# Compact Hashing for Mixed Image-Keyword Query over Multi-Label Images

Xianglong Liu[†], Yadong Mu[‡], Bo Lang[†], Shih-Fu Chang[‡]
[†]State Key Lab of Software Development Environment, Beihang University, Beijing, China
[‡]Columbia University, New York, NY 10027, U.S.A.
{xlliu, langbo}@nlsde.buaa.edu.cn ym2372@columbia.edu sfchang@ee.columbia.edu

## ABSTRACT

Recently locality-sensitive hashing (LSH) algorithms have attracted much attention owing to its empirical success and theoretic guarantee in large-scale visual search. In this paper we address the new topic of *hashing with multi-label data*, in which images in the database are assumed to be associated with missing or noisy multiple labels and each query consists of a query image and several textual search terms, similar to the new "Search with Image" function introduced by the Google Image Search. The returned images are judged based on the combination of visual similarity and semantic information conveyed by search terms. In most of the state-of-the-art approaches, the learned hashing functions are universal for all labels. To further enhance the hashing efficiency for such multi-label data, we propose a novel scheme "*boosted shared hashing*". Our basic observation is that image labels typically form cliques in the feature space. Hashing efficacy can be greatly improved by making each hashing function more targeted at and only shared across such cliques instead of all labels in conventional hashing methods. In other words, each hashing function is deliberately designed such that it is especially effective for a subset of labels. The targeted, but sparse association between labels and hash bits reduces the computation and storage when indexing a new datum, since only a small number of relevant hashing functions become active given the labels. We develop a Boosting-style algorithm for simultaneously optimizing the label subset and hashing function in a unified framework. Experimental results on standard image benchmarks like CIFAR-10 and NUS-WIDE show that the proposed hashing scheme achieves substantially superior performances over conventional methods in terms of accuracy under the same hash bit budget.

## Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing; I.2.10 [**Artificial Intelligence**]: Vision and Scene Understanding


*ICMR* '12, June 5-8, Hong Kong, China


## General Terms

Algorithms, Experimentation

## Keywords

Multi-label, Boosting, Locality-Sensitive Hashing

## 1. INTRODUCTION

With the rapid growth of multimedia data on the Internet and personal albums, storing, indexing and retrieving these data (usually represented by high dimensional features, such as SIFT or SURF) have become one of the major challenges in the multimedia community. Recently, a large amount of research attentions have been arisen by the problem of *approximate nearest neighbor* (ANN) search. On the one hand, many multimedia tasks (*e.g.*, content-based image retrieval) can be formulated as a nearest neighbor search problem. On other hand, an empirical observation is that exact nearest neighbor is computationally expensive yet unnecessary. In fact, approximate nearest neighbor gives a good balance between the performance of multimedia system and computational complexity. Among those ANN methods developed in the past decade, *locality-sensitive hashing* (LSH) [9] is one of the most popular methods owing to its empirical success and elegant theoretic guarantee.

The complications of applying LSH algorithms onto multimedia data stem from the diverse multimedia semantics. Conventional LSH algorithms are mostly targeting the unsupervised input, and mild conditions are postulated (*e.g.*, uniform data distribution [9]). However, the multimedia data show several important characteristics, including 1) **multiple categories**: the number of semantic categories varies with respect to different image benchmarks and multimedia tasks. For example, the MNIST digit dataset contains 60,000 images corresponding to digits '0'-'9'. ImageNet dataset [6] contains more than 10,000 object categories, which significantly complicates the visual retrieval operation; and 2) **multiple labels**: namely an image or video is associated with several semantic labels, as seen in the large-scale image benchmark NUS-WIDE [4]. Such data are more general and receiving increasing attention now in various areas [2, 19, 10]. Note that these two notations "multiple categories" and "multiple labels" are distinguished by subtle differences, especially the possible number of labels associated with each image (single label for the former and arbitrarily many for the latter). For brevity, we abuse the notation "multiple labels" to denote both (since multiple categories can be regarded as a special case in some sense)

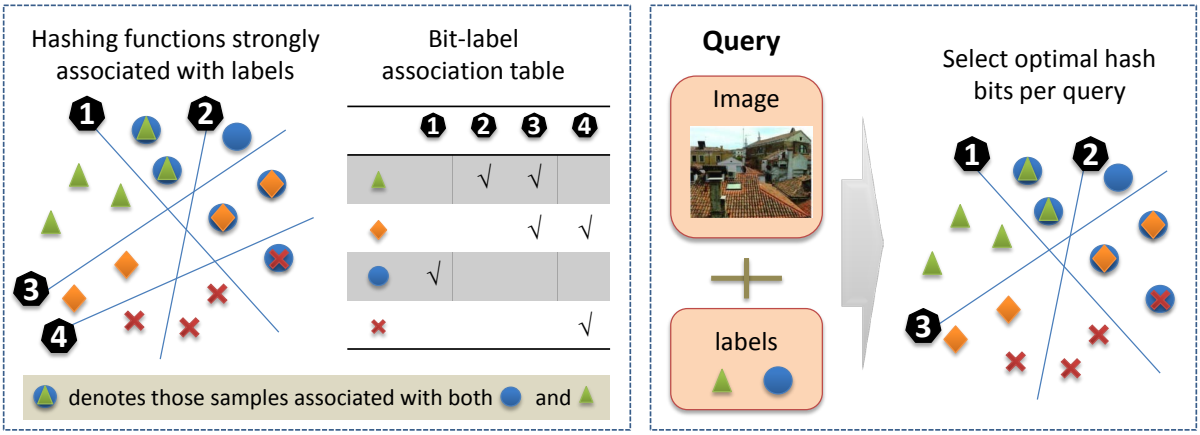



**Figure 1: We propose a novel learning framework to design compact hashing functions with strong association with labels. The left panel shows several hash projection functions, each of which targets at preserving consistence of only a different subset of labels (*e.g.*, hashing function 1 preserves the Blue label completely while function 3 preserves Green and Orange labels). The final association relations between labels and hashing functions are summarized in the table. At the query time, the specific query image and query labels are used to select a subset of hashing functions that are most effective in retrieving similar images for the given query.**

until otherwise mentioned.

