Comparing Apples to Oranges: A Scalable Solution with Heterogeneous Hashing

Mingdong Ou[1], Peng Cui[1], Fei Wang[2], Jun Wang[2], Wenwu Zhu[1], Shiqiang Yang[1]
[1]Tsinghua National Laboratory for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University, Beijing, China
[2]IBM Watson Research Center, Yorktown Heights, NY, U.S.
oumingdong@gmail.com,cuip@tsinghua.edu.cn, {fwang,wangjun}@us.ibm.com,
wwzhu@tsinghua.edu.cn, yangshq@tsinghua.edu.cn

ABSTRACT

Although hashing techniques have been popular for the large scale similarity search problem, most of the existing methods for designing optimal hash functions focus on homogeneous similarity assessment, i.e., the data entities to be indexed are of the same type. Realizing that heterogeneous entities and relationships are also ubiquitous in the real world applications, there is an emerging need to retrieve and search similar or relevant data entities from multiple heterogeneous domains, e.g., recommending relevant posts and images to a certain `Facebook` user. In this paper, we address the problem of "comparing apples to oranges" under the large scale setting. Specifically, we propose a novel *Relation-aware Heterogeneous Hashing* (RaHH), which provides a general framework for generating hash codes of data entities sitting in multiple heterogeneous domains. Unlike some existing hashing methods that map heterogeneous data in a common Hamming space, the RaHH approach constructs a Hamming space for each type of data entities, and learns optimal mappings between them simultaneously. This makes the learned hash codes flexibly cope with the characteristics of different data domains. Moreover, the RaHH framework encodes both homogeneous and heterogeneous relationships between the data entities to design hash functions with improved accuracy. To validate the proposed RaHH method, we conduct extensive evaluations on two large datasets; one is crawled from a popular social media sites, `Tencent Weibo`, and the other is an open dataset of `Flickr`(NUS-WIDE). The experimental results clearly demonstrate that the RaHH outperforms several state-of-the-art hashing methods with significant performance gains.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

Keywords

Heterogeneous Hashing; Heterogeneous Similarity Search; Heterogeneous Network; Scalability; Big Data

1. INTRODUCTION

With the fast growth of heterogeneous networks, especially social media networks like `Facebook`, `Flickr` and `Twitter`, it has attracted increasing attention to study and explore the interactions across heterogeneous domains. For example, image tagging aims at giving descriptive textual tags to images, recommendation on microblogging service focuses on providing relevant posts to certain users, and the targeted advertising is to send advertisements to potential customers via tracking their online traits. In general, these applications tend to find similar or relevant entities across different relational domains in heterogeneous networks. As those networks are often huge and associated with big data, there is an emerging need to design an efficient and scalable mechanism to evaluate heterogeneous similarity and search heterogeneous entities.

Hashing is a highly scalable indexing strategy for approximate nearest neighbor search [1, 13, 23, 25]. It encodes data entities into binary hash codes in Hamming space, where the search can be extremely efficient. In addition, the learned hash functions are usually in a simple form and the generation of hash codes can be done in a realtime manner. However, most of the existing hashing technologies are designed for homogeneous data, i.e., the data points indexed by a hash table should be of the same type. There are critical challenges for indexing heterogeneous data entities. First, different domains often have different characteristics, thus the corresponding mapped Hamming spaces should be also different. How to effectively bridge the gaps between those Hamming spaces and perform search across different hash tables? Second, it is often to see that there exist heterogeneous relationships and connections between the data entities from different domains. How to leverage such relationships into the hashing function learning process? Although there are a few recent studies designing new hashing techniques to index multi-modal data entities into a common Hamming space [4, 26, 15, 20, 27], such a single-Hamming space mapping strategy could be problematic and does not fit into real world scenarios due to the intrinsic heterogeneity of the multi-modal representations. In addition, the heterogeneous relationships are very critical but have not been considered by most of the existing approaches. Hence, rapidly searching similar data entities over heterogeneous domains still remains as an open issue.

To address these challenges, in this paper, we propose a novel hashing technique, namely *Relation-aware Heterogeneous Hashing* (RaHH), for indexing and searching large scale heterogeneous data. It utilizes the data features, the homogeneous relationship within each single domain, and the heterogeneous relationship across different domains to learn hash functions for each type of data entities, as well as optimal mappings between the hash codes of different types of data entities. In particular, we formulate the learning procedure as a joint optimization problem with respect to both hash functions and mapping functions. An efficient *Block-Coordinate Descent* (BCD) strategy [21] is applied to derive optimal solutions. Finally we validate the performance of the RaHH approach on two large datasets; one is crawled from Tencent Weibo, and the other is an open dataset of `Flickr`, i.e. NUS-WIDE[5].

The rest of this paper is organized as follows. Section 2 briefly reviews the related works. The detailed formulation and solution of RaHH is presented in Section 3. Experimental validation on two real world datasets is presented in Section 4, followed by our conclusion in Section 5.

2. RELATED WORK

The rapid growth of the applications with big data in many areas, including social media, genomics, sensor networks, and even business analytics, promotes the study of large scale search and retrieval. Due to computational and memory efficiency, hashing based indexing techniques have attracted more attentions in the recent decade. In particular, many new methods are developed through leveraging sophisticated machine learning and data mining algorithms to boost up the search efficiency and accuracy. In this section, we will briefly survey the existing hashing techniques and motivate our study for designing heterogeneous hashing technique.

The earliest hashing methods, including the well-known locality sensitive hashing (LSH) [10] and MinHash [3], are based on either random projections or permutations, resulting in data-independent hash functions. Although the asymptotic property is theoretically guaranteed, the practical performance is often limited [24, 25]. Realizing the limitations of the random techniques, many new hashing methods are designed through integrating either data properties or supervision information to achieve compact hash codes with improved accuracy. Representative unsupervised methods include spectral hashing [25], graph hashing [17], iterative quantization hashing [8], isotropic hashing [12], angular quantization hashing [7], spherical hashing [9], and so on. The key idea for those data-dependent hashing methods lies in the exploration of data properties for hash function design. For instance, spectral hashing explores the data distributions and the graph hashing utilizes the geometric structure of data for designing data-dependent hash functions. Supervised learning paradigms, ranging from kernel learning to metric learning to deep learning, have been exploited to learn binary codes and many supervised hashing methods are proposed [14, 16, 19, 22]. In addition, the semi-supervised hashing method was recently proposed to achieve accurate yet balanced hash codes [24].

