CROSS-MODAL HASHING THROUGH RANKING SUBSPACE LEARNING

Kai Li, Guojun Qi, Jun Ye, Kien A. Hua

Department of Computer Science, University of Central Florida, USA
kaili@eecs.ucf.edu, guojun.qi@ucf.edu, jye@eecs.ucf.edu, kienhua@eecs.ucf.edu

ABSTRACT

Hashing has been widely used for approximate nearest neighbor search of high-dimensional multimedia data. In this paper, we propose a novel hash learning framework that maps high-dimensional multimodal data into a common Hamming space where the cross-modal similarity can be measured using Hamming distance. Unlike existing cross-modal hashing methods that learn hash functions in the form of numeric quantization of linear projections, the proposed hash learning algorithm encodes features' ranking properties and takes advantage of rank correlations which are known to be scale-invariant, numerically stable and highly nonlinear. Specifically, we learn two groups of subspaces jointly, one for each modality, so that the ranking orders in those subspaces maximally preserve the cross-modal similarity. Extensive experiments on realworld datasets demonstrate superiority of the proposed methods compared to state-of-the-arts.

Index Terms— Multimedia retrieval, crossmodal hashing, ranking subspace learning, WTA hash

1 Introduction

The explosive growth of multimedia data needs efficient techniques to support content-based similarity search. Hashing has been widely adopted for its storage and computation efficiency [1, 2]. Most hashing techniques are designed for single-modal data thus not supporting cross-modal similarity search. Since multimedia data usually comes in different modalities, it is desirable to design hashing techniques to enable similarity search using data from another modality. This is crucial to many practical applications [1, 3].

Cross-modal hashing is a challenging problem because data from different modalities generally have distinct representations with incomparable space structures and dimensionalities. Existing algorithms [4, 5, 6] generally follow two steps: first, features from different modalities are mapped into a common feature space to minimize some cross-correlation error; second, hash codes are generated by quantizing the feature space obtained by a linear or nonlinear transformation of the original features.

The different hashing techniques usually differ in the first step where different cross-correlation errors are defined. As for the second step, they are very similar. In contrast to those feature quantization-based hashing algorithms, we propose to learn a new family of cross-modal hashing functions, based on features' relative ranking order.

Ranking-based *randomized* hashing functions have been explored in the single-modality setting. Representative works include Winner-Take-all (WTA) Hash [7] and Min-wise Hash (MinHash) [8]. WTA generates a compact representation of the input features by ranking the random permutations of input features and outputs the index of the maximum feature dimension as the hash code. MinHash is a special case of WTA for binary input features. Those ranking-based hashing algorithms are known to be invariant to feature scale and resilient to numeric noises, prevalent in practical applications [7, 8, 9], because relative ranking order of features is much less sensitive to noises. However, since existing ranking-based hashing algorithms are based on random permutations, they typically have to generate long codes through a large number of permutations to achieve desirable performance [7]. Moreover, those techniques are only applicable to single-modal data, and do not naturally fit the multimodal settings.

To address the above challenges, we propose to generate hash codes by ranking optimized linear subspaces instead of the random permutations of the features. Specifically, we learn two groups of linear subspaces jointly, one for each modality, such that the ranking order in one subspace is maximally aligned with that of the other subspace. Ranking hash codes learned in this way are much more compact and highly optimized for cross-modal similarity search while retaining the benefits of noise resilience and scale invariance inherent in rank correlation measures.

In the remainder of this paper, we first review the related work in Section 2. The proposed cross-modal ranking subspace learning problem is formulated in Section 3. In Section 4, we solve the optimization problem and present the learning algorithm. We discuss the experiment results in Section 5. Finally, we conclude the paper in Section 6.

2 Related Work

Cross-modal hashing aligns heterogeneous feature spaces by transforming multimodal data into a common Hamming space to enable cross-modal similarity search. Cross-Modal Similarity Sensitive Hashing (CMSSH) [4] and Cross-View

Hashing (CVH) [10] are early examples of this approach. CMSSH sequentially constructs two groups of linear hash functions and explicitly minimizes the distances between the data's Hamming spaces embeddings. CVH extends the unimodal spectral hashing (SH) to consider both intra-view and inter-view similarities through an eigen-system formulation.

Several new methods were proposed after CMSSH and CVH. Iterative Multi-View Hashing (IMVH) [11] learns discriminative hash functions by solving a series of binary label assignment problems. Co-Regularized Hashing (CRH) [12] learns single-bit cross-modal hash functions by solving DC (i.e. difference of convex function) programs; and multiple bits are sequentially learned using boosting. The same authors also propose Multimodal Latent Binary Embedding (MLBE) [13], which takes a probabilistic generative approach and generates competitive performance. The prohibitive computational costs for out-of-sample extensions, however, limit its scalability. Several other methods based on neural networks such as Cross-Media Neural Network Hashing (CMNNH) [5] and Multimodal Similarity-Preserving Hashing (CMSPH) [6] also suffer from the scalability issue.

