Learning to Hash with Partial Tags: Exploring Correlation between Tags and Hashing Bits for Large Scale Image Retrieval

Qifan Wang[1], Luo Si[1], and Dan Zhang[2]

[1] Department of Computer Science, Purdue University
West Lafayette, IN, USA, 47907-2107
{wang868,lsi}@purdue.edu

[2] Facebook Incorporation, Menlo Park, CA 94025, USA
danzhang@fb.com

Abstract. Similarity search is an important technique in many large scale vision applications. Hashing approach becomes popular for similarity search due to its computational and memory efficiency. Recently, it has been shown that the hashing quality could be improved by combining supervised information, e.g. semantic tags/labels, into hashing function learning. However, tag information is not fully exploited in existing unsupervised and supervised hashing methods especially when only partial tags are available. This paper proposes a novel semi-supervised tag hashing (SSTH) approach that fully incorporates tag information into learning effective hashing function by exploring the correlation between tags and hashing bits. The hashing function is learned in a unified learning framework by simultaneously ensuring the tag consistency and preserving the similarities between image examples. An iterative coordinate descent algorithm is designed as the optimization procedure. Furthermore, we improve the effectiveness of hashing function through orthogonal transformation by minimizing the quantization error. Extensive experiments on two large scale image datasets demonstrate the superior performance of the proposed approach over several state-of-the-art hashing methods.

Keywords: Hashing, Tags, Similarity Search, Image Retrieval.

1 Introduction

Due to the explosive growth of the Internet, a huge amount of image data has been generated, which indicates that efficient similarity search becomes more important. Traditional similarity search methods are difficult to be directly used for large scale datasets since the computational cost of similarity calculation using the original visual features is impractical for large scale applications. Recently, hashing has become a popular approach in large scale vision problems including image retrieval [5], object recognition [20], image matching [19], etc. Hashing methods design compact binary codes for a large number of images so that visually similar images are mapped into similar codes. In the retrieving process, these hashing methods first transform query images into the corresponding

D. Fleet et al. (Eds.): ECCV 2014, Part III, LNCS 8691, pp. 378–392, 2014.

hashing codes and then similarity search can be simply conducted by calculating the Hamming distances between the codes of available images and the query, and selecting images within small Hamming distances.

Recently hashing methods have shown that the hashing performance could be boosted by leveraging supervised information into hashing function learning such as semantic tags/labels. Although existing hashing methods generate promising results in large scale similarity search, tag information is not fully exploited especially when tags are incomplete and noisy. Most of the existing hashing methods only utilize a small portion of the information contained in tags, e.g., pairwise similarity or listwise ranking information, which might not be accurate or reliable under the situation where only partial tags are available. There are three main challenges to incorporate tags into hashing function learning: (1) we have no knowledge about how tags are related to the hashing bits; (2) we need to deal with noisy and incomplete tags when only partial tags are available; (3) we need to deal with the ambiguity of semantically similar tags.

This paper proposes a novel semi-supervised tag hashing (SSTH) approach to fully exploits tag information in learning effective hashing function by modeling the correlation between tags and hashing bits. The hashing function is learned in a unified framework by simultaneously ensuring the tag consistency and preserving the similarities between image examples. In particular, the objective function of the proposed SSTH approach is composed of two parts: (1) Tag consistency term (supervised), which ensures the hashing codes to be consistent with the observed tags via modeling the correlation between tags and hashing bits. (2) Similarity preservation term (unsupervised), which aims at preserving the visual similarity between images in the learned hashing codes. An iterative algorithm is then derived based on the relaxed objective function using a coordinate descent optimization procedure. We further improve the quality of hashing function by minimizing the quantization error.

We summarize the contributions in this paper as follows: (1) propose a unified framework to incorporate the supervised tag information for jointly learning effective hashing function and correlation between tags and hashing codes; (2) propose a coordinate descent method for the relaxed joint optimization problem; (3) prove the orthogonal invariant property of the optimal relaxed solution and learn an orthogonal matrix by minimizing the quantization error to further improve the code effectiveness.

2 Related Work

Hashing methods [1,15,16,17,24,25,27,30,32] are proposed to generate reasonably accurate search results in a fast process with compact binary vector representation. Hashing based fast similarity search methods transform the original visual features into a low dimensional binary space, while at the same time preserve the visual similarity between images as much as possible. Existing hashing methods can be divided into two groups: unsupervised and supervised/semi-supervised hashing methods.

Among the unsupervised hashing approaches, Locality-Sensitive Hashing (LSH) [3] is one of the most popular methods, which uses random linear projections to map images from a high dimensional Euclidean space to a binary space. This method has been extended to Kernelized LSH [7] by exploiting kernel similarity. Traditional dimensionality reduction methods try to solve the hashing problem based on the original feature information via simple thresholding. For example, the Principle Component Analysis (PCA) Hashing [8] method represents each example by coefficients from the top k principal components of the training set, and the coefficients are further binarized to 1 or -1 based on the median value. Restricted Boltzman Machine (RBM) is used in [17] to generate compact binary hashing codes. Recently, Spectral Hashing (SH) [28] is proposed to design compact binary codes with balanced and uncorrelated constraints in the learned codes that preserve the similarity between data examples in the original space. The work in [11] proposes a graph-based hashing method to automatically discover the neighborhood structure inherent in the data to learn appropriate compact codes. More recently, a bit selection method [12] has been proposed to select the most informative hashing bits from a pool of candidate bits generated from different hashing methods.