Most of the aforementioned hashing techniques are designed for single type homogeneous features. Note that

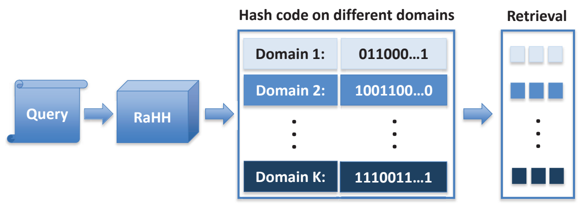


Figure 1: The flowchart of how RaHH is applied to a new query and get its similar items from different domains.

many practical applications involve the usage of heterogeneous feature representations. Recently, several new hashing methods are proposed to index the data points using multi-modal or multi-view feature representations [26, 27, 20, 4, 15]. For instance, Zhen et al. proposed to use a co-regularization framework to generate binary codes from each modality, while enforcing the agreement between the hash codes from different feature modalities [27]. As mentioned earlier, most of these existing heterogeneous hashing methods attempt to project all the heterogeneous features or entities to a common Hamming space. However, such a Hamming embedding often results in poor indexing performance due to the lack of commonality among the heterogeneous domains. In addition, besides heterogeneous entities, a typical heterogeneous network, like `Facebook` and `Twitter`, also contains heterogeneous connections or relationships between those entities, which have not been really considered during the design of hash functions by most existing works. Hence, for such a social network, it remains as an open issue to design an efficient and accurate hashing technique which could leverage all available homogeneous and heterogeneous information into the learning process. To address the above issues, in the following section, we will present a heterogeneous hashing technique.

3. THE METHODOLOGY

In this section we will introduce our *Relation-aware Heterogeneous Hashing* (RaHH) method in detail. As stated in the introduction section, the goal of RaHH is to learn a Hamming embedding for each type of data, and mappings between different Hamming embeddings such that we can get the corresponding hash codes in its relational domains. In this way, given a new query, we can use RaHH to fast retrieve similar entities in its own data domain as well as similar data entities from other relational domains. Fig.1 provides a conceptual diagram of the procedure of using RaHH to retrieve similar items from heterogeneous domains. We formulate the RaHH as a joint optimization problem over the homogeneous data hash codes and heterogeneous hash code mappings, in which we utilize data features, homogeneous and heterogeneous relationships. Fig.2 demonstrates the leveraged information for learning the RaHH functions. In the following, the notations and symbols are first introduced and will be used throughout the paper. Then, we give the detailed formulation of the proposed RaHH method, followed by an efficient optimization strategy using the block coordinate descent approach. We also provide an out-of-sample extension for calculating the hash codes in an online setting. Finally, the complexity analysis for both offline training and online hash code generation are discussed.

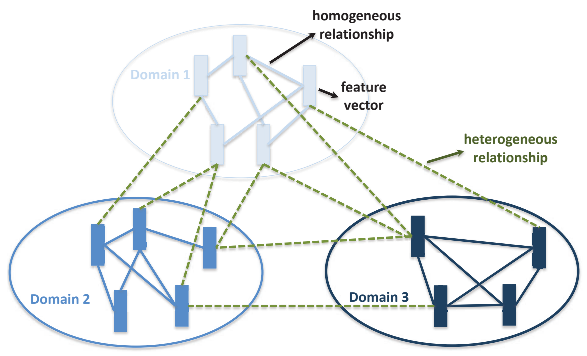


Figure 2: Different types of information we used in RaHH.

3.1 Notations and Problem Statement

Suppose we have a set of data items $\mathcal{V} = \{\mathcal{V}^p\}_{p=1}^P$ from P relational domains, where $\mathcal{V}^p = \{v_i^p\}_{i=1}^{m_p}$ is the dataset in the p-th domain with v_i^p being the i-th datum. We use $\mathbf{X}^p = [\mathbf{x}_1^p, \mathbf{x}_2^p, \cdots, \mathbf{x}_{m_p}^p] \in \mathbb{R}^{d_p \times m_p}$ to represent the data matrix of the p-th domain, and d_p is the dimensionality of the feature space of the p-th domain. $\mathbf{H}^p = [\mathbf{h}_1^p, \mathbf{h}_2^p, \cdots, \mathbf{h}_{m_p}^p] \in \{-1, 1\}^{r_p \times m_p}$ is the hash code matrix for the data in the p-th domain, with $\mathbf{h}_i^p$ being the hash code vector for v_i^p. In addition, $\mathbf{R}^p \in \mathbb{R}^{m_p \times m_p}$ denotes the homogeneous relationship matrix of the p-th domain, and $\mathbf{R}^{pq} \in \mathbb{R}^{m_p \times m_q}$ is the heterogeneous relationship matrix between the p-th domain and the q-th domain. We assume $\mathbf{H}^p$ can be mapped to $\mathbf{H}^q$ via a linear mapping $\mathbf{W}^{pq} \in \mathbb{R}^{r_p \times r_q}$. The goal of RaHH is to obtain the optimal $\{\mathbf{H}^p\}_{p=1}^P$ and $\{\mathbf{W}^{pq}\}_{p,q=1}^P$ via solving the following optimization problem

$$\min_{\{\mathbf{H}^p\}_{p=1}^P, \{\mathbf{W}^{pq}\}_{p,q=1}^P} \mathcal{J}^{ho}(\{\mathbf{H}^p\}) + \beta \mathcal{J}^{he}(\{\mathbf{H}^p\}, \{\mathbf{W}^{pq}\}) \quad (1)$$
$$s.t. \quad \mathbf{H}^p \in \Omega^p, \forall\ p = 1, 2, \cdots, P.$$

Here $\mathcal{J}^{ho}$ is the homogeneous loss term, and $\mathcal{J}^{he}$ is the heterogeneous loss term. $\beta > 0$ is the tradeoff parameter. Ω^p is the set imposing constraints on $\mathbf{H}^p$. Now we will introduce in detail how these terms are defined in RaHH.

3.2 Homogeneous Loss

In order to construct $\mathcal{J}^{ho}$, we assume that: (1) data objects with similar individual features are similar to each other; (2) data objects with strong relationships are similar to each other; (3) similar data objects tend to have similar hash codes. For item (1), we can use the data inner product matrices $\{(\mathbf{X}^p)^\top \mathbf{X}^p | p = 1, 2, \cdots, P\}$ as the feature based similarity matrices for each domain if we assume the data in all domains are normalized to unit norm. For item (2), we can use the homogeneous relationship matrix $\mathbf{R}^p$ to capture the data similarity. Then we can construct the following composite data similarity matrix to encode the pairwise data similarities

$$\mathbf{A}^p = (\mathbf{X}^p)^\top \mathbf{X}^p + \alpha \mathbf{R}^p, \quad (2)$$

where the constant $\alpha > 0$ is the combination weight.

