \mathbf{0}, \end{aligned} \tag{1}$$

where $\mathbf{B}$ means the hash code matrix for the input, $\mathbf{W}_i$ is the projection matrix of the i^{th} view, and $\mathbf{f}$ is a non-negative vector which consists of weights $f_t,\ t = 1, \cdots, K$ that combine features of different views. Then $T(\mathbf{B}, \tilde{\mathbf{W}}, \mathbf{f})$ is defined as:

$$\begin{aligned} T(\mathbf{B},\{\mathbf{W}_i\}_{i=1}^K,\mathbf{f}) = C_1 \cdot trace(\mathbf{B}^T \sum_{t=1}^K \tilde{\mathbf{L}}^{(t)}\mathbf{B}) + \sum_{t=1}^K ||\mathbf{W}_t||^2 \\ + C_2 \sum_{j=1}^n ||\mathbf{b}_j - \sum_{i=1}^K f_i \cdot \mathbf{W}_i^T \mathbf{x}_j^{(i)}||^2,\ C_1, C_2 \geq 0, \end{aligned} \tag{2}$$

where $\mathbf{b}_j$ is the j^{th} column of $\mathbf{B}$, and $\tilde{\mathbf{L}}^{(t)}$ is a Laplacian matrix for the t^{th} view. The first term means two binary codes should be similar if they are nearby in any view in the input space. The third term is to quantify the loss on binary operation using the sign operation.

B. Sequential Spectral Learning to Hash with Multiple Representations(SU-MVSH) [12]

SU-MVSH extends spectral hashing from single view to multiple views by introducing a weighted similarity matrix $\mathbf{S}^*$ and learning binary vector for each sample from each view:

$$\arg \min_{\{\mathbf{b}_i^{(t)}\}_{i=1}^n} \sum_{i=1}^n \sum_{j=1}^n \mathbf{S}_{ij}^* \sum_{t,t'} ||\mathbf{b}_i^{(t)} - \mathbf{b}_j^{(t')}||^2, \tag{3}$$

where $\mathbf{S}^*$ is the α-Average similarity matrix, and $\mathbf{b}_i^{(t)}$ is the binary code vector for $\mathbf{x}_i^{(t)}$, the t^{th} view of the i^{th} sample. An α-Average similarity matrix means the α-divergence from a set of distance matrices. To form $\mathbf{S}^*$, the average distance matrix $\mathbf{D}^*$ is calculated by first minimizing the α-divergence from view-specific distance matrices $\{\mathbf{D}^{(1)}, ..., \mathbf{D}^{(K)}\}$ [1], [12], and then $\mathbf{S}^*$ is formulated by:

$$\mathbf{S}_{ij}^* = \exp\{-{\mathbf{D}_{ij}^*}^2\}. \tag{4}$$

C. Multi-view anchor graph hashing(MVAGH) [11]

Multi-view anchor graph hashing is an unsupervised method based on anchor graph hashing. In MVAGH, the similarity between data points are measured by a set of anchor points. MVAGH defines a multi-view anchor graph and uses it to define multi-view similarity between data points. For $\mathbf{x}_i$ and $\mathbf{x}_j$, the similarity $\mathbf{S}_{ij}^*$ is formulated by:

$$\mathbf{S}_{ij}^* = \sum_{k=1}^K \sum_{m=1}^M p(\mathbf{x}_j^{(k)}|\mu_m^{(k)}) p(\mu_m^{(k)}|\mathbf{x}_i^{(k)}), \tag{5}$$

where k is the view index of input data, m indicates the m-th anchor point, $p(\mathbf{x}_j^{(k)}|\mu_m^{(k)}) = \frac{\mathbf{Z}_{jm}^{(k)}}{\sum_{j=1}^N \mathbf{Z}_{jm}^{(k)}}$ and $p(\mu_m^{(k)}|\mathbf{x}_i^{(k)}) = \mathbf{Z}_{im}^{(k)}$. $\mathbf{Z}_{im}^{(k)}$ is defined as:

$$\mathbf{Z}_{im}^{(k)} = \frac{k(\mathbf{x}_i^{(k)}, \mu_m^{(k)})}{\sum_{k=1}^K \sum_{j\in[i]} k(\mathbf{x}_i^{(k)}, \mu_j^{(k)})}, \forall m \in [i], \tag{6}$$

where $\{\mu_j^{(k)}\}_{j=1}^M$ is the k^{th} views anchor points, [i] contains the indices of the l-nearest anchors of $\mathbf{x}_i^{(k)}$, and $k(\cdot,\cdot)$ is kernel function. After defining the similarity matrix, MVAGH performs the eigenvalue decomposition of matrix $\mathbf{S}^*$ and obtains its eigenvector matrix $\tilde{\mathbf{Y}}$, so that the binary code is computed by sgn function: $\mathbf{B} = sgn(\tilde{\mathbf{Y}})$

D. Multi-view Alignment Hashing(MAH) [18]

Multi-view alignment hashing forms a regularized kernel non-negativity matrix factorization (NMF) to find a semantic representation of the joint probability distribution of data. The objective function of MAH is:

$$\begin{aligned} \arg \min_{\mathbf{U},\mathbf{B},\alpha_i} & ||\sum_{i=1}^K \alpha_i \mathbf{K}_i - \mathbf{U}\mathbf{B}||^2 \\ & + \gamma \sum_{i=1}^n \alpha_i \cdot trace(\mathbf{B}\mathbf{L}_i\mathbf{B}^T) + \eta||\mathbf{U}^T\mathbf{U} - \mathbf{I}||^2, \\ s.t. & \sum_{i=1}^n \alpha_i = 1, \quad \alpha_i \geq 0 \text{ for } \forall i, \end{aligned} \tag{7}$$

where $\mathbf{L}_i$ is the Laplacian matrix, $\mathbf{U}$ is the basis matrix of NMF, $\mathbf{B}$ is the binary code matrix, and $\mathbf{K}_i$ is the kernel matrix defined as below:

$$\mathbf{K}_i(\mathbf{x}_p^{(i)}, \mathbf{x}_q^{(i)}) = exp(\frac{-||\mathbf{x}_p^{(i)} - \mathbf{x}_q^{(i)}||^2}{2\tau^2}), \forall p, q. \quad (8)$$

E. Deep Multimodal Hashing with Orthogonal Regularization(DMHOR) [31]

Deep multimodal hashing with orthogonal regularization is a multi-view hashing using deep learning method. It applies deep learning to generate binary code by exploiting the intra-modality and inter-modality correlations and incorporating data from different views. The objective function to minimize is formed below:

$$\min_{\theta} \quad L(\mathbf{X}_v, \mathbf{X}_t; \theta) = L_1 + \gamma||\mathbf{W}_v^{(m_v+1)}\mathbf{W}_t^{(m_t+1)^T}||^2 \quad (9)$$
$$+ \sum_{l=1}^{m_v+1} \alpha_l||\mathbf{W}_v^{(l)T}\mathbf{W}_v^{(l)} - \mathbf{I}||^2 + \sum_{l=1}^{m_t+1} \beta_l||\mathbf{W}_t^{(l)T}\mathbf{W}_t^{(l)} - \mathbf{I}||^2,$$

where L_1 is the loss function on multimodal autoencoder (MAE) and cross-modality autoencoder (CAE) [31], $\mathbf{X}_t$ and $\mathbf{X}_i$ are the input from text and image, respectively, and $\mathbf{W}^{(l)}$ is the weight matrix for the l-th layer. The second term of function imposes the orthogonal regularization on weight matrix from different views of the same layer. The last two terms guarantee the orthogonality in each view.

F. Extension of Unsupervised Multi-view Hash Models to Semi-supervised Case

Although not directly reported in existing literatures, it is not an obstacle to extend most of the above unsupervised multi-view hash models for semi-supervised modeling. Most of the unsupervised multi-view hash models define the similarity between two samples, for example the $\mathbf{S}_{ij}^*$ in MVAGH. Hence, one can extend the definition of the similarity to integrate label information. Taking MVAGH as an example, $\mathbf{S}_{ij}^*$ can be redefined in Eq. (10). The MGH [8] is a semi-supervised extension of MVAGH in such a similar way with extra weight learning on combination of similarity matrices from different views. MAH can also be extended to the semi-supervised case when $\mathbf{K}_i(\mathbf{x}_p^{(i)}, \mathbf{x}_q^{(i)})$ is redefined in Eq. (11). However, we show in our experiments that such a straightforward extension is not effective as they do not measure the loss on semantic search directly and do not explicitly quantify the relation between the extracted features from different views during semi-supervised learning.

III. Approach

A. A Composite Discrete Multi-view Hash Modeling

We present the proposed semi-supervised multi-view discrete hash model (SSMDH) in this section. Suppose each data sample consists of K views. Given n training samples from L classes, we form K data matrices for each view $\mathbf{X} = \left\{\mathbf{X}^{(i)}\right\}_{i=1}^{K}$, where $\mathbf{X}^{(i)}$ is the i^{th} view data matrix and its j^{th} column is the feature vector of the j^{th} data sample from this view. Suppose that data matrix $\mathbf{X}^{(i)}$ consists of two parts: $\mathbf{X}^{(i)} = [\mathbf{X}_\ell^{(i)}, \mathbf{X}_{n\ell}^{(i)}]$, where the first ℓ columns form the labeled matrix $\mathbf{X}_\ell^{(i)}$, while the rest are unlabeled data samples. Let $\mathbf{Y}$ be the label matrix corresponding to matrix $\mathbf{X}_\ell^{(i)}$, where its j^{th} column is the label vector of the j^{th} data sample. In this work, each label vector is a L-dimensional binary vector where the ℓ^{th} entry is one if it is from the ℓ^{th} class otherwise it is zero. We list the terms and notations used in this section in Table I

TABLE I
TERMS AND DEFINITION

symbols	definition
n	number of samples
K	number of views of data sample
$\mathbf{C}$	classification matrix
$\mathbf{W}_i$	projection matrix for view i
$\mathbf{S}^{(i)}$	similarity matrix for view i
$\mathbf{X}^{(i)}$	data matrix of input view i
$\mathbf{Y}$	label matrix w.r.t $\mathbf{X}^{(i)}$
$\mathbf{B}_\ell$	binary code matrix for labeled data
$\mathbf{B}_{n\ell}$	binary code matrix for unlabeled data

