Scalable Heterogeneous Translated Hashing

Ying Wei[§], Yangqiu Song[†], Yi Zhen[‡], Bo Liu[§], Qiang Yang[§,¶]
[§]Hong Kong University of Science and Technology, Hong Kong
[†]University of Illinois at Urbana-Champaign, Urbana, IL, USA
[‡]Duke University, Durham, NC, USA
[¶]Huawei Noah's Ark Lab, Hong Kong
[§]{yweiad,bliuab,qyang}@cse.ust.hk, [†]yqsong@illinois.edu, [‡]zhenyisx@gmail.com

ABSTRACT

Hashing has enjoyed a great success in large-scale similarity search. Recently, researchers have studied the multi-modal hashing to meet the need of similarity search across different types of media. However, most of the existing methods are applied to search across multi-views among which explicit bridge information is provided. Given a heterogeneous media search task, we observe that abundant multi-view data can be found on the Web which can serve as an auxiliary bridge. In this paper, we propose a *Heterogeneous Translated Hashing* (HTH) method with such auxiliary bridge incorporated not only to improve current multi-view search but also to enable similarity search across heterogeneous media which have no direct correspondence. HTH simultaneously learns hash functions embedding heterogeneous media into different Hamming spaces, and translators aligning these spaces. Unlike almost all existing methods that map heterogeneous data in a common Hamming space, mapping to different spaces provides more flexible and discriminative ability. We empirically verify the effectiveness and efficiency of our algorithm on two real world large datasets, one publicly available dataset of Flickr and the other MIRFLICKR-Yahoo Answers dataset.

Categories and Subject Descriptors: H.3 [Information Storage and Retrieval]: Information Search and Retrieval; H.4 [Information Systems Applications]: Miscellaneous.

Keywords: Hash Function Learning; Heterogeneous Translated Hashing; Scalability.

1. INTRODUCTION

With the explosive growth of data on and off the Web, heterogeneity arising from different data sources has become ubiquitous. There exist numerous interactions among a diverse range of heterogeneous media: summarizing a piece of video with textual keywords, displaying advertisements by understanding the content of a mobile game, and recommending products based on social activities including sending messages, checking in at places, adding friends, and posting pictures. Figure 1 shows a simple example of leveraging images to provide more accurate answers in Question-Answering systems. All these applications boil down to a fundamental problem: *similarity search across heterogeneous modalities*.

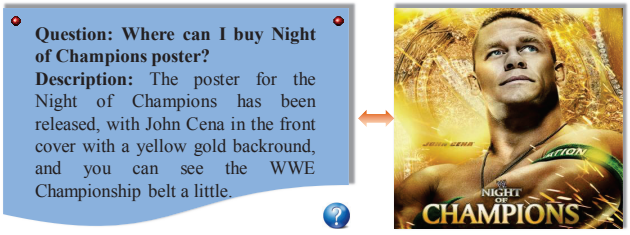


Figure 1: An example of using images to help better question-answering. With a specific poster of "Night of Champions", the answer to where to buy can be more precise.

The challenges of similarity search across heterogeneous modalities are two-fold: 1) how to efficiently perform the computation to meet the large amount of data available; and 2) how to effectively compare the similarity with the existence of heterogeneity. A brute force similarity comparison between the examples from different media is prohibitively expensive for large-scale datasets. Traditional space partitioning methods which accelerate similarity search, such as KD-trees [2] and Metric trees [24], have poor performance in high dimensional spaces [26]. Due to their constant or sub-linear query speed and low storage cost, hashing based methods initiated by locality sensitive hashing (LSH) [1, 7], have aroused more and more interest and become a main-stream technique for fast approximate nearest neighbour (ANN) search. The key principle of hashing is to learn compact binary codes that can preserve similarity. In other words, similar points in the original feature space are projected to similar hash codes in the Hammming space. However, these methods all work with homogeneous data points.

To apply hashing across heterogeneous media is a non-trivial task. First, data from different media sources have incommensurable representation structures. Second, besides preserving homogeneous media similarity in the way as traditional hashing does, heterogeneous media similarity should be preserved simultaneously. Heterogeneous media similarity is defined as semantic relatedness between a pair of entities in different modalities. For instance, a query image and a document in database are similar if they derive from the same topic, e.g., "sports". Such heterogeneous correspondence data that are labelled as similar or dissimilar are the "bridge" to search across heterogeneous media. However, in the task of illustrating questions with pictures as Figure 1 shows, questions as queries do not have any correspondence with the pre-defined database of images. Thus, the third challenge is that in most of practical applications, the explicit relationships between query en-

KDD'14, August 24–27, 2014, New York, NY, USA.

http://dx.doi.org/10.1145/2623330.2623688.

tities (in one domain) and the database entities (in another domain) probably do not apparently exist.

So far limited attempts have been made towards hashing across heterogeneous media. The existing works such as [4, 12, 32, 15] all focus on the situation where explicit relationships are given. For example, the method proposed in [12] assumes that the data is formatted in a multi-view fashion, i.e., each data instance in the database should have a representation in each view. Therefore, explicit relationship is clearly given for each data instance. Particularly, the method proposed in [15] even relies on the explicit relationships between queries and the database in the testing phase.

Moreover, most existing approaches embed multiple media data into a common Hamming space, thus generating hash codes with the same number of bits for all modalities. However, such an embedding is unreasonable because different media types usually have different dimensionality and distributions. In fact, researchers have argued that in uni-modal hashing using the same number of bits for all projected dimensions is unsound because dimensions of larger variances carry more information [8, 13]. Analogously, heterogeneous media data with incommensurable representations and distributions also carry different amounts of information so that they should not be collectively hashed into binary codes of the same length. Otherwise, the equal treatment of different modalities can deteriorate the hashing performance. To the best of our knowledge, the work of [15] is among the first to adopt different bits for different modalities and correlating these bits with mapping functions. However, as mentioned above, it re-learns hash codes for out-of-sample data highly reliant on the given relationships between queries and the database, which is neither practical nor efficient.

In this paper, we propose a novel learning method to enable translation-based hashing across heterogeneous media called *Heterogeneous Translated Hashing* (HTH) to address these limitations. Given a heterogeneous media search task, we observe that some multi-modal data are available on the Web which can serve as a bridge to preserve heterogeneous media similarity, while massive uncorrelated examples in each individual modality can be incorporated to enhance homogeneous media similarity preservation. Learning from such auxiliary heterogeneous correspondence data and homogeneous unlabelled data, HTH generates a set of hash functions for each modality that can project entities of each media type onto an individual Hamming space. All of the Hamming spaces are aligned with a learned translator. In practice, we formulate the above learning procedure as a joint optimization model. Despite the non-convex nature of the learning objective, we express it as a difference of two convex functions to enable the application of the concave-convex procedure (CCCP) [29] iteratively. Then we employ the stochastic sub-gradient strategy [20] to efficiently find the local optimum in each CCCP iteration. Finally, we conduct extensive experiments on two real-world large scale datasets and demonstrate our proposed method to be both effective and efficient.

The remainder of this paper is organized as follows. We review the related work in Section 2. In Section 3, we present the formulation and optimization details of the proposed method. Experimental results and analysis on two real-world datasets are shown in Section 4. Finally, Section 5 concludes the paper.

2. RELATED WORK

In this section, we briefly review the related work in two categories. We first introduce the recently developed learning to hash methods, which is the background of our approach. Then we review several current state-of-the-art hashing methods across heterogeneous modalities.

2.1 Learning to Hash

LSH [1, 7] and its variations [6, 10, 11, 17] , as the earliest exploration of hashing, generate hash functions from random projections or permutations. Nevertheless, these data-independent hash functions may not confirm to every application, and hence require very long hash codes, which increases costs of storage and online query, to achieve acceptable performances. Recently, data-dependent learning to hash methods attempt to alleviate the problem via learning hash functions from data. Unsupervised learning (spectral hashing (SH) [27], self-taught hashing (STH) [31], anchor graph hashing (AGH) [13]), supervised learning (Boosting [19], semantic hashing [18], LDAHash [23]) and semi-supervised learning (semi-supervised hashing [25]) have been explored since then. These approaches have significantly improved the hashing results for many specific tasks.

2.2 Hash acorss Heterogeneous Modalities

To the best of our knowledge, only a few research attempts towards multi-modal hashing have been made to speed up similarity search across different feature spaces or modalities.

Bronstein et al. [4] first explored the cross-modality similarity search problem and proposed cross-modal similarity sensitive hashing (CMSSH). It embeds multi-modal data into a common Hamming space. Later, several works [12, 28, 30, 32, 33, 34] were proposed. Both cross-view hashing (CVH) [12] and inter-media hashing (IMH) [22] extend spectral hashing to preserve intra-media and inter-media similarity simultaneously. Nevertheless, CVH enables cross-view similarity search given multi-view data whereas IMH adds a linear regression term to learn hash functions for efficient code generation of out-of-sample data. Zhen et al. [32] extended label-regularized max-margin partition (LAMP) [14] to the multi-modal case. Multimodal latent binary embedding (MLBE) [33] presents a probabilistic model to learn binary latent factors which are regarded as hash codes in the common Hamming space. Parametric local multimodal hashing (PLMH) [30] extends MLBE and learns a set of local hash functions for each modality. Recently, Zhu et al. [34] and Wu et al. [28] presented two new techniques for obtaining hash codes in multi-modal hashing. [34] obtains k-bit hash codes of a specific data point via thresholding its distances to k cluster centres while [28] thresholds the learned sparse coefficients for each modality as binary codes.

