# Compact Kernel Hashing with Multiple Features

Xianglong Liu[†], Junfeng He[‡], Di Liu[§], Bo Lang[†]
[†]State Key Lab of Software Development Environment, Beihang University, Beijing, China
[‡]Columbia University, New York, NY 10027, U.S.A.
[§]China Academy of Telecommunication Research of MIIT, Beijing, China
{xlliu, langbo}@nlsde.buaa.edu.cn jh2700@columbia.edu liudi@mail.ritt.com.cn

## ABSTRACT

Hashing methods, which generate binary codes to preserve certain similarity, recently have become attractive in many applications like large scale visual search. However, most of state-of-the-art hashing methods only utilize single feature type, while combining multiple features has been proved very helpful in image search. In this paper we propose a novel hashing approach that utilizes the information conveyed by different features. The multiple feature hashing can be formulated as a similarity preserving problem with optimal linearly-combined multiple kernels. Such formulation is not only compatible with general types of data and diverse types of similarities indicated by different visual features, but also helpful to achieve fast training and search. We present an efficient alternating optimization to learn the hashing functions and the optimal kernel combination. Experimental results on two well-known benchmarks CIFAR-10 and NUS-WIDE show that the proposed method can achieve 11% and 34% performance gains over state-of-the-art methods.

## Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; I.2.6 [**Artificial Intelligence**]: Learning

## General Terms

Algorithms, Experimentation

## Keywords

Multiple Feature, Multiple Kernel, Compact Hashing

## 1. INTRODUCTION

The explosive growth of the vision data motivates the recent studies on hash based nearest neighbor search. These methods have show promising performance in many applications. One of the most well-know methods is locality-sensitive hashing (LSH) [1]. It generates binary codes by projecting data on random vectors such that points within small distances share the same codes with a high probability. Following LSH, many well-designed hashing methods are proposed to solve different problems under various situations like unsupervised [11, 7] and supervised [9, 8] settings.

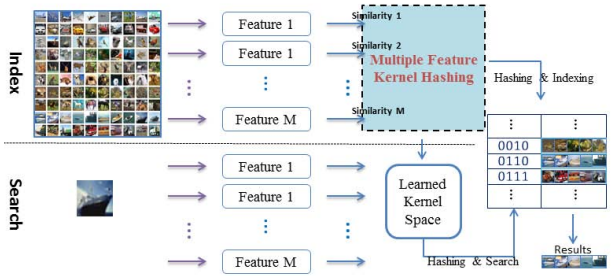



**Figure 1: Illustration of the proposed method.**

Despite of the aforementioned progress in hash based similarity search, most existing hashing methods still have one important limitation that only one visual feature is utilized. It is widely accepted that, in many applications instead of using a single feature type, a better way is to adaptively combine a set of diverse and complementary features to discriminate each data. For instance, in the literature content-based image retrieval systems gain significant performance improvement by fusing multiple features like color and texture [10]. Recently, local features combining with the global shape feature gives promising performance in mobile product search [4]. Feature combinations (feature fusion) is also very helpful in other domains like image classification [2].

To our best knowledge, there is very few works that learn hash functions incorporating multiple features expect [12] and [10]. In [12] the output of the hashing function is the convex output combination of linear hash functions on different sources, while [10] concatenates all features as one and projects it using the learned hashing hyperplane.

Although achieving promising improvement than single feature, these methods still either simply post-combine linear outputs of each feature type or equally concatenate all features as one. Different features may convey unbalanced and different information, which may be complementary to each other under different similarity space. Therefore it would be better to exploit the correlation between features. Moreover, these methods are not compatible with other data


*MM'12,* October 29–November 2, 2012, Nara, Japan.

types except vector type and different similarity measures widely involving in vision area. The correlation and importance of each feature type are still not fully exploited. In addition, partially due to features concatenation, these methods are computationally expensive in both training and searching (comparison details shown in Table 1), and therefore cannot be applied to solving large scale image search with a number of high-dimensional visual features.

In this paper, we propose a novel hashing approach that utilizes the similarities conveyed by different features. With concatenation of different features embedded into their similarity kernel space, the hashing problem can be formulated as a similarity preserving hashing with linearly combined multiple kernels. Kernel tricks are often more natural to gauge the similarity of general data types, where the underlying data embedding to the high-dimensional space is not known explicitly. Many hashing methods benefit from the use of domain specific kernel functions [6] [9] [5]. In this formulation, instead of a kernel function for single visual feature, a set of kernels for different features are combined and more discriminative ones in the combination are selected automatically. This is very similar to the popular method in computer vision named Multiple Kernel Learning (MKL) [3], which has shown to be able to better balance the similarities and improve the performance.