In order to capture neighborhood structure of the input data, we use anchor graph to map input data into a neighborhood model. For a single-view data sample $\mathbf{x}^{(i)}$, we map it to $\phi_i(\mathbf{x}^{(i)})$, a m-dimensional vector represented by m anchor points. More specifically, a set of anchor points $\{\mathbf{a}_t^{(i)}\}_{t=1..m}$ are randomly selected for each view and use these points to formulate $\phi_i(\mathbf{x}^{(i)})$:

$$\phi_i(\mathbf{x}^{(i)}) = [exp\frac{||\mathbf{x}^{(i)} - \mathbf{a}_1^{(i)}||^2}{\sigma_i^2}, ..., exp\frac{||\mathbf{x}^{(i)} - \mathbf{a}_m^{(i)}||^2}{\sigma_i^2}]^T, \quad (12)$$

where σ_i^2 is the standard variation of data of the i^{th} view

Our proposal is to learn a hash function that is able to fuse multiple view information to generate a binary code for fast search in Hamming space under the semi-supervised setting. In more details, we will learn a hash projection for each view, where we denote the i^{th} view hash projection as $\mathbf{W}_i$. And then, we predict the binary code $\mathbf{b}$ by fusing the feature information from K views below:

$$\mathbf{b} = sgn(\sum_{i=1}^{K} \mathbf{W}_i^T \phi_i(\mathbf{x}^{(i)})). \quad (13)$$

We denote the output binary code matrix as $\mathbf{B}$ where its j^{th} column is the binary code of the j^{th} data sample.

Multi-view Discrete Modeling. To facilitate learning projection $\mathbf{W}_i$ in Eq. (13), we will relax the sgn function in the optimization objective function. However, the discrepancy between the optimized binary code and the learned relaxed code remains. Hence it is necessary to minimize the discrepancy between the output binary code $\mathbf{B}$ and the predicted approximate one $\sum_{i=1}^{K} \mathbf{W}_i^T \phi_i(\mathbf{X}^{(i)})$ as follows:

$$\min_{\mathbf{B}, \{\mathbf{W}_i\}_{i=1}^{K}} L = ||\mathbf{B} - \sum_{i=1}^{K} \mathbf{W}_i^T \phi_i(\mathbf{X}^{(i)})||^2. \quad (14)$$

$$\mathbf{S}^*_{ij} = \begin{cases} \sum_{k=1}^{K}\sum_{m=1}^{M} p(\mathbf{x}_j^{(k)}|\mu_m^{(k)})p(\mu_m^{(k)}|\mathbf{x}_i^{(k)}), & \text{if either } \mathbf{x}_j^{(k)} \text{ or } \mathbf{x}_i^{(k)} \text{ is unlabeled,} \\ 1, & \text{if both } \mathbf{x}_j^{(k)} \text{ and } \mathbf{x}_i^{(k)} \text{ are labeled from the same class,} \\ 0, & \text{if both } \mathbf{x}_j^{(k)} \text{ and } \mathbf{x}_i^{(k)} \text{ are labeled but from different classes} \end{cases} \tag{10}$$

$$\mathbf{K}_i(\mathbf{x}_p^{(i)}, \mathbf{x}_q^{(i)}) = \begin{cases} exp(\frac{-||\mathbf{x}_p^{(i)} - \mathbf{x}_q^{(i)}||^2}{2\tau^2}), & \text{if either } \mathbf{x}_j^{(i)} \text{ or } \mathbf{x}_i^{(i)} \text{ is unlabeled,} \\ 1, & \text{if both } \mathbf{x}_p^{(i)} \text{ and } \mathbf{x}_q^{(i)} \text{ are labeled from the same class,} \\ 0, & \text{if both } \mathbf{x}_p^{(i)} \text{ and } \mathbf{x}_q^{(i)} \text{ are labeled but from different classes} \end{cases} \tag{11}$$

In our work, $\phi_i(\mathbf{X}^{(i)})$ is a matrix, column of which is the corresponding column of $\mathbf{X}^{(i)}$ after the mapping $\phi_i(\cdot)$.

Regression on Class Label Vectors. In Eq. (14), we optimize the hash projection matrices $\mathbf{W}_i$ and the binary output $\mathbf{B}$ together. Without any constraint on $\mathbf{B}$, trivial solutions will be gained. Since a handful of labeled data samples are available, we ensure that the output binary codes corresponding to the labeled data are suitable for semantic search. To this end, we predict the class label vector $\tilde{\mathbf{y}}$ for each labeled binary code vector $\mathbf{b}$ by

$$\tilde{\mathbf{y}} = G(\mathbf{b}) = \left[\mathbf{C}_1^T\mathbf{b}, ..., \mathbf{C}_L^T\mathbf{b}\right]^T, \tag{15}$$

where L is the number of classes as mentioned. Let $\tilde{\mathbf{Y}} = \mathbf{C}^T\mathbf{B}_\ell$ be the predicted label matrix of all labeled data samples, where $\mathbf{B}_\ell$ is the portion of binary code matrix corresponding to labeled data. We quantify the loss on the label prediction between $\mathbf{Y}$ and $\tilde{\mathbf{Y}}$ and seek optimal binary output $\mathbf{B}_\ell$ and weighting matrix $\mathbf{C}$ by the minimization problem below:

$$\min_{\mathbf{B}_\ell, \mathbf{C}} \frac{1}{n_\ell}||\mathbf{Y} - \mathbf{C}^T\mathbf{B}_\ell||^2 + \theta||\mathbf{C}||^2, \tag{16}$$

where θ is the regularization parameter.

Composite Locality Preserving. Since there is only limited labeled data samples available, we further utilize unlabeled data to make the hash bits more compact. In order to do so, we evaluate the following composite local data variation:

$$\begin{aligned} \mathbf{L} =& \frac{1}{n^2}\sum_{r=1}^{n}\sum_{t=1}^{n} S(r,t)\big(\sum_{i=1}^{K}\mathbf{W}_i^T(\phi_i(\mathbf{x}_r^{(i)}) - \phi_i(\mathbf{x}_t^{(i)}))\big) \\ & \times \big(\sum_{i=1}^{K}\mathbf{W}_i^T(\phi_i(\mathbf{x}_r^{(i)}) - \phi_i(\mathbf{x}_t^{(i)}))\big)^T \\ =& \frac{1}{n^2}\sum_{r=1}^{n}\sum_{t=1}^{n} S(r,t)\big(\sum_{i=1}^{K}\sum_{j=1}^{K}\mathbf{W}_i^T(\phi_i(\mathbf{x}_r^{(i)}) - \phi_i(\mathbf{x}_t^{(i)})) \\ & \times (\phi_j(\mathbf{x}_r^{(j)}) - \phi_j(\mathbf{x}_t^{(j)}))^T\mathbf{W}_j\big) \\ =& \frac{1}{n^2}\sum_{i=1}^{K}\sum_{j=1}^{K}\mathbf{W}_i^T\big(\sum_{r=1}^{n}\sum_{t=1}^{n} S(r,t)(\phi_i(\mathbf{x}_r^{(i)}) - \phi_i(\mathbf{x}_t^{(i)})) \\ & \times (\phi_j(\mathbf{x}_r^{(j)}) - \phi_j(\mathbf{x}_t^{(j)}))^T\big)\mathbf{W}_j \\ =& \sum_{i=1}^{K}\sum_{j=1}^{K}\mathbf{W}_i^T\mathbf{L}_{ij}\mathbf{W}_j \end{aligned} \tag{17}$$

where

$$\begin{aligned} \mathbf{L}_{ij} =& \frac{1}{n^2}\sum_{r=1}^{n}\sum_{t=1}^{n} S(r,t)(\phi_i(\mathbf{x}_r^{(i)}) - \phi_i(\mathbf{x}_t^{(i)}))(\phi_j(\mathbf{x}_r^{(j)}) - \phi_j(\mathbf{x}_t^{(j)}))^T \\ =& \frac{1}{n^2}\sum_{r=1}^{n}\sum_{t=1}^{n}\{S(r,t)\phi_i(\mathbf{x}_r^{(i)})\phi_j(\mathbf{x}_r^{(j)})^T \\ & - S(r,t)\phi_i(\mathbf{x}_r^{(i)})\phi_j(\mathbf{x}_t^{(j)})^T \\ & - S(r,t)\phi_i(\mathbf{x}_t^{(i)})\phi_j(\mathbf{x}_r^{(j)})^T \\ & + S(r,t)\phi_i(\mathbf{x}_t^{(i)})\phi_j(\mathbf{x}_t^{(j)})^T\} \\ =& \frac{2}{n^2}\phi_i(\mathbf{X}^{(i)})(\mathbf{D} - \mathbf{S})\phi_j(\mathbf{X}^{(j)})^T, \end{aligned} \tag{18}$$

where $\mathbf{D}$ is a diagonal matrix with the diagonal terms $\mathbf{D}_{rr} = \sum_t S(r,t)$. For a labeled pair between the r^{th} sample and the t^{th} sample, no matter from which view, the function $S(r,t)$ is defined as

$$S(r,t) = \begin{cases} 1, & if\ \mathbf{x}_r^{(i)} \text{ and } \mathbf{x}_t^{(j)} \text{ are from the same class for any } i, j; \\ 0, & if\ \mathbf{x}_r^{(i)} \text{ and } \mathbf{x}_t^{(j)} \text{ are not from the same class for any } i, j. \end{cases}$$

If one of the r^{th} sample and the t^{th} sample is unlabeled, we formulate the function $S(r,t)$ below:

$$S(r,t) = \frac{1}{K}\sum_{i=1}^{K} exp(-||\mathbf{x}_r^{(i)} - \mathbf{x}_t^{(i)}||^2/2\sigma_i^2). \tag{19}$$

In order to minimize the composite local data variation, the following is concerned, i.e.,

$$\min_{\{\mathbf{W}_i\}_{i=1}^K} \ trace(\sum_{i=1}^{K}\sum_{j=1}^{K}\mathbf{W}_i^T\mathbf{L}_{ij}\mathbf{W}_j). \tag{20}$$

This minimization can be useful for locality preserving of data when learning binary codes in Hamming space. Imposing the locality preserving is popularly used in existing literatures [37] [38]. The novelty here is to model it in a composite way for multi-view data and integrate it to enhance the proposed composite discrete multi-view hash model.