All these methods assume that the hashed data reside in a common Hamming space. However, this may be inappropriate especially when the modalities are quite different. Relational-aware heterogeneous hashing (RaHH) [15] addresses this problem by generating hash codes with different lengths (one for each modality) together with a mapping function. Unfortunately, RaHH has to adopt a fold-in scheme to generate hash codes for out-of-sample data, which is time-consuming, because it learns codes directly instead of explicit hash functions. In the next section, we elaborate our approach to eliminate these restrictions.

3. HETEROGENEOUS TRANSLATED HASHING

In this section, we present our approach in detail. We first introduce the general framework which consists of an offline training phase and an online querying phase. After introducing the notations and problem definitions, we show that HTH can be achieved by solving a novel optimization problem, and we develop an effective and efficient algorithm accordingly. The whole algorithm and the complexity analysis are given at the end of this section.

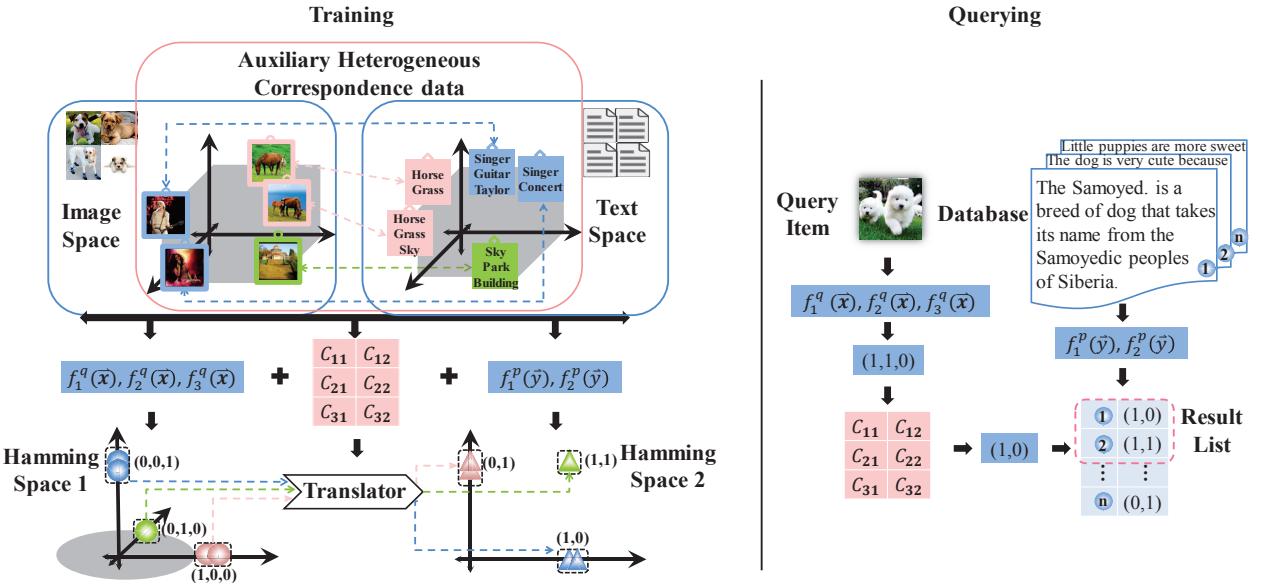


Figure 2: The flowchart of the proposed Heterogeneous Translated Hashing framework. The left corresponds to the offline training of hash functions and the translator, and the right summarizes the process of online querying.

3.1 Overview

We illustrate the HTH framework in Figure 2. HTH involves two phases: an offline training phase (left) and an online querying phase (right). For the simplicity of presentation, we focus on two heterogeneous media types, namely, images and text documents. Nevertheless, it is straightforward to extend HTH to more general cases with three or more types of media. During the offline training phase, HTH learns: 1) the hash functions for each media type to map the data to their individual Hamming space, which has dimensions of the number equalling the code length; and 2) a translator to align the two Hamming spaces. Since effective hash codes should simultaneously preserve homogeneous and heterogeneous media similarity, the correspondence between different domains is needed. In this work, we use auxiliary tagged images crawled from Flickr as the "bridge" shown in the central pink bounding box in Figure 2 which encloses images, documents and their relationships. Meanwhile, a proportion of queries, e.g., images, are incorporated to enhance intra-similarity preservation together with auxiliary images as the left blue bounding box which encloses all images shows. The same applies to text documents. With hash functions, similar homogeneous instances of each media type should be hashed into the same or close bucket in its Hamming space as displayed in Figure 2. Moreover, hash codes of one media type can be translated into the other Hamming space so that mutually correlated data points across different domains are expected to have small Hamming distances.

In the online querying phase, the database, e.g., a pile of text documents, is pre-encoded into a hash table via applying corresponding learned hash functions. When a new query instance comes, we first generate its hash codes using the domain specific hash functions. Subsequently, the hash codes are translated to the Hamming space of the database via the learned translator. Using existing hardware techniques such as bit operations, we can compute the Hamming distances between the query and all database instances, and retrieve its nearest neighbours efficiently.

3.2 Notations and Problem Definition

Suppose we are given a few query data instances $\tilde{\mathbf{X}}^q = \{\tilde{\mathbf{x}}_i\}_{i=1}^N$ and a large database $\tilde{\mathbf{Y}}^p = \{\tilde{\mathbf{y}}_j\}_{j=1}^M$, where $\tilde{\mathbf{x}}_i \in \mathbb{R}^{d_q}$ is a d_q dimensional feature vector in the query domain and $\tilde{\mathbf{y}}_j \in \mathbb{R}^{d_p}$ represents a d_p-dimensional vector in the feature space of the database. In addition, we are given a set of auxiliary data points from both modalities and their relationships which are expressed as a triple set $\mathcal{A}_{xy} = \cup_{i=1}^{N_1} \cup_{j=1}^{N_2} \{\mathbf{x}_i^*, \mathbf{y}_j^*, s_{ij}\}$, in which $s_{ij} = 1$ indicates that the instances $\mathbf{x}_i^*$ and $\mathbf{y}_j^*$ are correlated while $s_{ij} = 0$ otherwise. We construct the training set $\mathcal{T}_x = \{\mathbf{x}_i\}_{i=1}^{N_x}$ of the query domain as follows: randomly sample n instances from $\tilde{\mathbf{X}}^q$ and select all auxiliary data points corresponding to the query domain, i.e., $N_x = n + N_1$. Similarly, $\mathcal{T}_y = \{\mathbf{y}_j\}_{j=1}^{N_y}$ with $N_y = m + N_2$. Our goal is to learn two sets of hash functions and a translator from the training set $\mathcal{A}_{xy}$, $\mathcal{T}_x$ and $\mathcal{T}_y$. The two sets of hash functions, $\mathcal{F}^q(\mathbf{x}) = \{f_k^q(\mathbf{x})\}_{k=1}^{k_q}$ and $\mathcal{F}^p(\mathbf{y}) = \{f_l^p(\mathbf{y})\}_{l=1}^{k_p}$, project the query and database domain into a

Table 1: Definition of Notations

Notation	Description	Number	Set Notation
	Input		
$\tilde{\mathbf{x}}_i$	ith query instance	N	$\tilde{\mathbf{X}}^q = \{\tilde{\mathbf{x}}_i\}_{i=1}^N$
$\tilde{\mathbf{y}}_j$	jth database instance	M	$\tilde{\mathbf{Y}}^p = \{\tilde{\mathbf{y}}_j\}_{j=1}^M$
$\{\mathbf{x}_i^*, \mathbf{y}_j^*, s_{ij}\}$	a triple set of auxiliary pairs	$N_{xy} = N_1 \times N_2$	$\mathcal{A}_{xy} = \cup_{i=1}^{N_1} \cup_{j=1}^{N_2} \{\mathbf{x}_i^*, \mathbf{y}_j^*, s_{ij}\}$
$\mathbf{x}_i$	ith training instance in query domain	N_x	$\mathcal{T}_x = \{\mathbf{x}_i\}_{i=1}^{N_x}$
$\mathbf{y}_j$	jth training instance in database domain	N_y	$\mathcal{T}_y = \{\mathbf{y}_j\}_{j=1}^{N_y}$
	Output		
$f_k^q(\mathbf{x})$	kth hash function in query domain	k_q	$\mathcal{F}^q(\mathbf{x}) = \{f_k^q(\mathbf{x})\}_{k=1}^{k_q}$
$f_l^p(\mathbf{y})$	lth hash function in database domain	k_p	$\mathcal{F}^p(\mathbf{y}) = \{f_l^p(\mathbf{y})\}_{l=1}^{k_p}$
$\mathbf{C}^{k_q \times k_p}$	the $k_q \times k_p$ translator		
$\mathbf{h}_k^q$	kth hash code in query domain	k_q	$\mathbf{H}^q = \{\mathbf{h}_k^q\}_{k=1}^{k_q}$
$\mathbf{h}_l^p$	lth hash code in database domain	k_p	$\mathbf{H}^p = \{\mathbf{h}_l^p\}_{l=1}^{k_p}$