For item (3), we can construct a smoothness term for the data hash codes to enforce that similar data would have similar codes. Specifically, we design the following $\mathcal{J}^{ho}$.

$$\mathcal{J}^{ho}(\{\mathbf{H}^p\}) = \frac{1}{2} \sum_{p=1}^{P} \sum_{i,j=1}^{m_p} A_{ij}^p \|\mathbf{h}_i^p - \mathbf{h}_j^p\|^2, \quad (3)$$

where A_{ij}^p is the (i, j)-th element of $\mathbf{A}^p$. Clearly, when minimizing $\mathcal{J}^{ho}(\{\mathbf{H}^p\})$, a larger A_{ij}^p will cause a closer $\mathbf{h}_i^p$ and $\mathbf{h}_j^p$. We further construct the constraint set Ω^p as

$$\begin{aligned} \Omega^p \quad = \quad & \{\mathbf{H}^p | \mathbf{H}^p \in \{-1, 1\}^{r_p \times m_p}, \\ & \mathbf{H}^p \mathbf{1} = \mathbf{0}, \mathbf{H}^p (\mathbf{H}^p)^\top = m_p \mathbf{I}\}. \end{aligned} \quad (4)$$

We impose the constraint $\mathbf{H}^p \mathbf{1} = \mathbf{0}$ to preserve the balance of each bit, and $\mathbf{H}^p (\mathbf{H}^p)^\top = m_p \mathbf{I}$ to enforce that different bits capture complementary information, as suggested in [24, 25].

3.3 Heterogeneous Loss

As the data from multiple domains might be associated with different metric spaces, we cannot measure the similarity between heterogeneous items directly. To search similar items from relational domains, RaHH first assumes that the hash code for a datum in domain p can be linearly mapped to the Hamming space of a relational domain q. Then the mapped hash code is used to search nearest neighbors in the domain q. More concretely, RaHH maps $\mathbf{H}^p$ to each bit(row) of $\mathbf{H}^q$ respectively through utilizing the heterogeneous relation matrix $\mathbf{R}^{pq} \in \mathbb{R}^{m_p \times m_q}$. By treating $\mathbf{H}^p$ as a feature matrix and H_k^q as class labels, we cast the mapping problem as a series of binary classification problems and define $\mathcal{J}^{he}(\{\mathbf{H}^p\}, \{\mathbf{W}^{pq}\})$ as

$$\mathcal{J}^{he}(\{\mathbf{H}^p\}, \{\mathbf{W}^{pq}\}) = \sum_{p \sim q} \sum_k \sum_{i,j} l_{ijk}^{pq} + \lambda \|\mathbf{w}_k^{pq}\|^2, \quad (5)$$

where $p \sim q$ indicates domain p has relationship with domain q, and the logistic loss

$$l_{ijk}^{pq} = ln(1 + e^{-R_{ij}^{pq} H_{kj}^q (\mathbf{w}_k^{pq})^\top \mathbf{h}_i^p}) \quad (6)$$

measures the prediction loss after we map the hash code of v_i^p to the k-th bit on the q-th domain. To minimize the loss, H_{kj}^q and $(\mathbf{w}_k^{pq})^\top \mathbf{h}_i^p$ needs to be close for a large R_{ij}^{pq}, which suggests that for strongly associated v_i^p and v_j^q, the mapped hash code of v_i^p in the domain q should be as similar as the hash code of v_j^q.

3.4 Final Cost and Optimization Strategy

Bringing Eq.(3), Eq.(4) and Eq.(5) together into the original cost function defined in Eq. (1), we can derive the final cost function. Due to the binary constraint expressed in Ω^p, the cost function in Eq.(1) is not differentiable. Moreover, the balance constraint also makes problem 1 NP-hard to solve [25]. Therefore, we propose to relax those hard constraints and convert them into soft penalty terms. Specifically, we add the following three regularizers to the cost function,

$$\theta_1(\{\mathbf{H}^p\}) = \sum_{p=1}^{P} \|\mathbf{H}^p \odot \mathbf{H}^p - \mathbf{E}\|_F^2 \quad (7)$$

$$\theta_2(\{\mathbf{H}^p\}) = \sum_{p=1}^{P} \|\mathbf{H}^p \mathbf{1}\|^2 \quad (8)$$

$$\theta_3(\{\mathbf{H}^p\}) = \sum_{p=1}^{P} \|\mathbf{H}^p(\mathbf{H}^p)^\top - m_p\mathbf{I}\|_F^2, \quad (9)$$

where $\mathbf{E} \in \mathbb{R}^{r_p \times m_p}$ is an all-one matrix, $\mathbf{1} \in \mathbb{R}^{m_p \times 1}$ is an all-one vector, and $\mathbf{I}$ is an identity matrix. It is easy to see that these three regularizers correspond to the three relaxed constraint sets in $\{\Omega^p\}$. Then the relaxed version of the original cost function is

$$\begin{aligned}\mathcal{J} = \; & \mathcal{J}^{ho}(\{\mathbf{H}^p\}) + \beta\mathcal{J}^{he}(\{\mathbf{H}^p\},\{\mathbf{W}^{pq}\}) \\ & +\gamma_1\theta_1(\{\mathbf{H}^p\}) + \gamma_2\theta_2(\{\mathbf{H}^p\}) + \gamma_3\theta_3(\{\mathbf{H}^p\}) \quad (10)\end{aligned}$$

To minimize the above cost $\mathcal{J}$, we will present a *Block Coordinate Descent* (BCD) approach, as described in the following.

3.4.1 *Block Coordinate Descent*

Since the final cost function in Eq. (10) is not jointly convex with respect to all the variables, here we use BCD method [21] to search a local optimal solution. Specifically, the gradients are calculated as

$$\frac{\partial \mathcal{J}}{\partial \mathbf{w}_k^{pq}} = \sum_i \sum_j \frac{-R_{ij}^{pq} H_{kj}^q \mathbf{h}_i^p}{1 + e^{R_{ij}^{pq} H_{kj}^q (\mathbf{w}_k^{pq})^\top \mathbf{h}_i^p}} + 2\lambda \mathbf{w}_k^{pq} \quad (11)$$

$$\begin{aligned}\frac{\partial \mathcal{J}}{\partial H_{ki}^p} = & \frac{\partial \mathcal{J}^{ho}(\{\mathbf{H}^p\})}{\partial H_{ki}^p} + \beta \frac{\partial \mathcal{J}^{he}(\{\mathbf{H}^p\},\{\mathbf{W}^{pq}\})}{\partial H_{ki}^p} \\ & +\gamma_1 \frac{\partial \theta_1(\{\mathbf{H}^p\})}{\partial H_{ki}^p} + \gamma_2 \frac{\partial \theta_2(\{\mathbf{H}^p\})}{\partial H_{ki}^p} + \gamma_3 \frac{\partial \theta_3(\{\mathbf{H}^p\})}{\partial H_{ki}^p} (12)\end{aligned}$$

