

Robust Hashing for Multi-View Data: Jointly Learning Low-Rank Kernelized Similarity Consensus and Hash Functions

Lin Wu[‡], Yang Wang[†]

[‡]The University of Adelaide, Australia

[†]The University of New South Wales, Kensington, Sydney, Australia

lin.wu@adelaide.edu.au wangy@cse.unsw.edu.au

Abstract

Learning hash functions/codes for similarity search over multi-view data is attracting increasing attention, where similar hash codes are assigned to the data objects characterizing consistently neighborhood relationship across views. Traditional methods in this category inherently suffer three limitations: 1) they commonly adopt a two-stage scheme where similarity matrix is first constructed, followed by a subsequent hash function learning; 2) these methods are commonly developed on the assumption that data samples with multiple representations are noise-free, which is not practical in real-life applications; 3) they often incur cumbersome training model caused by the neighborhood graph construction using all N points in the database ($O(N)$). In this paper, we motivate the problem of jointly and efficiently training the robust hash functions over data objects with multi-feature representations which may be noise corrupted. To achieve both the robustness and training efficiency, we propose an approach to effectively and efficiently learning low-rank kernelized¹ hash functions shared across views. Specifically, we utilize landmark graphs to construct tractable similarity matrices in multi-views to automatically discover neighborhood structure in the data. To learn robust hash functions, a latent low-rank kernel function is used to construct hash functions in order to accommodate linearly inseparable data. In particular, a latent kernelized similarity matrix is recovered by rank minimization on multiple kernel-based similarity matrices. Extensive experiments on real-world multi-view datasets validate the efficacy of our method in the presence of error corruptions.

¹We use kernelized similarity rather than kernel, as it is not a squared symmetric matrix for data-landmark affinity matrix.

1 Introduction

Hashing is dramatically efficient for similarity search over low-dimensional binary codes with low storage cost. Intensive hashing methods valid on single data source have been proposed which can be classified into *data-independent* hashing such as locality sensitive hashing (LSH) [Datar *et al.*, 2004] and *data-dependent* hashing or learning based hashing [Weiss *et al.*, 2008; Wang *et al.*, 2010].

In real-life situations, data objects can be decomposed of multi-view (feature) spaces where each view can characterize its individual property, *e.g.*, an image can be described by color histograms and textures, and the two features turn out to be complementary to each other [Wang *et al.*, 2014; Wang *et al.*, 2013; Wang *et al.*, 2015c; Wu *et al.*, 2013; Wu *et al.*, 2016; Wang *et al.*, 2016b; Wang *et al.*, 2015d; Wang *et al.*, 2015e; Wang *et al.*, 2016a]. Consequently, a wealth of multi-view hashing methods [Zhang *et al.*, 2011; Kim *et al.*, 2012; Masci *et al.*, 2014; Song *et al.*, 2011; Liu *et al.*, 2012b; Shen *et al.*, 2015] are developed in order to effectively leverage complementary priors from multi-views to achieve performance improvement in similarity search. The critical issue is to ensure the learned hash codes can well preserve the original data similarities regarding view-dependent feature representations. To be specific, similar hash codes are assigned to data objects that consistently capture nearest neighborhood structure across all views.

1.1 Motivation

Despite improved performance delivered by existing multi-view hashing methods [Zhang *et al.*, 2011; Kim *et al.*, 2012; Masci *et al.*, 2014; Song *et al.*, 2011; Liu *et al.*, 2012b; Shen *et al.*, 2015], some fundamental limitations can be identified:

- The learning process is conducted by a two-stage mechanism where hash functions are learned based on pre-constructed data similarity matrix. Their methods commonly assume that data samples are noise-free under multiple views whereas in real-world applications input data objects may be noisy (*e.g.*, missing values in pixels), resulting in corresponding similarity matrices being corrupted by considerable noises [Wang *et al.*, 2015d; Xia *et al.*, 2014]. Moreover, the recovery of *consen-*

sus or *requisite* similarity values across views in the presence of noise contamination remains an unresolved challenge in multi-view data analysis [Li *et al.*, 2015; Zheng *et al.*, 2015; Ye *et al.*, 2012].

This motivates us to deliver a framework to jointly and effectively learn similarity matrices and *robust* hash functions with kernel functions plugged because the kernel trick is able to tackle linearly inseparable data [Liu *et al.*, 2012a]. To this end, a *latent* kernelized similarity matrix is recovered shared across views by using low-rank representation (LRR) [Liu *et al.*, 2013] which is robust to corrupted observations. The recovered *low-rank* kernelized similarity matrix is consensus-reaching across views and can reveal the true underlying structures in data points.

- State-of-the-art multi-view hashing methods is less efficiency in their learning procedure because the learning is performed by building and accessing a neighborhood graph using all N points ($O(N^2)$). This action is intractable in off-line training when N is large.

To this end, we are further motivated to employ an *landmark graph* to build an approximate neighborhood graph using landmarks [Liu *et al.*, 2011; Liu *et al.*, 2010b], in which the similarity between a pair of data points is measured with respect to a small number of landmarks (typically a few hundred). The resulting graph is built in $O(N)$ time and sufficiently sparse with performance approaching to true k -NN graphs as the number of landmarks increases [Liu *et al.*, 2011].

1.2 Our Method

In this paper, we propose a novel approach to robust multi-view hashing by effectively and efficiently learning a set of hash functions and a low-rank kernelized similarity matrix shared by multiple views.