Hashing with multi-label data is well motivated by recent advance in content-based image retrieval. A typical example is the "search by image" function[1] released by Google in 2011, whose input is the hybrid of a query image and several describing textual words. These words could be either provided by the users or inferred by the retrieval system. The final rank of an image in the database is determined by both the visual proximity to the query image and matching the image surrounding text with the input query words. In practice the additional textual hint greatly helps reduce the ambiguity of user intention and therefore refines the retrieval accuracy. This new paradigm of image retrieval naturally proposes the challenge of effective indexing multi-label multimedia data. Prior research on multi-label hashing can be roughly cast into two directions as below:

- **Learn hashing functions universal for all labels**. Several works have been devoted to indexing multi-category image data, which are also immediately extended for multi-label data. Representative works include kernel-based hashing with weak supervision [15] and semi-supervised hashing [17]. The key idea in these works is pushing same-label samples as close as possible in the hash-bit Hamming space, and simultaneously maximizing the separation between different semantic labels. The work of Mu et al. [15] also supports sparsely-specified pairwise similarity (or dissimilarity) constraints, which are common in many real-world applications. However, since most of previous approaches adopt linear hashing functions, for complex data it is difficult to have each hashing function significantly benefit all labels. In multi-label setting, *query-adaptive* hashing schemes are more desired, namely the hashing functions are dependent on the labels of the query (they are assumed to be known as in Google "search by image").

[1]http://www.google.com/searchbyimage

**Table 1: Description of data setting in the adopted image retrieval paradigm. See text for the definitions of three data categories.**

| | Labels | Visual Features |
|---|---|---|
| Training Data | √ | √ |
| Database Data | × | √ |
| Query | √ | √ |

- **Hash bit selection and reweighing**. It represents the most relevant works to our work. The key idea is building a large pool of random hashing functions and selecting a subset of optimal hash bits (possibly reweighing them) for each label. The selection procedure is independent for different labels, therefore this method is scalable for large set of labels and extensible to those labels unseen during training. Representative works include [14] and [11]. The over-complete hash bit pool is a double-edged sword. On the one hand, it is effortlessly adaptive to semantic changes by adjusting selected hash bits and corresponding weights. However, on other hand, it is inefficient when processing the multi-label data. As afore-mentioned, this method relies on a large pool of over-complete hash bits and selects a subset for each label. Typically the subsets of different labels seldom intersect with each other and each hashing function is only targeted at a single label. Suppose a sample is associated with $k$ labels, and each label is indexed using $L$ bits. In the worst case, $kL$ hashing bits will be required to store the sample, which is far from being optimal considering the heavy redundancy within semantic labels.

In this paper we propose a new hashing scheme to enhance the performance of multi-label data indexing. The key idea is akin to the works in [14, 11], as illustrated in Figure 1. Following the image retrieval paradigm as in Google "search by image", it is assumed that both the visual feature and a

few user-specified labels are bundled together as the query, as seen in the right panel of Figure 1. As Table 1 shows, the label information of the database set is assumed to be mostly unknown, partially missing and too noisy to be reliable as in the real-world image corpus on Flickr[2]. To learn compact hash codes, a relatively-small training set is collected and manually annotated. Then both the hashing functions and their sparse association with labels are learned sequentially. During the retrieval stage, a query-adaptive subset of the hash bits are selected to be active (algorithmic details are postponed to Section 2) for image retrieval.

We later show that the proposed scheme is able to enhance both the computational efficiency (fewer active hash bits) and the retrieval accuracy (query-adaptive). In brief, our contributions lie in three-folds, summarized as following:

1. Rare works have been devoted to the problem of hashing with multi-label image data. Our proposed new scheme, to the best of our knowledge, is the first compact hashing technique for mixed image-keyword search over multi-labeled images. It exploits the shared subspaces between the labels to help index multi-label data. Specifically, each hash function projects the data onto a 1-D linear subspace, where a subset of labels (those marked with ticks in the left panel of Figure 1) are well separated from other labels. In this way it sparsifies the association between the hashing functions and labels. The sparsification is crucial for improving the multi-label hashing performance since it enables query-adaptive hashing via label-oriented hash bit selection.

2. We develop an efficient Boosting-style algorithm to sequentially learn the hashing functions for multi-label image data. In the proposed scheme, each hashing function is expected to be active for a subset of the labels (referred to as *active labels* hereafter). Therefore, compared with the over-complete hash pool used in [14, 11], our constructed hash codes demonstrate higher compactness at the cost of tolerable training effort. The proposed algorithm is able to simultaneously optimize the label subset and hashing function in a unified framework.

3. Conventional content-based image retrieval takes only images as query. In our study we show both the feasibility of label-aided retrieval and the scalability of multi-label hashing. The evaluation conducted on the large-scale multi-label image benchmark NUS-WIDE contrasts not only different image retrieval paradigms, but also different multi-label indexing schemes.