Due to the kernel form, the combination of multiple features embedded into their kernel space individually doesn't bring more computation than a single feature type. With this formulation, we efficiently and alternately optimize the hashing functions and the kernel combination using eigen-decomposition and quadratic programming respectively. It is worth highlighting that the proposed method: **1. is formulated in kernel form and thus compatible with general types of data with any kernel function. 2. combines diverse types of similarities indicated by different visual features, and preserves consistency of semantic similarity. 3. achieves fast training, indexing and search speed**.

The rest of the paper is organized as follows: We present details of our approach in Section 2. Section 3 describes settings of our experiments and discusses the experimental results. Finally, we conclude in Section 4.

## 2. PROPOSED APPROACH

The key idea of our proposed approach is to utilize a set of different features and their similarities by kernel functions. This can be formulated as a similarity preserving hashing with the optimal multiple kernels for visual features. In this section, we will first give the notations and formulation, and then present details how to alternately learn the optimal hashing functions and kernel combination coefficients.

### 2.1 Formulation

In this paper, we are given a set of $N$ training examples with $M$ visual features. The $m$-th feature ($d_m$ dimension) of $n$-th sample can be represented as $X_n^{(m)} \in R^{d_m \times 1}$. Then $X^{(m)} = [X_1^{(m)}, X_2^{(m)}, \ldots, X_N^{(m)}] \in R^{d_m \times N}$ is the $m$-th feature matrix of all training data.

In order to learn $P$ hashing functions $\{h_1, \ldots, h_P\}$, we give a formulation similar to [11] and [5]. The hash codes $Y$ ($P \times N$ matrix) for all training data are learned to preserve the semantic similarity $S_{ij}$ between $i$-th and $j$-th data points (usually $S$ is a sparse matrix), meanwhile satisfying balance and uncorrelated constrains. Unlike the previous kernelized methods, in this paper the hash codes are implicitly related to a series of embedding functions $\varphi_m(\cdot)$ corresponding to each visual feature by defining $\varphi(X_i) = [\mu_1^{\frac{1}{2}}\varphi_1^T(X_i^{(1)}), \ldots, \mu_M^{\frac{1}{2}}\varphi_M^T(X_i^{(M)})]^T$, which is comprised of $M$ individual embedded feature $\varphi_m(X_i^{(m)})$ weighted by $\mu_m^{1/2}$. Later we will show that it maps data point into the space defined by a convex combinations of kernels on different features. With the embedding function $\varphi(\cdot)$, the $p$-th hash function is defined as a linear projection:

$$h_p(\cdot) = sign(V_p^T\varphi(\cdot) + b_p),\ p = 1, \ldots, P. \tag{1}$$

Thus the $p$-th code for $i$-th sample will be $Y_{pi} = h_p(X_i)$.

The hyperplane vector $V_p$ in kernel space can be represented as a combination of $L$ landmarks $Z_l$ embedded in corresponding kernel space [6, 5, 9]:

$$V_p = \sum_{l=1}^{L} W_{lp}\varphi(Z_l),\ l = 1, \ldots, L. \tag{2}$$

Here $W$ is a $L \times P$ weight matrix, and the landmarks can be clutter centers obtained by clustering [7] or random subsamples [5]. For each feature, let $K^{(m)}$ denote the kernel corresponding to its embedding function $\varphi_m(\cdot)$, which means that $K_{ij}^{(m)} = \varphi_m(X_i^{(m)})^T\varphi_m(X_j^{(m)})$. Then

$$\begin{aligned} K_{ij} &= \varphi(X_i)^T\varphi(X_j) \\ &= \sum_{m=1}^{M}(\mu_m^{\frac{1}{2}}\varphi_m(X_i^{(m)}))^T(\mu_m^{\frac{1}{2}}\varphi_m(X_j^{(m)})) \\ &= \sum_{m=1}^{M}\mu_m K_{ij}^{(m)}. \end{aligned} \tag{3}$$

Therefore $K = \sum_{m=1}^{M}\mu_m K^{(m)}$, namely $\varphi(\cdot)$ actually defines a kernel linearly combined by kernels for each feature. Then codes $Y_i$ of $i$-th data can be rewritten in kernel form:

$$Y_i = sign(W^T K_i + b), \tag{4}$$

where $K_i$ is the $i$-th column of $L \times N$ kernel matrix $K_{L\times N}$ between $L$ landmarks and $N$ samples, and $b = [b_1, b_2, \ldots, b_P]$.