We remark that our method is fundamentally different from existing multi-view hashing methods that are conditioned on corruption-free similarities, which has diminished their application to real-world tasks. Instead, we propose to learn hash functions and kernel-based similarities under a more realistic scenario with noisy observations. Our method is advantageous in the aspect of efficiency due to the employment of approximate neighborhood with landmark graphs. We clarify the recovered low-rank similarity matrix in kernel functions to be the kernelized rather than kernel since it is not a symmetric matrix yet characterizes non-linear similarities. The proposed method is also different from partial view study [Kumar and III, 2011; Wang *et al.*, 2015a], where they consider the case that data examples with some modalities are missing. Our approach follows the setting of multi-view learning which aims to improve existing single view model by learning a model utilizing data collected from multiple channels [Xu *et al.*, 2013; Zheng *et al.*, 2015; Kumar *et al.*, 2011; Wang *et al.*, 2014; Wang *et al.*, 2015b; Xia *et al.*, 2014] where all data samples have full information in all views.

In our framework, the low rank minimization is enforced to yield a consensus-reaching, kernelized similarity matrix

shared by multiple views where larger similarity values indicate corresponding data objects from the same cluster, while smaller similarity values imply those come from distinct clusters. Thus, the learned low-rank similarity matrix against multi-views can reflect the underlying clustering information.

Technically, a nonlinear kernelized similarity matrix in the m -th view, denoted as $K^{(m)}$, can be decomposed into three components: (1) A latent low-rank kernelized similarity matrix \hat{K} , representing the nonlinear requisite or consensus similarities shared across views; (2) a view-dependent redundancy characterizing its individual similarities; and (3) possible error corruptions for view-specific representations. We unify view redundancy and errors into $E^{(m)}$ and impose an $\ell_{2,1}$ -norm constraint on it, denoted as $\|E^{(m)}\|_{2,1}$. This is because view redundancy and disturbing errors are always sparsely distributed, and minimizing $\|E^{(m)}\|_{2,1}$ is able to identify non-zero sparse columns revealing corresponding redundancy/errors. Note that in this work, “error” generally refers to error corruptions or perturbation, *e.g.*, noise or missing values, in view-dependent feature values. These principles are formulated into an objective function, which is optimized based on the inexact Augmented Lagrangian Multiplier (ALM) scheme [Lin *et al.*, 2010]. It allows us to jointly learn a latent low-rank nonlinear similarity with corruption free and optimal hash functions for multi-view data, where hash codes are restricted to well preserve local (neighborhood) geometric structures in each view. We remark that several cross-view semantic hashing algorithms [Ou *et al.*, 2013; Wei *et al.*, 2014; Kumar and Udupa, 2011] have been developed to embed multiple high dimensional features from *heterogeneous* data sources into one Hamming space, while preserving their original similarities. Our setting is fundamentally different from cross-view/modal hashing in the aspect that we aim to leverage multiple features to jointly learn hash functions and a latent nonlinear similarity matrix over a *homogeneous* data source. To the best of our knowledge, we are the first to systematically address the problem of multi-view hashing with possible data error corruptions.

1.3 Contributions

The major contributions of this paper are three-fold.

- We motivate the problem of robust hashing over multi-view data with nonlinear data distribution, and propose to learn the robust hash functions and a low-rank kernelized similarity matrix shared by views.
- An iterative low-rank recovery optimization technique is proposed to learn the robust hashing functions. For the sake of efficiency, the neighborhood graph is approximated by using landmark graphs with sparse connection between data points.
- Extensive experiments conducted on real-world multi-view datasets validate the efficacy of our method in the presence of error corruptions for multi-view feature representations.

2 Related Work

2.1 Multi-view Learning based Hashing

The purpose of multi-view learning based hashing is to learn better hash codes by leveraging multiple views. Some recent representative works include Multiple Feature Hashing (MFH) [Song *et al.*, 2011], Composite Hashing with Multiple Sources (CHMS) [Zhang *et al.*, 2011], Compact Kernel Hashing with multiple features (CKH) [Liu *et al.*, 2012b], and Multi-view Sequential Spectral Hashing (SSH) [Kim *et al.*, 2012]. However, these methods have common drawbacks that they typically apply spectral graph technique (*e.g.*, k -NN graph) to model a similarities between data points. In general, the complexity of constructing the similarity matrix is $\mathcal{O}(N^2)$ for N data points, which is not pragmatic in large-scale applications. Moreover, the similarity matrix induced by graph construction is very sensitive to noise corruptions. To avoid the construction of similarity matrix, Shen *et al.* [Shen *et al.*, 2015] present a Multi-View Latent Hashing (MVLH) to learn hash codes by performing matrix factorization on a unified kernel feature space over multiple views. Nonetheless, there are significant differences between MVLH and our approach. First, matrix factorization is performed on a unified kernel space which is formed by simply concatenating multiple kernel feature spaces. This would discard distinct local structures in individual views. By contrast, the kernelized similarity matrix is constructed with respect to the distinct characteristic in each view. Second, MVLH neglects the case of potential noise corruption in data samples. In this aspect, we attentively employ the low-rank representation (LRR) [Liu *et al.*, 2013] to recover latent subspace structures from corrupted data.