In order to balance performance and computational complexity, (PLMH) [14] extends MLBE to learn parameterized hash functions as the linear combination of a small set of anchor points. Similar ideas have also been exploited in Linear Cross-Modal Hashing (LCMH) [15]. Semantic Correlation Maximization (SCM) [16] integrates semantic label information into a learning procedure with closed-form solutions and it avoids explicitly computing pairwise similarity matrix. Inter-Media Hashing (IMH) [1] incorporates both labeled and unlabeled data to explore correlations among multiple media types from large-scale data sources. More recently, Sparse Multi-Modal Hashing (SM^{2}H) [17] is proposed to obtain sparse codesets for data objects across different modalities through joint multi-modal dictionary learning. Latent Semantic Sparse Hashing (LSSH) [3] and Collective Matrix Factorization Hashing (CMFH) [2] use sparse coding and matrix factorization to capture the latent semantic features of different modalities.

3 Problem Formulation

3.1 Mathematical Notations

Suppose we have the data sets from two modalities $\mathcal{X}$ and $\mathcal{Y}$. Let $\mathcal{D}_\mathcal{X}$ be a set of $d_\mathcal{X}$-dimensional data points $\{\mathbf{x}_i\}_{i=1}^{N_\mathcal{X}}$ from modality $\mathcal{X}$ and $\mathcal{D}_\mathcal{Y}$ be a set of $d_\mathcal{Y}$-dimensional data points $\{\mathbf{y}_i\}_{i=1}^{N_\mathcal{Y}}$ from modality $\mathcal{Y}$. In addition, we have a set of inter-modality similarity labels $\mathcal{S} = \{s_{ij}|s_{ij} \in \{1, 0\}\}$ indicating whether the cross-modal pair $(\mathbf{x}_i, \mathbf{y}_j)$ describe the same concept or not. The similarity labels can be obtained either through semantic labels or by thresholding certain metric distances when semantic information is not available. Our objective is to learn two sets of hash functions $H_* = \{h_*^{(1)}(\cdot), h_*^{(2)}(\cdot), \cdots, h_*^{(L)}(\cdot)\}$ with $*$ being a place holder for $\mathcal{X}$ or $\mathcal{Y}$, so that data from both modalities can be projected into a common Hamming space.

3.2 Ranking-based Hash Function

We consider a family of hash functions based on the partial ordering of features projected into L linear subspaces. Specifically, we define $h_*^{(l)}(\cdot)$ as

$$h_*^{(l)}(\mathbf{z}_*; \mathbf{W}_*) = \arg \max_{1 \le k \le K} (\mathbf{w}_*^k)^T \mathbf{z}_*, \quad (1)$$

where $\mathbf{z}_* \in \mathcal{D}_*$, $\mathbf{w}_*^k \in \mathbb{R}^{d_*}, 1 \le k \le K$ are projection directions, and $\mathbf{W}_* = [\mathbf{w}_*^1, \mathbf{w}_*^2, \cdots, \mathbf{w}_*^K]^T$ defines a linear subspace for ranking features. Note that we have omitted the superscript l on $\mathbf{W}_*$ for notation simplicity.

Briefly, the hash function defined in (1) encodes an input data point as the index of the maximum feature value in a K-dimensional linear subspace of $\mathbb{R}^{d_*}$ obtained by the projection matrix $\mathbf{W}_*$. This encoding is entirely based on feature comparison and relative order rather than the feature value itself. It can be seen as a non-linear ordinal embedding that is scale-invariant. It is also resistant against feature noise since the generated ranking codes do not change as long as the noises are not larger than the gap between the leading and second largest features in the projection subspace.

Obviously, each hash code generated in this way requires only $\lceil \log_2(K) \rceil$ binary bits of storage and therefore a L-bit K-ary ranking hash code can be compactly represented using $L \times \lceil \log_2(K) \rceil$ binary bits.

Note that the values of K can range from 2 to $\min\{d_\mathcal{X}, d_\mathcal{Y}\}$. $K = 2$ produces the pairwise orders between the projected features while a larger K results in higher-order comparison between the features. In this spirit, larger values of K places emphasis on a more global comparison between the orders of the features in the subspace.

3.3 The Connection with WTA

The proposed ranking-based multimodal hashing is closely related to WTA Hash [7]. Indeed, WTA Hash can be seen as a special case of the hash function defined in (1) when the linear subspaces are defined by axis-aligned projections. The restriction to ranking only original feature dimensions greatly limits the flexibility of WTA to discover the potential discriminativity of ranking properties hidden in arbitrary linear subspaces. In addition, since WTA is based on random selection of feature dimensions, the hash codes obtained in heterogeneous feature spaces are incomparable making WTA not applicable to multi-modal hashing. In comparison, the generalized ranking-based hash function can be flexibly tuned to rank arbitrary feature subspaces and discover the rank correlation measures across heterogeneous data modalities.