The remaining sections are organized as follows. Section 2 provides a sketch of the basic idea, and then elaborates on the algorithm design, especially the Boosting-style multi-label boosted-shared hashing algorithm and a quadratic complexity method for optimized label subset selection. Section 3 reviews related works. In Section 4, comprehensive experimental results are presented to demonstrate the effectiveness. Finally Section 5 concludes this paper.

# 2. MULTI-LABEL SHARED HASHING

[2]http://www.flickr.com

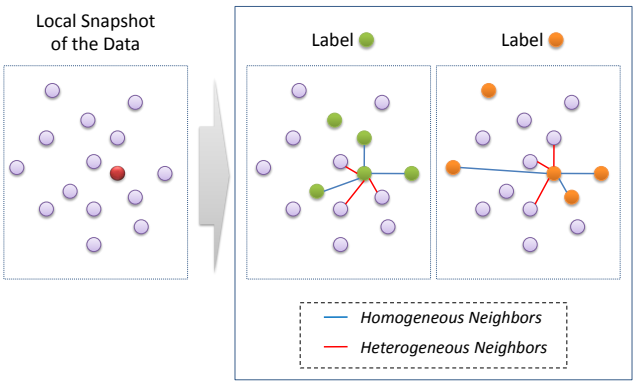



**Figure 2: Illustration of multi-label graph construction. The selected sample in the left sub-figure (highlighted in red) has two labels. For each label, two kinds of neighborhoods (homogeneous and heterogenous) are respectively constructed. Better viewed in color.**

In this section we elaborate on the proposed multi-label hashing method. As aforementioned, the key idea is to encourage sparse association between hashing functions and labels by exploiting shared subspaces among the labels. To generate compact hash codes, Section 2.1 presents a formulation in Boosting style. In each iteration, when the active labels are known, the pursuit of optimal hashing function is shown to boil down to an eig-decomposition problem, which is efficient to solve. Section 2.2 furthers describes a quadratic-complexity strategy to determine the active labels for each iteration.

## 2.1 Formulation

We first clarify the notations used in this section. Denote $\langle x_i, l_i \rangle \in \mathbb{R}^D \times \{0,1\}^L,\ i = 1 \dots N$ to be the $i$-th feature vectors together with the corresponding label vectors, where $D$, $N$, $L$ are the feature dimension, the numbers of data and labels respectively. Note that $l_i$ is binary vector. Let $l_i(k)$ to be the $k$-th element of $l_i$. $l_i(k) = 1$ reflects that $x_i$ is associated with the $k$-th label. For multi-label data, the number of nonzero elements is allowed to be more than one.

**Hash function definition**: Our goal is to optimally learn $B$ hash bits for the data. Let $h_b(\cdot)$ be the $b$-th hashing function. For tractability we adopt the linear form, *i.e.*,

$$h_b(x) = sign\left(w^T x\right), \tag{1}$$

where $sign(\cdot)$ is the signum function and $w$ is the normal vector of the hashing hyperplane.

**Multi-label graph construction**: The hashing vector $w$ in Equation (1) is typically randomly generated in original LSH algorithms [5]. Recent years have witnessed the generation of compact hash codes by optimizing $w$ according to specific criterion [17, 8]. We propose to use a multi-graph based learning framework to squeeze the most out of each hash bit. Figure 2 illustrates the construction of our proposed multi-label graphs. Specifically, for the $k$-th label, assume the $i$-th sample has $l_i(k) = 1$, and let $\mathcal{N}_k^o(i)$ denote its *homogeneous neighborhood*, which comprises the indices of its $k^o$-nearest neighbors of the same label, and $\mathcal{N}_k^e(i)$ be the *heterogenous neighborhood* containing its $k^e$-nearest neighbors of different labels. The subscript $k$ in $\mathcal{N}_k^o(i)$, $\mathcal{N}_k^e(i)$

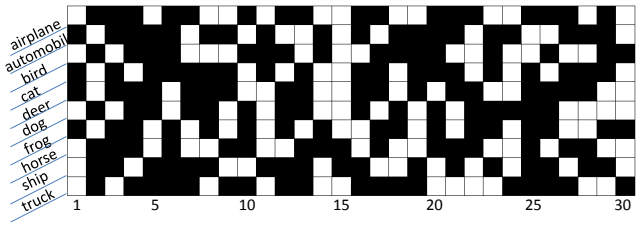



**Figure 3: Illustration of "active label set" (white cells) on CIFAR-10. The horizontal axis is the index of 30 hash bits learned by the proposed algorithm, and the vertical axis corresponds to ten semantic labels.**

reflects that they are pertained to the graph of the $k$-th label. Figure 2 shows the case for a sample with two labels.

**Active label set**: One defining feature of the proposed method is the sparse association between hash bits and the labels. Hereafter we use the term "active label set" and variable $\mathcal{S}_b$ to indicate those labels that are associated with the $b$-th hash bit. Figure 3 gives such an example on real-world image benchmark by applying the proposed hashing scheme. We postpone to present algorithmic details of choosing active labels to Section 2.2. In the following presentation, $\mathcal{S}_b$ is assumed to be known for clarity.