For the supervised/semi-supervised hashing methods, a Canonical Correlation Analysis with Iterative Quantization (CCA-ITQ) method has been proposed in [4,5] which treats the image features and tags as two different views. The hashing function is then learned by extracting a common space from these two views. The work in [26] combines tag information with topic modeling by extracting topics from texts for document retrieval. Recently, several pairwise hashing methods have been proposed. The semi-supervised hashing (SSH) method in [22] utilizes pairwise knowledge between image examples besides their visual features for learning more effective hashing function. A kernelized supervised hashing (KSH) framework proposed in [10] imposes the pairwise relationship between image examples to obtain good hashing codes. Complementary Hashing (CH) [29] uses pairwise information to learn multiple complementary hash tables in a boosting manner. Most recently, a ranking-based supervised hashing (RSH) [23] method is proposed to leverage the listwise ranking information to improve the search accuracy. However, these pairwise/listwise information is usually extracted from the image tags, and thus only represents a small portion of tag information rather than the complete supervised information contained in tags. Moreover, tags may have different representations for a similar semantic meaning (e.g.,‘car’ versus ‘automobile’) and could be missing or incomplete, which makes the pairwise/listwise information not reliable.

3 Semi-Supervised Tag Hashing

3.1 Problem Setting

Assume there are total n training image examples. Let us denote their features as: $\boldsymbol{X} = \{x_1, x_2, \ldots, x_n\} \in R^{d\times n}$, where d is the dimensionality of the visual feature. Denote the observed/partial tags as: $\boldsymbol{T} = \{t_1, t_2, \ldots, t_l\} \in \{0,1\}^{n\times l}$, where l is the total number of possible tags for each image. A label 1 in $\boldsymbol{T}$ means an image is associated with a certain tag, while a label 0 means a missing tag or the tag is not associated with that image. The goal is to obtain a linear hashing function $f : \mathbb{R}^d \to \{-1, 1\}^k$, which maps image examples $\boldsymbol{X}$ to their binary hashing codes $\boldsymbol{Y} = \{y_1, y_2, \ldots, y_n\} \in \{-1, 1\}^{k \times n}$ (k is the length of hashing code). The linear hashing function is defined as:

$$y_i = f(x_i) = sgn(\boldsymbol{W}^T x_i) \tag{1}$$

where $\boldsymbol{W} \in \mathbb{R}^{d \times k}$ is the coefficient matrix representing the hashing function and sgn is the sign function. $y_i \in \{-1, 1\}^k$ is the binary hashing code of x_i.

The objective function of SSTH is composed of two components: (1) Tag consistency term, the supervised part which ensures that the hashing codes are consistent with the observed tags. (2) Similarity preservation term, the unsupervised part which aims at preserving the visual similarity in the learned hashing codes. In the rest of this section, we will present the formulation of these two components respectively. Then in the next section, we will describe the optimization algorithm together with a scheme that can further improve the quality of the hashing function by minimizing the quantization error.

3.2 Tag Consistency

Image data is often associated with various tags in many vision applications. These tag information provides useful supervised knowledge in learning effective hashing function. Therefore, it is necessary to design a scheme for leveraging tag information. There are three main challenges to incorporate tags. (1) We have no knowledge about how tags are related to the hashing bits. Therefore, we need to explore the correlation between them in order to bridge tags with hashing codes. (2) Tags could be partial and missing, and we need to deal with the situation of incomplete tags. (3) We need to deal with the ambiguity of semantically similar tags (e.g., ‘human’ versus ‘people’, ‘car’ versus ‘automobile’).

In this work, we propose to model the consistency between tags and hashing codes via matrix factorization using the latent factor model [21]. Semantically similar tags are represented by different tags (e.g., ‘human’ and ‘people’ are two distinct tags) in our model and we will discuss how this issue can be addressed later. In the latent factor model, a set of latent variables c_j for each tag t_j is first introduced to model the correlation between tags and hashing bits, where $j \in \{1, 2, \ldots, l\}$ and c_j is a $k \times 1$ vector indicating the correlation between the j-th tag and k hashing bits. Then a tag consistency component can be naturally formulated as:

$$\sum_{i=1}^{n} \sum_{j=1}^{l} \|\boldsymbol{T}_{ij} - y_i^T c_j\|^2 + \alpha \sum_{j=1}^{l} \|c_j\|^2 \tag{2}$$

here $\boldsymbol{T}_{ij}$ is the label of j-th tag on the i-th image. Intuitively, $y_i^T c_j$ can be essentially viewed as a weighted sum that indicates how the j-th tag is related to the i-th image, and this weighted sum should be consistent with the observed label $\boldsymbol{T}_{ij}$ as much as possible. $\sum_{j=1}^{l} \|c_j\|^2$ is a regularizer to avoid overfitting and α is the trade-off parameter. In this way, the latent correlation between tags and hashing bits can be learned by ensuring this consistency term.

The ambiguity issue for semantically similar tags is addressed by the latent factor model since these tags often appear in common images, and thus the learned corresponding latent variables will be similar by ensuring the tag consistency term. This can also be explained by the formulation above, which ensures the consistency between tag t and $\boldsymbol{Y}c$ (i.e., $t \approx \boldsymbol{Y}c$). Therefore, if two tags t_i and t_j are associated with similar images, their corresponding c_i and c_j will be close as well. In the extreme case, if two tags appear in exactly the same set of images, their latent variables will be identical.

An importance matrix $\boldsymbol{I} \in R^{n\times l}$ is introduced to deal with the missing tag problem. As mentioned above, $\boldsymbol{T}_{ij} = 0$ can be interpreted into two ways: *j-th* tag on the *i-th* image is either missing or not related. Therefore, we set $\boldsymbol{I}_{ij} = a$ with a higher value when $\boldsymbol{T}_{ij} = 1$ than $\boldsymbol{I}_{ij} = b$ when $\boldsymbol{T}_{ij} = 0$, where a and b are parameters satisfying $a > b > 0$[1]. Then the whole tag consistency term becomes:

$$\sum_{i=1}^{n}\sum_{j=1}^{l} \boldsymbol{I}_{ij}\|\boldsymbol{T}_{ij} - y_i^T c_j\|^2 + \alpha\sum_{j=1}^{l}\|c_j\|^2 \tag{3}$$

By substituting Eqn.1, the above equation can be rewritten as a compact matrix form:

$$\|\boldsymbol{I}^{\frac{1}{2}} \cdot (\boldsymbol{T} - sgn(\boldsymbol{X}^T\boldsymbol{W})\boldsymbol{C})\|_F^2 + \alpha\|\boldsymbol{C}\|_F^2 \tag{4}$$

where $\boldsymbol{I}^{\frac{1}{2}}$ is the element-wise square root matrix of $\boldsymbol{I}$, and $\cdot$ is the element-wise matrix multiplication. $\|\|_F$ is the matrix *Frobenius* norm and $\boldsymbol{C}$ is a $k \times l$ correlation matrix bridging the hashing codes with tags. By minimizing this term, the consistency between tags and the learned hashing codes is ensured.