k_q dimensional and a k_p dimensional Hamming space respectively. The translator $\mathbf{C}^{k_q \times k_p}$ aligns the two Hamming spaces in a bitwise manner. Based on the hashing functions and the translator, we can generate hash codes $\mathbf{H}^q = \{\mathbf{h}_k^q\}_{k=1}^{k_q} \in \{-1,+1\}^{N \times k_q}$ in the query domain and $\mathbf{H}^p = \{\mathbf{h}_l^p\}_{l=1}^{k_p} \in \{-1,+1\}^{M \times k_p}$ in the database domain and perform accurate nearest neighbour retrieval across different media types. For brevity, we summarize these notations in Table 1.

3.3 Learning Hash Functions and Translators

In this section, we introduce the objective function of our proposed HTH integrating both the homogeneous similarity preservation term and the heterogeneous similarity preservation term.

3.3.1 Homogeneous media similarity preservation

A core criterion to preserve homogeneous media similarity is that similar data points in the original space should share similar hash codes within each single media type. To meet this criterion, in this work we first define the kth (lth) bit hash function of the query domain (the database domain) $f_k^q(\mathbf{x})$ ($f_l^p(\mathbf{y})$) as a linear projection which has been widely adopted in existing related works [22, 31, 32]:

$$f_k^q(\mathbf{x}) = sgn((\mathbf{w}_k^q)^T\mathbf{x}) \text{ and } f_l^p(\mathbf{y}) = sgn((\mathbf{w}_l^p)^T\mathbf{y}), \tag{1}$$

where $sgn(\cdot)$ is the sign function, and $\mathbf{w}_k^q$ and $\mathbf{w}_l^p$ denote projection vectors for the kth and lth bit hash codes in the query and database domain respectively.

In each domain, we can treat each hash function above as a binary classifier and each bit $h_k^{q(i)} \in \{-1,+1\}$ of the ith query data point as a binary class label. The goal is to learn binary classifiers $f_1^q, \cdots, f_{k_q}^q$ to predict k_q labels (bits) $h_1^q, \cdots, h_{k_q}^q$ for any query item $\mathbf{x}$. Moreover, we train binary classifiers for all the bits independently because different bits $h_1^q, \cdots, h_{k_q}^q$ should be uncorrelated. We propose to learn the hash function for the kth bit by solving the following optimization problem:

$$\mathcal{J}_{\mathbf{w}_k^q}^{ho} = \frac{1}{N_x}\sum_{i=1}^{N_x}\ell((\mathbf{w}_k^q)^T\mathbf{x}_i) + \gamma_q\Omega(\|\mathbf{w}_k^q\|_{\mathcal{H}}), \tag{2}$$

where $\ell(\cdot)$ denotes the loss function on one data point and Ω is a regularization term about functional norm $\|\mathbf{w}_k^q\|_{\mathcal{H}}$ in Hilbert spaces. Inspired by the large-margin criterion adopted by Support Vector Machine (SVM), we define ℓ using the hinge loss function $\ell((\mathbf{w}_k^q)^T\mathbf{x}_i) = [1 - h_k^{q(i)}(\mathbf{w}_k^q)^T\mathbf{x}_i]_+$, where $[a]_+$ returns a if $a >= 0$ and 0 otherwise. Ω is commonly defined as the L_2-norm $\frac{1}{2}\|\mathbf{w}_k^q\|^2$. Note that $h_k^{q(i)} = f_k^q(\mathbf{x}_i) = sgn((\mathbf{w}_k^q)^T\mathbf{x}_i)$, the optimization objective (2) can be rewritten as:

$$\mathcal{J}_{\mathbf{w}_k^q}^{ho} = \frac{1}{N_x}\sum_{i=1}^{N_x}[1 - |(\mathbf{w}_k^q)^T\mathbf{x}_i|]_+ + \frac{\gamma_q}{2}\|\mathbf{w}_k^q\|^2 + [\frac{1}{N_x}|\sum_{i=1}^{N_x}(\mathbf{w}_k^q)^T\mathbf{x}_i| - \delta]_+, \tag{3}$$

where γ_q is a balancing parameter controlling the impact of the regularization, and the last term is to avoid a trivially optimal solution which assigns all N_x data points to the same bit. Without the last constraint, the data points may be classified into the same side with large $|(\mathbf{w}_k^q)^T\mathbf{x}_i|$ value so that $[1 - |(\mathbf{w}_k^q)^T\mathbf{x}_i|]_+$ equals to 0 for all N_x data points which is meaningless in hashing. Thus we enforce $-\delta \le \frac{1}{N_x}|\sum_{i=1}^{N_x}(\mathbf{w}_k^q)^T\mathbf{x}_i| \le \delta$ with a pre-defined constant δ.

Similarly, we can learn the hash functions for the database domain by minimizing the following objective function:

$$\mathcal{J}_{\mathbf{w}_l^p}^{ho} = \frac{1}{N_y}\sum_{j=1}^{N_y}[1 - |(\mathbf{w}_l^p)^T\mathbf{y}_j|]_+ + \frac{\gamma_p}{2}\|\mathbf{w}_l^p\|^2 + [\frac{1}{N_y}|\sum_{j=1}^{N_y}(\mathbf{w}_l^p)^T\mathbf{y}_j| - \delta]_+, \tag{4}$$

where γ_p controls the impact of regularization. To learn hash functions from both query and database domains that preserve the homogeneous similarity, we combine (3) and (4) and derive the following objective function:

$$\begin{aligned}
&\mathcal{J}^{ho}(\mathbf{W}^q, \mathbf{W}^p) \\
&= \sum_{k=1}^{k_q}\left\{\frac{1}{N_x}\sum_{i=1}^{N_x}[1 - |(\mathbf{w}_k^q)^T\mathbf{x}_i|]_+ + [\frac{1}{N_x}|\sum_{i=1}^{N_x}(\mathbf{w}_k^q)^T\mathbf{x}_i| - \delta]_+\right\} \\
&+ \sum_{l=1}^{k_p}\left\{\frac{1}{N_y}\sum_{j=1}^{N_y}[1 - |(\mathbf{w}_l^p)^T\mathbf{y}_j|]_+ + [\frac{1}{N_y}|\sum_{j=1}^{N_y}(\mathbf{w}_l^p)^T\mathbf{y}_j| - \delta]_+\right\} \\
&+ \frac{\gamma_q}{2}\|\mathbf{W}^q\|_F^2 + \frac{\gamma_p}{2}\|\mathbf{W}^p\|_F^2,
\end{aligned} \tag{5}$$

where $\mathbf{W}^q = \{\mathbf{w}_1^q, \cdots, \mathbf{w}_{k_q}^q\}$, $\mathbf{W}^p = \{\mathbf{w}_1^p, \cdots, \mathbf{w}_{k_p}^p\}$ and $\|\cdot\|_F^2$ denotes the Frobenius norm.

3.3.2 Heterogeneous media similarity preservation

In the last subsection, we learn the hash codes that can preserve homogeneous media similarity for each media type. For the sake of flexibility and discrimination between two modalities, we adopt hash codes with different numbers of bits for different domains. To perform similarity search across different Hamming spaces, in this subsection, we introduce a translator $\mathbf{C}^{k_q \times k_p}$ to map the hash codes from a k_q-dimensional Hamming space to a k_p-dimensional Hamming space or vice versa. We also show that $\mathbf{C}$ can be learned from auxiliary heterogeneous pairs $\mathcal{A}_{xy} = \cup_{i=1}^{N_1}\cup_{j=1}^{N_2}\{\mathbf{x}_i^*, \mathbf{y}_j^*, s_{ij}\}$.