Extracting Statistically Uncorrelated View-specific Features. For multi-view learning, we wish that features extracted from different views should be complementary to each other so as to preserve as much information as possible in Hamming space. In order to do so, we investigate extracting statistically decorrelated view features. That is, features extracted from different views should be statistically uncorrelated. To that

end, for $i \neq j$, we form the cross-covariance matrix between view i and view j as follows:

$$\begin{aligned}
&\mathbb{E}_{\{(\mathbf{x}^{(i)},\mathbf{x}^{(j)}) \mid \mathbf{W}_i,\ \mathbf{W}_j\}} \left\{ \mathbf{W}_i^T(\mathbf{x}^{(i)} - \mathbb{E}_{\mathbf{x}^{(i)}})(\mathbf{x}^{(j)} - \mathbb{E}_{\mathbf{x}^{(j)}})^T \mathbf{W}_j \right\} \\
\approx &\frac{1}{n}\sum_{r=1}^{n} \mathbf{W}_i^T(\mathbf{x}_r^{(i)} - \mathbf{u}^{(i)})(\mathbf{x}_r^{(j)} - \mathbf{u}^{(j)})^T \mathbf{W}_j \\
= &\mathbf{W}_i^T \left\{ \frac{1}{n}\sum_{r=1}^{n} (\mathbf{x}_r^{(i)} - \mathbf{u}^{(i)})(\mathbf{x}_r^{(j)} - \mathbf{u}^{(j)})^T \right\} \mathbf{W}_j \\
= &\mathbf{W}_i^T Cov_{ij} \mathbf{W}_j,
\end{aligned} \tag{21}$$

where $Cov_{ij} = \frac{1}{n}\sum_{r=1}^{n}(\mathbf{x}_r^{(i)} - \mathbf{u}^{(i)})(\mathbf{x}_r^{(j)} - \mathbf{u}^{(j)})^T$, and $\mathbf{u}^{(i)}$ is the data mean of the i^{th} view.

Eq. (21) is to calculate the cross-correlation between two types of view-specific feature vectors. For example, the entry of the p^{th} row and the q^{th} column of matrix $\mathbf{W}_i^T Cov_{ij} \mathbf{W}_j$ (i.e., $\left(\mathbf{W}_i^T Cov_{ij} \mathbf{W}_j\right)_{pq}$) measures the statistical correlation between the p^{th} entry of $\mathbf{W}_i^T \mathbf{x}^{(i)}$ and the q^{th} entry of $\mathbf{W}_j^T \mathbf{x}^{(j)}$. And thus minimizing the following Eq. (22) would limit all the statistical correlation between different entries of $\mathbf{W}_i^T \mathbf{x}^{(i)}$ and of $\mathbf{W}_j^T \mathbf{x}^{(j)}$, and we call the matrix computed by Eq. (21) as the cross-covariance matrix between view i and view j. Hence, for this purpose, our objective is to minimize the following:

$$\min_{\{\mathbf{W}_i\}_{i=1}^K} \sum_{i=1}^{K}\sum_{j\neq i}^{K} ||\mathbf{W}_i^T Cov_{ij} \mathbf{W}_j||^2. \tag{22}$$

Objective Function. Finally, we combine Criteria (14), (16), (20) and (21), and present our objective function for *semi-supervised multi-view discrete hash* (SSMDH) model as follows:

$$\begin{aligned}
\min_{\mathbf{B},\mathbf{C},\{\mathbf{W}_i\}_{i=1}^K} L = &\frac{1}{n_\ell}||\mathbf{Y} - \mathbf{C}^T\mathbf{B}_\ell||^2 \\
&+ \frac{v}{n}||\mathbf{B} - \sum_{i=1}^{K} \mathbf{W}_i^T \phi_i(\mathbf{X}^{(i)})||^2 \\
&+ \frac{\alpha}{2}\sum_{i=1}^{K}\sum_{j\neq i}^{K} ||\mathbf{W}_i^T Cov_{ij} \mathbf{W}_j||^2 \\
&+ \beta \cdot trace\big(\sum_{i=1}^{K}\sum_{j=1}^{K} \mathbf{W}_i^T \mathbf{L}_{ij} \mathbf{W}_j\big) + \theta||\mathbf{C}||^2 \\
s.t. \quad &\tilde{\mathbf{Y}} = \mathbf{C}^T\mathbf{B}_\ell,\ \mathbf{B} \in \{-1, 1\}^{L*n}.
\end{aligned} \tag{23}$$

where v, α and β are the balance parameters on the binary discrepancy loss, cross-view correlation loss, and composite locality preserving loss, respectively. In the above, we split the binary matrix $\mathbf{B}$ into two parts $\mathbf{B} = [\mathbf{B}_\ell,\ \mathbf{B}_{n\ell}]$, and $\mathbf{B}_\ell$ is the one corresponding to label data. In the above criterion, we learn all view-specific projection matrices $\mathbf{W}_i$ jointly rather than separately, and our formulation enables a *semi-supervised composite discrete hashing* on multi-view data. We show later in our experiments that such a modeling would make clear benefit for multi-view learning.

Discussion. The proposed semi-supervised multi-view discrete hash is related to the supervised discrete hash (SDH) [27] on handling fully supervised data samples. Apart from extending SDH from single-view modeling to multi-view joint learning by composite modeling and developing a semi-supervised model, we further propose to quantify the relation between different views by extracting statistically uncorrelated view-specific features and propose a composite locality preserving criterion to constrain the local variation of hash codes. We show that the unification of the composite modeling, statistical uncorrelation between views, and composite locality preserving is important for semi-supervised multi-view learning in our experiments (see Sec.IV-E).

Algorithm 1 SSMDH

Input: Training dataset $\{\mathbf{X}^{(i)}\}$, class label matrix $\mathbf{Y}$, number of anchor points in each view i, binary code length r, maximum iteration number t, and Parameters $\alpha,\ v,\ \theta$

Output: Projection matrix $\{\mathbf{W}_i\}_{i=1}^K$. Binary code matrix $\mathbf{B}$

1) Initialize binary code matrix $\mathbf{B}$ and anchor point set;
2) **while** not reaching maximum iteration **do**
3) Apply Eq.(24) to compute classification matrix $\mathbf{C}$;
4) Apply Eq.(26) for each view to compute $\mathbf{W}_i$;
5) Compute the joint binary codes by Eq.(31) and (30);
6) **end**
7) Apply Eq.(13) to compute $\mathbf{B}$;

B. Optimization

Since optimizing all variables simultaneously in Eq.(23) is not convex, we present an alternating procedure for optimization. That is for learning all variables $\mathbf{B}$, $\mathbf{W}_i$ and $\mathbf{C}$, we fix other variables and optimize one single variable at one time, and this is repeated for each variable. The overview of the optimization procedure is presented in Algorithm 1.

1) Fixing $\mathbf{B}_\ell$ and Optimizing $\mathbf{C}$: If we fix $\mathbf{B}_\ell$ in Eq.(23), the optimization for $\mathbf{C}$ is independent of other terms in the objective function except for the first and the last ones, so we have:

$$\mathbf{C} = (\mathbf{B}_\ell \mathbf{B}_\ell{}^T + n_\ell \theta \mathbf{I})^{-1} \mathbf{B}_\ell \mathbf{Y}^T. \tag{24}$$

2) Fixing $\mathbf{B}$ and Optimizing $\mathbf{W}_i$: Since the projection matrix of each view is dependent on the others, for optimizing projection matrix of the i^{th} view, we fix $\mathbf{B}$ and all $\mathbf{W}_{j(j\neq i)}$ except for $\mathbf{W}_i$. Then the optimization for $\mathbf{W}_i$ is equivalent to minimizing the function below:

$$\begin{aligned}
O(\mathbf{W}_i) = &v||\mathbf{B}||^2 - 2v \cdot trace(\mathbf{W}_i^T \phi_i(\mathbf{X}^{(i)}) \mathbf{B}^T) \\
&+ v \cdot trace(\phi_i(\mathbf{X}^{(i)})^T \mathbf{W}_i \mathbf{W}_i^T \phi_j(\mathbf{X}^{(i)})) \\
&+ 2v \sum_{j\neq i} trace(\phi_i(\mathbf{X}^{(i)})^T \mathbf{W}_i \mathbf{W}_j^T \phi_j(\mathbf{X}^{(j)})) \\
&+ n\alpha \sum_{j\neq i}^{K} trace(\mathbf{W}_i^T Cov_{ij} \mathbf{W}_j \mathbf{W}_j^T Cov_{ij}^T \mathbf{W}_i) \\
&+ n\beta \cdot trace\big(\mathbf{W}_i^T \mathbf{L}_{ii} \mathbf{W}_i\big) + 2n\beta \cdot trace\big(\sum_{j\neq i} \mathbf{W}_i^T \mathbf{L}_{ij} \mathbf{W}_j\big).
\end{aligned} \tag{25}$$

$\mathbf{W}_i$ can be computed by:

$$\mathbf{W}_i = (v\phi_i(\mathbf{X}^{(i)})\phi_i(\mathbf{X}^{(i)})^T + n\alpha\sum_{j\neq i}^{K} Cov_{ij}\mathbf{W}_j\mathbf{W}_j^T Cov_{ij}^T + n\beta\mathbf{L}_{ij})^{-1} (v\phi_i(\mathbf{X}^{(i)})\mathbf{B}^T - v\sum_{j\neq i}\phi_i(\mathbf{X}^{(i)})\phi_j(\mathbf{X}^{(j)})^T W_j - n\beta\sum_{j\neq i}\mathbf{L}_{ij}\mathbf{W}_j). \tag{26}$$

3) Fixing other variables and Optimizing $\mathbf{B}$*:* It is hard to solve binary code by regularized least squares problem and it is a NP hard question. In order to optimize $\mathbf{B}$, we have to optimize $\mathbf{B}_\ell$ and $\mathbf{B}_{n\ell}$ separately. That is we only need to minimize part of Eq.(23) as follow:

$$\min_{\mathbf{B}} \frac{n}{n_\ell}||\mathbf{Y} - \mathbf{C}^T\mathbf{B}_\ell||^2 + v||\mathbf{B}_\ell - \sum_{i=1}^{K}\mathbf{W}_i^T\phi_i(\mathbf{X}_\ell^{(i)})||^2 + v||\mathbf{B}_{n\ell} - \sum_{i=1}^{K}\mathbf{W}_i^T\phi_i(\mathbf{X}_{n\ell}^{(i)})||^2, \tag{27}$$

where, as mentioned, $\mathbf{X}_\ell^{(i)}$ is the labeled data matrix and $\mathbf{X}_{n\ell}^{(i)}$ is the unlabeled data matrix.