3.3 Similarity Preservation

One of the key problems in hashing algorithms is similarity preserving, which indicates that visually similar images should be mapped to similar hashing codes within a short Hamming distance. The Hamming distance between two binary codes y_i and y_j can be calculated as $\frac{1}{4}\|y_i - y_j\|^2$. To measure the similarity between image examples represented by the binary hashing codes, one natural way is to minimize the weighted average Hamming distance as follows:

$$\sum_{i,j} \boldsymbol{S}_{ij}\|y_i - y_j\|^2 \tag{5}$$

Here, $\boldsymbol{S}$ is the similarity matrix, which can be calculated from image features $\boldsymbol{X}$. In this paper, we adopt the local similarity [31], due to its nice property in many machine learning applications. To meet the similarity preservation criterion, we seek to minimize this quantity since it incurs a heavy penalty if two similar images are mapped far away.

[1] In our experiments, we set the importance parameters a=1 and b=0.01.

By introducing a diagonal $n \times n$ matrix $\boldsymbol{D}$, whose entries are given by $\boldsymbol{D}_{ii} = \sum_{j=1}^{n} \boldsymbol{S}_{ij}$. Eqn.5 can be rewritten as:

$$tr\left(\boldsymbol{Y}(\boldsymbol{D}-\boldsymbol{S})\boldsymbol{Y}^T\right) = tr\left(\boldsymbol{Y}\boldsymbol{L}\boldsymbol{Y}^T\right) = tr\left(sgn(\boldsymbol{W}^T\boldsymbol{X})\boldsymbol{L}sgn(\boldsymbol{X}^T\boldsymbol{W})\right) \tag{6}$$

where $\boldsymbol{L}$ is called graph *Laplacian* [28] and $tr()$ is the matrix trace function. The similarity preservation term plays an important role in hashing function learning especially when the supervised information is limit due to noisy and incomplete tags. By minimizing this term, the similarity between different image examples can be preserved in the learned hashing codes.

3.4 Overall Objective

The entire objective function consists of two components: the tag consistency term in Eqn.4 and the visual similarity preservation term given in Eqn.6 as follows:

$$\begin{gathered}\min_{\boldsymbol{W},\boldsymbol{C}} \|\boldsymbol{I}^{\frac{1}{2}} \cdot (\boldsymbol{T} - sgn(\boldsymbol{X}^T\boldsymbol{W})\boldsymbol{C})\|_F^2 + \gamma\ tr\left(sgn(\boldsymbol{W}^T\boldsymbol{X})\boldsymbol{L}sgn(\boldsymbol{X}^T\boldsymbol{W})\right) + \alpha\|\boldsymbol{C}\|_F^2 \\ s.t. \quad \boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I_k}\end{gathered} \tag{7}$$

where α and γ are trade-off parameters to balance the weights among the terms. The hard orthogonality constraints enforce the hashing bits to be uncorrelated with each other and therefore the learned hashing codes can hold least redundant information.

4 Optimization Algorithm

4.1 Relaxation

Directly minimizing the objective function in Eqn.7 is intractable since it is a constrained integer programming, which is proven to be NP-hard to solve. Therefore, we first convert the hard constraints into a soft penalty term by adding a regularizer to the objective and use the signed magnitude instead of the sign function as suggested in [10,23]. Then the relaxed objective function becomes:

$$\min_{\tilde{\boldsymbol{W}},\boldsymbol{C}} \|\boldsymbol{I}^{\frac{1}{2}} \cdot (\boldsymbol{T} - \boldsymbol{X}^T\tilde{\boldsymbol{W}}\boldsymbol{C})\|_F^2 + \gamma\ tr\left(\tilde{\boldsymbol{W}}^T\tilde{\boldsymbol{L}}\tilde{\boldsymbol{W}}\right) + \alpha\|\boldsymbol{C}\|_F^2 + \beta\|\tilde{\boldsymbol{W}}^T\tilde{\boldsymbol{W}} - \boldsymbol{I_k}\|_F^2 \tag{8}$$

where $\tilde{\boldsymbol{L}} \equiv \boldsymbol{X}\boldsymbol{L}\boldsymbol{X}^T$ and can be pre-computed. However, even after the relaxation, the objective function is still difficult to optimize since $\tilde{\boldsymbol{W}}$ and $\boldsymbol{C}$ are coupled together and it is non-convex with respect to $\tilde{\boldsymbol{W}}$ and $\boldsymbol{C}$ jointly. We propose to split the optimization problem into two simpler sub-problems. The idea is that given $\tilde{\boldsymbol{W}}$, $\boldsymbol{C}$ has a closed form solution with respect to $\tilde{\boldsymbol{W}}$ (see details in $SP2$ below). Thus we split the relaxed objective with respect to $\tilde{\boldsymbol{W}}$ and $\boldsymbol{C}$ and solve the two sub-problems iteratively using coordinate descent method. The two sub-problems are given as:

$$SP1 : \min_{\tilde{\boldsymbol{W}}} \|\boldsymbol{I}^{\frac{1}{2}} \cdot (\boldsymbol{T} - \boldsymbol{X}^T\tilde{\boldsymbol{W}}\boldsymbol{C})\|_F^2 + \gamma\ tr\left(\tilde{\boldsymbol{W}}^T\tilde{\boldsymbol{L}}\tilde{\boldsymbol{W}}\right) + \beta\|\tilde{\boldsymbol{W}}^T\tilde{\boldsymbol{W}} - \boldsymbol{I_k}\|_F^2 \quad (9)$$

$$SP2 : \min_{\boldsymbol{C}} \|\boldsymbol{I}^{\frac{1}{2}} \cdot (\boldsymbol{T} - \boldsymbol{X}^T\tilde{\boldsymbol{W}}\boldsymbol{C})\|_F^2 + \alpha\|\boldsymbol{C}\|_F^2 \quad (10)$$