A good translator should have the following three properties: 1) semantically related points across different domains should have similar hash codes after translation; 2) semantically uncorrelated points across different domains should be far away from each other in the translated Hamming space; 3) it should have good generalization power. To obtain such a good translator, we propose to minimize the following heterogeneous loss function:

$$\mathcal{J}^{he} = \sum_{i,j}^{N_{xy}}[s_{ij}d_{ij}^2 + (1 - s_{ij})\tau(d_{ij})] + \frac{\gamma_C}{2}\|\mathbf{C}\|_F^2, \tag{6}$$

where $d_{ij} = \sum_{l=1}^{k_p}[\sum_{k=1}^{k_q}C_{kl}(\mathbf{w}_k^q)^T\mathbf{x}_i^* - (\mathbf{w}_l^p)^T\mathbf{y}_j^*]^2$ represents the distance in the Hamming space of the database between the ith translated hash code from the query domain and the jth code string from the database domain. $\tau(\cdot)$ is the SCISD [16] function specified by two parameters a and λ:

$$\tau(d_{ij}) = \begin{cases} -\frac{1}{2}d_{ij}^2 + \frac{a\lambda^2}{2} & \text{if} \quad 0 \le |d_{ij}| \le \lambda \\ \frac{d_{ij}^2 - 2a\lambda|d_{ij}| + a^2\lambda^2}{2} & \text{if} \quad \lambda < |d_{ij}| \le a\lambda \\ 0 & \text{if} \quad |d_{ij}| > a\lambda \end{cases}. \tag{7}$$

Note that if two data points are semantically similar, that is, $s_{ij} = 1$, we require that they have small d_{ij}; if they are semantically dissimilar, we require that they have small SCISD value which implies that they are far apart in the Hamming space.

3.3.3 Overall optimization problem

Combining the objective functions introduced in the previous two subsections, the overall optimization problem of HTH can be written as follows:

$$\min_{\mathbf{W}^q, \mathbf{W}^p, \mathbf{C}} \quad \mathcal{J}^{ho} + \beta\mathcal{J}^{he} \tag{8}$$

where β is a trade-off parameter between homogeneous and heterogeneous loss functions.

3.4 Optimization

Problem (8) is non-trivial to solve because it is discrete and non-convex *w.r.t.* $\mathbf{W}^q$, $\mathbf{W}^p$ and $\mathbf{C}$. In the following, we develop an alternating algorithm to solve this problem which converges to a local minimum very quickly.

We first describe how to learn the projection vector $\mathbf{w}_k^q$ for the kth bit while fixing other variables. Note that projection vectors for different bits can be learned independently using the same algorithm. The objective function *w.r.t.* $\mathbf{w}_k^q$ is:

$$\mathcal{J}_{\mathbf{w}_k^q} = \frac{1}{N_x}\sum_{i=1}^{N_x}[1-|(\mathbf{w}_k^q)^T\mathbf{x}_i|]_+ + [\frac{1}{N_x}|\sum_{i=1}^{N_x}(\mathbf{w}_k^q)^T\mathbf{x}_i| - \delta]_+ + \beta\sum_{i,j}^{N_{xy}}[s_{ij}d_{ij}^2 + (1-s_{ij})\tau(d_{ij})] + \frac{\gamma_q}{2}\|\mathbf{w}_k^q\|^2. \quad (9)$$

Although (9) is not convex, it can be expressed as the differences of two convex functions, and hence can be minimized efficiently using *constrained concave-convex-procedure* (CCCP) [29].

We briefly introduce the idea of CCCP here. Given an optimization problem in the form of $\min_x f(x) - g(x)$ where f and g are real-valued convex functions, the key idea of CCCP is to iteratively evaluate an upper bound of the objective function by replacing g with its first-order Taylor expansion around the current solution, x_t, i.e., $\mathcal{R}(g(x_t)) = g(x_t) + \partial_x g(x_t)(x - x_t)$. Then the relaxed sub-problem $f(x) - \mathcal{R}(g(x_t))$ is in convex form and can be solved by off-the-shelf convex solvers. The solution sequence $\{x_t\}$ obtained by CCCP is guaranteed to reach a local optimum.

Specifically, the upper bound of (9) in the tth CCCP iteration is:

$$\mathcal{J}_{\mathbf{w}_k^q}^{(t)} = \quad (10)$$
$$\frac{1}{N_x}\sum_{i=1}^{N_x}[f_1(\mathbf{w}_k^q) - \mathcal{R}(g_1(\mathbf{w}_k^{q(t)}))] + [\frac{1}{N_x}|\sum_{i=1}^{N_x}(\mathbf{w}_k^q)^T\mathbf{x}_i| - \delta]_+ + \beta\sum_{i,j}^{N_{xy}} s_{ij}d_{ij}^2 + \beta\sum_{i,j}^{N_{xy}}(1-s_{ij})[\tau_1(d_{ij}) - \mathcal{R}(\tau_2(d_{ij}^{(t)}))] + \frac{\gamma_q}{2}\|\mathbf{w}_k^q\|^2,$$

where $f_1(\mathbf{w}_k^q) = 1 + [|(\mathbf{w}_k^q)^T\mathbf{x}_i| - 1]_+$, $g_1(\mathbf{w}_k^{q(t)}) = |(\mathbf{w}_k^{q(t)})^T\mathbf{x}_i|$, $\tau_2(d_{ij}) = \frac{1}{2}d_{ij}^2 - \frac{a\lambda^2}{2}$ and

$$\tau_1(d_{ij}) = \begin{cases} 0 & \text{if} \quad 0 \le |d_{ij}| \le \lambda \\ \frac{ad_{ij}^2 - 2a\lambda|d_{ij}| + a\lambda^2}{2(a-1)} & \text{if} \quad \lambda < |d_{ij}| \le a\lambda \\ \frac{1}{2}d_{ij}^2 - \frac{a\lambda^2}{2} & \text{if} \quad |d_{ij}| > a\lambda \end{cases}. \quad (11)$$

The Taylor expansion of $g_1(\cdot)$ and $\tau_2(\cdot)$ around the value of $\mathbf{w}_k^q$ in the tth iteration are $\mathcal{R}(g_1(\mathbf{w}_k^{q(t)})) = |(\mathbf{w}_k^{q(t)})^T\mathbf{x}_i| + sgn((\mathbf{w}_k^{q(t)})^T\mathbf{x}_i)\cdot\mathbf{x}_i(\mathbf{w}_k^q - \mathbf{w}_k^{q(t)})$ and $\mathcal{R}(\tau_2(d_{ij}^{(t)})) = \frac{1}{2}d_{ij}^{(t)2} - \frac{a\lambda^2}{2} + d_{ij}^{(t)}\frac{\partial d_{ij}^{(t)}}{\mathbf{w}_k^q}(\mathbf{w}_k^q - \mathbf{w}_k^{q(t)})$ respectively. Note that

$$d_{ij}^{(t)} = \sum_{l=1}^{k_p}[\sum_{k=1}^{k_q} C_{kl}(\mathbf{w}_k^{q(t)})^T\mathbf{x}_i^* - (\mathbf{w}_l^p)^T\mathbf{y}_j^*]^2. \quad (12)$$

However, minimizing (10) is time-consuming if the data dimensionality is high. As a result, we employ Pegasos [21] which is a sub-gradient based solver and reported to be one of the fastest gradient-based solvers, to solve the problem. In each Pegasos iteration, the key step is to evaluate the sub-gradient of $\mathcal{J}_{\mathbf{w}_k^q}^{(t)}$ *w.r.t.* $\mathbf{w}_k^q$ from l_1 random homogeneous data points and l_2 random heterogeneous pairs:

Algorithm 1 Heterogeneous Translated Hashing (HTH)

Input:
$\mathcal{T}_x = \{\mathbf{x}_i\}_{i=1}^{N_x}$ – query training set
$\mathcal{T}_y = \{\mathbf{y}_j\}_{j=1}^{N_y}$ – database training set
$\mathcal{A}_{xy} = \cup_{i=1}^{N_1}\cup_{j=1}^{N_2}\{\mathbf{x}_i^*, \mathbf{y}_j^*, s_{ij}\}$ – auxiliary heterogeneous data
$\beta, \gamma_q, \gamma_p, \gamma_C$ – regularization parameters
δ – partition balance parameter
a, λ – SCISD function parameter
k_q; k_p – length of hash codes
Output:
$\mathbf{W}^q$, $\mathbf{W}^p$, $\mathbf{C}$

```
Initialize W^q, W^p with CVH and C = I;
while W^q, W^p and C are not converged do
    Fix W^p and C, optimize W^q:
    for k = 1 ··· k_q do
        for t = 1 ··· t_max do
            w_k^{q(t+1)} = arg min J^{(t)}_{w_k^q};
        end for
    end for
    Fix W^q and C, optimize W^p
    for k = 1 ··· k_p do
        for t = 1 ··· t_max do
            w_l^{p(t+1)} = arg min J^{(t)}_{w_l^p};
        end for
    end for
    Fix W^q and W^p, optimize C:
    for t = 1 ··· t_max do
        solve C^{(t+1)} = arg min J_C^{(t)};
    end for
end while
```