3.4 The Objective Function

For each cross-modal training pair $(\mathbf{x}_i, \mathbf{y}_j)$ with a similarity label s_{ij}, we define an error term incurred by the hash func-

tion in (1) as

$$\mathrm{erf}(h^i_{\mathcal{X}}, h^j_{\mathcal{Y}}, s_{ij}) = \begin{cases} \alpha\, \mathrm{I}(h^i_{\mathcal{X}} \neq h^j_{\mathcal{Y}}), & s_{ij} = 1 \\ \beta\, \mathrm{I}(h^i_{\mathcal{X}} = h^j_{\mathcal{Y}}), & s_{ij} = 0 \end{cases} \tag{2}$$

where $\mathrm{I}(\cdot)$ is the indicator function that equals 1 when the condition holds and 0 otherwise; $h^i_{\mathcal{X}}$ and $h^j_{\mathcal{Y}}$ are short for $h_{\mathcal{X}}(\mathbf{x}_i; \mathbf{W}_{\mathcal{X}})$ and $h_{\mathcal{Y}}(\mathbf{y}_j; \mathbf{W}_{\mathcal{Y}})$; and α and β are two hyper-parameters controlling the penalty for miss-assigned pairs.

Intuitively, this error function penalizes similar pairs with different hash bits or dissimilar pairs with the same hash bit, with α and β controlling the false negative and false positive punishment respectively. The overall learning objective is to find $\mathbf{W}_{\mathcal{X}}$ and $\mathbf{W}_{\mathcal{Y}}$ to minimize the cumulative errors over all the training data:

$$E(\mathbf{W}_{\mathcal{X}}, \mathbf{W}_{\mathcal{Y}}) = \sum_{s_{ij} \in \mathcal{S}} \mathrm{erf}(h^i_{\mathcal{X}}, h^j_{\mathcal{Y}}, s_{ij}) \tag{3}$$

Note that $\mathbf{W}_{\mathcal{X}}$ and $\mathbf{W}_{\mathcal{Y}}$ factor into the above objective function because $h^i_{\mathcal{X}}$ and $h^j_{\mathcal{Y}}$ are functions of $\mathbf{W}_{\mathcal{X}}$ and $\mathbf{W}_{\mathcal{Y}}$.

4 Optimization

4.1 Reformulation

The objective function in (3) is hard to optimize due to the $\arg\max$ terms, which are typically discontinuous and non-convex. We seek a linear upper bound of $E(\mathbf{W}_{\mathcal{X}}, \mathbf{W}_{\mathcal{Y}})$ and attempt to minimize this upper bound instead.

Note that each hash function $h^{(l)}_*(\cdot)$ in (1) can be equivalently formulated as $\mathbf{h}_*$ as shown below

$$\begin{aligned} &\mathbf{h}_*(\mathbf{z}_*; \mathbf{W}_*) = \arg\max_{\mathbf{g}_*} \mathbf{g}_*^T \mathbf{W}_* \mathbf{z}_*, \\ &\text{s.t. } \mathbf{g}_* \in \{0,1\}^K, \mathbf{1}^T \mathbf{g}_* = 1, \end{aligned} \tag{4}$$

where the 1-of-K coding scheme is used to represent the K-ary hash code for an input feature. To see the equivalence, the constrained hash code acts as a dimension selector of the maximum features of $\mathbf{W}_*\mathbf{z}_*$, where the maximum value can only be obtained by setting the bit of $\mathbf{g}_*$ corresponding to the maximum dimension to 1. We also note that the reformulation is only for optimization convenience. It does not increase the binary coding length of a ranking hash code, i.e., only $\lceil \log K \rceil$ bits are needed to encode a K-ary code bit.

Given a cross-modal pair $(\mathbf{x}_i, \mathbf{y}_j)$, let $\mathbf{h}^i_{\mathcal{X}}$ and $\mathbf{h}^j_{\mathcal{Y}}$ be their hash codes obtained through (4) (i.e. $\mathbf{h}^i_{\mathcal{X}} \equiv \mathbf{h}_{\mathcal{X}}(\mathbf{x}_i; \mathbf{W}_{\mathcal{X}})$ and $\mathbf{h}^j_{\mathcal{Y}} \equiv \mathbf{h}_{\mathcal{Y}}(\mathbf{y}_j; \mathbf{W}_{\mathcal{Y}})$). Then, the error function (2) can be upper bounded by

$$\begin{aligned} &\mathrm{erf}(\mathbf{h}^i_{\mathcal{X}}, \mathbf{h}^j_{\mathcal{Y}}, s_{ij}) \leq \\ &\max_{\mathbf{g}^{ij}_{\mathcal{X}}, \mathbf{g}^{ij}_{\mathcal{Y}}} [\mathrm{erf}(\mathbf{g}^{ij}_{\mathcal{X}}, \mathbf{g}^{ij}_{\mathcal{Y}}, s_{ij}) + (\mathbf{g}^{ij}_{\mathcal{X}})^T \mathbf{W}_{\mathcal{X}} \mathbf{x}_i + (\mathbf{g}^{ij}_{\mathcal{Y}})^T \mathbf{W}_{\mathcal{Y}} \mathbf{y}_j] \\ &\quad - (\mathbf{h}^i_{\mathcal{X}})^T \mathbf{W}_{\mathcal{X}} \mathbf{x}_i - (\mathbf{h}^j_{\mathcal{X}})^T \mathbf{W}_{\mathcal{Y}} \mathbf{y}_j \end{aligned} \tag{5}$$