The gradient components in Eq.(12) are given as follows

$$\begin{aligned}\frac{\partial \mathcal{J}^{ho}(\{\mathbf{H}^p\})}{\partial H_{ki}^p} &= \sum_j A_{ij}^p (H_{ki}^p - H_{kj}^p) \\ &= (\mathbf{H}^p diag(A^p\mathbf{1}) - (\mathbf{H}^p(\mathbf{X}^p)^\top)\mathbf{X}^p - \alpha\mathbf{H}^p\mathbf{R})_{ki}\end{aligned}$$

$$\frac{\partial \mathcal{J}^{he}(\{\mathbf{H}^p\},\{\mathbf{W}^{pq}\})}{\partial H_{ki}^p} = \sum_q \sum_j \left[\sum_g \frac{-R_{ij}^{pq} H_{gj}^q W_{kg}^{pq}}{1 + e^{R_{ij}^{pq} H_{gj}^q (\mathbf{w}_g^{pq})^\top \mathbf{h}_i^p}} + \frac{-R_{ji}^{qp} (\mathbf{w}_k^{qp})^\top \mathbf{h}_j^q}{1 + e^{R_{ji}^{qp} H_{ki}^p (\mathbf{w}_k^{qp})^\top \mathbf{h}_j^q}} \right]$$

$$\frac{\partial \theta_1(\{\mathbf{H}^p\})}{\partial H_{ki}^p} = 4((H_{ki}^p)^2 - 1)H_{ki}^p = 4((\mathbf{H}^p \odot \mathbf{H}^p - \mathbf{E}) \odot \mathbf{H}^p)_{ki}$$

$$\frac{\partial \theta_2(\{\mathbf{H}^p\})}{\partial H_{ki}^p} = 2\sum_j H_{kj}^p = 2(\mathbf{H}^p\mathbf{1})_k$$

$$\begin{aligned}\frac{\partial \theta_3(\{\mathbf{H}^p\})}{\partial H_{ki}^p} &= 4\sum_j (H_k^p (H_j^p)^\top - m_p I_{kj}) H_{kj}^p \\ &= 4((\mathbf{H}^p(\mathbf{H}^p)^\top - m_p\mathbf{I})\mathbf{H}^p)_{ki}\end{aligned}$$

where $\{\mathbf{H}^p(\mathbf{X}^p)^\top\}$, $\{\mathbf{H}^p\mathbf{1}\}$ and $\{(\mathbf{H}^p(\mathbf{H}^p)^\top - m_p\mathbf{I})\}$ are three statistics denoted by $\mathbf{S}$, which will be used to accelerate the optimization algorithm.

Algorithm 1 Relation-aware Heterogeneous Hashing (RaHH)

```
Require: $\{\mathbf{X}^p\}$, $\{\mathbf{R}^p\}$, $\{\mathbf{R}^{pq}\}$
Ensure: $\{\mathbf{H}^p\}$, $\{\mathbf{W}^{pq}\}$
 1: initialize $\{\mathbf{H}^p\}$ by CVH and $\{\mathbf{W}^{pq}\}$ as identity matrix
 2: initialize $\mathbf{S}$
 3: while the value of objective function don't converge do
 4:   for each domain $p$ do
 5:     for each entity $i$ in domain $p$ do
 6:       calculate the gradients with respect to $\mathbf{h}_i^p$
 7:       update $\mathbf{h}_i^p$ by one step gradient descent
 8:       update statistics $\mathbf{S}$
 9:     end for
10:     for each domain $q$ do
11:       for each bit k of domain $q$ do
12:         calculate gradients with respect to $\mathbf{w}_k^{pq}$
13:         update $\mathbf{w}_k^{pq}$ by one step gradient descent
14:       end for
15:     end for
16:   end for
17: end while
```

Algorithm 2 Out-of-sample Extension for Relation-aware Heterogeneous Hashing

```
Require: statistics $\mathbf{S}$, $\mathbf{x}_{m_p+1}^p$, $\mathbf{r}_{m_p+1}^p$ and $\{\mathbf{r}_{m_p+1}^{pq}\}_{q=1}^P$ connected with the out-of-sample entity $v_{m_p+1}^p$
Ensure: $\mathbf{h}_{m_p+1}^p$
 1: initialize $\mathbf{h}_{m_p+1}^p$ by CVH
 2: while the value of objective function don't converge do
 3:   calculate gradients with respect to $\mathbf{h}_{m_p+1}^p$
 4:   update $\mathbf{h}_{m_p+1}^p$ by one step gradient descent
 5: end while
```

Finally, we optimize the objective function by iteratively updating $\mathbf{H}$ and $\mathbf{W}$ until the value of objective function converges. We describe the training procedure in Algorithm 1.

3.5 Out of Sample Extension

It is critical to derive the out of sample extension for computing the hash code for any query datum in an online setting. In the formulation of the proposed RaHH, we can easily compute the hash code for an out-of-sample entity v_i^p by minimizing Eq. (10). Since the hash tables are constructed and the mappings $\{\mathbf{W}^{pq}\}_{p,q=1}^P$ are learned during the offline training process, we only need to minimize the cost in Eq. (10) with respect to the new entry v_i^p. Similar to the method introduced in Section 3.4, a gradient descent can be applied to efficiently compute the optimal hash code for the entity v_i^p. The detailed procedure for out of sample extension is described in Algorithm 2.

3.6 Complexity Analysis

We first analyze the online complexity for computing the hash code of an out-of-sample query point v_i^p. With the statistics $\mathbf{S}$ calculated previously, the time complexity of calculating gradients and updating for a single entity v_i^p is

$$O(d_p r_p + r_p s_i^p + r_p \sum_q r_q s_i^{pq}),$$

where s_i^p is the number of homogeneous relations connected with v_i^p, s_i^{pq} is the number of heterogeneous relations connected with v_i^p in domain q, and d_p is the dimensionality of the features in domain p. We can see that the time complexity for generating the hash code for v_i^p is linear with respect to the number of relations connected with it. As the relations are often sparse, RaHH can be very efficient to generate the hash codes and can be applied to large scale and real-time applications.

During the training procedure, besides the computational cost of updating for single entities, the additional time complexity for updating statistics $\mathbf{S}$ is $O(d_p r_p + (r_p)^2)$. Finally, the time complexity of calculating gradients and updating $\mathbf{W}^{pq}$ is $O(s^{pq} r_p r_q)$, where s^{pq} is the number of heterogeneous relations across domain p and q. Hence, the total time complexity of training RaHH is

$$O(\sum_p ((d_p r_p + (r_p)^2)m_p + r_p s^p + r_p \sum_q r_q s^{pq})),$$

where s^p is the number of homogeneous relations in domain p. We can see that the training time complexity is linear to the size of the training set and the number of relations. In practice, all the scale-free networks, like `Facebook` and `Twitter`, are often sparsely connected. Early study shows that the total connections for a reliable scale-free network is sublinear to the number of entities [6]. Therefore, in practice, the training cost is with linear complexity to the number of training entities, and the worst case in theory is with quadratic complexity when the network is fully connected. In summary, the proposed RaHH method has affordable offline training cost and the online hash code generation is extremely fast.