**Exponential-loss oriented cost function**: Based on the multi-label graphs and active label set $\mathcal{S}_b$, we further define the cost function to be optimized. Intuitively, a sample is encouraged to share the same hash values with its homogeneous neighbors (in other words, small Hamming distance between their hash codes), and have significant different hash codes compared with its heterogenous neighbors. This likelihood can be measured via the function below:

$$F(x_i, x_j, k) = \sum_{b=1}^{B} \delta\left[k \in \mathcal{S}(b)\right] \cdot h_b(x_i) \cdot h_b(x_j), \tag{2}$$

where $k$ denotes one of the labels that $x_i$ is associated with, and $\delta(x)$ is the delta function that returns 1 if $x$ is true, otherwise 0. Obviously the delta function serves as the gating function for hash bit selection, which is consistent to the idea illustrated in Figure 3. Accumulating over all neighboring samples, the cost defined on $x_i$ (denoted as $c(x_i)$) is as following:

$$\sum_{\forall k, l_k(i)=1} \left( \frac{1}{k^o} \sum_{j \in \mathcal{N}_k^o(i)} e^{-F(x_i,x_j,k)} + \frac{1}{k^e} \sum_{j \in \mathcal{N}_k^e(i)} e^{F(x_i,x_j,k)} \right)$$

Exponential function is adopted in the Equation above owing to its robustness and the convenience for incremental update. Let $\mathcal{N}_k(i) = \mathcal{N}_k^o(i) \cup \mathcal{N}_k^e(i)$. For conciseness consideration, we ignore the effect of $k^o, k^e$ and introduce an auxiliary variable $z_{ij}$ that takes value -1 if $j \in \mathcal{N}_k^e(i)$, otherwise 1. The cost function for all samples can be represented as below:

$$\mathcal{J} = \sum_{i=1}^{N} c(x_i) = \sum_{i=1}^{N} \sum_{\forall k, l_k(i)=1} \sum_{j \in \mathcal{N}_k(i)} e^{-z_{ij} F(x_i,x_j,k)}, \tag{3}$$

$$= \sum_{(i,j,k) \in \mathcal{I}} e^{-z_{ij} F(x_i,x_j,k)}, \tag{4}$$

where the variable $\mathcal{I}$ is introduced to denote the index set of all valid triple sets $(i, j, k)$.

**Taylor expansion and updating rule**: Following the spirit of the well-known AdaBoost algorithm [7], the hashing function $h_b(\cdot)$, $b = 1 \ldots B$ is learned in an incremental manner. According to Equation 2, the function $F(x_i, x_j, k)$ is additive, and thus at the $b$-th iteration it can be written as:

$$F^{(b)}(x_i, x_j, k) = F^{(b-1)}(x_i, x_j, k) + f^{(b)}(x_i, x_j, k), \tag{5}$$

where $f^{(b)}(x_i, x_j, k) = \delta\left[k \in \mathcal{S}(b)\right] \cdot h_b(x_i) \cdot h_b(x_j)$. Then any binary term in Equation (4) can be therefore simplified as following:

$$\begin{aligned} e^{-z_{ij} F^{(b)}(x_i,x_j,k)} &= e^{-z_{ij} F^{(b-1)}(x_i,x_j,k)} \cdot e^{-z_{ij} f^{(b)}(x_i,x_j,k)}, \\ &= \pi^{(b-1)}(x_i, x_j, k) \cdot e^{-z_{ij} f^{(b)}(x_i,x_j,k)}, \end{aligned} \tag{6}$$

where the variable $\pi$ is introduced to maintain a probabilistic distribution over all neighbor pairs and can be estimated from previous $b-1$ iterations. From the Taylor expansion, we can further obtain (the approximation below is resulted from abandoning high-order terms in Taylor expansion):

$$e^{-z_{ij} f^{(b)}(x_i,x_j,k)} \approx 1 - z_{ij} f^{(b)}(x_i, x_j, k) + \frac{z_{ij}^2 \left(f^{(b)}(x_i, x_j, k)\right)^2}{2}.$$

For $k \in \mathcal{S}(b)$, both the variable $z_{ij}$ and function $f^{(b)}(x_i, x_j, k)$ are discrete, ranging over $\{-1, 1\}$. Therefore the equation above can be further written as:

$$e^{-z_{ij} f^{(b)}(x_i,x_j,k)} \approx -z_{ij} f^{(b)}(x_i, x_j, k) + const. \tag{7}$$

Finally, we relax the signum function to be continuous, *i.e.*, $f^{(b)}(x_i, x_j, k) = sign(w^T x_i) \cdot sign(w^T x_j) \approx w^T x_i x_j^T w$. The optimization problem for the $b$-th iteration is as below:

$$\begin{aligned} \min_w \ \mathcal{J}^c &= - \sum_{(i,j,k) \in \mathcal{I}, k \in \mathcal{S}(b)} \pi^{(b-1)}(x_i, x_j, k) \cdot z_{ij} \cdot w^T x_i x_j^T w, \\ &= w^T \left( - \sum_{(i,j,k) \in \mathcal{I}, k \in \mathcal{S}(b)} \pi^{(b-1)}(x_i, x_j, k) \cdot z_{ij} \cdot x_i x_j^T \right) w. \end{aligned} \tag{8}$$

It can be efficiently solved by eig-decomposition. The whole boosted shared hashing algorithm is shown by Algorithm 1.

## 2.2 Active label set selection

In Section 2.1, the active label set $\mathcal{S}(b)$ is assumed to be known for clear presentation. This section elaborates on a scalable algorithm for selecting $\mathcal{S}(b)$ in $O(L^2)$ complexity (recall that $L$ is the number of labels), rather than exhaustively comparing all possible $2^L$ subsets. The scalability to large label set mainly stems from a greedy selection strategy. At the beginning, the active label set $\mathcal{S}_b$ is initialized to be the label that best minimizes Problem (8). The optimal $w^*$ is recorded. At each following iteration, it expands $\mathcal{S}_b$ by adding another label. Specifically, for each un-selected label $k$, it calculates the value of $\mathcal{J}$ in Equation (4) with the relaxation in Equation (7), using active label set $\mathcal{S}(b) \cup \{k\}$ and current $w^*$. The one that most decreases the objective value is selected, and $w^*$ is afterwards re-calculated by optimizing Problem (8). The procedure is terminated when the gain is only incremental (say, below 5% compared with the previous iteration).