2.2 Low-rank Modeling

Low-rank modeling in attracting increasing attention due to its capability of recovering the underlying structure among data objects [Wright *et al.*, 2009a; Candes and Recht, 2009; Li *et al.*, 2015; Zhang *et al.*, 2016; Zhang *et al.*, 2014; Zhang *et al.*, 2015]. It has striking success in many applications such as data compression [Wright *et al.*, 2009a], subspace clustering [Liu *et al.*, 2013; Deng *et al.*, 2013; Zhang *et al.*, 2014], and image processing [Zhou *et al.*, 2013; Zhang *et al.*, 2016; Zhang *et al.*, 2015]. For instance, in [Zhang *et al.*, 2016], Zhang *et al.* consider a joint formulation of recovering low-rank and sparse subspace structures for robust representation.

Nowadays, data are usually collected from diverse domains or obtained from various feature extractors, and each group of features can be regarded as a particular view [Xu *et al.*, 2013]. Moreover, these data can be easily corrupted by potential noises (*e.g.*, missing pixels or outliers), or large variations (*e.g.*, post variations in face images) in real applications. In practice, the underlying structure of data could be multiple subspaces, and thus Low-Rank Representation (LRR) is designed to find subspace structures in noisy data [Liu *et al.*, 2013; Zhang *et al.*, 2014]. The multi-view low-rank analysis [Li *et al.*, 2015] is a recently proposed multi-view learning approach, which introduces low-rank constraint to reveal the

intrinsic structure of data, and identifies outliers for the representation coefficients in low-rank matrix recovery.

In this paper, we are the first to apply low-rank learning to reveal structured kernelized similarity among multi-view data, and scale it up well to large-scale applications.

3 Robust Multi-view Hashing

3.1 Preliminary and Problem Definition

Let $\phi(\cdot) = \{\phi_1(\cdot), \dots, \phi_M(\cdot)\}$ be the embedding function for M nonlinear feature spaces, each of which corresponds to one view. Following the Kernelized Locality Sensitive Hashing [Kulis and Grauman, 2009], we uniformly select R samples from the training set X , denoted by Z_r ($r = 1, \dots, R$), to construct kernelized similarity matrices under multiple views. Given a sample represented by its feature x_i , the p -th hash bit can be generated via the linear projection:

$$h_p(x_i) = \text{sign}(C_p^T \phi(x_i) + b_p), \quad (1)$$

where $\text{sign}(\cdot)$ denotes the element-wise function, which is 1 if it is larger or equal to 0 and -1 otherwise. $C_p = \sum_{r=1}^R W_{rp} \phi(Z_r)$ indicates the linear combination of R landmarks, which can be the cluster centers [Liu *et al.*, 2011] via scalable R -means clustering over the feature space with d dimensions. $b_p \in \mathbb{R}$ is a bias term. Then, we have

$$h_p(x_i) = \text{sign}\left(\sum_{r=1}^R W_{rp} K_i + b_p\right), \quad (2)$$

where K_i denotes the i -th column of $K \in \mathbb{R}^{R \times N}$, such that $K = \sum_{m=1}^M K^{(m)}$, and $K^{(m)} \in \mathbb{R}^{R \times N}$ ($m = 1, \dots, M$) denotes the kernelized similarity matrix between R landmarks and N samples corresponding to the kernelized representation $\phi^{(m)}(\cdot)$. Accordingly, the hash code of x_i can be rewritten via the kernel form,

$$y_i = \text{sign}(W^T K_i + b), \quad (3)$$

where $W \in \mathbb{R}^{R \times P}$ and $b = [b_1, \dots, b_P]$.

Given a set of training samples $X = [x_1, \dots, x_N]$ that may contain errors, $x_i^{(m)} \in \mathbb{R}^{d_m \times 1}$ denotes the m -th feature of x_i , and d_m is the dimensionality for the feature space regarding the m -th view. Then $X^{(m)} = [x_1^{(m)}, x_2^{(m)}, \dots, x_N^{(m)}] \in \mathbb{R}^{d_m \times N}$ is the view matrix corresponding to the m^{th} feature of all training data. $x_i = [(x_1^T)^T, \dots, (x_M^T)^T]^T \in \mathbb{R}^{d \times 1}$ is the vector representation of the i^{th} training data using all features where $d = \sum_{m=1}^M d_m$, and M is the number of views. We denote $Y = [y_1, y_2, \dots, y_N] \in \mathbb{R}^{P \times N}$ as the hash codes of the training samples corresponding to all features, and $Y^{(m)} = [y_1^{(m)}, \dots, y_N^{(m)}] \in \mathbb{R}^{P \times N}$ as the hash codes of the training data for the m -th view. We aim to learn a latent low-rank kernel matrix \hat{K} shared across multiple kernels, and construct a set of robust hashing functions $H = \{h_1(\cdot), \dots, h_P(\cdot)\}$ for multi-view data where $h_p: \mathbb{R}^d \mapsto \{1, -1\}$ ($p = 1, 2, \dots, P$), and P is the number of hashing functions, *i.e.*, the hash code length. The kernel function is plugged into hash function because the kernel trick has been theoretically and empirically proved to be able to tackle the data distribution that is almost linearly inseparable [Liu *et al.*, 2012a].