$SP1$ is still non-convex, but it is smooth and differentiable which enables gradient descent methods for efficient optimization. The gradient of $SP1$ is calculated as follows:

$$\partial\frac{SP1}{\tilde{\boldsymbol{W}}} = 2\boldsymbol{X}(\boldsymbol{I} \cdot (\boldsymbol{X}^T\tilde{\boldsymbol{W}}\boldsymbol{C} - \boldsymbol{T}))\boldsymbol{C}^T + 2\gamma\tilde{\boldsymbol{L}}\tilde{\boldsymbol{W}} + 4\beta\tilde{\boldsymbol{W}}(\tilde{\boldsymbol{W}}^T\tilde{\boldsymbol{W}} - \boldsymbol{I_k}) \quad (11)$$

With this obtained gradient, L-BFGS quasi-Newton method [9] is applied to solve $SP1$.

By taking the derivative of $SP2$ w.r.t. $\boldsymbol{C}$ and setting it to $\mathbf{0}$, we can obtain the closed form solution of $SP2$ below:

$$\begin{aligned}\partial\frac{SP2}{\boldsymbol{C}} &= 2\tilde{\boldsymbol{W}}^T\boldsymbol{X}(\boldsymbol{I} \cdot (\boldsymbol{X}^T\tilde{\boldsymbol{W}}\boldsymbol{C} - \boldsymbol{T})) + 2\alpha\boldsymbol{C} = \mathbf{0} \\ \Rightarrow c_j &= (\tilde{\boldsymbol{W}}^T\boldsymbol{X}\boldsymbol{I}_j\boldsymbol{X}^T\tilde{\boldsymbol{W}} + \alpha\boldsymbol{I_k})^{-1}\tilde{\boldsymbol{W}}^T\boldsymbol{X}\boldsymbol{I}_j\boldsymbol{T}_j\end{aligned} \quad (12)$$

where $\boldsymbol{I}_j$ is a $n \times n$ diagonal matrix with $\boldsymbol{I}_{ij}, i = 1, 2, \ldots, n$ as its diagonal elements and $\boldsymbol{T}_j = (\boldsymbol{T}_{ij}), i = 1, 2, \ldots, n$ is a $n \times 1$ label vector of *j-th* tag.

We alternate the process of updating $\tilde{\boldsymbol{W}}$ and $\boldsymbol{C}$ for several iterations to find a locally optimal solution. In practice, we have found that a reasonable small number of iterations (i.e., 30 in our experiments) can achieve good performance.

4.2 Orthogonal Transformation

After obtaining the optimal hashing function $\tilde{\boldsymbol{W}}$ for the relaxation, the hashing codes $\boldsymbol{Y}$ can be generated using Eqn.1. It is obvious that the quantization error can be measured as $\|\boldsymbol{Y} - \tilde{\boldsymbol{W}}^{\boldsymbol{T}}\boldsymbol{X}\|_F^2$. Inspired by [5], we propose to further improve the hashing function by minimizing this quantization error using an orthogonal transformation. We first prove the following orthogonal invariant theorem.

Theorem 1. *Assume* $\boldsymbol{Q}$ *is a* $k \times k$ *orthogonal matrix, i.e.,* $\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I_k}$*. If* $\tilde{\boldsymbol{W}}$ *and* $\boldsymbol{C}$ *are an optimal solution to the relaxed problem in Eqn.8, then* $\tilde{\boldsymbol{W}}\boldsymbol{Q}$ *and* $\boldsymbol{Q}^T\boldsymbol{C}$ *are also an optimal solution.*

Proof. By substituting $\tilde{\boldsymbol{W}}\boldsymbol{Q}$ and $\boldsymbol{Q}^T\boldsymbol{C}$ into Eqn.8, we have:
$\|\boldsymbol{I}^{\frac{1}{2}} \cdot (\boldsymbol{T} - \boldsymbol{X}^T\tilde{\boldsymbol{W}}\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{C})\|_F^2 = \|\boldsymbol{I}^{\frac{1}{2}} \cdot (\boldsymbol{T} - \boldsymbol{X}^T\tilde{\boldsymbol{W}}\boldsymbol{C})\|_F^2$,
$tr\left((\tilde{\boldsymbol{W}}\boldsymbol{Q})^T\tilde{\boldsymbol{L}}\tilde{\boldsymbol{W}}\boldsymbol{Q}\right) = tr\left(\boldsymbol{Q}^T\tilde{\boldsymbol{W}}^T\tilde{\boldsymbol{L}}\tilde{\boldsymbol{W}}\boldsymbol{Q}\right) = tr\left(\tilde{\boldsymbol{W}}^T\tilde{\boldsymbol{L}}\tilde{\boldsymbol{W}}\right)$, $\|\boldsymbol{Q}^T\boldsymbol{C}\|_F^2 = \|\boldsymbol{C}\|_F^2$
and $\|(\tilde{\boldsymbol{W}}\boldsymbol{Q})^T\tilde{\boldsymbol{W}}\boldsymbol{Q} - \boldsymbol{I_k}\|_F^2 = \|\boldsymbol{Q}^T(\tilde{\boldsymbol{W}}^T\tilde{\boldsymbol{W}} - \boldsymbol{I_k})\boldsymbol{Q}\|_F^2 = \|\tilde{\boldsymbol{W}}^T\tilde{\boldsymbol{W}} - \boldsymbol{I_k}\|_F^2$.
Thus, the value of the objective function in Eqn.8 does not change by the orthogonal transformation.