4. EXPERIMENTS

In this section we will present the experimental results on applying our RaHH algorithm to two real world data sets. First we will introduce the basic information of them. We run experiments implemented by Matlab on a machine running Windows Server 2008 with 12 2.4GHz cores, 32GB memory.

4.1 Dataset Information

Tencent Weibo Dataset is crawled from Tencent Weibo[1] which is one of the biggest microblogging service in China. We use two domains, users and posts, in our evaluation. The dataset contains 34,676 users and 439,509 posts. Each user has 3 user labels[2] at least, and there are 4,385 user labels in total. Each post contains at most 140 Chinese characters. We use probability distribution on 50 topics detected by Latent Dirichlet Allocation (LDA) [2] from user labels as user feature vector, and friendship as homogeneous relationship for user domain. The post feature vector is constructed with the probability distribution of 50 topics detected by LDA on post, and no homogeneous relationship in the post domain is used. We use user-post adoption/rejection behaviors as positive/negative heterogeneous relationships between users and posts. User-post adoption behaviors are recorded when users post or forward posts. However, user-post rejection behavior, which is defined as a user does not like a post, cannot be observed explicitly, as we are not sure a user did not adopt a post because he(she) did not like it or just did not see it. Just like [11], based on the assumption that users will see the posts around the adopted posts, we assume t (which is set to 5 in our experiment) nearest unadopted posts around adopted post on user's Timeline[3] as rejected. We obtain 483,038 positive heterogeneous relationships and 1,039,441 negative heterogeneous relationships.

NUS-WIDE Dataset is a fully labeled web image dataset. It contains about 260,000 images from flickr. Each image is labeled by at least one concept, and also described by text tags. We use the probability distribution of 100 LDA topics on text tags as feature vector for text tags, and 500-dimensional Shift Invariant Feature Transform (SIFT) feature [18] as feature vector for images. No homogeneous relationship is used in either image or text tag domains.

4.2 Baseline Methods

We compare RaHH with the following baseline algorithms.

Cross View Hashing(CVH)[15] performs multi-modal hashing via a Canonical Correlation Analysis procedure. In our implementation, we treat each positive relation as a data object, and the two domains as two modalities. For novel queries, CVH can obtain their codes in both modalities based on their features.

Modified Spectral Hashing(MSH) is a straightforward extension of Spectral Hashing(SH)[25]. We construct a unified similarity matrix. Similarity between entities in same domain is $\{\mathbf{A}^p\}$, and similarity between heterogeneous entities is positive part of $\mathbf{R}^{pq}$(i.e. $(\mathbf{R}^{pq} + |\mathbf{R}^{pq}|)/2$). For out-of-sample data, in order to exploit relation, we calculate hash codes by Nyström method.

Multimodal Latent Binary Embedding(MLBE)[26] is a probabilistic model which formulates hash codes as latent variables and learns them in discrete form. It can preserve both homogeneous similarity and heterogeneous relation.

RaHH_NR is a variant of RaHH which does not exploit heterogeneous relation in out-of-sample extension.

RaHH_NC is a variant of RaHH without any regularizers in both training and out-of-sample extension procedures.

Here CVH is a representative purely feature-based method. MSH and MLBE are representative relation-aware methods. RaHH_NR and RaHH_NC are used to demonstrate the effectiveness of heterogeneous relations and regularizers.

4.3 Evaluation Metric

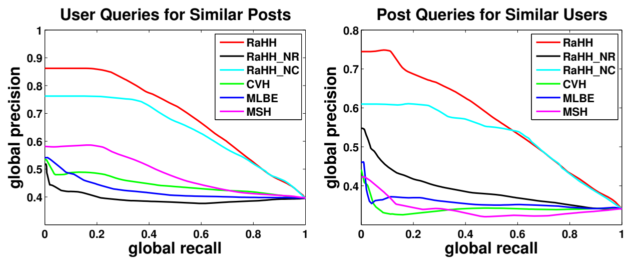
We use *precision, recall* and *Mean Average Precision* (MAP) as our evaluation metrics. We propose two ways (i.e. global and entity-wise) to calculate precision and recall. Specifically, let $\mathcal{P}$ and $\mathcal{N}$ be the set of positive and negative pairs of heterogeneous entities respectively, $HD(i,j)$ be the Hamming Distance between data entities v_i and v_j, then the metrics are defined below.

- Global Precision and Global Recall, or GP and GR for short, are defined as the precision and recall of retrieval

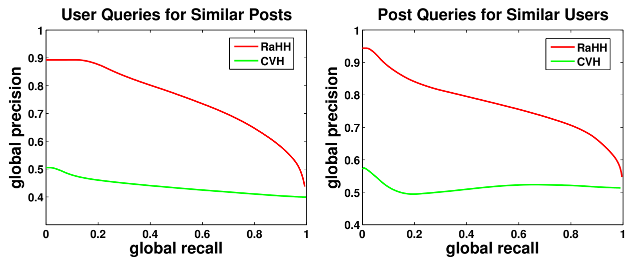
[1]`http://t.qq.com`

[2]The user labels are specified by the users themselves according to their personal interests and biography.

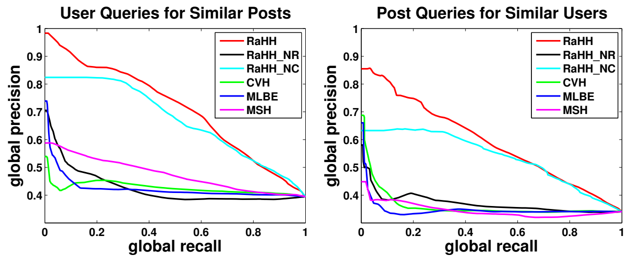
[3]Posts are pushed to users in chronological order in Tencent Weibo, and the set of posts pushed to users in this order is called Timeline