**Algorithm 1** Boosted Shared Hashing.

1: Initialize the weights $\pi^{(0)}(i,j,k) = 1$ for $(i,j,k) \in \mathcal{I}$ and normalize them to form a probability distribution. Set $F(x_i, x_j, k) = 0$.
2: // Learn the hash bits
3: **for** $b = 1, 2, \ldots, B$ **do**
4: // Learn the active label set
5: $\mathcal{S}(b) = \emptyset$;
6: **while** $|\mathcal{S}(b)| <= L$ **do**
7: **for** any un-selected label $k$ **do**
8: Calculate the decrease amount of $\mathcal{J}$ in Equation (4) with active label set $\mathcal{S}(b) \cup \{k\}$;
9: **end for**
10: Select the label with maximal decrease. Terminate if the decrease is below pre-specified threshold;
11: **end while**
12: Calculate the hashing vector $w_b$ by optimizing Problem (8);
13: Update the weights and normalize them:

$$\pi^{(b)}(i,j,k) = \pi^{(b-1)}(i,j,k) \cdot e^{-z_{ij} f^{(b)}(x_i, x_j, k)}.$$

14: Update $F(x_i, x_j, k)$ via Equation (5).
15: **end for**

## 2.3 The retrieval stage

Sections 2.1 and 2.2 have presented the algorithms for learning the hashing functions and active label sets, which encourage sparse association between the hash bits and labels. We denote the learned bit-label association matrix to be $A \in \{0,1\}^{L \times B}$. When a query $(x_q, l_q)$ comes, the retrieval task boils down to selecting specific number of hash bits from the learned $B$ hash bits.

Unfortunately, this task is non-trivial since

- A natural idea is finding the associated bits for each textual query word and taking their union. However, this strategy fails when $\|l_q\|_0$ is large. In the extreme case, all $B$ bits will be selected.
- The number of associated bits in matrix $A$ is varying for different labels, which complicates adaptively choosing proper number of hash bits for the retrieval.

We propose to use a simple matching-score based bit ranking method for bit selection. A matching score is computed between each bit-label association vector $a$ (any column of matrix $A$) and $l_q$, as following:

$$s_J(a, l_q) = \frac{|a \cap l_q|}{|a \cup l_q|}, \tag{9}$$

where $|\cdot|$ denotes the number of nonzero elements in a set. $\cap$, $\cup$ represent the intersection and union operations on two sets respectively. Recall that both $a$ and $l_q$ are binary vectors. Intuitively, $s_J$ calculates the shared labels between them, penalized by their union. It is known as the Jaccard index in the hashing literature. Given a query $(x_q, l_q)$, the algorithm greedily selects specific number of hash bits from $A$ by sorting the scores $s_J(a, l_q)$ in descending order.

# 3. RELATED WORKS

The seminal work of Indyk et al. [9] attacks the problem of approximate nearest neighbor retrieval for binary vectors equipped with Hamming distance. The so-called *locality-sensitive* property refers to the fact that LSH algorithms embed similar data under specific metric into similar codes in Hamming space. They use the data structure of multiple hash table to accomplish a constructive proof for the sub-linear complexity. The most important factor in LSH algorithm design is the underlying metric to gauge data similarity. In the past decade researchers from the theoretic computer science community have explored hashing on a diverse set of metrics, including $l_p$-norm ($p \in (0, 2]$) [5], Jaccard index [1] etc. The idea of LSH is also tailored for other scenarios, like kernelized hashing [12], learning-based compact hashing [18], graph-based unsupervised hashing [13], and semi-supervised hashing [17, 15].

Two lines of works are most relevant to our work in this paper. The first one is *query-adaptive hashing*, *i.e.*, the retrieval system has auxiliary information (*e.g.*, associated tags or labels) of the query image, which is expected to significantly clarify user's intention and motivates recent research on hash bit selection [14, 11]. The key idea in these works is to maintain a large pool of over-complete hash bits and adaptively select a small subset of bits according to the given auxiliary tags or labels. The other line of work is *shared subspace learning for multi-label data*. Early investigation on shared subspace utilized the multi-task learning framework [2]. The idea is also exploited in other applications like the boosted multi-class object detector, which is proposed by Torralba et al [16] to learn the shared feature across different classes. Moreover, since multi-label data are abundant in the field of multimedia, researchers also propose shared subspace learning methods to facilitate multi-label image classification or annotation. For example, Yan et al. [19] have proposed a model-shared subspace boosting methods as an attempt to reduce the information redundancy in the learning process and Ji et al. [10] extract the shared structure in multiple categories.

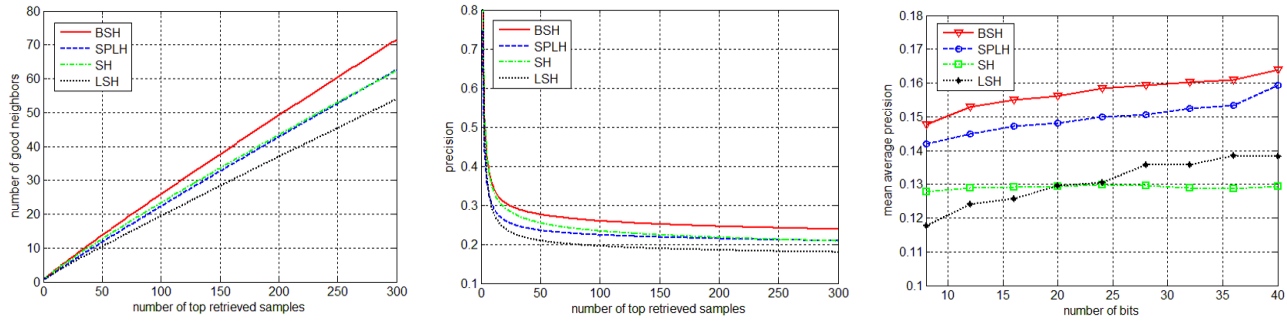
Our work is a marriage between these two ideas. It is motivated by the label-aided image retrieval paradigm and significantly reduces the inter-bit redundancy in the bit-selection based hashing methods [14, 11] through the idea of shared subspace learning. As far as we know, this is the first work that learns the shared hash functions for multi-class data to make more compact hashing bits.