According to the above theorem, we propose to find a better hashing function $\boldsymbol{W} = \tilde{\boldsymbol{W}}\boldsymbol{Q}$ by minimizing the quantization error between the binary hashing codes and the orthogonal transformation of the relaxed solution as follows:

$$\begin{aligned} &\min_{\boldsymbol{Y},\boldsymbol{Q}} \|\boldsymbol{Y} - (\tilde{\boldsymbol{W}}\boldsymbol{Q})^T\boldsymbol{X}\|_F^2 \\ s.t. \qquad &\boldsymbol{Y} \in \{-1, 1\}^{k\times n}, \quad \boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I_k} \end{aligned} \tag{13}$$

Intuitively, we seek binary codes that are close to some orthogonal transformation of the relaxed solution. The orthogonal transformation not only preserves the optimality of the relaxed solution but also provides us more flexibility to achieve better hashing codes with low quantization error. The idea of orthogonal transformation is also utilized in ITQ [5]. However, ITQ method is not designed for incorporating partial tag information into learning effective hashing function and it does not preserve the local similarities among data examples. The above optimization problem can be solved by minimizing Eqn.13 with respect to $\boldsymbol{Y}$ and $\boldsymbol{Q}$ alternatively as follows:

Fix Q **and update** Y. The closed form solution can be expressed as:

$$\boldsymbol{Y} = sgn\left((\tilde{\boldsymbol{W}}\boldsymbol{Q})^T\boldsymbol{X}\right) = sgn(\boldsymbol{W}^T\boldsymbol{X}) \tag{14}$$

which is identical with our linear hashing function in Eqn.1.

Fix Y **and update** Q. The objective function becomes:

$$\min_{\boldsymbol{Q}^T\boldsymbol{Q}=\boldsymbol{I_k}} \|\boldsymbol{Y} - \boldsymbol{Q}^T\tilde{\boldsymbol{W}}^T\boldsymbol{X}\|_F^2 \tag{15}$$

In this case, the objective function is essentially the classic Orthogonal Procrustes problem [18], which can be solved efficiently by singular value decomposition using the following theorem (we refer to [18] for the detailed proof).

Theorem 2. *Let* $\boldsymbol{S\Lambda V}^T$ *be the singular value decomposition of* $\boldsymbol{YX}^T\tilde{\boldsymbol{W}}$. *Then* $\boldsymbol{Q} = \boldsymbol{VS}^T$ *minimizes the objective function in Eqn.15.*

We then perform the above two steps alternatively to obtain the optimal hashing codes and the orthogonal transform matrix. In our experiments, we find that the algorithm usually converges in about 40∼60 iterations. The full learning algorithm is described in Algorithm 1.

4.3 Complexity Analysis

This section provides some analysis on the training cost of the optimization algorithm. The optimization algorithm of SSTH consists of two main loops. In the first loop, we iteratively solve $SP1$ and $SP2$ to obtain the optimal solution, where the time complexities for solving $SP1$ and $SP2$ are bounded by $O(nlk + nkd + nk^2)$ and $O(nk^2 + nkl)$ respectively. The second loop iteratively optimizes the binary hashing codes and the orthogonal transformation matrix, where the

Algorithm 1. Semi-Supervised Tag Hashing (SSTH)

```
Input: Images $\boldsymbol{X}$, Observed Tags $\boldsymbol{T}$ and trade-off parameters
Output: Hashing function $\boldsymbol{W}$, Hashing codes $\boldsymbol{Y}$ and Correlation $\boldsymbol{C}$
  Initialize $\boldsymbol{C} = \mathbf{0}$ and $\boldsymbol{Q} = \boldsymbol{I_k}$, Calculate $\tilde{\boldsymbol{L}}$.
  repeat
      Optimize $SP1$ using Eqn.11 and update $\tilde{\boldsymbol{W}}$
      Optimize $SP2$ using Eqn.12 and update $\boldsymbol{C}$
  until the solution converges
  repeat
      Update $\boldsymbol{Y}$ using Eqn.14
      Update $\boldsymbol{Q} = \boldsymbol{V}\boldsymbol{S}^T$ according to Theorem 2.
  until the solution converges
  Compute hashing function $\boldsymbol{W}$=$\tilde{\boldsymbol{W}}\boldsymbol{Q}$.
```

time complexities for updating $\boldsymbol{Y}$ and $\boldsymbol{Q}$ are bounded by $O(nk^2 + nkd + k^3)$. Moreover, both two loops take less than 60 iterations to converge as mentioned before. Thus, the total time complexity of the learning algorithm is bounded by $O(nlk + nkd + nk^2 + k^3)$, which scales linearly with n given $n \gg l > d > k$. For each query, the hashing time is constant $O(dk)$.

5 Experimental Results

5.1 Datasets

We evaluate our method for large scale image retrieval on two image benchmarks: *NUS-WIDE* and *FLICKR-1M*. *NUS-WIDE* [2] is created by NUS lab for evaluating image annotation and retrieval techniques. It contains $270k$ images associated with $5k$ unique tags. 500-dimensional visual features are extracted using a bag-of-visual-word model with local SIFT descriptor [13]. We randomly partition this dataset into two parts, $1k$ for testing and around $269k$ for training. *FLICKR-1M* [6] is collected from Flicker images for image retrieval tasks. This benchmark contains 1 million image examples associated with more than $7k$ unique tags. 512-dimensional GIST descriptors [14] are extracted from these images and are used as image features for hashing function learning. We randomly choose $990k$ image examples as the training set and $10k$ for testing.

We implement our algorithm using Matlab on a PC with Intel Duo Core i5-2400 CPU 3.1GHz and 8GB RAM. The parameters α, β and γ in SSTH are tuned by 5-fold cross validation on the training set.

5.2 Evaluation Method

To conduct fair evaluation, we follow two criteria which are commonly used in the literature [5,10,23]: *Hamming Ranking* and *Hash Lookup*. *Hamming Ranking* ranks all the points in the database according to their Hamming distance from the query and the top k points are returned as the desired neighbors.