This upper bound directly follows from the following inequality

$$\begin{aligned} &\max_{\mathbf{g}^{ij}_{\mathcal{X}}, \mathbf{g}^{ij}_{\mathcal{Y}}} [\mathrm{erf}(\mathbf{g}^{ij}_{\mathcal{X}}, \mathbf{g}^{ij}_{\mathcal{Y}}, s_{ij}) + (\mathbf{g}^{ij}_{\mathcal{X}})^T \mathbf{W}_{\mathcal{X}} \mathbf{x}_i + (\mathbf{g}^{ij}_{\mathcal{Y}})^T \mathbf{W}_{\mathcal{X}} \mathbf{x}_j] \\ &\quad \geq \mathrm{erf}(\mathbf{h}^i_{\mathcal{X}}, \mathbf{h}^j_{\mathcal{Y}}, s_{ij}) + (\mathbf{h}^i_{\mathcal{X}})^T \mathbf{W}_{\mathcal{X}} \mathbf{x}_i + (\mathbf{h}^j_{\mathcal{X}})^T \mathbf{W}_{\mathcal{Y}} \mathbf{y}_j \end{aligned}$$

Note that the above $\max$ should be taken with the constraints in (4). We do not write them underneath the $\max$ function to avoid notational clutter.

Thus, our objective boils down to minimizing the following function with respect to $\mathbf{W}_{\mathcal{X}}$ and $\mathbf{W}_{\mathcal{Y}}$

$$\begin{aligned} &\Theta(\mathbf{W}_{\mathcal{X}}, \mathbf{W}_{\mathcal{Y}}) = \\ &\sum_{s_{ij} \in \mathcal{S}} \Big\{ \max_{\mathbf{g}^{ij}_{\mathcal{X}}, \mathbf{g}^{ij}_{\mathcal{Y}}} [\mathrm{erf}(\mathbf{g}^{ij}_{\mathcal{X}}, \mathbf{g}^{ij}_{\mathcal{Y}}, s_{ij}) + (\mathbf{g}^{ij}_{\mathcal{X}})^T \mathbf{W}_{\mathcal{X}} \mathbf{x}_i \\ &\quad + (\mathbf{g}^{ij}_{\mathcal{Y}})^T \mathbf{W}_{\mathcal{Y}} \mathbf{y}_j] - (\mathbf{h}^i_{\mathcal{X}})^T \mathbf{W}_{\mathcal{X}} \mathbf{x}_i - (\mathbf{h}^j_{\mathcal{X}})^T \mathbf{W}_{\mathcal{Y}} \mathbf{y}_j \Big\} \end{aligned} \tag{6}$$

4.2 The Learning Algorithm

The upper bound in (6) is convex-concave and piece-wise linear with respect to $\mathbf{W}_{\mathcal{X}}$ and $\mathbf{W}_{\mathcal{Y}}$. It is not differentiable because both the $\max$ term and $(\mathbf{h}^i_{\mathcal{X}}, \mathbf{h}^j_{\mathcal{Y}})$ depend on those projections. Also note that $\mathbf{W}_{\mathcal{X}}$ and $\mathbf{W}_{\mathcal{Y}}$ are not independent of each other because the cross-modal influence is propagated through the $\max$ term.

Our learning algorithm involves the following alternating optimization steps.

First, consider $\mathbf{W}_{\mathcal{X}}$ and $\mathbf{W}_{\mathcal{Y}}$ are fixed. We need to solve the $\max$ problem of the augmented error in the square brackets: $\mathrm{erf}(\mathbf{g}^{ij}_{\mathcal{X}}, \mathbf{g}^{ij}_{\mathcal{Y}}, s_{ij}) + (\mathbf{g}^{ij}_{\mathcal{X}})^T \mathbf{W}_{\mathcal{X}} \mathbf{x}_i + (\mathbf{g}^{ij}_{\mathcal{Y}})^T \mathbf{W}_{\mathcal{Y}} \mathbf{y}_j$. This discrete-optimization admits a global optimal solution. Specifically, it is not hard to see that the solution corresponds to the maximum entry in the following matrix with index p and q