# 4. EXPERIMENTS

In this section, we evaluate the performances of our proposed boosted-shared hashing (denoted as BSH hereafter) on both multi-category (*i.e.*, each image is associated with only one semantic category) and multi-label image benchmarks. Specifically, we choose two large-scale image benchmarks: CIFAR-10[3] (60K images) for multi-category retrieval, and the multi-label benchmark NUS-WIDE [4], which is crawled from Flickr and consists of about 270K images. As the pre-processing, feature vectors are all centered and unit-normalized. In all experiments, the parameters of nearest neighbor number $k^o$, $k^e$ are set to be 15, 30 respectively without fine tuning. Following previous work [20], the parameter $k^e$ is set twice larger than $k^o$ to compensate the extreme unbalance between the numbers of homogeneous neighbor pairs and heterogenous neighbor pairs.

It is well-known that when comparing different hashing

[3]http://www.cs.utoronto.ca/~kriz/cifar.html




(a) Good neighbors under 24 hash bits (b) Precision under 24 hash bits (c) Mean-Average-Precision (0-1)

**Figure 4: Performances of three hashing schemes: our proposed BSH, SPLH and SH on CIFAR-10. The sub-figures in (a) and (b) are the results of using 24 hash bits per query, and sub-figure (c) presents the performances with varying numbers of hash bits. See text for more details.**

methods, the performance heavily relies on the number of hash bits used for a query. Therefore, towards a fair comparison, by default all hashing schemes in the experiments are contrasted with respect to the same budget of hash bits per query. Moreover, since there is no hashing scheme specially designed for multi-label data, we adopt three state-of-the-art hashing methods, *sequential projection learning for hashing* (SPLH) [17], *spectral hashing* (SH) [18] and random projection based locality sensitive hashing (LSH) [3] as the baseline algorithms. All methods are properly tailored before they are applied to the multi-label data. To reduce the effect of randomness, we estimate the performance by averaging 10 independent runs.

## 4.1 Multi-Category Benchmark

**CIFAR-10** is a subset of the 80-million tiny image[4]. It is originally constructed for learning meaningful recognition-related image filters whose responses resemble the behavior of human visual cortex. The dataset is comprised of 60,000 $32 \times 32$ color images from 10 semantic categories (*e.g.*, airplane, frog, truck etc.). There are 6,000 images for each category, from which 300 images are kept for training the hashing functions. We also randomly select in total 1,000 samples as the queries. Regarding the visual features, 387-D GIST features are extracted from these images.

Generally, the performance of a hashing scheme is gauged based on various statistics about "good neighbors". On CIFAR-10, we define "good neighbor" as those samples from the same category to the query. Figure 4 shows our experimental results. In Figure 4(a) and (b), the performance with 24 hash bits for any query are reported, in terms of the number of "good neighbors" and precision in top 300 returned samples. Moreover, it is also important to investigate the tendency of the performance with respect to the number of hash bits per query. Figure 4(c) summarizes their performances using hashing bits ranging from 8 bits to 40 bits. It is observed that our proposed scheme is consistently superior to others. In all cases, the proposed BSH learns 100 hash bits from the procedure as described in Section 2.1, from which the bits used for retrieval are selected using the method in Section 2.3.

Significant performance gaps between the proposed BSH and other schemes like SPLH are observed in Figure 4. On

[4]http://groups.csail.mit.edu/vision/TinyImages

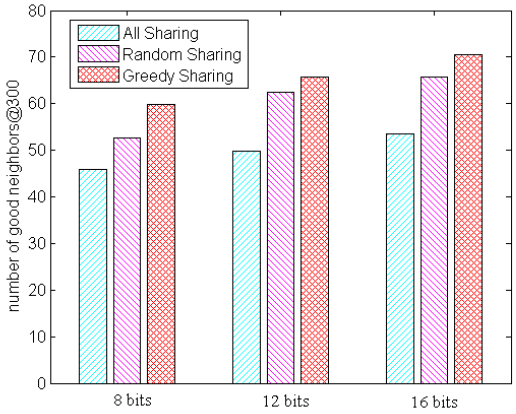
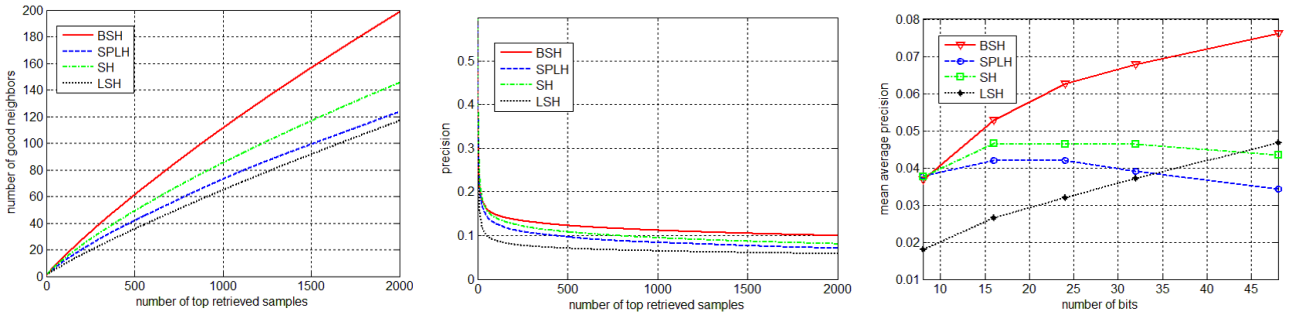