Now we give our formulation of multiple feature hashing:

$$\begin{aligned} \min_{W,b,\mu}\ & \frac{1}{2}\sum_{i,j=1}^{N} S_{ij}\|Y_i - Y_j\|^2 + \lambda\|V\|_F^2 \\ s.t.\ & Y_i \in \{-1, 1\}^P \\ & \sum_{i=1}^{N} Y_i = 0,\ \frac{1}{N}\sum_{i=1}^{N} Y_i Y_i^T = I \\ & \mathbf{1}^T\mu = 1,\ \mu \succeq 0. \end{aligned} \tag{5}$$

Such formulation forces the learned hash functions to preserve the given similarity $S$ as much as possible, by optimizing both the hyperplane vectors $W$ and the kernel combination weights $\mu$. Although in Problem 5, $\varphi(X_i)$ are formed by concatenating $\varphi_m(X_i^{(m)})$, but the final dimension after embedded into the kernel space is only $L$. Hence, the computation will be reduced much compared to methods [12, 10] which concatenate multiple raw features as one feature. The complexity comparison details are shown in Table 1,

**Table 1: Complexity of Different Multiple Feature Hashing Methods**

| methods | training | search |
|---|---|---|
| CHMS [12] | $O(T(D^3 + D^2N + DN_s))$ | $O(PD)$ |
| MFH [10] | $O(D^3 + D^2N + DN_s)$ | $O(PD)$ |
| Proposed | $O(T(LN_s + P^3))$ | $O(PM + ML)$ |

[1]Here $P, M, T \ll D, N_s \ll N^2$ and $L < D$ ($D = \sum_{m=1}^{M} d_m$).

where $T$ is the iteration number. The sparsity of $S$ helps reduce the computation involving $S$ from $O(N^2)$ to $O(N_s)$ ($N_s \ll N^2$) in all three methods.

## 2.2 Optimization

Due to the discrete constraints and non-convexity, the above optimization problem is difficult to solve. Similar to spectral hashing, the discrete constrains of $Y_i \in \{-1, 1\}^P$ can be relaxed as $Y_i = W^T K_i + b$.

Note that with either $W$ or $\mu$ fixed, fortunately the problem is convex with respect to the other. Therefore we present an alternating optimizing way that can efficiently find optimum in a few steps. First, we will show that given $\mu$, the optimal $W$ and $b$ of closed form can be obtained elegantly by eigen-decomposition.

**(1)** Given the fixed $\mu$, the optimal $W$ can be obtained by solving the following problem:

$$\begin{aligned} \min_{W} \quad & \mathrm{tr}(W^T C W) \\ s.t. \quad & W^T G W = I \end{aligned} \tag{6}$$

where,

$$C = K_{L\times N}(\Delta - S)K_{L\times N}^T + \lambda K_{L\times L}$$

$$G = \frac{1}{N} K_{L\times N}(I - \frac{1}{N}\mathbf{1}\mathbf{1}^T)K_{L\times N}^T.$$

Here $\Delta = \mathrm{diag}(S\mathbf{1})$, $b = -\frac{1}{N} W^T K_{L\times N}\mathbf{1}$. Such problem can be optimized efficiently by eigen-decomposition as [5] did.

**(2)** Given $W$ and $b$, the optimization with respect to $\mu$ can be formulated as a quadratic programming problem:

$$\begin{aligned} \min_{\mu} \quad & \frac{1}{2}\mu^T E \mu + h^T \mu \\ s.t. \quad & \mathbf{1}^T \mu = 1, \mu \succeq 0 \end{aligned} \tag{7}$$

where,

$$E_{ij} = 2\mathrm{tr}(W^T K_{L\times N}^{(i)}(\Delta - S){K_{L\times N}^{(j)}}^T W),\ i, j = 1, \ldots, M$$

$$h_i = \lambda \mathrm{tr}(W^T K_{L\times L}^{(i)} W),\ i = 1, \ldots, M.$$

Again, for space limit we omit the derivation. Finally the optimal solution of Problem 5 can be obtained by repeating the above two steps until it converges. In our experiments, it takes less than 10 iterations. For a novel sample $x$, its hash bits can be computed as

$$y = sign(W^T[K(x, Z_1), \ldots, K(x, Z_L)]^T + b). \tag{8}$$

The whole proposed algorithm is listed in Algorithm 1.