We can rewrite Eq.(27) if we only optimize $\mathbf{B}_{n\ell}$ as follow:

$$\begin{aligned} \min_{\mathbf{B}_{n\ell}} \quad & v(||\mathbf{B}_{n\ell}||^2 - 2tr(\mathbf{B}_{n\ell}^T\mathbf{P}) + ||\mathbf{P}||^2) \\ s.t. \quad & \mathbf{P} = \sum_{i=1}^{K}\mathbf{W}_i^T\phi_i(\mathbf{X}^{(i)}). \end{aligned} \tag{28}$$

That is equal to addressing the following maximization problem:

$$\begin{aligned} \max_{\mathbf{B}_{n\ell}} \quad & 2v\cdot tr(\mathbf{B}_{n\ell}^T\mathbf{P}) \\ s.t. \quad & \mathbf{P} = \sum_{i=1}^{K}\mathbf{W}_i^T\phi_i(\mathbf{X}^{(i)}). \end{aligned} \tag{29}$$

Eq.(29) can be easily computed if we fix other columns of $\mathbf{B}_{n\ell}$ and optimize the i^{th} column $\mathbf{b}_i$ at a time:

$$\mathbf{b}_i = sgn(\mathbf{p}_i), \tag{30}$$

where $\mathbf{p}_i$ is the i^{th} column of $\mathbf{P}$.

Meanwhile, $\mathbf{B}_\ell$ in Eq.(27) can be optimized bit by bit by using discrete cyclic coordinate descent(DCC) [27]. Let $\mathbf{z}_i^T$ be the i^{th} row of $\mathbf{B}_\ell$, and then it can be optimized by:

$$\mathbf{z}_i = sgn(\mathbf{q}_i - \frac{n}{n_\ell}\mathbf{B}_{\ell,-i}^T\mathbf{C}_{-i}\mathbf{v}), \tag{31}$$

where $\mathbf{B}_{\ell,-i}$ is the matrix $\mathbf{B}_\ell$ excluding $\mathbf{z}_i$, $\mathbf{q}_i^T$ is the i^{th} row of $\mathbf{Q}$ where $\mathbf{Q} = \frac{n}{n_\ell}\mathbf{C}\mathbf{Y} + v\sum_{i=1}^{K}\mathbf{W}_i^T\phi_i(\mathbf{X}^{(i)})$ and $\mathbf{v}^T$ is the i^{th} row of $\mathbf{C}$, and $\mathbf{C}_{-i}$ is $\mathbf{C}$ excluding the i^{th} row.

IV. Experiments

In our work, we mainly performed the experimental analysis on three widely used datasets. We evaluated our method and compared with related multi-view hash models and several state-of-the-art semi-supervised hash models.

A. Datasets and Settings

The three widely employed datasets for main experimental analysis are introduced below.

CIFAR-10 [13] . It consists of 60000 32x32 color images from 10 classes. There are 6000 images for each class, and each image has only one class label. In our experiment, we selected 59000 images to form the training set and 1000 images for testing. The training set was used to learn hash function and construct hash look-up table. To form multi-view data, a 512-dimensional GIST [24] descriptor and a 496-dimensional HOG [5] descriptor were extracted from each image. For verifying the performance of semi-supervised multi-view hash models, we randomly selected 4k labeled data from training set and the remained were treated as unlabeled data samples. For SSH and SCPH, we randomly generated similar and dissimilar constraints from the labeled data that we selected.

WIKI [25]. It consists of 2866 documents provided by Wikipedia. Each document contains an image and is also annotated with 10 semantic labels. Two documents were considered similar if they shared common label. For each document, there are a 128-dimensional SIFT [22] descriptor and a 10-dimensional LDA [2] feature. 80% samples of the dataset were randomly selected to form the training set and the remained formed the testing set. For verifying the performance of semi-supervised multi-view hash models, 200 labeled data samples were used and the rest were considered as unlabeled data. For SSH and SCPH, we generated similar and dissimilar constraints from the labeled data that we selected.

NUS-WIDE [4]. It is a widely used dataset that contains 269648 images from Flickr. The dataset is constituted of 81 ground-truth concepts and each image is labeled by at least one concept, and multiple concept can be assigned to each image. We randomly selected 180k images from 10 largest concept. Besides, each image was represented by a 500-dimensional bag of words feature based on SIFT [22] descriptor, and each text was represented by 1000-dimensional tag vector. Two images were considered as matched if they shared at least one of the concept. The training set contains 175k samples, and the rest were used to form the testing set. For verifying the performance of semi-supervised multi-view hash models, we randomly selected 10k labeled data from the training set and the rest training data samples were treated as unlabeled.

An extra dataset called "ILSVRC-150K" will be employed and introduced later for a further experimental analysis on the scalability of the proposed method over many classes and under imbalanced case.

Default Parameter Setting. The default values of the parameters in Criterion (23) were set below in our experiments: $v = 0.1, \alpha = 4, \beta = 5, \theta = 0.1$. We will report the importance of parameter values in Sec. IV-E3.

B. Evaluation Criteria

To evaluate the performance of different algorithms, we reported several measurements: precision, recall, and mean average precision(MAP). The evaluation criteria are defined as follows:

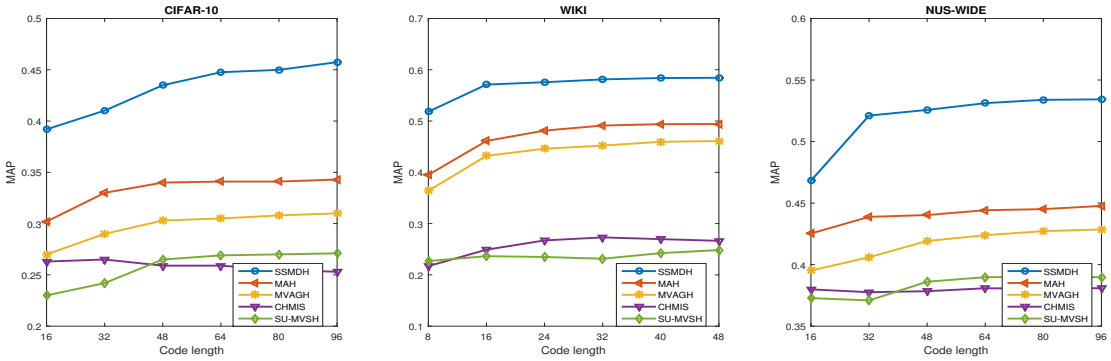

Fig. 1. MAP comparison with unsupervised multi-view hash models with different code lengths.

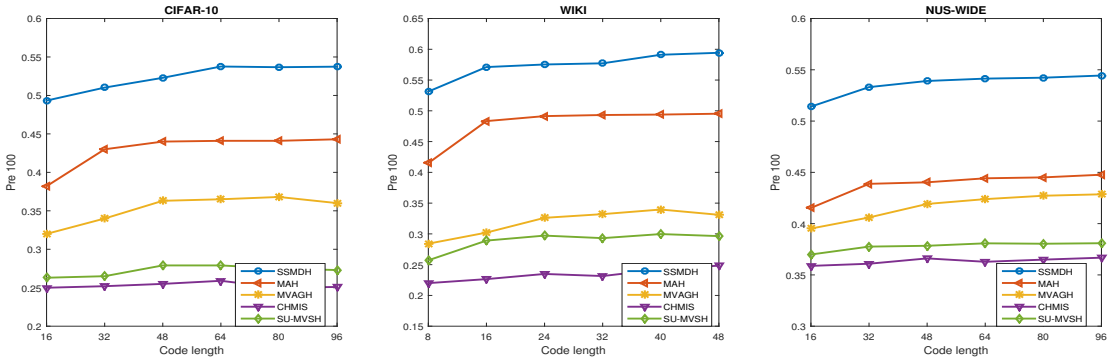

Fig. 2. Precision-100 comparison with unsupervised multi-view hash models with different code lengths.

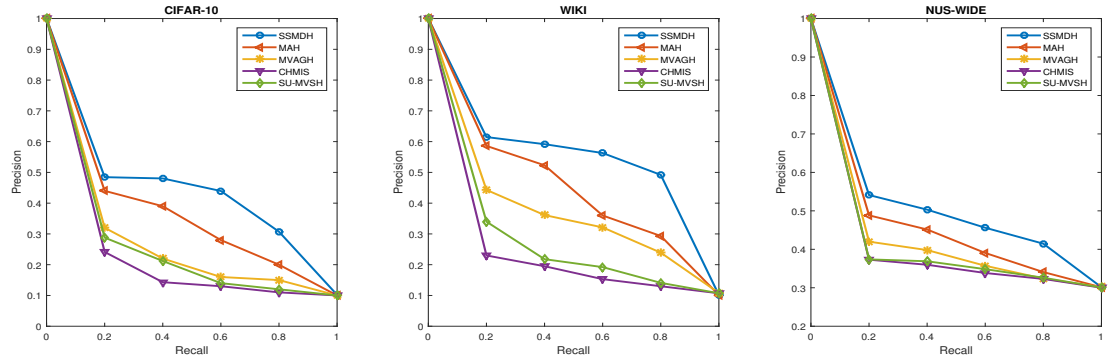

Fig. 3. Precision Recall Curves comparison with unsupervised multi-view hash models with different code lengths.