$$\frac{\partial\mathcal{J}_{\mathbf{w}_k^q}^{(t)}}{\mathbf{w}_k^q} = \frac{1}{N_x}\sum_{i=1}^{N_x}[\frac{\partial(f_1(\mathbf{w}_k^q))}{\mathbf{w}_k^q} - sgn((\mathbf{w}_k^{q(t)})^T\mathbf{x}_i)\mathbf{x}_i] + \eta_{\mathbf{w}_k^q} + \gamma_q\mathbf{w}_k^q + 2\beta\sum_{i,j}^{N_{xy}} s_{ij}d_{ij}\frac{\partial d_{ij}}{\mathbf{w}_k^q} + \beta\sum_{i,j}^{N_{xy}}(1-s_{ij})(\frac{\partial\tau_1(d_{ij})}{\mathbf{w}_k^q} - d_{ij}^{(t)}\frac{\partial d_{ij}^{(t)}}{\mathbf{w}_k^q}), \quad (13)$$

where

$$\frac{\partial(f_1(\mathbf{w}_k^q))}{\mathbf{w}_k^q} = \begin{cases} 0 & \text{if} \quad |(\mathbf{w}_k^q)^T\mathbf{x}_i| \le 1 \\ sgn((\mathbf{w}_k^q)^T\mathbf{x}_i)\mathbf{x}_i & \text{otherwise,} \end{cases} \quad (14)$$

$$\eta_{\mathbf{w}_k^q} = \begin{cases} 0 & \text{if} \quad \frac{1}{N_x}|(\sum_{i=1}^{N_x}\mathbf{w}_k^q)^T\mathbf{x}_i| \le \delta \\ sgn(\frac{1}{N_x}\sum_{i=1}^{N_x}(\mathbf{w}_k^q)^T\mathbf{x}_i - \delta)\cdot\frac{1}{N_x}\sum_{i=1}^{N_x}\mathbf{x}_i & \text{otherwise,} \end{cases} \quad (15)$$

$$\frac{\partial d_{ij}}{\mathbf{w}_k^q} = \sum_{l=1}^{k_p}\{2\cdot[\sum_{k=1}^{k_q} C_{kl}(\mathbf{w}_k^q)^T\mathbf{x}_i^* - (\mathbf{w}_l^p)^T\mathbf{y}_j^*]\cdot C_{kl}\mathbf{x}_i^*\}, \quad (16)$$

$$\frac{\partial d_{ij}^{(t)}}{\mathbf{w}_k^q} = \sum_{l=1}^{k_p}\{2\cdot[\sum_{k=1}^{k_q} C_{kl}(\mathbf{w}_k^{q(t)})^T\mathbf{x}_i^* - (\mathbf{w}_l^p)^T\mathbf{y}_j^*]\cdot C_{kl}\mathbf{x}_i^*\}, \quad (17)$$

$$\frac{\partial\tau_1(d_{ij})}{\mathbf{w}_k^q} = \frac{\partial d_{ij}}{\mathbf{w}_k^q}\cdot\begin{cases} 0 & \text{if} \quad 0 \le |d_{ij}| \le \lambda \\ \frac{ad_{ij} - a\lambda sgn(d_{ij})}{a-1} & \text{if} \quad \lambda < |d_{ij}| \le a\lambda \\ d_{ij} & \text{if} \quad |d_{ij}| > a\lambda. \end{cases} \quad (18)$$

Similarly, the objective function and sub-gradient *w.r.t.* $\mathbf{w}_l^p$ at the tth CCCP iteration are :

$$\mathcal{J}_{\mathbf{w}_l^p}^{(t)} = \frac{1}{N_y}\sum_{j=1}^{N_y}[f_1(\mathbf{w}_l^p) - \mathcal{R}(g_1(\mathbf{w}_l^{p(t)}))] + [\frac{1}{N_y}|\sum_{j=1}^{N_y}(\mathbf{w}_l^p)^T\mathbf{y}_j| - \delta]_+ + \beta\sum_{i,j}^{N_{xy}} s_{ij}d_{ij}^2 + \beta\sum_{i,j}^{N_{xy}}(1-s_{ij})[\tau_1(d_{ij}) - \mathcal{R}(\tau_2(d_{ij}^{(t)}))] + \frac{\gamma_p}{2}\|\mathbf{w}_l^p\|^2, \tag{19}$$

and

$$\frac{\partial\mathcal{J}_{\mathbf{w}_l^p}^{(t)}}{\mathbf{w}_l^p} = \frac{1}{N_y}\sum_{j=1}^{N_y}[\frac{\partial(f_1(\mathbf{w}_l^p))}{\mathbf{w}_l^p} - sgn((\mathbf{w}_l^{p(t)})^T\mathbf{y}_j)\mathbf{y}_j]] + \eta_{\mathbf{w}_l^p} + \gamma_p\mathbf{w}_l^p + 2\beta\sum_{i,j}^{N_{xy}} s_{ij}d_{ij}\frac{\partial d_{ij}}{\mathbf{w}_l^p} + \beta\sum_{i,j}^{N_{xy}}(1-s_{ij})(\frac{\partial\tau_1(d_{ij})}{\mathbf{w}_l^p} - d_{ij}^{(t)}\frac{\partial d_{ij}^{(t)}}{\mathbf{w}_l^p}), \tag{20}$$

where

$$\frac{\partial d_{ij}}{\mathbf{w}_l^p} = 2\cdot[\sum_{k=1}^{k_q} C_{kl}(\mathbf{w}_k^q)^T\mathbf{x}_i^* - (\mathbf{w}_l^p)^T\mathbf{y}_j^*]\cdot\mathbf{y}_j^*. \tag{21}$$

Note that both $\frac{\partial d_{ij}^{(t)}}{\mathbf{w}_l^p}$ and $\frac{\partial\tau_1(d_{ij})}{\mathbf{w}_l^p}$ can easily be derived and we omit them here due to space limitations.

To update the translator $\mathbf{C}$, we also use CCCP. The objective function and sub-gradient *w.r.t.* every element C_{kl} in the tth CCCP iteration:

$$\mathcal{J}_{C_{kl}}^{(t)} = \beta\sum_{i,j}^{N_{xy}} s_{ij}d_{ij}^2 + \beta\sum_{i,j}^{N_{xy}}(1-s_{ij})[\tau_1(d_{ij}) - \mathcal{R}(\tau_2(d_{ij}^{(t)}))] + \frac{\gamma_C}{2}\|\mathbf{C}\|_F^2,$$

and

$$\frac{\partial\mathcal{J}_{C_{kl}}^{(t)}}{C_{kl}} = \beta\sum_{i,j}^{N_{xy}}(1-s_{ij})(\frac{\partial\tau_1(d_{ij})}{C_{kl}} - d_{ij}^{(t)}\frac{\partial d_{ij}^{(t)}}{C_{kl}}) + \gamma_C C_{kl} + 2\beta\sum_{i,j}^{N_{xy}} s_{ij}d_{ij}\frac{\partial d_{ij}}{C_{kl}},$$

where

$$\frac{\partial d_{ij}}{C_{kl}} = 2\cdot[\sum_{k=1}^{k_q} C_{kl}(\mathbf{w}_k^q)^T\mathbf{x}_i^* - (\mathbf{w}_l^p)^T\mathbf{y}_j^*]\cdot(\mathbf{w}_k^q)^T\mathbf{x}_i^*.$$

The overall procedure of the HTH method, alternating learning $\mathbf{W}^q$, $\mathbf{W}^p$ and the translator $\mathbf{C}$ with CCCP and Pegasos, is presented in Algorithm 1.

3.5 Complexity Analysis

The computational cost of the proposed algorithm comprises three parts: updating $\mathbf{W}^q$, $\mathbf{W}^p$ and $\mathbf{C}$. Hence, the total time complexity of training HTH is $O(k_q d_q(l_1+l_3) + k_p d_p(l_2+l_3) + l_3 k_p k_q)$ where l_1 and l_2 are the numbers of stochastically selected training data points in the query domain and database domain by the Pegasos solver. l_3 is the number of randomly sampled auxiliary data pairs from N_{xy} auxiliary heterogeneous co-occurrence data pairs. Clearly, the time complexity for our algorithm scales linearly with the number of training data points and quadratic with the length of hash codes. In practice, the code length is short, otherwise, the technique of "hashing" will be meaningless. Hence our algorithm is very computationally efficient.

During the online query phase, given a query instance $\tilde{\mathbf{x}}_i \in \tilde{\mathbf{X}}^q$, we apply our learned hash functions for the query domain to it by performing two dot-product operations, $\mathbf{H}_i^q = \mathbf{x}_i\cdot\mathbf{W}^q$ and translation $\mathbf{H}_i^q\cdot\mathbf{C}$, which are quite efficient. The translated query hash codes are then compared with the hash codes of the database by quick XOR and bit count operations. These operations enjoy the sub-linear time-complexity *w.r.t.* the database size.

4. EXPERIMENTS

In this section, we evaluate the performance of HTH on two real world datasets and compare it with the state-of-the-art multi-modal hashing algorithms.

4.1 Experimental Settings

4.1.1 Datasets

In this work, we use two real world datasets, NUS-WIDE[1] and MIRFLICKR-Yahoo Answers.