$$m_{pq} = \begin{cases} \bar{x}^{(p)}_i + \bar{y}^{(q)}_j + \alpha(1 - s_{ij}) & \text{if } p = q \\ \bar{x}^{(p)}_i + \bar{y}^{(q)}_j + \beta s_{ij} & \text{otherwise} \end{cases} \tag{7}$$

where $1 \leq p, q \leq K$ and $\bar{x}^{(p)}_i$ and $\bar{y}^{(q)}_j$ denote the p^{th} and q^{th} dimension of $\mathbf{W}_{\mathcal{X}}\mathbf{x}_i$ and $\mathbf{W}_{\mathcal{Y}}\mathbf{y}_j$, respectively. Assuming that the entry at (p^*, q^*) of the matrix attains the maximum value, the maxima of the augmented error, denoted by $(\widehat{\mathbf{g}}^{ij}_{\mathcal{X}}, \widehat{\mathbf{g}}^{ij}_{\mathcal{Y}})$, are 1-of-$K$ binary vectors with the p^*th and the q^*th dimension set to 1. On the other hand, $(\mathbf{h}^i_{\mathcal{X}}, \mathbf{h}^j_{\mathcal{Y}})$ are the hashing codes selecting the maximal entries in the projected vectors $\mathbf{W}_{\mathcal{X}}\mathbf{x}_i$ and $\mathbf{W}_{\mathcal{Y}}\mathbf{y}_j$.

Now, considering that $(\widehat{\mathbf{g}}^{ij}_{\mathcal{X}}, \widehat{\mathbf{g}}^{ij}_{\mathcal{Y}})$ and $(\mathbf{h}^i_{\mathcal{X}}, \mathbf{h}^j_{\mathcal{Y}})$ are fixed. Then, $\Theta(\mathbf{W}_{\mathcal{X}}, \mathbf{W}_{\mathcal{Y}})$ becomes a linear function of $\mathbf{W}_{\mathcal{X}}$ and $\mathbf{W}_{\mathcal{Y}}$ and the following negative gradients can be computed

accordingly:

$$
\begin{aligned}
-\Delta \mathbf{W}_{\mathcal{X}} &= \sum_{i,j} (\mathbf{h}_{\mathcal{X}}^{i} - \widehat{\mathbf{g}}_{\mathcal{X}}^{ij}) \mathbf{x}_i^T \\
-\Delta \mathbf{W}_{\mathcal{Y}} &= \sum_{i,j} (\mathbf{h}_{\mathcal{Y}}^{j} - \widehat{\mathbf{g}}_{\mathcal{Y}}^{ij}) \mathbf{y}_j^T .
\end{aligned} \tag{8}
$$

The gradients in (8) can be used to update $\mathbf{W}_{\mathcal{X}}$ and $\mathbf{W}_{\mathcal{Y}}$ only when the data can fit in the physical memory all at once; otherwise, batch updates can be applied using a subset of the training ensemble. Specifically, $\mathbf{W}_{\mathcal{X}}$ and $\mathbf{W}_{\mathcal{Y}}$ can be updated with one training pair at a time, leading to the following online learning procedure:

$$
\begin{aligned}
\mathbf{W}_{\mathcal{X}} &\leftarrow \mathbf{W}_{\mathcal{X}} + \eta (\mathbf{h}_{\mathcal{X}}^{i} - \widehat{\mathbf{g}}_{\mathcal{X}}^{ij}) \mathbf{x}_i^T \\
\mathbf{W}_{\mathcal{Y}} &\leftarrow \mathbf{W}_{\mathcal{Y}} + \eta (\mathbf{h}_{\mathcal{Y}}^{j} - \widehat{\mathbf{g}}_{\mathcal{Y}}^{ij}) \mathbf{y}_j^T ,
\end{aligned} \tag{9}
$$

where η is the learning rate.

The above learning procedure is used to learn hash function for one code bit at a time. To learn L ranking hash codes, the procedure can be repeated L times. The convex-concave nature of the objective function means that there are typically multiple local minima instead of a global one. This is a desirable property in our application because each local minima corresponds to a hash function and multiple functions reveal complementary ranking structure to each other, leading to discriminative ranking hash codes. However, independently learned hash functions may be redundant because they may correspond to the same local minima. In order to minimize code redundancy, we use Adaboost to learn multiple hash functions sequentially. The benefits of boosting in learning sequential hash functions have been verified in previous studies such as [12] and [4]. Briefly, each training pair is assigned a weight which is updated based on errors incurred by current hash function. Since boosting is not the contribution of this paper, we do not elaborate on it.