Precision-Recall. Precision and Recall are defined as below

$$Precison = \frac{tp}{tp + fp} \tag{32}$$

$$Recall = \frac{tp}{tp + fn} \tag{33}$$

where tp is the number of similar points, fp is the number of non-similar points and fn is the number of similar points that are not retrieved.

MAP. MAP has a good stability to evaluate the performance of Hamming ranking. It is defined as follows:

$$mAP = \frac{1}{Q}\sum_{i=1}^{Q} AP(q_i) \tag{34}$$

$$AP(q) = \frac{1}{M}\sum_{r=1}^{R} P_q(r)\mu(r) \tag{35}$$

where Q is the number of queries and P_q is the precision for query q when the top r^{th} neighbors returned, $\mu(r)$ is an indication function which is 1 when the r^{th} result has the same class label with q and otherwise 0, M is the number of true neighbors of query q, and R is the size of dataset.

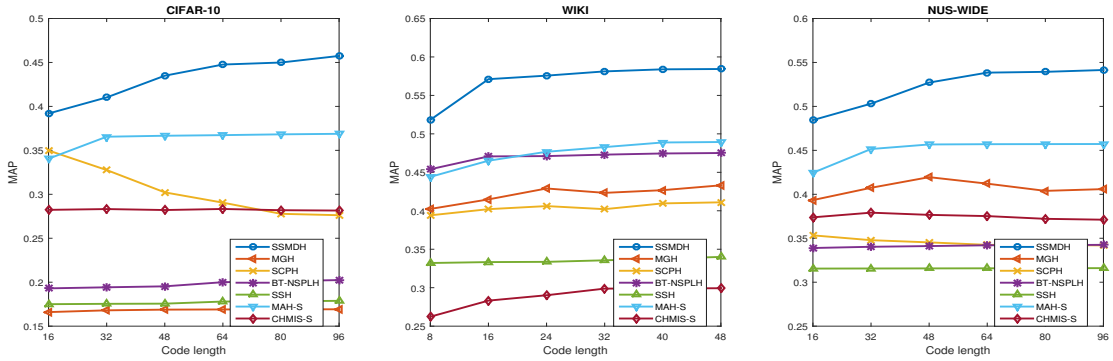

Fig. 4. MAP comparison with semi-supervised hash models with different code lengths.

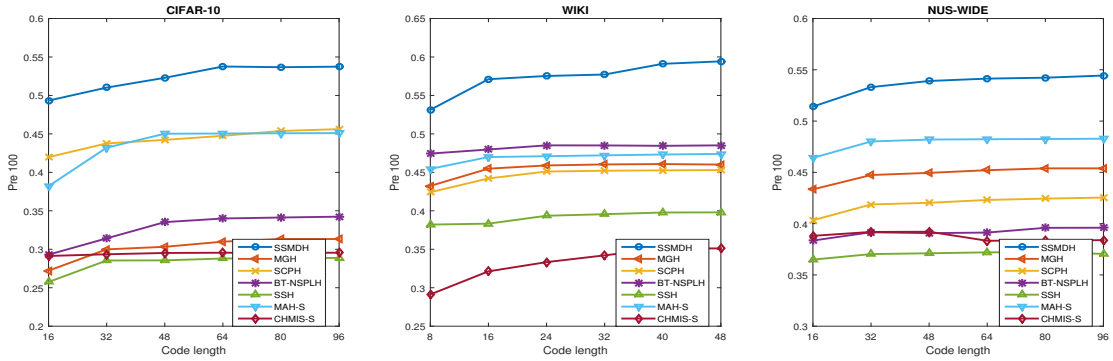

Fig. 5. Precision-100 comparison with semi-supervised hash models with different code lengths.

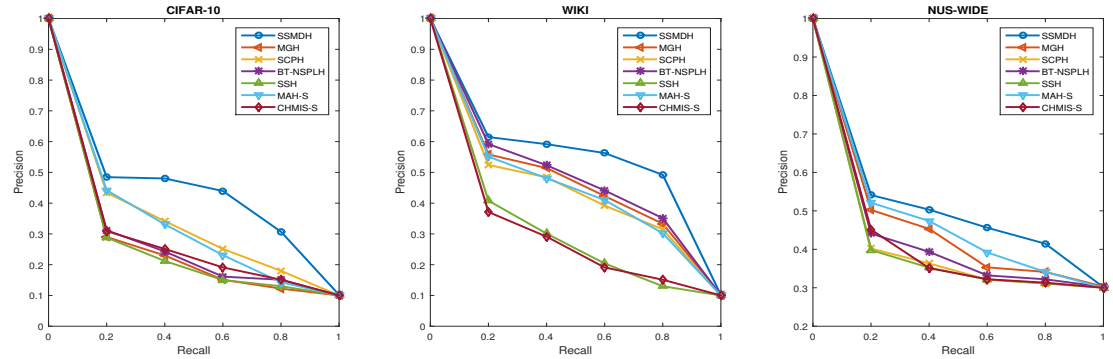

Fig. 6. Precision-Recall Curves comparison with semi-supervised hash models with different code lengths.

We reported the experiments on datasets CIFAR-10 and NUS-WIDE by varying the bits from 16 to 96, and 8 to 48 bits for dataset WIKI since WIKI only contains thousands of training samples. All experiments were independently run 10 times, and the average results were reported. All experiments were all run on a workstation with 24 Intel(R) Xeon(R) E5-2620@2.0GHz CPUs, 96GB RAM and 64-bit Ubuntu system, and all methods were evaluated under the same measurement.

C. Comparison with Unsupervised Multi-view Hash Models

We first mainly compared several state-of-the-art unsupervised multi-view hashing algorithms: (1) MAH [18], (2) MVAGH [11], (3) CHMIS [38], and (4) SU-MVSH [12]. For all compared methods, we implemented them using the recommended parameters provided by authors.

We reported the results in Figures 1, 2 and 3, and our method achieved the best performance on three datasets. For example, on CIFAR-10, our model achieved 44.7% on MAP when the code length was 64 bits while the second highest MAP was 34.2%. It is reasonable since a set of labeled data was used for our multi-view modeling in the experiments, and indeed the results suggest using a small set of labeled data can improve the accuracy of fast search a lot. We also compared MAH visually on the retrieved images in Figure 12. The results showed that our results were more likely to be similar to

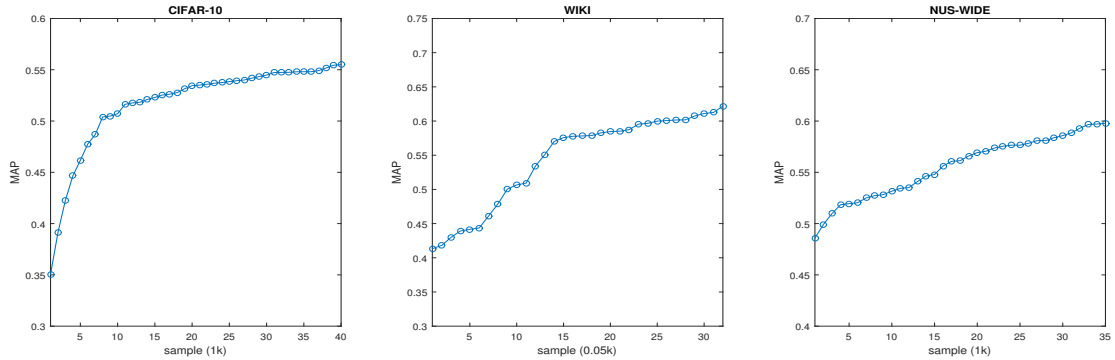

Fig. 7. Illustration of the effect of number of labeled samples on the search performance (MAP).

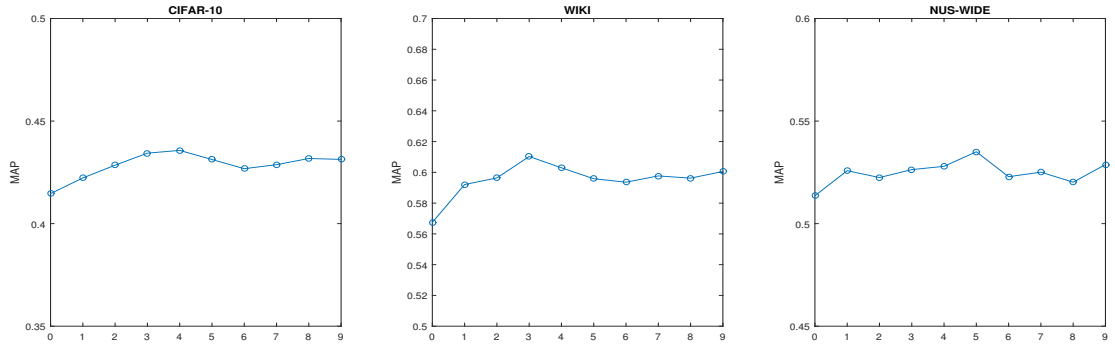

Fig. 8. Illustration of the effect of trade-off parameter α on the search performance (MAP).