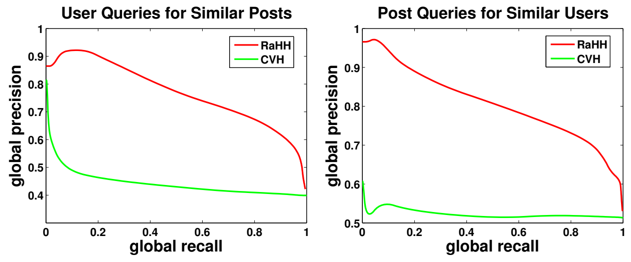
(a) Small scale dataset,$r = 8$



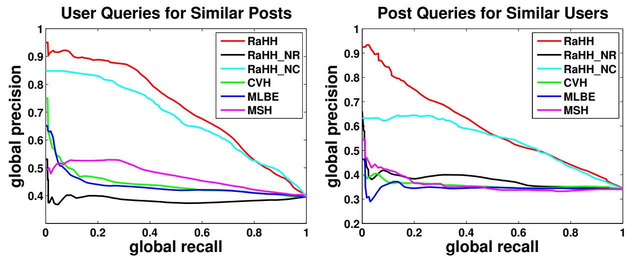
(b) Large scale dataset,$r = 8$



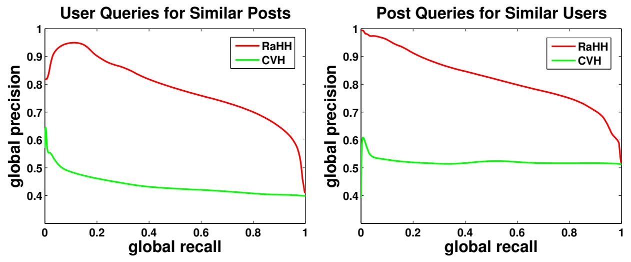
(c) Small scale dataset,$r = 16$



(d) Large scale dataset,$r = 16$



(e) Small scale dataset,$r = 32$



(f) Large scale dataset,$r = 32$

Figure 3: Results of two settings, i.e., user queries for similar posts and post queries for similar users, on the Tencent Weibo dataset using various numbers of hash bits ($r = 8, 16, 32$).

results within Hamming radius d for the query set:

$$GP(d) = \frac{|\{(i,j)|(i,j) \in \mathcal{P}, HD(i,j) \leq d\}|}{|\{(i,j)|(i,j) \in \mathcal{P} \cup \mathcal{N}, HD(i,j) \leq d\}|}$$

$$GR(d) = \frac{|\{(i,j)|(i,j) \in \mathcal{P}, HD(i,j) \leq d\}|}{|\mathcal{P}|}$$

We obtain the precision-recall curve by series of precision-recall pairs on different Hamming distance.

- Query-wise Precision (QP) is defined as the precision of retrieval results within Hamming radius d for query i:

$$QP(d,i) = \frac{|\{(i,j)|(i,j) \in \mathcal{P}, HD(i,j) \leq d\}|}{|\{(i,j)|(i,j) \in \mathcal{P} \cup \mathcal{N}, HD(i,j) \leq d\}|}$$

If the retrieval of similar entities for a query fails, the precision will be assigned to zero. We use the mean precision within Hamming radius 2 to evaluate the retrieval accuracy.

- Mean Average Precision(MAP) of Hamming ranking:

$$AP(i) = \frac{\sum_j P_i(j)\delta_i(j)}{|\{(i,j)|(i,j) \in \mathcal{P}\}|}$$

$$MAP = \frac{\sum_i AP(i)}{m}$$

where $P_i(j)$ denotes the precision of the top j retrieved entities for query i, $\delta_i(j) = 1$ if the relation between j-th entity and query i is positive and $\delta_i(j) = 0$ otherwise, and m is the number of query entities.

4.4 Results on Tencent Weibo Dataset

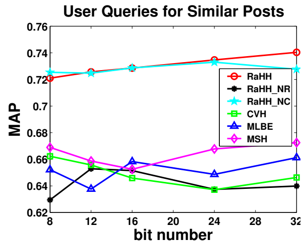
We use two settings to evaluate the effectiveness of RaHH: (1) given user as query, and retrieve similar posts; (2) given post as query, and retrieve similar users. We conduct experiment on ten small scale subsets and one large scale subset. Each small set contains about 500 users and 500 posts, and the large scale set contains 19,330 users and 169,696 posts. We adopt an iterative reduction strategy to obtain the datasets. All users and posts that are not involved in any adoption behaviors are deleted from the original data set. For all those data sets, we use 90% data as the training set, and the rest 10% data as the query set.

4.4.1 Parameter Setting

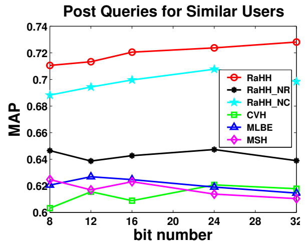
There are four tradeoff parameters in the objective of RaHH as can be seen from Eq.(10). β controls the weight of heterogeneous relationship term, and γ_1, γ_2, γ_3 control the weight of regularizer terms. To obtain a good group of parameters, we did grid search, and change those four parameters independently. Table 1 shows the mean MAP of the two retrieval settings using various numbers of hash bits. Parameter group 4 achieves the best performance, and we adopt those parameter values in the rest of the experiments.

Table 1: Mean MAP of RaHH for different parameter groups on small scale subsets of Tencent Weibo Dataset using various numbers of hash bits.

group	β	γ_1	γ_2	γ_3	mean MAP
1	10	1	0.0003	0.3	0.708
2	10	10	0.003	3	0.680
3	100	1	0.0003	0.3	0.722
4	100	10	0.003	3	**0.725**
5	100	100	0.03	30	0.696
6	1000	10	0.003	3	0.720
7	1000	100	0.03	30	0.724

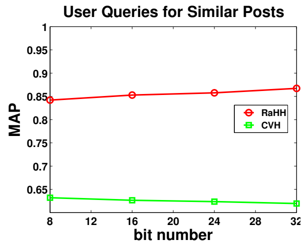


(a) User query results

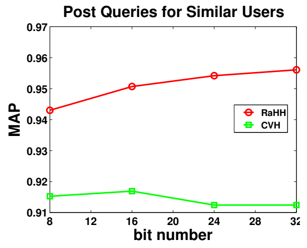


(b) Poster query results

Figure 4: MAP curve on small scale subsets of Tencent Weibo dataset by varying the numbers of hash bits.



(a) User query results



(b) Post query results

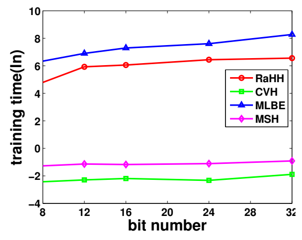
Figure 5: MAP curve on large scale subset of Tencent Weibo dataset by varying the numbers of hash bits.

4.4.2 *Results*

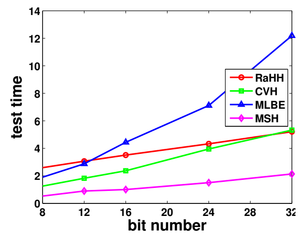
Figure 3 shows the precision-recall curve obtained on these datasets. Figure 4 and 5 show the MAP on those small scale subsets and large scale subset. MLBE and MSH requires large computational storage and cannot be tested on our large scale subset, we only report the results with RaHH and CVH. From these results we can see that RaHH significantly outperforms those baselines in all datasets. Some potential weakness of those baseline algorithms include:

- CVH only preserve homogeneous feature similarity in out-of-sample extension. It cannot capture the cross-domain relationships between queries and other domains.
- MSH does not differentiate homogeneous similarity and heterogeneous relationships which are essentially different.
- MLBE does not require the independency between different bits, and may generate highly redundant bits.