NUS-WIDE is a Flickr dataset containing 269,648 tagged images [5]. The annotation for 81 semantic concepts is provided for evaluation. We prune this dataset via keeping the image-tag pairs that belong to the ten largest concepts. For image features, 500 dimensional SIFT vectors are used. On the other side, a group of tags for an image composes a single text document. For each text document, we use the probability distribution of 100 Latent Dirichlet Allocation (LDA) [3] topics as the feature vector. Therefore, NUS-WIDE is a multi-view dataset. Each data instance has an image view and a text view. When searching images using text query or searching text documents using image query, the groundtruth is derived by checking whether an image and a text document share at least one of the ten selected largest concepts.

MIRFLICKR-Yahoo Answers is a heterogeneous media dataset consisting of images from MIRFLICKR-25000 [9] and QAs from Yahoo Answers. MIRFLICKR-25000 is another Flickr collection consisting of 25,000 images. We utilize 5,018 tags provided by NUS-WIDE to filter irrelevant pictures in MIRFLICKR-25000 by cross-checking tags of each image with these 5,018 tags. The 500 dimensional sift feature vector is also applied. Yahoo Answers are crawled from a public API of Yahoo Query Language (YQL)[2]. The 5,018 tags are taken as keywords to search relevant QAs on Yahoo Answers. For each keyword, we extract top 100 results returned by YQL. Finally, we obtain a pool of about 300,000 QAs, each of which is regarded as a text document in the experiment. Each QA is represented in a 100 dimensional LDA based feature vector. For the task using image query to retrieve QAs in the database, those QAs which share at least two words with tags corresponding to the image query (images in MIRFLICKR-25000 are also tagged) are labelled as the groundtruth. The groundtruth for the task using QA as query to retrieve the image database is obtained similarly. More importantly, we randomly select a number of multi-view instances, e.g., 2,000, in the NUS-WIDE dataset as the "bridge". As a result, we obtain $2,000^2 = 4\times10^6$ auxiliary heterogeneous pairs.

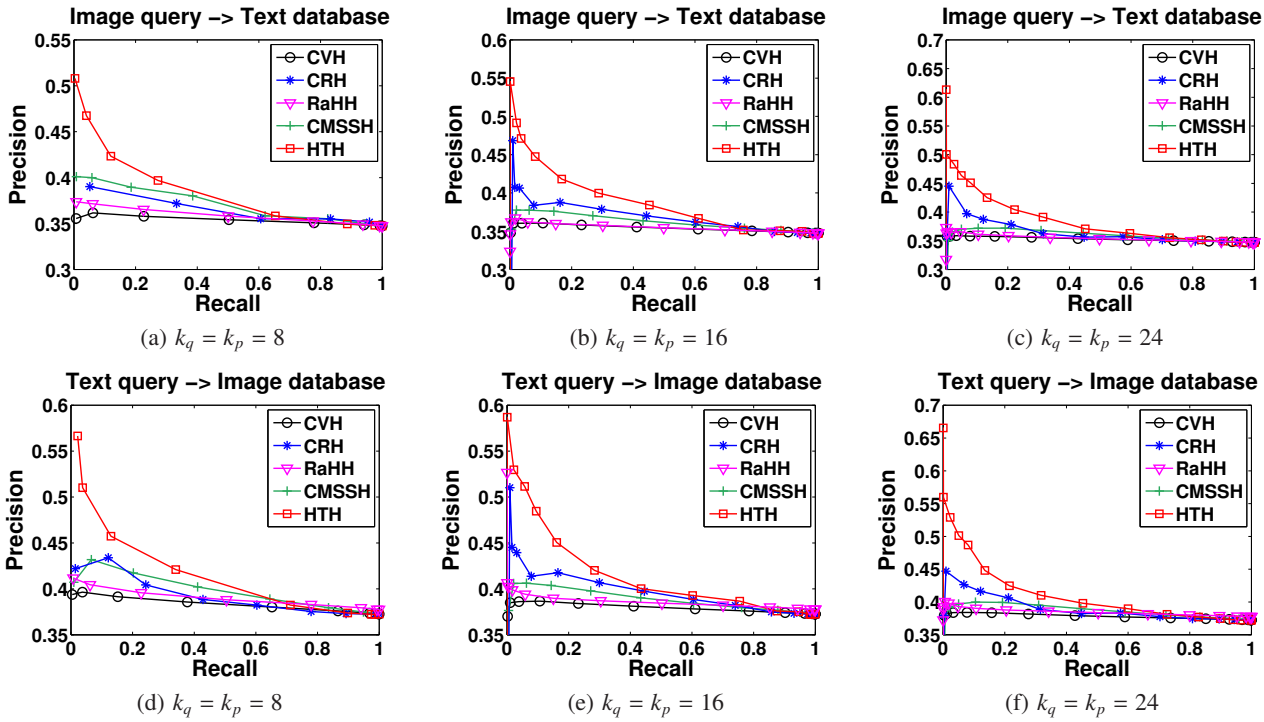
4.1.2 Baselines

We compare our method with the following four baseline algorithms.

Cross-modality similarity-sensitive hashing (CMSSH) [4], to the best of our knowledge, is the first approach that tackles hashing across multimodal data. CMSSH uses Adaboost to construct a group of hash functions sequentially for each modality while only preserving inter-modality similarity.

Cross-view hashing (CVH) [12] extends spectral hashing to the multi-view case via a CCA (canonical correlation analysis) alike

[1]http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm
[2]http://developer.yahoo.com/yql/



(a) $k_q = k_p = 8$ (b) $k_q = k_p = 16$ (c) $k_q = k_p = 24$

(d) $k_q = k_p = 8$ (e) $k_q = k_p = 16$ (f) $k_q = k_p = 24$

Figure 3: Precision-recall curves on NUS-WIDE dataset.

procedure. In our implementation, CVH learns two hash functions which can directly be applied to out-of-sample data.

Co-regularized hashing (CRH) [32] proposes a boosted co-regularization framework to learn two sets of hash functions for both the query and database domain.

Relation-aware heterogeneous hashing (RaHH) [15] adopts uneven bits for different modalities and a mapping function between them. During testing we add no heterogeneous relationship between queries and the database in our setting. In [15], however, they used the explicit relationship and attained higher accuracies.

4.1.3 Evaluation metric

In this paper, Mean Average Precision (MAP), precision and recall are adopted as our evaluation metrics of effectiveness.

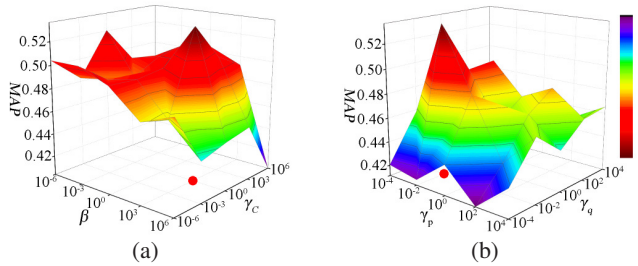
MAP stands out among performance measures in virtue of its competitive stability and discrimination. To compute MAP, Average Precision (AP) of top R retrieved documents for a single query is first calculated. $AP = \frac{1}{L}\sum_{r=1}^{R} P(r)\delta(r)$ where L is the number of groundtruth in the R retrieved set, $P(r)$ indicates the precision of top-r retrieved documents and $\delta(r) = 1$ denotes whether the rth retrieved document is a true neighbour otherwise $\delta(r) = 0$. MAP is then averaged over all queries' APs. The larger the MAP score, the better the retrieval performance. In our experiments, we set R= 50. The precision and recall scores reported in this paper are averaged over all queries. The larger the area under the curves, the better the achieved performance.

4.2 Results on NUS-WIDE Dataset

We perform two kinds of tasks on the NUS-WIDE dataset: 1) retrieving text documents by using images as queries; 2) retrieving images by using text documents as queries. In either task, we randomly select $300^2 = 90,000$ image-tag pairs from the NUS-WIDE dataset to be our training pairs. For the task of retrieving texts by image queries (retrieving images by text queries), we select 2,000 images (text documents) as queries and 10,000 text documents (images) to be the database. We perform our experiment on four such randomly sampled datasets and the average MAP results for all the compared algorithms are reported in Table 2. To be comparable with CVH, CMSSH and CRH, HTH adopts the same code length for different domains. From Table 2, we have the following observation. HTH outperforms all state-of-the-art methods in almost all settings at most 30%.