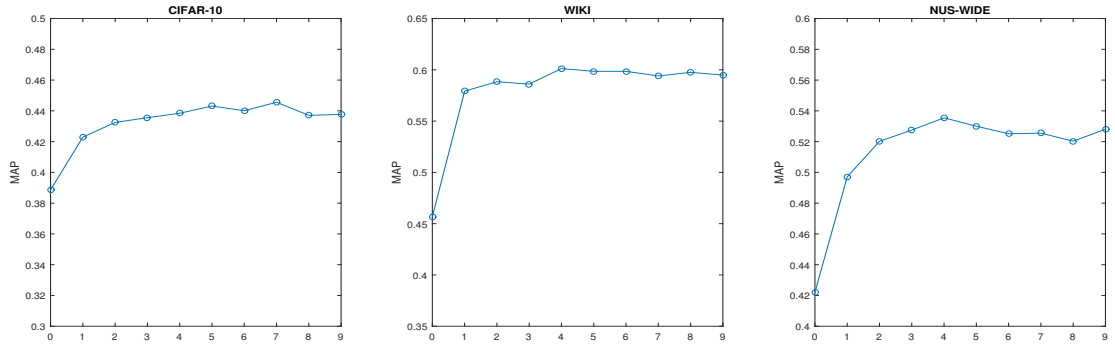

Fig. 9. Illustration of the effect of trade-off parameter β on the search performance (MAP).

the query ones. For the comparison with DMHOR [31], we compared SSMDH with it on WIKI, since only results on WIKI under the same setting were reported in [31]. On WIKI, the MAP of DMHOR is 0.3424, 0.489, and 0.5268 when the code length is 8, 16, and 32, respectively; in comparison, the MAP of the proposed SSMDH is 0.5182, 0.5711, and 0.5812 when the code length is 8, 16, and 32, respectively. It suggested that SSMDH performed much better when the code length is small.

D. Comparison with Semi-supervised Hash Models

We compared with several state-of-the-art semi-supervised hash methods, including SSH [33], BT-NSPLH [36], SCPH [32] and MGH [8]. We evaluated the "MAP vs. Code Length", "Precision 100 vs. Code Length", and "Precision vs. Recall" of each method on three datasets. For comparing with single-view semi-supervised hash models, we reported their results on the concatenation of "HOG" and "GIST", or the concatenation of "IMG" and "TXT". The results were shown in Figures 4, 5 and 6. They showed that our approach was clearly better than the compared methods. On "MAP vs. Code Length", the proposed method always outperformed the second best about 10% MAP rates; on "Precision 100 vs. Code Length", a similar margin was also observed; on "Precision vs. Recall", the precision of the proposed method degraded much less when the recall rate increased. On dataset

NUS-WIDE, MGH reached the second best accuracy since it used multi-view data to generate a multi-graph solution to optimize weights between different views/modalities on semi-supervised hashing. In addition, we found out that although SCPH, SSH, and BT-NSPLH have already directly incorporated the similar pairs and non-similar pairs into the modeling, SSH has lower performance due to the quantization error when converting into binary code, and BT-NSPLH ignores quantization error of the overwhelming majority unlabeled data points. We also compared with SSH visually about the retrieved images in Figure 12. The results showed that our results were more accurate. Even though an inaccurate retrieved image (a "bird") for the query (a "plane") was observed when using our proposed SSMDH, their appearances are very similar.

TABLE II
EVALUATION ON PROPOSED SSMDH: MAP ON CIFAR-10

	SSMDH	GIST	HOG	Naive Fusion
16 bits	**0.3919**	0.3632	0.3543	0.3575
32 bits	**0.4102**	0.3801	0.3892	0.3847
48 bits	**0.4349**	0.3905	0.3902	0.3921
64 bits	**0.4479**	0.3909	0.3924	0.3930
80 bits	**0.4574**	0.3980	0.3965	0.3992

TABLE III
EVALUATION ON PROPOSED SSMDH: MAP ON WIKI

	SSMDH	IMG(SIFT)	TXT	Naive Fusion
8 bits	**0.5182**	0.2524	0.4273	0.3075
16 bits	**0.5711**	0.2623	0.5054	0.4495
24 bits	**0.5757**	0.2618	0.5281	0.4405
32 bits	**0.5812**	0.2621	0.5317	0.4503
40 bits	**0.5839**	0.2629	0.5319	0.4543

TABLE IV
EVALUATION ON PROPOSED SSMDH: MAP ON NUS-WIDE

	SSMDH	IMG(SIFT)	TXT	Naive Fusion
16 bits	**0.4844**	0.4334	0.4726	0.4586
32 bits	**0.5031**	0.4547	0.4619	0.4644
48 bits	**0.5271**	0.4691	0.4664	0.4749
64 bits	**0.5384**	0.4752	0.4649	0.4723
80 bits	**0.5393**	0.4710	0.4683	0.4772

For more analysis, we also generalized the unsupervised multi-view hash model MAH and CHMIS to their semi-supervised cases, denoted by MAH-S and CHMIS-S, respectively. The results were also shown in Figures 4, 5 and 6. MAH performed the second best in the last section, and we compared CHMIS-S since CHMIS also estimates hash codes in a composite manner. MAH-S (CHMIS-S) was formed by re-defining the matrix $\mathbf{K}$(the similarity matrix $\mathbf{S}$), where the entry is 1 when two labeled samples are from the same class, 0 when two labeled samples are from different classes, and keeps the same as the case in MAH (CHMIS) in the other cases. Although MAH always performed as a second best unsupervised multi-view hash model, the straightforward semi-supervised extension MAH-S is not effective. In despite of sharing composite modeling between CHMIS and our proposed model, the semi-supervised extension CHIMIS-S did not perform well. On one hand, it is because MAH does not consider the relation of hash codes extracted between different views, while our proposed model does; and on the other hand, we propose a composite discrete hash model which is able to minimize the loss when using relaxation on learning hashing codes.

In summary, our method is more effective than the compared semi-supervised methods in the aspect of using multi-view data. Our results show that 1) a multi-view semi-supervised modeling is better than a single-view based, and 2) a joint multi-view quantification on "HOG" and "GIST" is more effective than simple concatenation of "HOG" and "GIST". Note that even though compared to our degraded semi-supervised multi-view discrete hashing on single view[1] in Tables II, III and IV, our degraded model still performed better overall. In addition, the performance of some straightforward extension of unsupervised multi-view methods is inferior to ours clearly.

E. More Evaluation on the Proposed Model

1) The Effect of Using Multi-view Data: We firstly directly evaluated our semi-supervised hashing when learned on each view (i.e., the "GIST", "HOG", "TXT") in Tables II, III and IV. That is we performed our model on each view when setting $\alpha = 0$. The results indicate that a clear improvement would be gained when multi-view information is used and learned jointly.

Secondly, we evaluated our joint multi-view learning. A naive way to perform semi-supervised multi-view hash model is to perform the semi-supervised hashing on each view and fuse them together. That is Eq. (23) would become

$$\begin{aligned}\min_{\mathbf{B},\mathbf{C},\{\mathbf{W}_i\}_{i=1}^K} &\sum_{i=1}^{K}\frac{1}{n_\ell}||\mathbf{Y}-\mathbf{C}^T\mathbf{B}_{i,\ell}||^2 \\ &+\frac{v}{n}||\mathbf{B}_i-\mathbf{W}_i^T\phi_i(\mathbf{X}^{(i)})||^2 \\ &+\beta\cdot trace\big(\sum_{i=1}^{K}\mathbf{W}_i^T\mathbf{L}_{ii}\mathbf{W}_i\big)+\theta||\mathbf{C}||^2,\end{aligned} \tag{36}$$

where $\mathbf{B}_i$ is the hash code matrix of the i^{th} view and $\mathbf{B}_{i,\ell}$ is the corresponding labeled portion. In the above, we model the local data variation in each view separately and set $\alpha = 0$, and finally the prediction model Eq. (13) is used for generating hash codes. We call this the naive semi-supervised multi-view hash model, denoted as "Naive Fusion". In comparison, our proposed model is a joint learning model of different views. We reported the comparison results in Tables II, III, and IV. The results suggest that a joint modeling is better than learning different views separately.

2) The Effect of Amount of Labeled Data: To further demonstrate the effectiveness and extension of our method, we evaluated the MAP against the number of labeled samples used in our semi-supervised hash model. The results were showed in Figure 7. On CIFAR-10 and WIKI, the performance of our method increased when more labeled data samples were used. For example, on CIFAR-10, the MAP was 0.43

[1]For implementation of our method on single view, we set $K = 1$, $\alpha = 1$ in the Criterion 23.

TABLE V
MAP EVALUATION ON CIFAR-10(CNN+HOG+GIST)

Code Length	16	32	48	64	80	96
SSMDH	**0.5918**	**0.6040**	**0.6351**	**0.6461**	**0.6471**	**0.6483**
MAH	0.5303	0.5467	0.5820	0.5924	0.5943	0.5961
MVAGH	0.3421	0.3915	0.3991	0.4038	0.4091	0.4131
CHMIS	0.3044	0.3623	0.3870	0.3896	0.3904	0.3910
SU-MVSH	0.2758	0.2754	0.2743	0.2731	0.2720	0.2727
MGH	0.1744	0.1805	0.2141	0.2219	0.2240	0.2255
SCPH	0.3952	0.3849	0.3753	0.3431	0.2716	0.2701
BT-NSPLH	0.2083	0.2157	0.2320	0.2405	0.2423	0.2431
SSH	0.1939	0.2048	0.2257	0.2333	0.2361	0.2380
MAH-S	0.3520	0.4097	0.4148	0.4281	0.4342	0.4388
CHMIS-S	0.2838	0.2873	0.2893	0.2901	0.2909	0.2915

TABLE VI
MAP EVALUATION ON NUS-WIDE(CNN+SIFT+TXT)

Code Length	16	32	48	64	80	96
SSMDH	**0.6383**	**0.6671**	**0.6895**	**0.6987**	**0.7073**	**0.7129**
MAH	0.5845	0.6149	0.6382	0.6502	0.6614	0.6687
MVAGH	0.5649	0.5893	0.5990	0.6032	0.6067	0.6091
CHMIS	0.4458	0.4531	0.4562	0.4580	0.4588	0.4592
SU-MVSH	0.4281	0.4366	0.4398	0.4403	0.4389	0.4371
MGH	0.5431	0.5586	0.5548	0.5560	0.5563	0.5567
SCPH	0.5459	0.5340	0.5295	0.5248	0.5224	0.5209
BT-NSPLH	0.5146	0.5263	0.5279	0.5281	0.5284	0.5283
SSH	0.4739	0.4752	0.4756	0.4760	0.4762	0.4758
MAH-S	0.5714	0.5925	0.6003	0.6041	0.6068	0.6072
CHMIS-S	0.4618	0.4780	0.4825	0.4841	0.4862	0.4867

when 4k training samples were used and it was 0.5 when 10k training samples were used. As shown by the figures, the MAP increased slightly when more than 15k training samples were used. Similar observation was found on WIKI. On NUS-WIDE, the performance disturbance happened when more labeled data samples were used. This might be because NUS-WIDE consists of multi-label data, and sometimes using more data samples will increase the diversity of labels of data.