**Algorithm 1** Multiple Feature Kernel Hashing (MFKH)

1: Initialize $\mu_i = \frac{1}{M}$, $i = 1, \ldots, M$.
2: **repeat**
3: Fix $\mu$, calculate $W$ and $b$ by solving Problem 6;
4: Fix $W$, calculate $\mu$ by solving Problem 7;
5: **until** converge
6: Generate $Y$ according to Equation 8.

# 3. EXPERIMENTS

In this section we evaluate the proposed method and discuss the impact of multiple features. There are very few works designing compact hashing with multiple features except the recently proposed composite hashing with multiple sources (CHMS) [12]. We will compare our method with CHMS and other state-of-the-art well-known hashing methods like local sensitive hashing (LSH) [1], spectral hashing (SH) [11], and optimal kernel hashing (OKH) [5]. As [12] suggests, we tune appropriate parameters $C_1$ and $C_2$ for CHMS. For LSH and SH, we concatenate multiple features as one feature. All methods in our experiments will be run 10 times to suppress the effect of randomness.

## 3.1 Data Sets

We choose two well known datasets: **CIFAR-10** (60K) and **NUS-WIDE** (270K) as our experimental data sets. For simplicity and space limit, and similar to previous works, we adapt two visual features for each set to verify the efficiency of our proposed method.

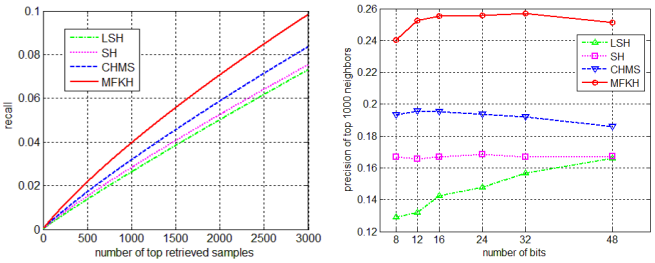
CIFAR-10 contains 60K $32 \times 32$ color images of 10 classes and 6K images in each class. For each image, we extract 384-D GIST feature and 300-D bag of visual words quantized from dense SIFT features of $8 \times 8$ patches with 4 space overlap. NUS-WIDE as one of largest real-world labeled image datasets comprises about 270K images with 81 ground truth concept tags, of which we consider 25 most frequent tags ('sky', 'animal', etc.). Besides, multiple visual features have been provided already in this set, and thus here we arbitrarily select two presentative features: 128-D wavelet texture and 225-D block-wise color moments. In all our experiments, we apply Gaussian RBF kernels for each feature.

For evaluation, we uniformly sample 3,000 and 5,000 images respectively as the training data for each dataset. To test the performance, we randomly select 1,000 and 3,000 query images respectively. The true neighbors are defined by whether two images share at least one common tag.

## 3.2 Results and Discussions

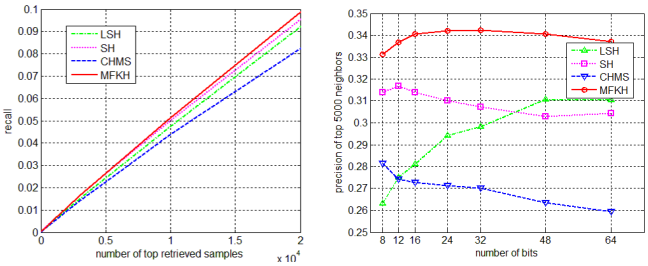
Figure 2 shows experimental results on CIFAR-10 including recall and precision of Hamming ranking. From Figure 2(a), it is clear that with 24 bits MFKH gives much better recall performance than other methods including CHMS, SH and LSH. To evaluate the impact of bit numbers, we increase $P$ from 8 bits to 48 bits. As Figure 2(b) depicts, increasing the bit number leads to increasing hamming ranking precision of top 1,000 results for all methods at first, and then a slight decrease for MFKH, SH and CHMS. This phenomenon is similar to those appearing in previous works [7], and we believe that one reason is that eigen-decomposition places most of the variance in top few principal directions, which substantially reduces the quality of following bits. However the proposed method outperforms others for all bits significantly, especially when using 32 bits (over 34% performance gain). This indicates that MFKH can provide very compact bits guaranteeing the performance.