4.3 Algorithm Complexity

Consider the complexity for the batch update, where each batch contains N training pairs. Note that each of the iterative learning step involves the alternating updates of $\mathbf{W}_*$ and the the hash codes. Specifically, the matrix-vector product $\mathbf{W}_{\mathcal{X}} \mathbf{x}_i$ and $\mathbf{W}_{\mathcal{Y}} \mathbf{y}_i$ can be computed at a cost of $O(NKd_{\mathcal{X}} + NKd_{\mathcal{Y}})$ for N training pairs. Computing $\mathbf{h}_{\mathcal{X}}^{i}$ and $\mathbf{h}_{\mathcal{Y}}^{j}$ takes $O(NK)$; $\widehat{\mathbf{g}}_{\mathcal{X}}^{ij}$ and $\widehat{\mathbf{g}}_{\mathcal{Y}}^{ij}$ each can be obtained at a cost of $O(NK^2)$. After obtaining those hash codes, the complexity for computing $\Delta \mathbf{W}_*$ is $O(N)$. Therefore, the overall costs for each iteration step is $O(NK^2 + NKd_{\mathcal{X}} + NKd_{\mathcal{Y}})$. Since K is typically very small (e.g. ≤ 8) compared with $N, d_{\mathcal{X}}$ and $d_{\mathcal{Y}}$, the iterative updates can be computed efficiently.

5 Experiments

5.1 Datasets

NUS-WIDE[1]. The NUS-WIDE dataset is a real-world image dataset that contains $269,648$ images downloaded from Flicker. Each image has a number of textual tags and is labeled with one or more of 81 concepts. We select the $186,577$ image-tag pairs belonging to the 10 largest concepts. The images are represented by 500-D bag-of-visual words (BOVW) and the image tags are represented by 1000-D tag occurrence feature vectors.

Wiki[2]. The wiki dataset is crawled from Wikipedia's "featured articles". It consists of $2,866$ documents which are image-text pairs and annotated with semantic labels of 10 categories. Each image is represented as 128-D bag-of-SIFT feature vector. As for text documents, we extract the 1000-D tf-idf features over the most representative words and further apply PCA to reduce its dimensionality to 150-D.

These two datasets, especially NUS-WIDE, have become the *de factor* benchmarks to evaluate performances of cross-modal hashing algorithms [1, 4, 18, 19].

5.2 Baselines and Settings

We compare with a range of state-of-the-art cross-modal hashing algorithms: Co-Regularized Hashing (CRH) [12], Cross-Modal Similarity Sensitive hashing (CMSSH) [4], Cross-View Hashing (CVH) [10], Inter-Media Hashing (IMH) [1] and Collective Matrix Factorization Hashing (CMFH) [2]. The publicly available implementations of the above algorithms were used in the experiments.

CMRSH takes three parameters, namely, the subspace dimension K and the two penalty coefficients α and β. We use linear search in log scale of K from 1 to 4 (i.e. $K = 2^1, \cdots, 2^4$). As for α and β, we use grid search among $\{0.2, 0.4, 0.6, 0.8, 1.0\}$. A five-fold cross-validation is performed on a heldout subset of the training set to find the best parameters. Note that since our hash code is K-ary, we set $L = \lfloor \frac{N_b}{\lceil \log_2(K) \rceil} \rfloor$ when comparing to other binary hashing codes at N_b bits to ensure fairness. For all compared algorithms, we use the parameters suggested by their authors.

For benchmarking, we consider two cross-modal retrieval tasks: 1) image-query-text, where an image is used to query relevant text in the text database; and 2) text-query-image, where textual queries are used to search for relevant images in the image database. Specifically, we follow the widely used [2, 3, 16] metrics: *mean Average Precision* (mAP), which is the area under precision-recall curve averaged over all test queries, and *top-k precision*, defined as the precision of k nearest neighbor measured in hamming distance.

[1] http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm
[2] http://www.svcl.ucsd.edu/projects/crossmodal/