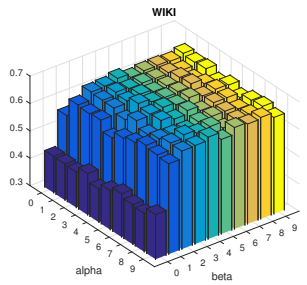

Fig. 10. MAP results against joint variation of α and β on WIKI.

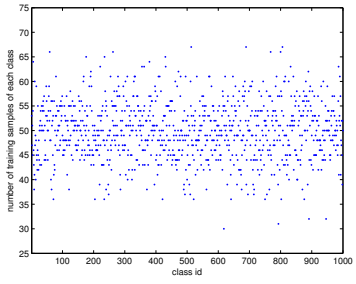

Fig. 11. Number of training data of each class on ILSVRC-150K in the imbalanced setting.

3) The Effect of Parameters: We evaluated the influence of two main parameters α and β of the proposed model on three datasets. When studying one parameter, the others were fixed as default values.

Parameter α: Figure 8 showed the MAP of SSMDH against parameter α from 0 to 9. We found that the performance increased when α was less than 4 on CIFAR-10. The result also showed that the best α for these three datasets was sightly different. On CIFAR-10 the best α value was 4, while on WIKI the best α value was 3. Generally speaking, we can set α between 3 and 5 for learning. The results also suggested that lowest MAP results were always obtained when the extracted features from different views were not decorrelated, and this showed the effectiveness of the statistical uncorrelation modeling.

Parameter β: Figure 9 showed the MAP of SSMDH against parameter β from 0 to 9. As shown, when β was 0, the performance was much lower than the ones when $\beta > 0$. It verified that making hash codes more compact by minimizing composite local data variation. In addition, a recommended β from the results is 4.

Finally, by taking WIKI dataset as example, we examined the effect of both parameters jointly in Figure 10. The results suggested the performance of our method was relatively reliable when not setting $\alpha = 0$ and $\beta = 0$, and the recommended setting is still around $\alpha = 3$ and $\beta = 4$.

F. Indepth Analysis

When Using Deep Features: We took CIFAR-10 and NUS-WIDE as examples to show below. For evaluation, we further employed the features extracted by CNN as the third view of the data and integrated it into our multi-view framework. The results are reported in Tables V and VI. Indeed, compared to Figures 1 and 4, the performance of some methods can be improved a lot when using deep features, and especially the proposed SSMDH gains 20% MAP rates more on CIFAR-10. The improvement of our method when using deep features is more clear than the others. The results again suggest that the proposed multi-view strategy is effective in utilizing multi-view features. The results also show that our proposed semi-supervised multi-view hash model still outperforms the compared semi-supervised and unsupervised multi-view hash models.

Searching on Many Classes: While the employed datasets in the previous experiments are popular for evaluation of hashing methods, the number of classes in those datasets is limited. To show the evaluation of SSMDH on fast search over many classes, by following [17], we formed a subset consists of 1000 classes, which are very diverse and from different categories, where the training set consists of 150k data samples (150 images per class) and the labeled data set consists of 10k

Fig. 12. Comparisons of some retrieval results among SSMDH, MAH and SSH on CIFAR-10. On each row, the left are the query, and the right are retrieved images ranked from left to right.

TABLE VII
MAP EVALUATION ON ILSVRC-150K

Code Length	32	64	96	128	160
SSMDH	**0.1120**	**0.2027**	**0.2418**	**0.2460**	**0.2571**
MAH	0.0897	0.1082	0.1124	0.1206	0.1258
MVAGH	0.0734	0.0859	0.0980	0.1064	0.1113
CHMIS	0.0342	0.0358	0.0366	0.0378	0.0375
SU-MVSH	0.0298	0.0303	0.0312	0.0316	0.0321
MGH	0.0699	0.0733	0.0758	0.0772	0.0778
SCPH	0.0645	0.0691	0.0739	0.0765	0.0773
BT-NSPLH	0.0378	0.0407	0.0454	0.0467	0.0479
SSH	0.0143	0.0159	0.0167	0.0171	0.0173
MAH-S	0.0972	0.1211	0.1259	0.1283	0.1301
CHMIS-S	0.0349	0.0368	0.0387	0.0392	0.0397

TABLE VIII
PRECISION-50 EVALUATION ON ILSVRC-150K

Code Length	32	64	96	128	160
SSMDH	**0.2373**	**0.2662**	**0.2810**	**0.2935**	**0.3033**
MAH	0.2024	0.2278	0.2409	0.2446	0.2492
MVAGH	0.1825	0.2074	0.2138	0.2170	0.2199
CHMIS	0.0782	0.0846	0.0878	0.0894	0.0925
SU-MVSH	0.0565	0.0603	0.0637	0.0658	0.0664
MGH	0.1737	0.1940	0.2013	0.2046	0.2058
SCPH	0.1479	0.1627	0.1801	0.1856	0.1879
BT-NSPLH	0.0541	0.0657	0.0687	0.0693	0.0706
SSH	0.0138	0.0142	0.0145	0.0145	0.0144
MAH-S	0.2295	0.2604	0.2786	0.2850	0.2881
CHMIS-S	0.0392	0.0451	0.0468	0.0470	0.0465

data samples (10 images per class). We denote this subset as "ILSVRC-150K". We extracted 17 and 18 layer features of VGG net in our experiment to form multi-view features. For single view methods, we concatenated layers 17 and 18 to form the feature representation. In our experiment, 1000 data samples were selected to form the testing dataset and the rest were considered as the training samples. The MAP results and the Precision-50 results in Tables VII and VIII show that our method also performed better than the compared ones on ILSVRC-150K.

When Samples are Imbalanced over Many Classes. For evaluation, we kept on conducting experiment on ILSVRC-150K. Since the training set of ILSVRC-150K is balanced, to make the training set imbalanced, we randomly selected 50k samples from the training set used in the balanced case to form the imbalanced training dataset and the same 10k samples as used in the balanced case to form the labeled dataset. The random selection made the amount of training samples of different classes imbalanced. The distribution of the number of training samples in each class is shown in Figure 11. The MAP results and the Precision-50 results are shown in Table IX and X. As shown, compared to Tables VII and VIII which report the results when sample distribution is balanced, SSMDH performed more stably in the imbalanced case.

TABLE IX
MAP EVALUATION ON IMBALANCED ILSVRC-150K

Code Length	32	64	96	128	160
SSMDH	**0.1017**	**0.1916**	**0.2439**	**0.2655**	**0.2710**
MAH	0.0901	0.1357	0.1422	0.1460	0.1473
MVAGH	0.0741	0.0949	0.1018	0.1040	0.1199
CHMIS	0.0287	0.0301	0.0325	0.0379	0.0404
SU-MVSH	0.0254	0.0331	0.0380	0.0392	0.0408
MGH	0.0534	0.0668	0.0722	0.0760	0.0752
SCPH	0.0514	0.0659	0.0723	0.0737	0.0745
BT-NSPLH	0.0328	0.0377	0.0391	0.0403	0.0410
SSH	0.0124	0.0129	0.0134	0.0138	0.0136
MAH-S	0.0954	0.1236	0.1259	0.1301	0.1306
CHMIS-S	0.0299	0.0307	0.0316	0.0331	0.0338

Comparison with Classifier-based Hashing. Recently, classifier based hashing draws attention [26]. Its main idea is to learn a classifier model based on the available labeled data samples, use it to predict a probability vector for each query, and then apply one-hot strategy or Locality Sensitive Hashing(LSH) to encode a short length binary code [26], where one-hot can encode a $log_2 m$ length code and m is the number of

TABLE X
PRECISION-50 EVALUATION ON IMBALANCED ILSVRC-150K

Code Length	32	64	96	128	160
SSMDH	**0.2315**	**0.2648**	**0.2791**	**0.2900**	**0.3002**
MAH	0.1832	0.1956	0.2027	0.2065	0.2077
MVAGH	0.1734	0.1886	0.2013	0.2091	0.2126
CHMIS	0.0592	0.0630	0.0679	0.0701	0.0724
SU-MVSH	0.0476	0.0493	0.0512	0.0528	0.0547
MGH	0.1640	0.1802	0.1935	0.1989	0.2017
SCPH	0.1326	0.1543	0.1674	0.1731	0.1778
BT-NSPLH	0.0347	0.0383	0.0406	0.0411	0.0427
SSH	0.0102	0.0124	0.0131	0.0134	0.0136
MAH-S	0.1873	0.1994	0.2108	0.2176	0.2204
CHMIS-S	0.0658	0.0670	0.0684	0.0692	0.0698

classes. We followed Jegou et al.'s work [26] to infer hash code from a classifier output. For CIFAR-10 and WIKI, we infer the hash code based on the multi-class logistic regression method [26]; for NUS-WIDE, a multi-label classifier [6] was used, since NUS-WIDE is a multi-label database; and for ILSVRC-150K, we applied the Multi-class SVM classifier due to the difficulty of learning a stable logistic regression model with large number of classes (1000 classes) and very limited labeled samples (only 10 labeled samples for each class). Note that for CIFAR-10 and NUS-WIDE, we used the same feature as used in Table V and Table VI in the main manuscript, respectively. The results are shown in Table XI. Indeed, the "Classifier+ONE-HOT" is useful (especially on NUS-WIDE and WIKI), and it outperformed many our compared methods in Figure 4 and Tables V and VI. But, the results also suggest that our proposed SSMDH can still perform better on CIFAR-10 and ILSVRC-150K, and perform comparably on WIKI and NUS-WIDE.