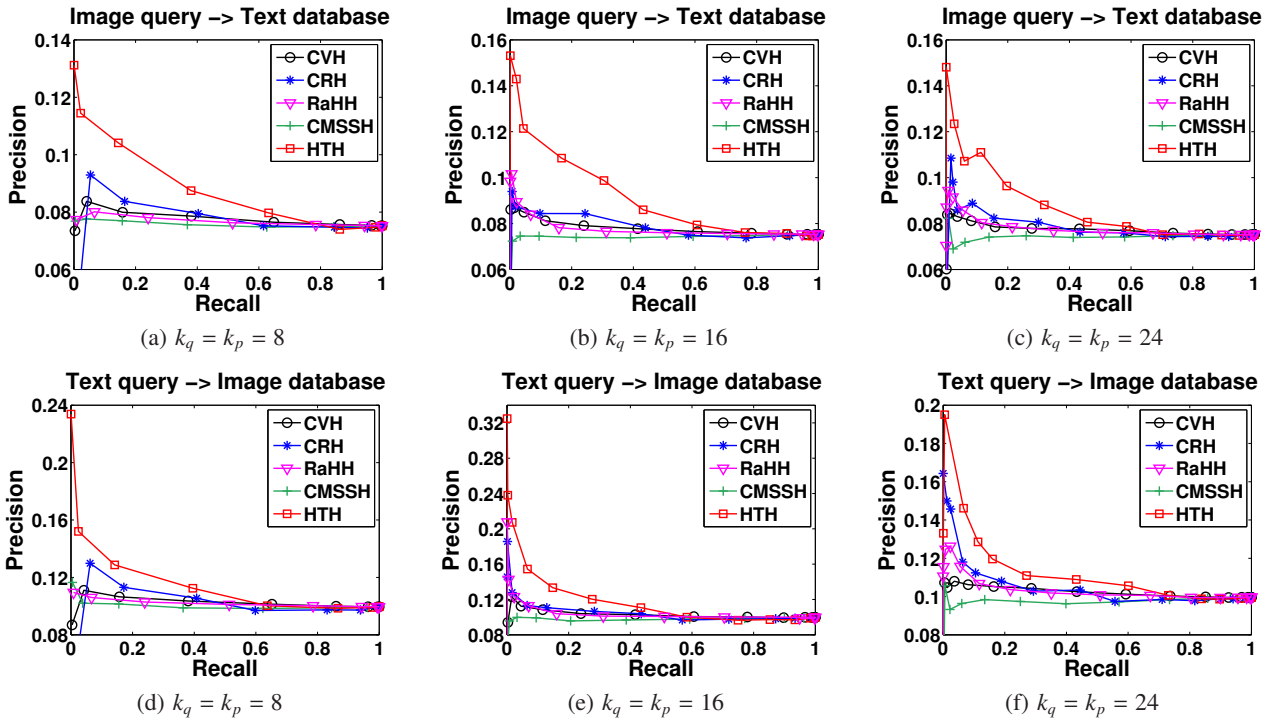
Table 2: MAP comparison on NUS-WIDE.

Task	Algorithm	Code Length ($k_q = k_p$)			
		8	16	24	32
Image→Text	CVH	0.4210	0.4085	0.4066	0.4112
	CMSSH	0.4447	0.4209	0.4109	0.4123
	CRH	0.4645	0.5003	0.5255	0.3207
	RaHH	0.4120	0.4122	0.4098	0.4182
	HTH	**0.5013**	**0.5357**	**0.5362**	**0.5151**
Text→Image	CVH	0.4483	0.4323	0.4296	0.4361
	CMSSH	0.4779	0.4485	0.4378	0.4373
	CRH	0.4986	0.5327	**0.5774**	0.3378
	RaHH	0.4595	0.4396	0.4351	0.4315
	HTH	**0.5398**	**0.5688**	0.5508	**0.5525**



(a) (b)

Figure 4: Study of parameter sensitivity on NUS-WIDE dataset. Parameter settings in our experiments are labelled in red dots and correspond to the best performances.

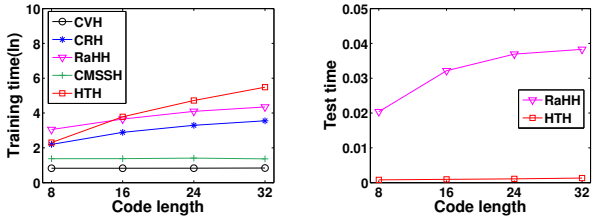


(a) $k_q = k_p = 8$ (b) $k_q = k_p = 16$ (c) $k_q = k_p = 24$

(d) $k_q = k_p = 8$ (e) $k_q = k_p = 16$ (f) $k_q = k_p = 24$

Figure 5: Precision-recall curves on MIRFLICKR-Yahoo Answer dataset.

The precision-recall curves for 8, 16 and 24 bits are plotted in Figure 3. The superior performance of HTH in precision-recall curves agrees with the results of MAP in Table 2.



(a) Training time (b) Test time

Figure 6: Time cost of training and testing on NUS-WIDE dataset with different code lengths. The time is measured in seconds. Y-axis in (a) is the natural logarithm of training time.

We also study the effect of different parameter settings on the performance of HTH. We fix the code lengths of both modalities to be 16. There are four trade-off parameters, β, γ_q, γ_p and γ_C as shown in objective function (8). We perform grid search on β and γ_C in the range of $\{10^{-6},10^{-3},10^{0},10^{3},10^{6}\}$ by fixing γ_q and γ_p. HTH gains the best MAP at $\beta = 1,000$, $\gamma_C = 1$ as Figure 4(a) shows. When fixing the β and γ_C, grid search of γ_q and γ_p in the range of $\{10^{-4},10^{-2},10^{0},10^{2},10^{4}\}$ shows that $\gamma_q = \gamma_p = 0.01$ performs the best. We adopt $\gamma_C = 1, \beta = 1,000, \gamma_q = 0.01, \gamma_p = 0.01$ in our experiments.

The time costs of HTH and other baselines are shown in Figure 6 as the code length changes. Since the training complexity of HTH is quadratic with respect to the code length, it takes more training time when the codes are longer. However, hashing with less bits is expected, thereby HTH is practical. In online querying phase, since CVH, CMSSH and CRH have the same time complexity as HTH, we only compare HTH with RaHH. The average query search time of HTH is much less than RaHH because RaHH does not learn explicit hash functions and has to adopt the fold-in scheme for out-of-sample data.

4.3 Results on MIRFLICKR-Yahoo Answers Dataset

We also report the results of the two tasks (using images to search text documents and using text documents to search images) on the MIRFLICKR-Yahoo Answers dataset which contains a larger number of images and text documents.

In this experiment, $N_{xy} = 2,000^2 = 4\times10^6$ image-tags pairs from NUS-WIDE dataset, 500 randomly selected images from MIRFLIC -KR as well as 500 sampled QAs from Yahoo Answers are chosen for training. In this case, these image-tag pairs from NUS-WIDE serve as the auxiliary bridge while queries and the database have no direct correspondence since MIRFLICKR images and Yahoo Answers are obtained independently.

Table 3: MAP comparison on MIRFLICKR-Yahoo Answers.

Task	Algorithm	Code Length($k_q = k_p$)			
		8	16	24	32
Image→QA	CVH	0.1331	0.1443	0.1460	0.1506
	CMSSH	0.1373	0.1268	0.1210	0.1295
	CRH	0.0979	0.1419	0.1317	0.1216
	RaHH	0.1346	0.1437	0.1474	0.1275
	HTH	**0.1577**	**0.1738**	**0.1824**	**0.1617**
QA→Image	CVH	0.1515	0.1758	0.1694	0.1721
	CMSSH	0.1735	0.1483	0.1518	0.1544
	CRH	0.1048	0.2234	0.1793	0.1862
	RaHH	0.1669	0.1731	0.1686	0.1452
	HTH	**0.1785**	**0.2460**	**0.2055**	**0.2324**

To apply CVH, CMSSH, CRH to this dataset, we simply train hash functions for images and texts using the image-tag pairs from NUS-WIDE and generate hash codes of images from MIRFLICKR and QAs from Yahoo Answers by directly applying corresponding hash functions. For RaHH, in the training phase, the data is the

same as those used in CVH, CMSSH and CRH. In the testing phase, we do not add any relationships between queries and database entities when applying the fold-in algorithm to generate hash codes for out-of-sample data. For HTH, we train the hash functions with the auxiliary heterogeneous pairs and unlabelled homogeneous data. In the testing phase, it is easy to apply the learned hash functions to out-of-sample data without any correspondence information.

Table 4: MAP of RaHH and HTH on MIRFLICKR-Yahoo Answer with different combinational code length.

k_q \ k_p	8		16		24		32	
	RaHH	HTH	RaHH	HTH	RaHH	HTH	RaHH	HTH
8	0.1346	0.1577	0.1315	0.1410	0.1442	0.1583	0.1366	0.1725
16	0.1346	0.1525	0.1437	0.1738	**0.1545**	**0.1894**	0.1274	0.1638
24	0.1346	0.1692	0.1437	0.1736	0.1474	0.1824	0.1378	0.1625
32	0.1346	0.1761	0.1437	0.1626	0.1474	0.1701	0.1275	0.1617

The MAP results are summarized in Table 3 with various code length settings. It shows that HTH outperforms all the other algorithms under all settings. This demonstrates that HTH shows more superiority in situations where queries and the database do not interrelate. Similarly, the precision-recall curves are plotted in Figure 5.