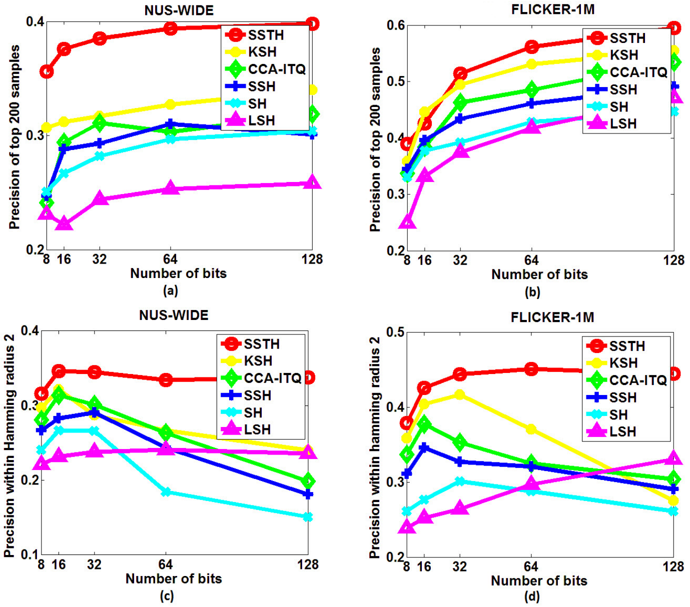

Fig. 1. Precision results on two datasets. (a)-(b): Precision of the top 200 returned examples using *Hamming Ranking*. (c)-(d): Precision within Hamming radius 2 using *Hash Lookup*.

Hash Lookup returns all the points within a small Hamming radius r of the query. The search results are evaluated based on whether the retrieved image and the query image share any ground-truth tags (i.e., if a returned image and the query image share any common semantic tags, then we treat this returned image as a true neighbor of the query image). We use several metrics to measure the performance of different methods. For *Hamming Ranking* based evaluation, we calculate the precision at top K which is the percentage of true neighbors among the top K returned examples, where we set K to be 200 in the experiments. We also compute the precision-recall value which is a widely used metric in information retrieval applications. A hamming radius of $R = 2$ is used to retrieve the neighbors in the case of *Hash Lookup*. The precision of the returned examples falling within Hamming radius 2 is reported.

5.3 Results and Discussion

The proposed SSTH approach is compared with five different algorithms, i.e., Spectral Hashing (SH) [28], Latent Semantic Hashing (LSH) [3], Canonical Correlation Analysis with Iterative Quantization (CCA-ITQ) [5,4], Semi-Supervised Hashing (SSH) [22] and Kernel Supervised Hashing (KSH) [10]. For LSH, we randomly select projections from a Gaussian distribution with zero-mean and identity covariance to construct the hash tables. For SSH and KSH, we sample $2k$

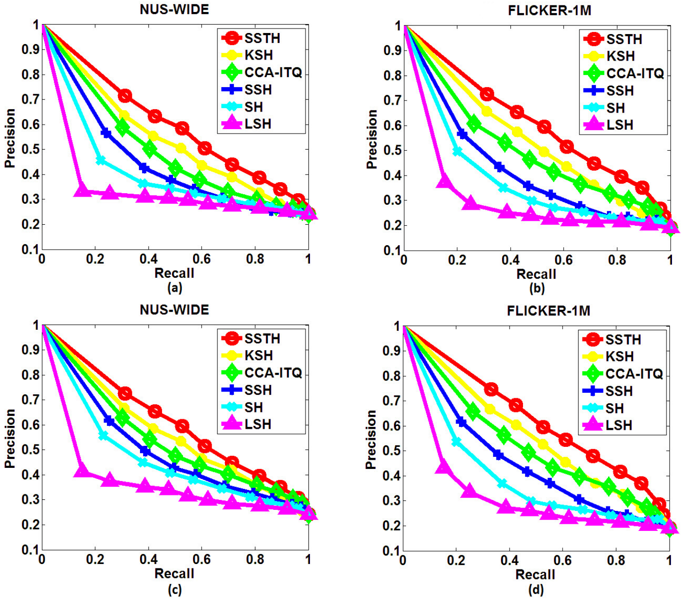

Fig. 2. Results of Precision-Recall behavior on two datasets. (a)-(b): Precision-Recall curve with 16 hashing bits. (c)-(d): Precision-Recall curve with 32 hashing bits.

random points from the training set to construct the pairwise constraint matrix. The reason we choose $2k$ points is that tags tend to be noisy and incomplete, and the constructed pairwise matrix based on these tags may be unreliable and inconsistent especially when tags are very sparse, resulting in even lower performance with more samples. Therefore, following the parameter settings in their papers, we also sample $2k$ points in our experiments.

In the first set of experiments, we report the precisions for the top 200 retrieved images and the precisions for retrieved images within Hamming ball with radius 2 by varying the number of hashing bits in the range of $\{8, 16, 32, 64, 128\}$ in Fig.1. The precision-recall curves with 16 and 32 hashing bits on both datasets are reported in Fig.2. From these comparison results, we can see that SSTH provides the best results among all six hashing methods on both benchmarks. LSH does not perform well in most cases since LSH method is data-independent, which may generate inefficient codes compared to those data-depend methods. The unsupervised SH method only tries to preserve image similarity in learned hashing codes, but does not utilize the supervised information contained in tags. SSH and KSH achieves better performance than SH and LSH due to the modeling of pairwise information. However, as pointed out in section 2, the coarse pairwise constraints generated from tags do not fully utilize tag information. The supervised method CCA-ITQ have similar performance to KSH since it also incorporates tags into learning better data representations. But in CCA-ITQ, it treats tags as another independent source where it may not be even reliable as tags can be incomplete, noisy and partially available. Moreover, the visual similarity is not well preserved in its hashing function learning. On the other hand, our SSTH not only exploits tag information via modeling the correlation between tags and hashing bits, but also preserves image similarity at the same time in the learned hashing function, which enables SSTH to generate higher quality hashing codes than the other supervised hashing methods. In Fig.1(c)-(d), we observe the precision of *Hash Lookup* for most of the compared methods decreases significantly with the increasing number of hashing bits. The reason is that the Hamming space becomes increasingly sparse with longer hashing bits and very few data points fall within the Hamming ball with radius 2, which makes many queries have 0 precision results. However, the precision of SSTH is still consistently higher than the other methods for *Hash Lookup*.