3.2 Low-rank Kernelized Similarity Recovery from Multi-views

Given a collection of high-dimensional multi-view data samples that may contain certain errors for each view-specific representation, we construct multiple nonlinear feature spaces $K^{(m)} (m = 1, \dots, M)$, each of which represents one feature view. To leverage multiple complementary representations, we propose to derive a consensus low-rank kernelized similarity matrix \hat{K} recovered from corrupted data objects, and shared across views. This low-rank nonlinear similarity matrix is considered as the most requisite component, whilst each view also contains individual non-requisite information including redundancy and errors. We explicitly model the redundancy via sparsity since multi-view study suggests that each individual view is sufficient to identify most of the similarity structure, and the deviation between requisite component and data sample is sparse [Kumar *et al.*, 2011]. In reality, data samples can be grossly corrupted due to the sensor failure or communication errors. Thus, an $\ell_{2,1}$ -norm is adopted to characterize errors since they usually cause column sparsity in an affinity matrix [Liu *et al.*, 2013].

In our framework, the low-rank similarity matrix is constructed to be sparse by considering data samples and landmarks, thus ascertaining the efficiency of our approach. Therefore, the latent low-rank kernelized similarity matrix \hat{K} can be recovered from $K^{(m)} (m = 1, \dots, M)$ through a low-rank constraint on \hat{K} and sparse constraint on each $E^{(m)}$, that is,

$$\min_{\hat{K}, E^{(m)}} \|\hat{K}\|_* + \lambda \sum_{m=1}^M \|E^{(m)}\|_{2,1}, \text{ s.t. } K^{(m)} = \hat{K} + E^{(m)}, \hat{K} \geq 0. \quad (4)$$

where λ is the trade-off parameter and $E^{(m)}$ encodes the summation of error corruption and possible noise information regarding the m -th view.

3.3 Objective Function

Many studies [Weiss *et al.*, 2008; Song *et al.*, 2011] have shown the benefits to exploit local structure of the training data to infer accurate and compact hash codes. However, all these algorithms are sensitive to error corruptions, hampering them to be effective in practical situations. By contrast, we propose to jointly learn hash codes by preserving local similarities in multiple views while being robust to errors. To exploit the local structure in each view, we define M affinity matrices $S^{(m)} \in \mathbb{R}^{N \times N} (m = 1, \dots, M)$, one for each view, that is,

$$S_{ij}^{(m)} = \begin{cases} 1, & x_i^{(m)} \in \mathcal{N}_k(x_j^{(m)}) \text{ or } x_j^{(m)} \in \mathcal{N}_k(x_i^{(m)}); \\ 0, & \text{else.} \end{cases}$$

where $\mathcal{N}_k(\cdot)$ is the k -nearest neighbor set, and the Euclidean distance is employed in each feature space to determine the neighborhood. A reasonable criteria of learning hash codes $y_i^{(m)}$ from the m -th view is to ensure similar objects in the original space should have similar binary hash codes. This

can be formulated as below:

$$\min \sum_{i,j=1}^N S_{ij}^{(m)} \|y_i^{(m)} - y_j^{(m)}\|_F^2. \quad (5)$$

Given a training sample x_i , we expect the optimal hash code y_i consistent with its distinct hash codes $y_i^{(m)}$ derived from each view. In this way, the local geometric structure in a single view can be globally optimized. Therefore, we have

$$\min \sum_{m=1}^M \left(\sum_{i,j=1}^N S_{ij}^{(m)} \|y_i^{(m)} - y_j^{(m)}\|_F^2 + \gamma \sum_{i=1}^N \|y_i - y_i^{(m)}\|_F^2 \right), \quad (6)$$

where γ is a trade-off parameter. The main bottleneck in the above formulation is computation where the cost of building the underlying graph and its associate affinity matrix $S^{(m)}$ is $O(d_m N^2)$, which is intractable for large N . To avoid the computational bottleneck, we employ a landmark graph by using a small set of L points called landmarks to approximate the data neighborhood structure [Liu *et al.*, 2011]. Similarities of all N database points are measured with respect to these L landmarks, and the true adjacency/similarity matrix $S^{(m)}$ in the m -th view is approximated using these similarities. First, K-means clustering² is performed on N data points to obtain L ($L \ll N$) clusters center $\mathcal{U} = \{u_j \in \mathbb{R}^{d_m}\}_{j=1}^L$ that act as landmark points. Next, the landmark graph defines the truncated similarities F_{ij} 's between all N data points and L landmarks as,

$$F_{ij} = \begin{cases} \frac{\exp(-\mathcal{D}^2(x_i, u_j)/t)}{\sum_{j' \in \langle i \rangle} \exp(-\mathcal{D}^2(x_i, u_{j'})/t)}, & \forall j \in \langle i \rangle \\ 0, & \text{elsewhere} \end{cases}$$

where $\langle i \rangle \subset [1 : L]$ denotes the indices of k ($k \ll L$) nearest landmarks of points x_i in \mathcal{U} according to a distance function $\mathcal{D}()$ such as ℓ_2 distance, and t denotes the bandwidth parameter. Note that the matrix $F \in \mathbb{R}^{N \times L}$ is highly sparse. Each row of F contains only k non-zero entries which sum to 1. Thus, the landmark graph provides a powerful approximation to the adjacency matrix $S^{(m)}$ as $\hat{S}^{(m)} = F \Lambda^{-1} F^T$ where $\Lambda = \text{diag}(F^T \mathbf{1}) \in \mathbb{R}^{L \times L}$ [Liu *et al.*, 2011].