Transfer case. In order to see how well a hash model learned on a set of classes can be used to conduct fast search on another separate set of new classes, we followed Jegou et al.'s work [26] to split dataset into two parts without class overlap: train75 and train25/test25, where train75 indicates the training set to train the hash models and train25/test25 is the gallery image set/query image set. We only conducted experiments on CIFAR-10 and ILSVRC-150K, since WIKI is too small (only 2866 samples) and it is hard to find two sets of classes without sample overlap due to the fact that each sample in NUS-WIDE belongs to several classes under a multi-label setting. For CIFAR-10, the train75 consists of data of 7 classes, and the data samples of the rest 3 classes form the train25/test25. For ILSVRC-150k, the train75 consists of data of 750 classes, and data samples of the rest 250 classes form the train25/test25. For each dataset, the test25 consists of 1000 query samples and the others form the train25. The MAP results are shown in Table XII below. The results suggest the proposed SSMDH still outperformed the compared methods under the transfer setting. Note that the MAP results in Table XII are larger than the ones reported in other tables, and it is because less classes are set to search in the testing stage under the transfer setting.

TABLE XI
MAP EVALUATION ON CLASSIFIER-BASED HASHING

Database	Method	Code Length	MAP
CIFAR-10	Classifier+ONE-HOT	4	0.4235
	Classifier+LSH	64	0.4542
	SSMDH	64	**0.6461**
WIKI	Classifier+ONE-HOT	4	**0.5938**
	Classifier+LSH	16	0.5816
	SSMDH	16	0.5711
NUS-WIDE	Classifier+ONE-HOT	4	0.6570
	Classifier+LSH	64	0.6394
	SSMDH	64	**0.6987**
ILSVRC-150K	Classifier+ONE-HOT	10	0.0536
	Classifier+LSH	64	0.0482
	SSMDH	64	**0.2027**

TABLE XII
MAP EVALUATION ON TRANSFER CASE

Database	CIFAR-10			ILSVRC-150K		
Code Length	32	64	96	32	64	96
SSMDH	**0.7200**	**0.7214**	**0.7227**	**0.4065**	**0.4144**	**0.4225**
MAH	0.6684	0.6732	0.6741	0.3253	0.3382	0.3402
MVAGH	0.6706	0.6683	0.6620	0.3447	0.3581	0.3639
CHMIS	0.4526	0.4569	0.4611	0.1841	0.1878	0.1890
SU-MVSH	0.4024	0.4110	0.4123	0.1394	0.1386	0.1381
MGH	0.4418	0.4450	0.4456	0.3062	0.3104	0.3149
SCPH	0.5315	0.5247	0.5174	0.2859	0.2900	0.2923
BT-NSPLH	0.4727	0.4739	0.4746	0.1455	0.1527	0.1540
SSH	0.4312	0.4427	0.4475	0.0942	0.0955	0.0959
MAH-S	0.6869	0.6904	0.6918	0.3438	0.3511	0.3537
CHMIS-S	0.4635	0.4658	0.4666	0.2003	0.2015	0.2019

V. CONCLUSION

We have proposed a semi-supervised composite multi-view discrete hash (SSMDH) model. SSMDH minimizes the loss jointly when using relaxation on learning hashing codes for similarity search on multi-view data, and meanwhile it increases the discriminant ability of the learned hash codes by reducing regression loss on a portion of labeled samples. In this development, all view-specific transformations are learned jointly over multi-view data, while reducing the statistical correlation between them and imposing composite locality preserving constraint on hash codes. In summary, through extensive experiments we find that:

1) A composite discrete hash model on multi-view data performs more effectively than learning discrete hash model for each view independently;
2) The proposed strategy of extracting features on multi-view data for hash model is more effective than the existing multi-view modeling for hashing under the semi-supervised setting;
3) It is empirically found that straightforward extension of some effective unsupervised multi-view hash models does not suit the semi-supervised hashing very well.

In the future, since incorporating deep features would clearly and sometimes significantly improve the performance of the proposed semi-supervised multi-view hashing, it would be interesting to investigate whether a completely end-to-end semi-supervised multi-view model would gain much more improvement.

ACKNOWLEDGMENTS

The authors would like to thank reviewers' constructive comments on improving the manuscript.

REFERENCES

[1] S.-i. Amari. Integration of stochastic models by minimizing α-divergence. *Neural Computation*, 19(10):2780–2796, 2007.
[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.
[3] M. M. B. P. F. Christoph Strecha, Alexander M Bronstein. Ldahash: Improved matching with smaller descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(1):66–78.
[4] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng. Nus-wide: a real-world web image database from national university of singapore. In *Proceedings of the ACM international conference on image and video retrieval*.
[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 886–893, 2005.
[6] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. 2001.
[7] K. He, F. Wen, and J. Sun. K-means hashing: An affinity-preserving quantization method for learning binary compact codes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2938–2945, 2013.
[8] P. L. M. W. H. L. Jian Cheng, Cong Leng. Semi-supervised multi-graph hashing for scalable similarity search. *Computer Vision and Image Understanding*, 124:12–21, 2014.
[9] Z. Jin, Y. Hu, Y. Lin, D. Zhang, S. Lin, D. Cai, and X. Li. Complementary projection hashing. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 257–264, 2013.
[10] Y. Kang, S. Kim, and S. Choi. Deep learning to hash with multiple representations. In *IEEE International Conference on Data Mining*, pages 930–935, 2012.
[11] S. Kim and S. Choi. Multi-view anchor graph hashing. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3123–3127, 2013.
[12] S. Kim, Y. Kang, and S. Choi. Sequential spectral learning to hash with multiple representations. In *European Conference on Computer Vision*, pages 538–551. 2012.
[13] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Toronto, 2009.
[14] B. Kulis and T. Darrell. Learning to hash with binary reconstructive embeddings. In *Advances in neural information processing systems*, pages 1042–1050, 2009.
[15] B. Kulis, P. Jain, and K. Grauman. Fast similarity search for learned metrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2143–2157, 2009.
[16] X. Li, G. Lin, C. Shen, A. Van den Hengel, and A. Dick. Learning hash functions using column generation. In *Proceedings of The 30th International Conference on Machine Learning*, pages 142–150, 2013.
[17] H. Liu, R. Wang, S. Shan, and X. Chen. Dual purpose hashing. *arXiv preprint arXiv:1607.05529*, 2016.
[18] L. Liu, M. Yu, and L. Shao. Multiview alignment hashing for efficient image search. *IEEE Transactions on Image Processing*, 24(3):956–966, 2015.
[19] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang. Supervised hashing with kernels. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2074–2081, 2012.
[20] W. Liu, J. Wang, S. Kumar, and S.-F. Chang. Hashing with graphs. In *Proceedings of the 28th international conference on machine learning*, pages 1–8, 2011.
[21] X. Liu, D. Tao, M. Song, Y. Ruan, C. Chen, and J. Bu. Weakly supervised multiclass video segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 57–64, 2014.
[22] D. G. Lowe. Object recognition from local scale-invariant features. In *The proceedings of the seventh IEEE international conference on Computer vision*, volume 2, pages 1150–1157, 1999.
[23] M. Norouzi and D. M. Blei. Minimal loss hashing for compact binary codes. In *Proceedings of the 28th international conference on machine learning*, pages 353–360, 2011.
[24] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3):145–175, 2001.
[25] N. Rasiwasia, J. Costa Pereira, E. Coviello, G. Doyle, G. R. Lanckriet, R. Levy, and N. Vasconcelos. A new approach to cross-modal multimedia retrieval. In *Proceedings of the international conference on Multimedia*, pages 251–260, 2010.
[26] A. Sablayrolles, M. Douze, H. Jégou, and N. Usunier. How should we evaluate supervised hashing? *CoRR*, abs/1609.06753, 2016.
[27] F. Shen, C. Shen, W. Liu, and H. T. Shen. Supervised discrete hashing. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 37–45, 2015.
[28] F. Shen, C. Shen, Q. Shi, A. Hengel, and Z. Tang. Inductive hashing on manifolds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1562–1569, 2013.
[29] D. Song, W. Liu, D. A. Meyer, D. Tao, and R. Ji. Rank preserving hashing for rapid image search. In *Data Compression Conference*, pages 353–362, 2015.
[30] D. Song, W. Liu, T. Zhou, D. Tao, and D. A. Meyer. Efficient robust conditional random fields. *IEEE Transactions on Image Processing*, 24(10):3124–3136, 2015.
[31] D. Wang, P. Cui, M. Ou, and W. Zhu. Deep multimodal hashing with orthogonal regularization. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 2291–2297, 2015.
[32] D. Wang, X. Gao, and X. Wang. Semi-supervised constraints preserving hashing. *Neurocomputing*, 167:230–242, 2015.
[33] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for scalable image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3424–3431, 2010.
[34] Y. Wang, X. Lin, L. Wu, W. Zhang, and Q. Zhang. Lbmch: Learning bridging mapping for cross-modal hashing. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 999–1002. ACM, 2015.
[35] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Advances in neural information processing systems*, pages 1753–1760, 2009.
[36] C. Wu, J. Zhu, D. Cai, C. Chen, and J. Bu. Semi-supervised nonlinear hashing using bootstrap sequential projection learning. *IEEE Transactions on Knowledge and Data Engineering*, 25(6):1380–1393, 2013.
[37] Y. H. P. N. H. Z. Xiaofei He, Shuicheng Yan. Face recognition using laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):328–340, 2005.
[38] D. Zhang, F. Wang, and L. Si. Composite hashing with multiple information sources. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 225–234, 2011.

Chenghao Zhang received the B.S. degree in computer science from Sun Yat-sen University in 2016. He is pursuing M.S. degree in Carnegie Mellon University and Sun Yat-sen University. His current research insterests include large-scale machine learning algorithms and action recognition on Austim children behavior analysis.

Wei-Shi Zheng received the PhD degree in applied mathematics from Sun Yat-sen University in 2008. He is a Professor with Sun Yat-Sen University. He has been a postdoctoral researcher on the EU FP7 SAMURAI Project with Queen Mary University of London. His recent research interests include person re-identification, action/activity recognition, and large-scale machine learning algorithms. He has joined Microsoft Research Asia Young Faculty Visiting Programme. He has outstanding reviewer award in ECCV 2016. He is a recipient of the Excellent Young Scientists Fund of the National Natural Science Foundation of China, and a recipient of Royal Society-Newton Advanced Fellowship.