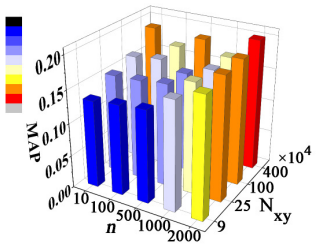
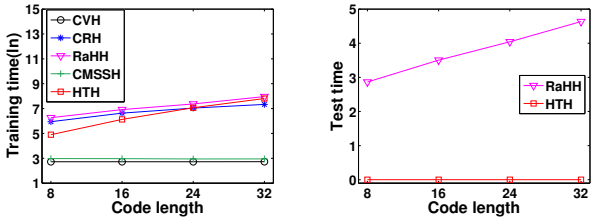


Figure 7: The influence of varying N_{xy}, the number of auxiliary heterogeneous pairs, and n, the number of added unlabelled images/QAs, on MAP performance.

More importantly, our proposed HTH method adopts uneven bits for different modalities so that it discriminates between the query and database domain flexibly. In Table 4, we compare HTH with RaHH, which also supports different code lengths. The row represents code length of images while the column is for that of QAs. HTH and RaHH both attain the best MAP results at $k_q = 16$ and $k_p = 24$. This code length combination is regarded as the best trade-off between effective translator learning and original information preservation. Moreover, images require less bits to achieve comparable performance compared to QAs because instances in text domain are more dispersed so that more bits are called for encoding all the instances.

The influence of N_{xy}, the number of auxiliary heterogeneous training pairs, and n, the number of added unlabelled images/QAs, on MAP performance is investigated in Figure 7. Reasonably, larger n and N_{xy} result in better MAP performance. In practice, we choose $N_{xy} = 4\times10^6$ and $n = 500$, which is competitive with $N_{xy} = 4\times10^6$ and $n = 2,000$, to be more efficient during training.

The time costs of HTH on MIRFLICKR-Yahoo Answers dataset is also reported in Figure 8. The results are similar to Figure 6 except that HTH is more efficient by contrast. This demonstrates that HTH is less sensitive to the size of the training data, which can be further proved in Figure 9. CVH and CMSSH rely on eigen-decomposition operations which are efficient especially when the dimensions of the dataset are comparatively low. However, they do not consider homogeneous similarity or the regularization of pa rameters, thus resulting in less accurate out-of-sample testing performance. Although HTH takes more training time than CRH and RaHH when the number of auxiliary heterogeneous pairs, N_{xy}, is



(a) Training time (b) Test time

Figure 8: Time cost of training and testing on MIRFLICKR-Yahoo Answers dataset with code lengths. The time is measured in seconds. Y-axis in (a) is the natural logarithm of training time.

small, it shows more efficiency compared with CRH and RaHH as N_{xy} increases. Therefore, HTH has good scalability and can be applied to large-scale datasets. Note that we do not report results of CMSSH when $N_{xy} = 2.5\times10^7, 10^8$ since the algorithm has "out-of-memory" at these scales.

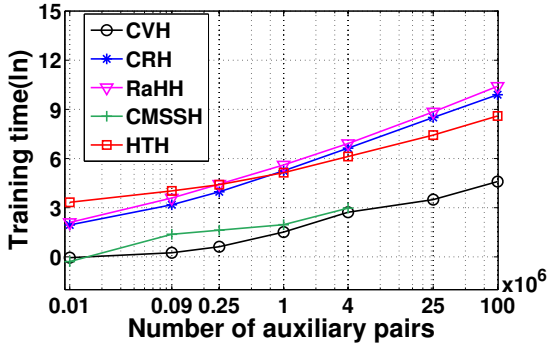


Figure 9: Scalability of training on MIRFLICKR-Yahoo Answers dataset as the number of auxiliary heterogeneous pairs, N_{xy}, increases. The time is measured in seconds. X-axis is in logarithmic scale. Y-axis is the natural logarithm of training time.

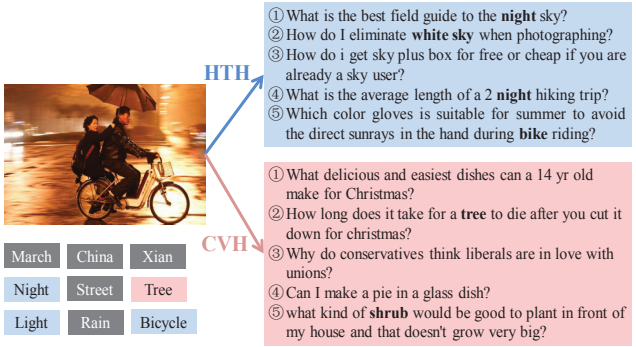


Figure 10: Given a picture in the MIRFLICKR dataset as query, we retrieve top-5 nearest questions from the Yahoo Answers dataset by HTH and CVH. Whether there exist corresponding keywords in a retrieved question to the labels of the picture indicates the relevance of this question to the picture.

We finally provide a similarity search example in Figure 10 to visually show the effectiveness of HTH. Given an image from MIRFLICKR, we compare the relevance of top-5 nearest questions to the image by HTH and CVH[3]. The retrieved questions via HTH are more relevant to this picture as shown in Figure 10.

[3]For space limitation, we only compare with CVH.

5. CONCLUSIONS

In this paper, we propose a novel heterogeneous translated hashing (HTH) model to perform similarity search across heterogeneous media. In particular, by leveraging auxiliary heterogeneous relationship on the web as well as massive unlabelled instances in each modality, HTH learns a set of hash functions to project instances of each modality to an individual Hamming space and a translator aligning these Hamming spaces. Extensive experimental results demonstrate the superiority of HTH over state-of-the-art multi-modal hashing methods. In the future, we plan to apply HTH to other modalities from social media and mobile computing, and to devise a more appropriate scheme to translate between different Hamming spaces, thereby further improving HTH.

6. ACKNOWLEDGEMENTS

We thank the reviewers for their valuable comments to improve this paper. The research has been supported in part by China National 973 project 2014CB340304 and Hong Kong RGC Projects 621013, 620812, and 621211.

7. REFERENCES

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

[2] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

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

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

[5] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y.-T. Zheng. Nus-wide: A real-world web image database from national university of singapore. In *VLDB*, pages 48:1–48:9, 2009.

[6] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *SCG*, pages 253–262, 2004.

[7] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *VLDB*, pages 518–529, 1999.

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

[9] M. J. Huiskes and M. S. Lew. The mir flickr retrieval evaluation. In *MIR*, 2008.

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

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

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

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

[14] Y. Mu, J. Shen, and S. Yan. Weakly-supervised hashing in kernel space. In *CVPR*, pages 3344–3351, 2010.

[15] M. Ou, P. Cui, F. Wang, J. Wang, W. Zhu, and S. Yang. Comparing apples to oranges: A scalable solution with heterogeneous hashing. In *KDD*, pages 230–238, 2013.

[16] N. Quadrianto and C. Lampert. Learning multi-view neighborhood preserving projections. In *ICML*, pages 425–432, 2011.

[17] M. Raginsky and S. Lazebnik. Locality-sensitive binary codes from shift-invariant kernels. In *NIPS*, pages 1509–1517. 2009.

[18] R. Salakhutdinov and G. Hinton. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969 – 978, 2009.

[19] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter-sensitive hashing. In *ICCV*, pages 750–757, 2003.

[20] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *ICML*, pages 807–814, 2007.

[21] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter. Pegasos: primal estimated sub-gradient solver for svm. *Mathematical Programming*, 127(1):3–30, 2011.

[22] J. Song, Y. Yang, Y. Yang, Z. Huang, and H. T. Shen. Inter-media hashing for large-scale retrieval from heterogeneous data sources. In *SIGMOD*, pages 785–796, 2013.

[23] C. Strecha, A. Bronstein, M. Bronstein, and P. Fua. Ldahash: Improved matching with smaller descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(1):66–78, 2012.

[24] J. K. Uhlmann. Satisfying general proximity/similarity queries with metric trees. *Information processing letters*, 40(4):175–179, 1991.

[25] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for scalable image retrieval. In *CVPR*, pages 3424–3431, 2010.

[26] R. Weber, H.-J. Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *VLDB*, pages 194–205, 1998.

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

[28] F. Wu, Z. Yu, Y. Yang, S. Tang, Y. Zhang, and Y. Zhuang. Sparse multi-modal hashing. *Multimedia, IEEE Transactions on*, 16(2):427–439, 2014.

[29] A. L. Yuille, A. Rangarajan, and A. Yuille. The concave-convex procedure (cccp). *Advances in neural information processing systems*, 2:1033–1040, 2002.

[30] D. Zhai, H. Chang, Y. Zhen, X. Liu, X. Chen, and W. Gao. Parametric local multimodal hashing for cross-view similarity search. In *IJCAI*, pages 2754–2760, 2013.

[31] D. Zhang, J. Wang, D. Cai, and J. Lu. Self-taught hashing for fast similarity search. In *SIGIR*, pages 18–25, 2010.

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

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

[34] X. Zhu, Z. Huang, H. T. Shen, and X. Zhao. Linear cross-modal hashing for efficient multimedia search. In *MM*, pages 143–152, 2013.