Table 1. Precision of the top 200 retrieved images under different training tag ratios on two datasets with 32 hashing bits

	NUS-WIDE					*FLICKR*-1*M*				
tag ratio	0.2	0.4	0.6	0.8	1.0	0.2	0.4	0.6	0.8	1.0
SSTH	**0.337**	**0.341**	**0.354**	**0.363**	**0.374**	**0.453**	**0.461**	**0.476**	0.480	**0.518**
SSTH^0	0.328	0.332	0.347	0.351	0.356	0.443	0.449	0.464	0.476	0.505
KSH [10]	0.288	0.296	0.301	0.308	0.316	0.422	0.448	0.459	**0.481**	0.484
CCA-ITQ [5,4]	0.287	0.290	0.305	0.330	0.348	0.410	0.427	0.445	0.467	0.494
SSH [22]	0.283	0.285	0.291	0.297	0.299	0.398	0.416	0.422	0.435	0.439

In the second set of experiments, we evaluate the effectiveness of the proposed SSTH if only partial tags are available. We progressively increase the number of training tags by varying the training tag ratio from {0.2, 0.4, 0.6, 0.8, 1}[2] and compare our SSTH with the other supervised hashing methods[3], CCA-ITQ, SSH and KSH on both datasets by fixing the hashing bits to 32. The precision results of top 200 retrieved images are reported in Table 1. We also evaluate our method without orthogonal transformation (by setting $\boldsymbol{Q} = \boldsymbol{I_k}$) and call this SSTH^0 in Table 1. It can be seen from the results that our SSTH gives the best performance among all supervised hashing methods in most cases. We also observe that the precision result of CCA-ITQ drops much faster than SSTH when the number of training tags decreases. Our hypothesis is that when training tags are very sparse, the common space learned from partial tags and visual features by CCA-ITQ is not accurate and reliable, resulting in low quality hashing codes. The comparison results of SSTH and SSTH^0 in Table 1 demonstrate that the orthogonal transformation can further improve the effectiveness of the hashing function, which is consistent with our expectation.

The third set of experiments demonstrate how the learned correlations, $\boldsymbol{C}$, can bridge tags and hashing codes. We conduct the experiments on *FLICKR*-1*M*

[2] Tags are randomly sampled from the training data based on the ratio.

[3] SH and LSH do not utilize tags and thus are not necessary to be compared here.

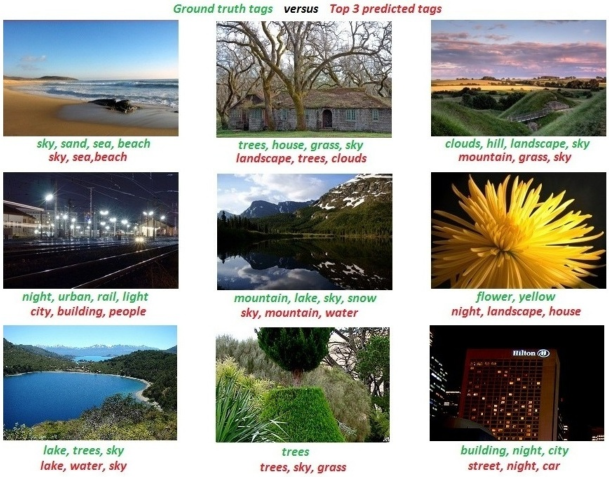

Fig. 3. Results of top 3 predicted tags on *FLICKR*-1*M*

to predict tags for query images based on their hashing codes. In particular, we first generate hashing code for each query image by $y_q = \boldsymbol{W}^T x_q$, and predict its tag vector using $t_q = \boldsymbol{C}^T y_q$. Then we select the top 3 tags with largest values in tag vector t_q as the predicted tags for the query image. The comparison results of the top 3 predicted tags with ground truth tags on several images are shown in Fig.3. From this figure we can see that our SSTH can generate reasonable accurate tags for query images. The reason is that our method not only incorporates tags in learning effective hashing function, but also extracts the correlation between tags and hashing bits. Therefore, the tag information is fully explored in our SSTH.

In the fourth set of experiments, the training time for learning hashing function and testing time for encoding each query image on both datasets (with 32 bits) are reported in Table 2. Note that we do not include the cross-validation time and any offline calculation cost in all methods for fair comparison. We can see from this table that the training time of SSTH is comparable with most of the other hashing methods and it is efficient enough in practice. The test time is

Table 2. Training and testing time (in second) on two datasets with 32 hashing bits

	NUS-WIDE		*FLICKR*-1*M*	
methods	training	testing	training	testing
SSTH	83.57	0.4x10^{-4}	219.03	0.6x10^{-4}
KSH [10]	248.85	2.4x10^{-4}	592.16	2.5x10^{-4}
CCA-ITQ [5,4]	46.13	0.5x10^{-4}	135.37	0.5x10^{-4}
SSH [22]	23.56	0.4x10^{-4}	40.83	0.5x10^{-4}
SH [28]	51.63	3.6x10^{-4}	173.68	4.1x10^{-4}
LSH [3]	2.75	0.4x10^{-4}	7.84	0.4x10^{-4}

sufficiently fast especially when compared to the nonlinear hashing method SH and kernel hashing method KSH.