Comparing the performance of RaHH and RaHH_NR, we can observe that heterogeneous relationships are complementary with homogeneous similarity and adding heterogeneous relationships yields great improvement of performance. Comparing the performance of RaHH and RaHH_NC, we can see that RaHH_NC overfits the training set, which leads to poor generalization performance on the query set.



(a) Training time



(b) Retrieval time

Figure 6: Time costs curve on small subsets of Tencent Weibo dataset by varying the numbers of hash bits. The time unit is second.

We also test the scalability of RaHH and baseline algorithms and the results are shown in Figure 6. Because of its quadratic complexity with respect to the training set size and sextic complexity with respect to the number of hash bits, MLBE costs most training time which is about one order of magnitude longer than our algorithm. RaHH costs more test time when the length of hash codes is small, however, as the number of bits grows, MLBE costs more test time.

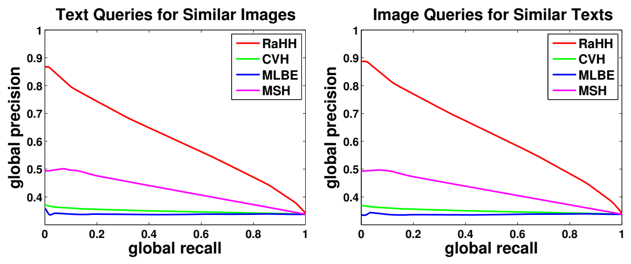
With the flexible mapping across domains, RaHH is allowed to learn hash codes with different lengths for different domains. Table 2 shows the empirical results on small subsets , where first column and first row are the lengths of hash codes for user domain and post domain respectively. We can see that MAPs in left down are larger than that in upper right which suggests that users tend to need more bits to represent than posts. One possible reason is that one user is often interested in more than one topic and one post often belongs to only one topic, users need more bits to represent their diverse interests. RaHH achieves best MAP when assign 32 bits for user domain and 16 bits for post domain. Comparing with the best result (assign 32 bits for each domain) in uniform code length setting, the best result with different code lengths can achieve higher MAP and save 50% storage in post domain.

Table 2: MAP of RaHH on small subsets of Tencent Weibo dataset for different pairs of bit numbers

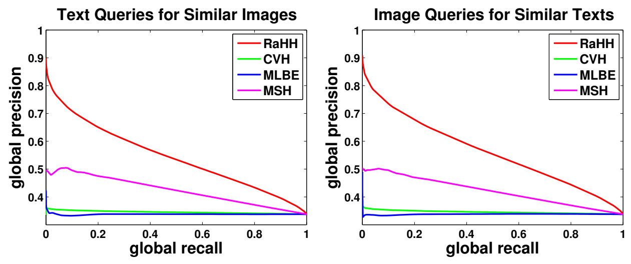
User \ Post	8	12	16	24	32
8	0.716	0.718	0.717	0.712	0.712
12	0.717	0.720	0.721	0.722	0.720
16	0.725	0.728	0.725	0.725	0.731
24	0.726	0.726	0.738	0.729	0.731
32	0.729	0.723	**0.739**	0.728	0.734

4.5 Results on NUS-WIDE Dataset

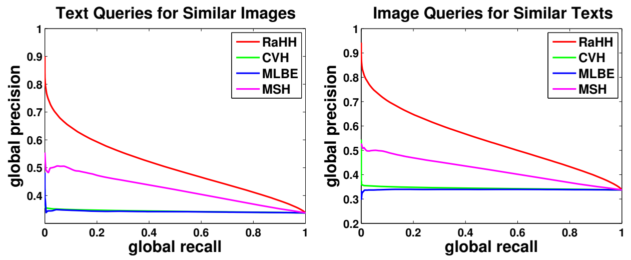
Similar to the experiment on Tencent Weibo Dataset, we also designed two cross-domain retrieval settings on NUS-WIDE dataset: (1) given text (set of tags) as query, and retrieve similar images; (2) given image as query, and retrieve similar texts. We select the images and their text tags



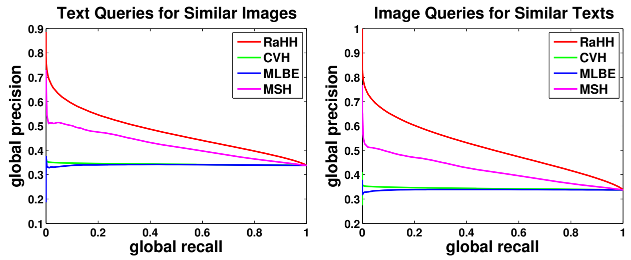
(a) r=8



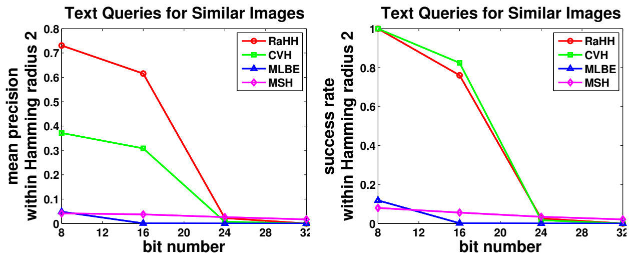
(b) r=16



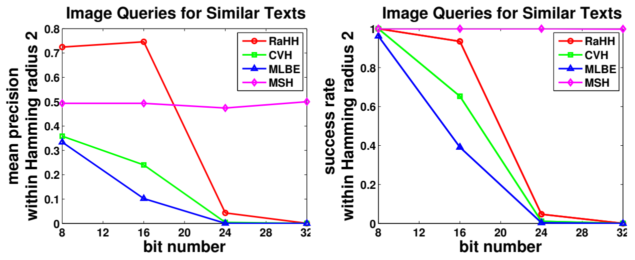
(c) r=24



(d) r=32



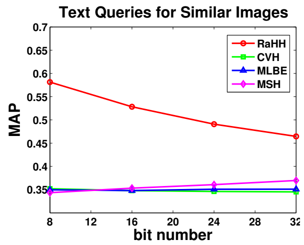
(e)



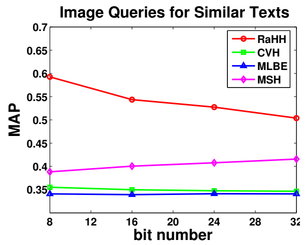
(f)