For ease of representation, we denote $\mathcal{L}(Y, Y^{(m)}) = \sum_{i,j=1}^N \hat{S}_{ij}^{(m)} \|y_i^{(m)} - y_j^{(m)}\|_F^2 + \gamma \sum_{i=1}^N \|y_i - y_i^{(m)}\|_F^2$. To learn a set of hashing functions and a consensus nonlinear representation in a joint framework, we formulate the objective function of robust multi-view hashing as follows

$$\begin{aligned} \min_{W, \hat{K}, b, E^{(m)}} \sum_{m=1}^M \mathcal{L}(Y, Y^{(m)}) + \alpha \|\hat{K}\|_* + \lambda \sum_{m=1}^M \|E^{(m)}\|_{2,1}, \\ \text{s.t. } K^{(m)} = \hat{K} + E^{(m)}, \hat{K} \geq 0, m = 1, \dots, M, \\ y_i = \text{sign}(W^T \hat{K}_i + b) \in \{-1, 1\}^P, Y Y^T = I, \end{aligned} \quad (7)$$

where α is a trade-off parameter, $y_i \in \{-1, 1\}^P$ enforces the hash code y_i to be binary codes, and the constraint $Y Y^T = I$

²In practice, running K-means algorithm on a small subsample of the database with very few iterations is sufficient.

is imposed to encourage bit de-correlations while avoiding the trivial solution. Due to the discrete constraints and non-convexity, the optimization problem in Eq.(7) is difficult to solve. Following spectral hashing [Weiss *et al.*, 2008], we relax the constraints $y_i \in \{-1, 1\}^P$ to be $y_i = W^T \hat{K}_i + b$, then we have

$$\begin{aligned} \min_{W, \hat{K}, b, E^{(m)}} & \sum_{m=1}^M \mathcal{L}(Y, Y^{(m)}) + \alpha \|\hat{K}\|_* + \lambda \sum_{m=1}^M \|E^{(m)}\|_{2,1}, \\ \text{s.t. } & K^{(m)} = \hat{K} + E^{(m)}, \hat{K} \geq 0, m = 1, \dots, M; \\ & y_i = W^T \hat{K}_i + b, YY^T = I. \end{aligned}$$

We rewrite the objective function by further minimizing the least square error regarding Y while regularizing W coupled with trade-off parameters β and δ , it then has

$$\begin{aligned} \min_{W, \hat{K}, b, E^{(m)}} & \sum_{m=1}^M \mathcal{L}(Y, Y^{(m)}) + \alpha \|\hat{K}\|_* + \lambda \sum_{m=1}^M \|E^{(m)}\|_{2,1} \\ & + \beta \left(\|\hat{K}^T W + \mathbf{1}b - Y\|_F^2 + \delta \|W\|_F^2 \right) \\ \text{s.t. } & K^{(m)} = \hat{K} + E^{(m)}, \hat{K} \geq 0, m = 1, \dots, M; YY^T = I. \end{aligned} \quad (8)$$

Eq.(8) is still non-convex due to orthogonal constraint $YY^T = I$. Fortunately with either W , b or \hat{K} , $E^{(m)}$ fixed, the problem is convex with respect to the other variables. Therefore, we present an alternating optimization way that can efficiently find the optimum in a few steps. First, given \hat{K} and $E^{(m)}$, we show that computation expressions of W and b can be obtained. To compute \hat{K} and $E^{(m)}$, we employ an efficient optimization technique, the inexact augmented Lagrange multiplier (ALM) algorithm [Lin *et al.*, 2010].

4 Optimization

4.1 Compute W and b

With other variables fixed, and setting the derivative of Eq.(8) w.r.t. b to zero, we get

$$\begin{aligned} \mathbf{1}^T (\hat{K}^T W + \mathbf{1}b - Y) &= 0 \\ \Rightarrow b &= \frac{1}{N} (\mathbf{1}^T Y - \mathbf{1}^T \hat{K}^T W). \end{aligned} \quad (9)$$

Setting the derivative of Eq.(8) w.r.t. W to zero, we yield

$$\hat{K}(\hat{K}^T W + \mathbf{1}b - Y) + \delta W = 0. \quad (10)$$

Substituting b in Eq.(9) into Eq.(10), we have

$$\begin{aligned} \hat{K} \hat{K}^T W + \hat{K} \mathbf{1} \left(\frac{1}{N} (\mathbf{1}^T Y - \mathbf{1}^T \hat{K}^T W) \right) - \hat{K} Y &= 0 \\ \Rightarrow W &= (\hat{K} L_c \hat{K}^T + \delta I)^{-1} + \hat{K} L_c Y, \end{aligned} \quad (11)$$

where $L_c = I - \frac{1}{N} \mathbf{1} \mathbf{1}^T$ is the centering matrix, and $L_c = L_c^T = L_c L_c^T$.