We present similar results on the NUS-WIDE shown in Figure 3. The recall curve of Hamming ranking using 24 bits is plotted in Figure 3(a). Again MFKH outperforms CHMS




(a) Recall (24 bits) (b) Precision vs. # bits

**Figure 2: Performances comparison on CIFAR-10.**



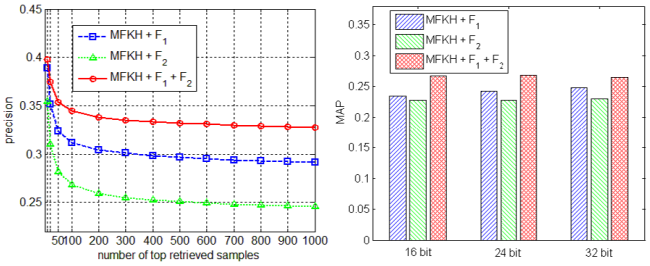

(a) Recall (24 bits) (b) Precision vs. # bits

**Figure 3: Performances comparison on NUS-WIDE.**

significantly, and also LSH and SH with multiple features. We show the top 5,000 precision in Figure 3(b) using varying number of bits. Due to similar reasons mentioned above, performance of all methods except CHMS increase at first when using more bits, and then decrease except LSH. But for all bits MFKH achieves the most superior performance consistently as on CIFAR-10 (over 11% performance gain).

We also compare performance of our method with multiple features and single features on NUS-WIDE. Note that with single feature, MFKH turns to be OKH [5]. From Figure 4, it can be observed that the performance with multiple features outperforms both single features ($F_1$: wavelet texture and $F_2$: color moments) as expected, which indicates that our multiple feature scheme helps improve the retrieval performance by incorporating the complementary information between features. The results are consistent with other related research on multiple feature fusion.

It should be noted that in all our experiments we just simply choose RBF kernels. More performance improvement might be archived, if the optimal kernels for different features are learned or chosen (for instance, Chi-Square kernels for histogram). The parameter $\lambda$ has slight effect on the performance according to our observation. Hence in all experiments we simply set $\lambda = 0.1$. Finally, in terms of training and search time, we use a workstation with 2.53 GHz Intel Xeon CPU and 10 GB Memory. On average CHMS takes more than 100 s to train on NUS-WIDE using 32 bits and 10 ms for each query, while MFKH archives much efficiency by only taking less than 15 s and 3 ms respectively.




(a) Precision (32 bits) (b) Precision vs. # bits

**Figure 4: Precision comparison of MFKH with single feature ($F_1$ and $F_2$) and multiple features ($F_1$+$F_2$).**

## 4. CONCLUSIONS

As described in this paper, we have proposed an efficient kernel hashing with multiple features. The hashing problem is formulated as a similarity preserving hashing with linearly combined multiple kernels, which is compatible with general data types and diverse similarities indicated by different visual features. Due to the efficient alternating optimizing way, our method achieves fast training, indexing and search speed. Experiment results on large-scale image retrieval prove the promising performance. Our work indicates that in the future more attention can be placed on how to utilize the information conveyed by different features.

## 5. ACKNOWLEDGMENTS

This work is supported 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).

## 6. REFERENCES

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

[2] P. Gehler and S. Nowozin. On feature combination for multiclass object classification. In *IEEE CVPR*, 2009.

[3] M. Gonen and E. Alpayd. Multiple kernel learning algorithms. *J. Mach. Learn. Res.*, 12, 2011.

[4] J. He, J. Feng, X. Liu, T. Cheng, T.-H. Lin, H. Chung, and S.-F. Chang. Mobile product search with bag of hash bits and boundary reranking. In *IEEE CVPR*, 2012.

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

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

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

[8] X. Liu, Y. Mu, B. Lang, and S.-F. Chang. Compact hashing for mixed image-keyword query over multi-label images. In *ACM ICMR*, 2012.

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

[10] J. Song, Y. Yang, Z. Huang, H. T. Shen, and R. Hong. Multiple feature hashing for real-time large scale near-duplicate video retrieval. In *ACM MM*, 2011.

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

[12] D. Zhang, F. Wang, and L. Si. Composite hashing with multiple information sources. In *ACM SIGIR*, 2011.