Figure 7: Results of two retrieval settings, i.e., text queries for similar images and image queries similar texts, on NUS-WIDE dataset using various numbers of hash bits($r = 8, 16, 24, 32$).

belong to 10 largest concepts as our experimental database, then randomly select 300 images as training set, 10000 images as test set, 2000 images as query set, and 5% heterogeneous relations for training. We applied experiment on 5 such datasets. Figure 8 shows the average MAP curves, our



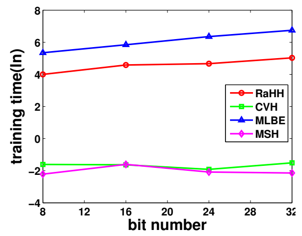
(a) Text query results



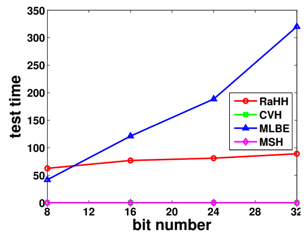
(b) Image query results

Figure 8: MAP curve on NUS-WIDE dataset by varying the number of hash bits.

algorithm outperforms the best baseline about 60% at most. Figure 7 shows precision-recall curve in the top two rows, and the curve of mean precision within Hamming radius 2 and corresponding success rate in the bottom row. Just like the results on Tencent Weibo dataset, the precision-recall curves of RaHH are much better than the others. When the length of hash codes is small(e.g. 8 bits and 16 bits), RaHH obtain better mean precision than the others due to both high success rate and precision in successful retrievals. MSH and MLBE obtain poor performance in retrieval setting(1) due to the low success rate. When the length of hash codes grows longer, as the success rate of retrieval decreases rapidly, the mean precision of most methods decrease to a low level that cannot be applied in practice. The mean precision of MSH in retrieval setting(2) didn't decrease, but, the mean precision of MSH is also too low to be applied in practice.



(a) Training time



(b) Retrieval time

Figure 9: Time costs curve on NUS-WIDE dataset by varying the number of hash bits. The time unit is second.

Figure 9 shows the training time and test time of different lengths of hash codes. MLBE is the most time-consuming method in most situations. The test time cost of one query in our method is about 30 milliseconds which is reasonable for real-time applications.

5. CONCLUSIONS

In this paper, we propose a *Relation-aware Heterogeneous Hashing* (RaHH) approach for efficient similarity search on large scale heterogeneous data. In particular, through leveraging both homogeneous and heterogeneous relationships, we learn a Hamming embedding to construct hash tables for each data domain. Meanwhile, linear mappings are simultaneously learned as bridges between different Hamming embeddings. To achieve efficient training process, a Block Coordinate Descent method is applied to derive optimal solutions. With this flexible hashing framework, we can capture the characteristics of different data domains to generate more effective and accurate hash codes. The empirical study demonstrates the superiority of the proposed RaHH method, compared to several state-of-the-art hashing techniques. Our future directions include applying sophisticated learning algorithms, e.g., kernel learning and metric learning, to further improve the performance with an affordable training cost.

6. ACKNOWLEDGEMENTS

This work is supported by National Natural Science Foundation of China, No. 61003097, No. 60933013, and No. 61210008; International Science and Technology Cooperation Program of China, No. 2013DFG12870; National Program on Key Basic Research Project, No. 2011CB302206.

7. REFERENCES

[1] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Proc. of FOCS*, pages 459–468, 2006.

[2] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[3] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher. Min-wise independent permutations. In *Proc. of STOC*, pages 327–336, 1998.

[4] M. Bronstein, A. Bronstein, F. Michel, and N. Paragios. Data fusion through cross-modality metric learning using similarity-sensitive hashing. In *Proc. of CVPR*, pages 3594–3601, 2010.

[5] 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 *Proc. of CIVR*, page 48, 2009.

[6] C. I. Del Genio, T. Gross, and K. E. Bassler. All scale-free networks are sparse. *Physical Review Letters*, 107(17):178701, 2011.

[7] Y. Gong, S. Kumar, V. Verma, and S. Lazebnik. Angular quantization-based binary codes for fast similarity search. In *Proc. of NIPS*, pages 1205–1213, 2012.

[8] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *Proc. of CVPR*, pages 817–824, 2011.

[9] J.-P. Heo, Y. Lee, J. He, S.-F. Chang, and S.-E. Yoon. Spherical hashing. In *Proc. of CVPR*, pages 2957–2964, 2012.

[10] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proc. of STOC*, pages 604–613, 1998.

[11] M. Jiang, P. Cui, F. Wang, Q. Yang, W. Zhu, and S. Yang. Social recommendation across multiple relational domains. In *Proc. of CIKM*, pages 1422–1431, 2012.

[12] W. Kong and W.-J. Li. Isotropic hashing. In *Proc. of NIPS*, pages 1655–1663, 2012.

[13] B. Kulis and K. Grauman. Kernelized locality-sensitive hashing for scalable image search. In *Proc. of ICCV*, pages 2130–2137, 2009.

[14] B. Kulis, P. Jain, and K. Grauman. Fast similarity search for learned metrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2143–2157, 2009.

[15] S. Kumar and R. Udupa. Learning hash functions for cross-view similarity search. In *Proc. of IJCAI*, pages 1360–1365, 2011.

[16] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang. Supervised hashing with kernels. In *Proc. of CVPR*, pages 2074–2081, 2012.

[17] W. Liu, J. Wang, S. Kumar, and S.-F. Chang. Hashing with graphs. In *Proc. of ICML*, pages 1–8, 2011.

[18] D. G. Lowe. Object recognition from local scale-invariant features. In *Proc. of ICCV*, pages 1150–1157, 1999.

[19] M. Norouzi, D. Fleet, and R. Salakhutdinov. Hamming distance metric learning. In *Proc. of NIPS*, pages 1070–1078, 2012.

[20] 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 *Proc. of SIGMM*, pages 423–432, 2011.

[21] D. Sontag, A. Globerson, and T. Jaakkola. Introduction to dual decomposition for inference. *In S. Sra, S. Nowozin, and S. Wright, Eds., Optimization for Machine Learning. MIT Press*, pages 219–252, 2010.

[22] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *Proc. of CVPR*, pages 1–8, 2008.

[23] J. Wang, S. Kumar, and S.-F. Chang. Sequential projection learning for hashing with compact codes. In *Proc. of ICML*, pages 1127–1134, 2010.

[24] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for large-scale search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(12):2393–2406, 12 2012.

[25] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Proc. of NIPS*, pages 1753–1760, 2008.

[26] Y. Zhen and D. Yeung. A probabilistic model for multimodal hash function learning. In *Proc. of SIGKDD*, pages 940–948, 2012.

[27] Y. Zhen and D.-Y. Yeung. Co-regularized hashing for multimodal data. In *Proc. of NIPS*, pages 1385–1393, 2012.