4.2 Compute \hat{K} and $E^{(m)}$

With variables W and b being fixed, the problem turns to be

$$\begin{aligned} \min_{\hat{K}, E^{(m)}} & \alpha \|\hat{K}\|_* + \lambda \sum_{m=1}^M \|E^{(m)}\|_{2,1}, \\ \text{s.t. } & K^{(m)} = \hat{K} + E^{(m)}, \hat{K} \geq 0, m = 1, \dots, M. \end{aligned} \quad (12)$$

The rank minimization problem has been well studied in literature [Liu *et al.*, 2010a; Wright *et al.*, 2009b]. By introducing an auxiliary variable Q such that $\hat{K} = Q$, Eq.(12) can be then converted into the following equivalent form:

$$\begin{aligned} \mathcal{D}(\hat{K}, Q, E^{(m)}) &= \alpha \|Q\|_* + \lambda \sum_{m=1}^M \|E^{(m)}\|_{2,1} \\ &+ \sum_{m=1}^M \left(\langle A^{(m)}, \hat{K} + E^{(m)} - K^{(m)} \rangle + \frac{\mu}{2} \|\hat{K} + E^{(m)} - K^{(m)}\|_F^2 \right) \\ &+ \langle B, \hat{K} - Q \rangle + \frac{\mu}{2} \|\hat{K} - Q\|_F^2, \end{aligned} \quad (13)$$

where $A^{(m)}$ and B represent the Lagrange multipliers, $\langle \cdot, \cdot \rangle$ denotes the inner product of matrices, and $\mu > 0$ is an adaptive penalty parameter. Next we will elaborate the update rules for each of \hat{K} , Q , and $E^{(m)}$ by minimizing \mathcal{D} while fixing the others.

Solving for Q When the other variables are fixed, the sub-problem w.r.t. Q is

$$\min_Q \|Q\|_* + \frac{\mu}{2\alpha} \|\hat{K} - Q + \frac{B}{\mu\alpha}\|_F^2. \quad (14)$$

It can be solved by the Singular Value Threshold method [Cai *et al.*, 2010]. More specifically, let $U\Sigma V^T$ be the SVD form of $(\hat{K} + \frac{B}{\mu\alpha})$, the updating rule of Q using the SVD operator in each iteration will be

$$Q = U \mathcal{S}_{1/\mu\alpha}(\Sigma) V^T, \quad (15)$$

where $\mathcal{S}_\varrho(x) = \max(x - \varrho, 0) + \min(x + \varrho, 0)$ is the shrinkage operator [Lin *et al.*, 2015].

Solving for $E^{(m)}$ The subproblem with respect to $E^{(m)}$, ($m = 1, \dots, M$) can be simplified as

$$\min_{E^{(m)}} \lambda \|E^{(m)}\|_{2,1} + \frac{\mu}{2} \|E^{(m)} - (K^{(m)} - \hat{K} - \frac{A^{(m)}}{\mu})\|_F^2, \quad (16)$$

which enjoys a closed form solution $E^{(m)} = \mathcal{S}_{\lambda/\mu}(K^{(m)} - \hat{K} - \frac{A^{(m)}}{\mu})$.

Solving for \hat{K} With the other variables being fixed, we update \hat{K} by solving

$$\hat{K} = \arg \min_{\hat{K}} \|\hat{K} + E^{(m)} - K^{(m)} + \frac{A^{(m)}}{\mu}\|_F^2 + \frac{\mu}{2} \|\hat{K} - Q + \frac{B}{\mu}\|_F^2 \quad (17)$$

For ease of representation, we define $C = \frac{1}{M}(Q - \frac{B}{\mu} + \sum_{m=1}^M (K^{(m)} - E^{(m)} - \frac{A^{(m)}}{\mu}))$. Then, the problem in Eq.(17) can be rewritten as

$$\hat{K} = \arg \min_{\hat{K}} \frac{1}{2} \|\hat{K} - C\|_F^2 = \arg \min_{\hat{K}_1, \dots, \hat{K}_N} = \frac{1}{2} \sum_{i=1}^N \|\hat{K}_i - C_i\|_F^2$$

$$s.t. \hat{K} \geq 0, \hat{K}_{i(i=1, \dots, N)} \geq 0. \quad (18)$$

Hence, the problem in Eq.(18) can be decomposed into N independent subproblems: $\min_{\hat{K}_i} \frac{1}{2} \|\hat{K}_i - C_i\|_F^2$, subject to $\hat{K}_i \geq 0$. Each subproblem is a proximal operator problem, which can be efficiently solved by the projection algorithm in [Duchi *et al.*, 2008].

4.3 Learning Hash Codes

Once the hashing function implemented by W and b is learned by exploiting the kernelized similarity consensus \hat{K} , we can generate hash codes for both database and query samples, denoted as x_t , via Eq. (19).

$$y_t = \text{sign}(W^T [\mathcal{K}(x_t, Z_1), \dots, \mathcal{K}(x_t, Z_R)]^T + b), \quad (19)$$

where $W \in \mathbb{R}^{R \times P}$, $\mathcal{K}(x_t, Z_i)$ represents the similarity between x_t and the i -th landmark using Gaussian RBF kernel over the concatenated feature space for all views.