6 Conclusions

This paper proposes a novel Semi-Supervised Tag Hashing (SSTH) framework that incorporates partial tag information by exploring the correlation between tags and hashing bits to fully exploit tag information. The framework simultaneously ensures the consistency between hashing codes and tags and preserves the similarities between images. Orthogonal transform is proposed for further improving the effectiveness of hashing bits. Experiments on two large scale datasets demonstrate the advantage of the proposed method over several state-of-the-art hashing methods. In future, we plan to investigate generalization error bound for the proposed learning method. We also plan to apply some sequential learning approach to accelerate the training speed of our method.

Acknowledgments. This work is partially supported by NSF research grants IIS-0746830, DRL-0822296, CNS-1012208, IIS-1017837, CNS-1314688 and a research grant from Office of Naval Research (ONR-11627465). This work is also partially supported by the Center for Science of Information (CSoI), an NSF Science and Technology Center, under grant agreement CCF-0939370.

References

1. Bergamo, A., Torresani, L., Fitzgibbon, A.W.: Picodes: Learning a compact code for novel-category recognition. In: NIPS, pp. 2088–2096 (2011)
2. Chua, T.S., Tang, J., Hong, R., Li, H., Luo, Z., Zheng, Y.: Nus-wide: A real-world web image database from national university of singapore. In: CIVR (2009)
3. Datar, M., Immorlica, N., Indyk, P., Mirrokni, V.S.: Locality-sensitive hashing scheme based on p-stable distributions. In: Symposium on Computational Geometry, pp. 253–262 (2004)
4. Gong, Y., Ke, Q., Isard, M., Lazebnik, S.: A multi-view embedding space for modeling internet images, tags, and their semantics. International Journal of Computer Vision 106(2), 210–233 (2014)
5. Gong, Y., Lazebnik, S., Gordo, A., Perronnin, F.: Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. IEEE TPAMI (2012)
6. Huiskes, M.J., Thomee, B., Lew, M.S.: New trends and ideas in visual concept detection: the mir flickr retrieval evaluation initiative. In: Multimedia Information Retrieval, pp. 527–536 (2010)
7. Kulis, B., Grauman, K.: Kernelized locality-sensitive hashing for scalable image search. In: ICCV, pp. 2130–2137 (2009)
8. Lin, R.S., Ross, D.A., Yagnik, J.: Spec hashing: Similarity preserving algorithm for entropy-based coding. In: CVPR, pp. 848–854 (2010)
9. Liu, D.C., Nocedal, J.: On the limited memory bfgs method for large scale optimization. Mathematical Programming 45, 503–528 (1989)

10. Liu, W., Wang, J., Ji, R., Jiang, Y.G., Chang, S.F.: Supervised hashing with kernels. In: CVPR, pp. 2074–2081 (2012)
11. Liu, W., Wang, J., Kumar, S., Chang, S.F.: Hashing with graphs. In: ICML, pp. 1–8 (2011)
12. Liu, X., He, J., Lang, B., Chang, S.F.: Hash bit selection: A unified solution for selection problems in hashing. In: CVPR, pp. 1570–1577 (2013)
13. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV 60(2), 91–110 (2004)
14. Oliva, A., Torralba, A.: Modeling the shape of the scene: A holistic representation of the spatial envelope. IJCV 42(3), 145–175 (2001)
15. Raginsky, M., Lazebnik, S.: Locality-sensitive binary codes from shift-invariant kernels. In: NIPS, pp. 1509–1517 (2009)
16. Rastegari, M., Farhadi, A., Forsyth, D.: Attribute discovery via predictable discriminative binary codes. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part VI. LNCS, vol. 7577, pp. 876–889. Springer, Heidelberg (2012)
17. Salakhutdinov, R., Hinton, G.E.: Semantic hashing. Int. J. Approx. Reasoning 50(7), 969–978 (2009)
18. Schonemann, P.: A generalized solution of the orthogonal procrustes problem. Psychometrika 31(1), 1–10 (1966)
19. Strecha, C., Bronstein, A.A., Bronstein, M.M., Fua, P.: Ldahash: Improved matching with smaller descriptors. IEEE TPAMI 34(1), 66–78 (2012)
20. Torralba, A., Fergus, R., Freeman, W.T.: 80 million tiny images: A large data set for nonparametric object and scene recognition. IEEE TPAMI 30(11), 1958–1970 (2008)
21. Wang, C., Blei, D.M.: Collaborative topic modeling for recommending scientific articles. In: KDD, pp. 448–456 (2011)
22. Wang, J., Kumar, S., Chang, S.F.: Semi-supervised hashing for large-scale search. IEEE TPAMI 34(12), 2393–2406 (2012)
23. Wang, J., Liu, W., Sun, A., Jiang, Y.G.: Learning hash codes with listwise supervision. In: ICCV (2013)
24. Wang, Q., Shen, B., Wang, S., Li, L., Si, L.: Binary codes emmbedding for fast image tagging with incomplete labels. In: ECCV (2014)
25. Wang, Q., Si, L., Zhang, Z., Zhang, N.: Active hashing with joint data example and tag selection. In: SIGIR (2014)
26. Wang, Q., Zhang, D., Si, L.: Semantic hashing using tags and topic modeling. In: SIGIR, pp. 213–222 (2013)
27. Wang, Q., Zhang, D., Si, L.: Weighted hashing for fast large scale similarity search. In: CIKM, pp. 1185–1188 (2013)
28. Weiss, Y., Torralba, A., Fergus, R.: Spectral hashing. In: NIPS, pp. 1753–1760 (2008)
29. Xu, H., Wang, J., Li, Z., Zeng, G., Li, S., Yu, N.: Complementary hashing for approximate nearest neighbor search. In: ICCV, pp. 1631–1638 (2011)
30. Ye, G., Liu, D., Wang, J., Chang, S.F.: Large scale video hashing via structure learning. In: ICCV (2013)
31. Zhang, D., Wang, J., Cai, D., Lu, J.: Self-taught hashing for fast similarity search. In: SIGIR, pp. 18–25 (2010)
32. Zhang, L., Zhang, Y., Tang, J., Lu, K., Tian, Q.: Binary code ranking with weighted hamming distance. In: CVPR, pp. 1586–1593 (2013)