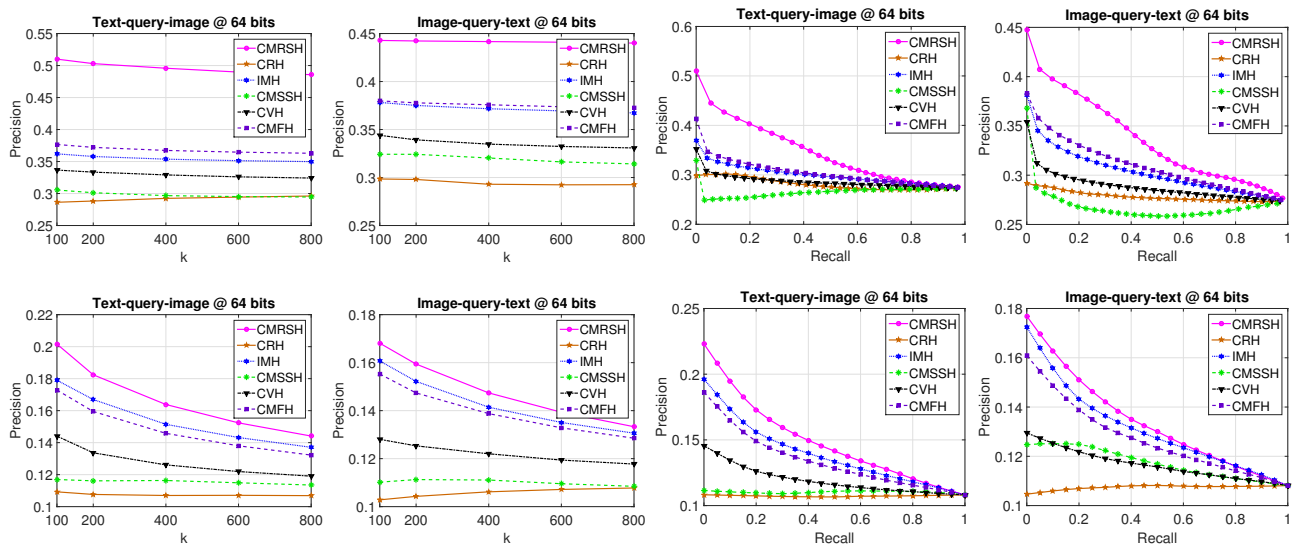

Fig. 1. Results of Top-k precision and precision-recall on NUS-WIDE (row 1) and Wiki (row 2).

Table 1. mAP comparison on NUS-WIDE.

Task	Method	Code Length		
		$N_b = 24$	$N_b = 48$	$N_b = 64$
Image query text	CVH	0.2984	0.2915	0.2893
	CMSSH	0.2780	0.2768	0.2766
	CRH	0.2856	0.2869	0.2872
	IMH	0.3132	0.3103	0.3079
	CMFH	0.3163	0.3164	0.3134
	CMRSH	**0.3284**	**0.3272**	**0.3300**
Text query image	CVH	0.2935	0.2984	0.2863
	CMSSH	0.2726	0.2728	0.2733
	CRH	0.2823	0.2833	0.2833
	IMH	0.3072	0.3039	0.3025
	CMFH	0.3094	0.3077	0.3072
	CMRSH	**0.3513**	**0.3680**	**0.3740**

Table 2. mAP comparison on Wiki.

Task	Method	Code Length		
		$N_b = 24$	$N_b = 48$	$N_b = 64$
Image query text	CVH	0.1254	0.1232	0.1236
	CMSSH	0.1466	0.1456	0.1475
	CRH	0.1140	0.1146	0.1147
	IMH	0.1588	0.1529	0.1515
	CMFH	0.1462	0.1448	0.1442
	CMRSH	**0.1743**	**0.1778**	**0.1823**
Text query image	CVH	0.1249	0.1254	0.1252
	CMSSH	0.1226	0.1187	0.1173
	CRH	0.1099	0.1105	0.1109
	IMH	0.1462	0.1467	0.1457
	CMFH	0.1388	0.1410	0.1416
	CMRSH	**0.1472**	**0.1558**	**0.1587**

5.3 Results on NUS-WIDE

We randomly select $2,000$ image-text pairs as test queries and the rest are used as the database. In order to test the out-of-sample extension capability of the compared algorithms, we randomly select $3,000$ instances from the database to construct the similarity matrix for hash function learning, and the learned hash functions are applied to the entire database to generate the database hash codes.

The test results, with the number of bits N_b varied from 24, 48 to 64, are presented in Table 1 and Figure 1 (row 1). They show that CMRSH outperforms the other state-of-the-art methods in both the image-query-text and text-query-image tasks. We note that CMRSH demonstrates consistent performance improvement with longer codes, which is generally not true for all the other methods. This is because some of the compared algorithms (e.g. CVH, IMH) involve some sort of eigenvalue decomposition sub-problems with orthogonality constraints on different bits. As a result, the first few bits have high variance and are quite discriminative. As the length of hash code increases, the hash bits gradually become dominated by low-variance bits, thus leading to unsatisfactory results.

It is worth noting that although CMSSH and CRH also use boosting to learn multiple hash bits sequentially, their performances are far below CMRSH. Overall, the performance superiority is mainly due to the effectiveness of the ranking-based hash learning framework. Actually, our settings for NUSWIDE are quite similar to real-world scenarios, where labeled data are limited compared to the entire data corpus. The good performance indicates that CMRSH can handle large-scale datasets well.

5.4 Results on Wiki

Wiki is much smaller than NUSWIDE; but it has been used in some previous works [4, 12] and we also report the major results on this dataset. We randomly select 20% from the entire dataset as the test queries, and a separate $1,000$ instances from the remaining 80% to construct the pairwise training set. Similar to NUSWIDE, we apply the hash functions learned on the training subset to the entire database to test the out-of-sample extension capability of the compared algorithms.