4.4 Out-of-Sample Extension

An essential part of hashing is to generate binary codes for new samples, which is known as out-of-sample problems. A widely used solution is the Nyström extension [Bengio *et al.*, 2004]. However, this is impractical for large-scale hashing since the Nyström extension is as expensive as doing exhaustive nearest neighbor search with a complexity of $\mathcal{O}(N)$ for N data points. In order to address the out-of-sample extension problem, we employ a non-parametric regression approach, inspired by Shen *et al.* [Shen *et al.*, 2013]. Specifically, given the hashing embedding $Y = \{y_1, y_1, \dots, y_N\}$ for the entire training set $X = \{x_1, x_2, \dots, x_N\}$, for a new data point x_q , we aim to generate a hashing embedding y_q while preserving the local neighborhood relationships among its neighbors $\mathcal{N}_k(x_q)$ in X . A simple inductive formulation can produce the embedding for a new data point by a sparse linear combination of the base embeddings:

$$y_q = \frac{\sum_{i=1}^N w(x_q, x_i) y_i}{\sum_{i=1}^N w(x_q, x_i)}, \quad (20)$$

where we define

$$w(x_1, x_i) = \begin{cases} \exp(-\|x_q - x_i\|^2 / \sigma^2), & x_i \in \mathcal{N}_k(x_q) \\ 0, & \text{elsewhere.} \end{cases}$$

However, Eq.(20) does not scale well for computing out-of-sample extension ($\mathcal{O}(N)$) for large-scale tasks. To this end, we employ a prototype algorithm [Shen *et al.*, 2013] to approximate y_q using only a small base set:

$$h(x_q) = \text{sign} \left(\frac{\sum_{j=1}^Z w(x_q, c_j) y_j}{\sum_{j=1}^Z w(x_q, c_j)} \right) \quad (21)$$

where $\text{sign}(\cdot)$ is the sign function, and $Y = \{y_1, y_2, \dots, y_Z\}$ is the hashing embedding for the base set $B = \{c_1, c_2, \dots, c_Z\}$ which is the cluster centers obtained by K-means. In this stage, the major computation cost comes from K-means clustering, which is $\mathcal{O}(dlZN)$ in time (d is the feature dimension, and l is the number of iterations in K-means). The iteration number l can be set less than 50, thus, the K-means only costs $\mathcal{O}(dZN)$. Considering that Z is much less than N , the total time is linear in the size of training set. The computation of distance between B and X cost $\mathcal{O}(dZN)$. Thus, the overall time cost is $\mathcal{O}(dZN + dZN) = \mathcal{O}(dZN)$.

5 Complexity Analysis

We analyze the time complexity regarding per iteration of the optimization strategy. The complexity of computing $\hat{K} L_c \hat{K}^T$ and $(\hat{K} L_c \hat{K}^T + \delta I)^{-1}$ in Eq.(11) is $\mathcal{O}(R^2 N)$ and $\mathcal{O}(R^3)$, respectively. Commonly, $R(\ll N)$ landmarks are generated off-line via scalable K-means clustering for less than 50 iterations, keeping the complexity of computing W to be $\mathcal{O}(R^2 N) + \mathcal{O}(R^3)$. The complexity of computing hash codes for a new sample is $\mathcal{O}(dZN)$. Overall, the time complexity is $\mathcal{O}(dZN + R^3 + R^2 N) \approx \mathcal{O}(dZN + R^2 N)$ in one iteration, which is linear with respect to the training size.

6 Experiments

6.1 Experimental Settings

Competitors We compare our method with recently proposed state-of-the-art multiple feature hashing algorithms:

- Multiple feature hashing (**MFH**) [Song *et al.*, 2011]: This method exploits local structure in each feature and global consistency in the optimization of hashing functions.
- Composite hashing with multiple sources (**CHMS**) [Zhang *et al.*, 2011]: This method treats a linear combination of view-specific similarities as an average similarity which can be plugged into a spectral hashing framework.
- Compact kernel hashing with multiple features (**CKH**) [Liu *et al.*, 2012b]: It is a multiple feature hashing framework where multiple kernels are linearly combined.
- Sequential spectral hashing with multiple representations (**SSH**) [Kim *et al.*, 2012]: This method constructs an average similarity matrix to assemble view-specific similarity matrices.
- Multi-View Latent Hashing (**MVLH**) [Shen *et al.*, 2015]: This is an unsupervised multi-view hashing approach where binary codes are learned by the latent factors shared by multiple views from an unified kernel feature space.

Datasets We conduct the experiments on two image benchmarks: CIFAR-10³ and NUS-WIDE.

³<http://www.cs.toronto.edu/~kriz/cifar.html>

- **CIFAR-10** consists of 60K 32×32 color images from ten object categories, each of which contains 6K samples. Every image is assigned to a mutually exclusive class label and for each image, we extract 512-dimensional GIST feature [Oliva and Torralba, 2001] and 300-dimensional bag-of-words quantized from dense SIFT features [Lowe, 2004] to be *two views*.
- **NUS-WIDE** [Chua *et al.*, 2009] contains 269,648 labeled images crawled from Flickr and is manually annotated with 81 categories. Three types of features are extracted: 128-dimensional wavelet texture, 225-dimensional block-wise color moments, and 500-dimensional bag-of-words to construct *three views*.

Multi-view Corruption Setting In **CIFAR-10**, considering that missing features may have some structure, we remove a square patch of pixels from each image covering 25% of the total number of pixels. The location of the patch is uniformly sampled for each image. This will naturally deteriorate view-dependent feature representations. In **NUS-WIDE**, we consider the scenario where 20% of feature values in each view are corrupted with perturbation noise following a standard Gaussian distribution.