**Figure 5: Performance comparison of different sharing methods.**

average, each hash bit learned by BSH is shared by 3.47 active labels. An example of bit-label association matrix is found in Figure 3, where white blob indicates specific bit and label are associated. Unlike conventional methods which learn the hash bits versatile for all labels, the bits by BSH are intentionally forced to focus on the selected active labels, which largely enhances the discriminative ability of the hash bits. The experimental results corroborate the effectiveness of our proposed scheme.

To further illustrate it, we perform another experiment as illustrated in Figure 5. Recall that we propose a greedy strategy for active label selection (denoted by "greedy sharing" in Figure 5), greatly accelerating the exhaustive search which has an exponential complexity $O(2^L)$ ($L$ is the label number). We contrast this strategy with two baseline strategies, *i.e.*, all labels are selected for each bit (*i.e.*, "all sharing") and randomly select a number of labels for each bit (*i.e.*, "random sharing"). The "all sharing" strategy is intrinsically to the one used by other conventional hashing schemes [17, 18] yet differs in the way to learn the hashing functions. The "random sharing" strategy uniformly samples specific number of labels to be active (we set the number to be 3, *i.e.*, the rounded averaged number of active labels per bit in BSH). The comparison is based on the number of good neighbors in top-300 retrieved samples for




(a) Good neighbors: two-label query (b) Precision: two-label query (c) MAP: two-label query

**Figure 6: Performances of BSH and baseline algorithms on NUS-WIDE image benchmark with two-label queries.**

all strategies. To make it comprehensive, the performances with different hash bits per query (specifically, 8, 12, 16 bits) are reported. From Figure 5, it is observed that both "random sharing" and "greedy sharing" outperform "all sharing", which indicates that the idea of sparsifying bit-label association is helpful. Moreover, our proposed "greedy sharing" is shown to be better than "random sharing".

## 4.2 Multi-Label Benchmark

We choose the large-scale NUS-WIDE benchmark for the multi-label image retrieval, where each image is associated with several tags and the number of tags varies for different images. A group of volunteers are solicited to manually annotate 81 tags (hereafter we use the term "tag" and "label" interchangeably until otherwise mentioned) across the entire dataset. In the experiments, we represent each image by concatenating 500-D bag-of-words feature built from SIFT local descriptors and 225-D block-wise color moment feature. We further select 25 most-frequent tags, like 'sky', 'clouds', 'person' etc., each of which has at least several thousand associated images. We randomly select 5,000 image to build a training set and keep another query set of 1,000 images. Again the methods of SPLH, SH and LSH are chosen as baselines. In the experiments, it is noticed that the regularization parameter $\eta$ in SPLH is critical for its performance. We set it to be $8 \times 10^{-3}$ after fine tuning. The label matrix $S$ in SPLH is created by summarizing all the neighbor relationships of different classes.

As above-mentioned, we adopt the retrieval paradigm that takes "image + labels" as the query. It models the user intention more accurately and proposes serious challenges to conventional hashing schemes which fail to take multi-label information into account. Illustrating examples are found in Figure 7, where two images associated with three tags are shown. Assume the users are only interested in part of the semantic tags associated with the query image (which is the common case in real-world image retrieval), conventional hashing schemes tend to fail since they are not query-adaptive. In Figure 7, for each image we generate two different queries, *e.g.*, "image visual feature + 'Ocean' + 'Sky'" or "image visual feature + 'Ocean' + 'Person'" for the second image. Top-10 returned images obtained by different hashing schemes are displayed. Our method is able to select query-adaptive hash bits and therefore gets more reasonable results.

Furthermore, to make quantitative study, we generate 1,000 random queries from those images associated with at least two tags. For each image, two tags are randomly selected from its tags to form a two-label query (we focus on two-label queries yet the proposed method is trivially extended to queries with single label or more than two labels). The ground truth for each query is defined as the set of images that satisfies two requirements:

- The image should be associated with the two tags in the query (note that these tag information of database images are kept unknown during the evaluation. They are only used to generate the ground truth).
- Moreover, the distance of their visual feature should be close enough, which is introduced to avoid a large set of ground truth images. In practice, we set a threshold to ensure at most 3,000 "good neighbors" for each query.

Figure 6 shows the performances of our proposed method and three baselines. Similar to Figure 4, the performances in terms of good neighbor number and precision under 32 hash bits per query are presented in Figure 6(a) and (b). And Figure 6(c) plots the mean-average-precision (MAP) values under varying hash bits per query. Suppose $k_b$ hash bits are used for each query ($k_b$ is from 8 to 48), our proposed BSH first learns $6k_b$ hash bits [5], which serve as the hash pool for query-adaptive hash bit selection. It is noticed that the proposed BSH outperforms all other three methods in all experiments.