Table 2 and Figure 1 (row 2) show the results on the Wiki dataset. We note that the overall performance of all methods are much lower that in NUSWIDE. This is mainly due to two reasons :1) less data are used for training; and 2) the semantic gaps between two views in Wiki is quite large [3]. However, CMRSH still consistently outperforms the other methods in this dataset with the performance gains up to 20% (image-query-text @ 64 bits). This indicates that the ranking structure exploited by CMRSH can be very useful in bridging the semantic gap between multimodal feature spaces. Similar to what we have seen in NUSWIDE, CMRSH demonstrates consistent improvement in performance with increasing code length; while the other methods' performance increase marginally or even drop as a result of not being able to generate enough discriminative hash bits. The overall leading performance further confirms the effectiveness of CMRSH.

6 Conclusion

In this paper, we propose a novel cross-modal hash learning framework, referred to as Cross-Modal Ranking Subspace Hashing (CMRSH), for large-scale cross-modal similarity search. Specifically, we exploit a new class of hash functions based on the ranking structure of feature subspaces. We explicitly minimize an empirical objective with respect to the ranking-based hash functions and provide an effective iterative learning algorithm. Extensive experiments on two widely used multimodal datasets demonstrate the superiority of CMRSH in generating highly discriminative compact hash codes for cross-modal retrieval tasks.

7 Acknowledgment

This material is based upon work partially supported by the NASA under Grant Number NNX15AV40A. Any opinions, findings, and conclusion or recommendations expressed in this materials are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Jingkuan Song, Yang Yang, Yi Yang, Zi Huang, and Heng Tao Shen, "Inter-media hashing for large-scale retrieval from heterogeneous data sources," in *ACM SIGMOD, 2013*, pp. 785–796.

[2] Guiguang Ding, Yuchen Guo, and Jile Zhou, "Collective matrix factorization hashing for multimodal data," in *CVPR, 2014*, pp. 2083–2090.

[3] Jile Zhou, Guiguang Ding, and Yuchen Guo, "Latent semantic sparse hashing for cross-modal similarity search," in *ACM SIGIR, 2014*, pp. 415–424.

[4] Michael M Bronstein, Alexander M Bronstein, Fabrice Michel, and Nikos Paragios, "Data fusion through cross-modality metric learning using similarity-sensitive hashing," in *CVPR, 2010*, pp. 3594–3601.

[5] Yueting Zhuang, Zhou Yu, Wei Wang, Fei Wu, Siliang Tang, and Jian Shao, "Cross-media hashing with neural networks," in *ACM MM, 2014*, pp. 901–904.

[6] Jonathan Masci, Michael M Bronstein, Alexander M Bronstein, and Jürgen Schmidhuber, "Multimodal similarity-preserving hashing," *TPAMI, 2014*.

[7] Jay Yagnik, Dennis Strelow, David A Ross, and Rueisung Lin, "The power of comparative reasoning," in *ICCV, 2011*, pp. 2431–2438.

[8] Andrei Z Broder, Moses Charikar, Alan M Frieze, and Michael Mitzenmacher, "Min-wise independent permutations," *Journal of Computer and System Sciences*, vol. 60, no. 3, pp. 630 – 659, 2000.

[9] Kai Li, Jun Ye, and Kien A. Hua, "What's making that sound?," in *ACM MM*, 2014.

[10] Shaishav Kumar and Raghavendra Udupa, "Learning hash functions for cross-view similarity search," in *IJCAI, 2011*, vol. 22, p. 1360.

[11] Novi Quadrianto and Christoph H Lampert, "Learning multi-view neighborhood preserving projections," in *ICML, 2011*, pp. 425–432.

[12] Yi Zhen and Dit-Yan Yeung, "Co-regularized hashing for multimodal data," in *NIPS, 2012*, pp. 1376–1384.

[13] Yi Zhen and Dit-Yan Yeung, "A probabilistic model for multimodal hash function learning," in *SIGKDD, 2012*.

[14] Deming Zhai, Hong Chang, Yi Zhen, Xianming Liu, Xilin Chen, and Wen Gao, "Parametric local multimodal hashing for cross-view similarity search," in *IJCAI, 2013*, pp. 2754–2760.

[15] Xiaofeng Zhu, Zi Huang, Heng Tao Shen, and Xin Zhao, "Linear cross-modal hashing for efficient multimedia search," in *ACM MM, 2013*, pp. 143–152.

[16] Dongqing Zhang and Wu-Jun Li, "Large-scale supervised multimodal hashing with semantic correlation maximization," in *AAAI, 2014*.

[17] Fei Wu, Zhou Yu, Yi Yang, Siliang Tang, Yin Zhang, and Yueting Zhuang, "Sparse multi-modal hashing," *IEEE TMM, 2014*.

[18] Zijia Lin, Guiguang Ding, Mingqing Hu, and Jianmin Wang, "Semantics-preserving hashing for cross-view retrieval," in *CVPR*, 2015, pp. 3864–3872.

[19] Botong Wu, Qiang Yang, Wei-Shi Zheng, Yizhou Wang, and Jingdong Wang, "Quantized correlation hashing for fast cross-modal search," in *IJCAI*, 2015.