# 5. CONCLUSIONS

In this paper, we propose an efficient multi-label hashing algorithm which exploits the shared subspaces among related labels in a boosting scheme. This method can automatically finds the similar label groups, and learns the shared hash functions for each group in the boosting procedure. Learning a hashing function for all data with different semantic similarities is very difficult, while the proposed method only focuses on the subset of each label group to overcome the difficulty and learns a more powerful hash functions for the label group. Our experimental results on two standard benchmarks, CIFAR-10 and NUS-WIDE, show that the proposed method can achieve better performance than state-of-the-art method when using the same total number of hash functions.

[5]This number of hash bits is a tunable parameter which is intrinsically determined by the sparsity of bit-label association matrix.

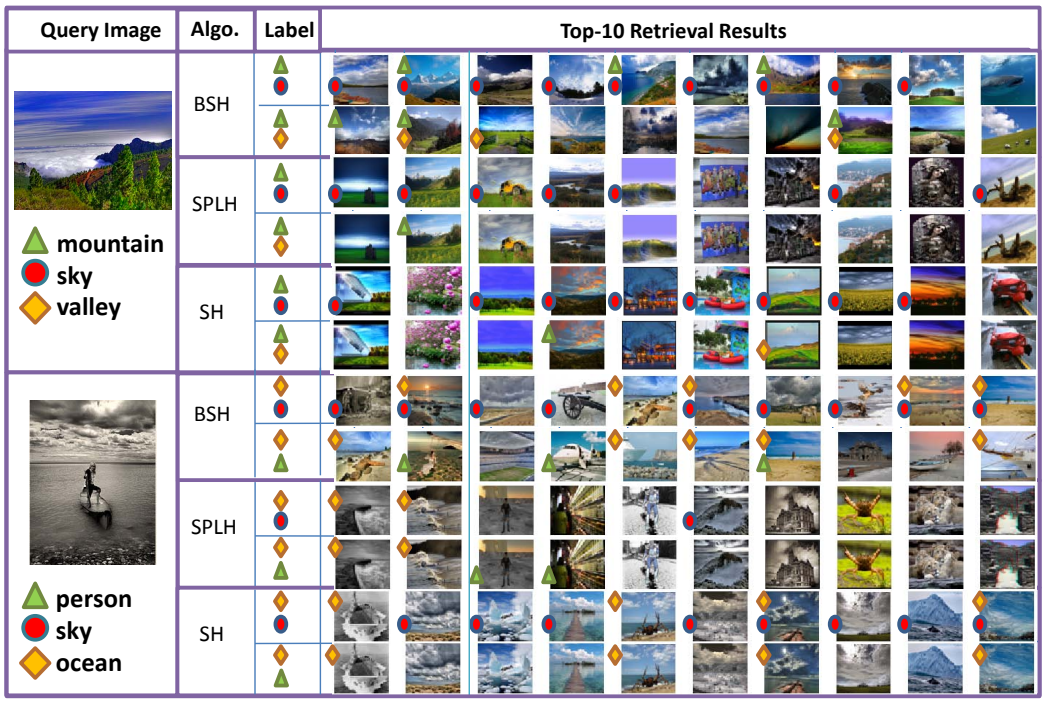



**Figure 7: Illustrating examples of image retrieval with multi-label data. The images and associated ground-truth tags are shown in the first column, and the tags that are selected for image retrieval (typically a subset of the ground-truth tags) are shown in the third column. Our proposed BSH gives much better results than other methods.**

## 6. ACKNOWLEDGEMENT

The first and third authors are supported in part by National Major Project of China "Advanced Unstructured Data Repository" (2010ZX01042-002-001-00) and Foundation of State Key Laboratory of Software Development Environment (SKLSDE-2011ZX-01).

## 7. REFERENCES

[1] A. Broder, M. Charikar, A. Frieze, and M. Mitzenmacher. Min-wise independent permutations. In *STOC*, 1998.

[2] R. Caruana. Multitask learning. *Mach. Learn.*, 28:41–75, 1997.

[3] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *STOC*, 2002.

[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 *CIVR*, 2009.

[5] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *SCG*, 2004.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009.

[7] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28, 1998.

[8] J. He, W. Liu, and S.-F. Chang. Scalable similarity search with optimized kernel hashing. In *ACM SIGKDD*, 2010.

[9] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *STOC*, 1998.

[10] S. Ji, L. Tang, S. Yu, and J. Ye. A shared-subspace learning framework for multi-label classification. *ACM Trans. Knowl. Discov. Data*, 4:8:1–8:29, 2010.

[11] Y.-G. Jiang, J. Wang, and S.-F. Chang. Lost in binarization: query-adaptive ranking for similar image search with compact codes. In *ICMR*, 2011.

[12] B. Kulis and T. Darrell. Learning to hash with binary reconstructive embeddings. In *NIPS*, 2009.

[13] W. Liu, J. Wang, S. Kumar, and S.-F. Chang. Hashing with graphs. In *ICML*, 2011.

[14] Y. Mu, X. Chen, T.-S. Chua, and S. Yan. Learning reconfigurable hashing for diverse semantics. In *ICMR*, 2011.

[15] Y. Mu, J. Shen, and S. Yan. Weakly-supervised hashing in kernel space. In *CVPR*, 2010.

[16] A. Torralba, K. Murphy, and W. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *CVPR*, 2004.

[17] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for scalable image retrieval. In *CVPR*, 2010.

[18] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *NIPS*, 2008.

[19] R. Yan, J. Tesic, and J. R. Smith. Model-shared subspace boosting for multi-label classification. In *ACM SIGKDD*, 2007.

[20] S. Yan, D. Xu, B. Zhang, H. Zhang, Q. Yang, and S. Lin. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(1):40–51, 2007.