Parameter Setting In the training phase, we uniformly sample 30K and 100K images as training data from both datasets, and generate 300 and 500 landmarks. That is, we fix the graph construction parameters $L = 300$, $k = 3$ on CIFAR-10, and $L = 500$, $k = 5$ on NUS-WIDE, respectively. In the testing phase, we randomly select 1,000 query images in which the true neighbors of each image are defined as the semantic neighbors which share at least one common semantic label. For our method and **CKH**, we use Gaussian RBF kernel $K^{(i)}(x, y) = \exp(-\|x - y\|_i^2 / 2\sigma^2)$ ($i = 1, \cdot, M$), where $\|\cdot\|_i^2$ represents the Euclidean distance within the i -th feature space. The parameter σ is learned via the self-tuning strategy [Zelnik-Manor and Perona, 2004].

In Eq.(8), there are five tunable parameters: γ , δ , α , β , and λ . Parameters γ and δ controlling global hash code learning and regularization on hashing functions are set as 10^{-4} and 10^{-6} , respectively. For α , β , and λ , we tune their optimal combination, that is, $\alpha = 10^{-1}$, $\beta = 10^0$, and $\lambda = 10^{-3}$, as conducted in section 6.3.

Evaluation Metric The mean precision-recall and mean average precision (MAP) are computed over the retrieved set consisting of the samples with the hamming distance [Liu *et al.*, 2012a] using 8 to 32 bits to a specific query. We carry out hash lookup within a Hamming radius 2 and report the mean hash lookup precision over all queries. For a query q , the average precision (AP) is defined as $AP(q) = \frac{1}{L_q} \sum_{z=1}^l P_q(z) \varpi_q(z)$, where L_q is the number of ground-truth neighbors of q in database, l is the number of entities in database, $P_q(z)$ denotes the precision of the top z retrieved entities, and $\varpi_q(z) = 1$ if the z -th retrieved entity is a ground-truth neighbor and $\varpi_q(z) = 0$, otherwise. Ground truth neighbors are defined as items which share at

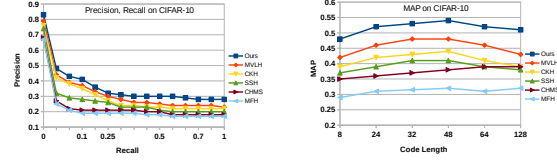


Figure 1: Performance comparison on CIFAR-10 database. Left: Mean precision-recall of Hamming ranking at 48 bits. Right: Mean average precision of Hamming ranking w.r.t. 8-128 bits.

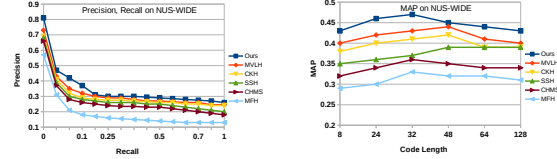


Figure 2: Performance comparison on NUS-WIDE database. Left: Mean precision-recall of Hamming ranking at 64 bits. Right: Mean average precision of Hamming ranking w.r.t. 8-128 bits.

least one semantic label. Given a query set of size F , the MAP is defined as the mean of the average precision for all queries: $MAP = \frac{1}{F} \sum_{i=1}^F AP(q_i)$.

6.2 Results

We report the mean precision-recall curves of Hamming ranking, and mean average precision (MAP) w.r.t. different number of hashing bits over 1K query images. Results are shown in Fig.1, which are computed from top-100 retrieved samples. It can be seen from top subfigure of Fig.1 that our method achieves a performance gain in both precision and recall over all counterparts and the second best is **MVLH**. This can demonstrate the superiority of using nonlinear hashing functions in nonlinear space. More importantly, the latent consensus kernelized similarity matrix by low-rank minimization is not only effective in leveraging complementary information from multi-views, but also robust against the presence of errors. The subfigure (bottom) in Fig.1 shows that as the hashing bit number varies, our method consistently keeps superior performance. Specifically, it reaches the highest precision value for 48 bits and shows a relatively steady performance with more hashing bits. The results from the NUS-WIDE database are shown in Fig.2. Once again we can see performance gaps in precision-recall between our approach and competitors, as illustrated in top subfigure of Fig.2. This validates the advantage of our method by exploiting consensus of kernelized similarity to learn robust nonlinear hashing functions. In subfigure (bottom) of Fig.2, as the number of hashing bit increases, our method is able to keep high and steady MAP values.

To evaluate the impact of hashing bit numbers on performance of hash lookup, in Table 1, we report hash lookup mean precision with standard deviation (mean \pm std) in the case of 8, 32, 48, 128 bits on both databases. Similar to

Table 1: Hash lookup precision (mean \pm std) with Hamming radius 2 on different databases.

Method	CIFAR-10				NUS-WIDE			
	P=8	P=32	P=48	P=128	P=8	P=32	P=48	P=128
MFH	23.31 \pm 0.71	28.19 \pm 0.48	26.38 \pm 0.68	23.68 \pm 0.71	23.52 \pm 0.72	26.49 \pm 0.85	33.55 \pm 0.49	34.97 \pm 0.81
CHMS	25.61 \pm 0.22	31.8 \pm 0.66	26.54 \pm 0.52	19.38 \pm 0.84	27.54 \pm 0.41	30.22 \pm 0.92	28.24 \pm 0.96	27.52 \pm 1.12
CKH	31.75 \pm 0.53	32.05 \pm 0.72	37.32 \pm 0.76	34.45 \pm 0.81	29.72 \pm 0.43	37.84 \pm 0.63	33.56 \pm 0.82	34.42 \pm 1.32
SSH	27.34 \pm 0.46	35.78 \pm 0.68	29.36 \pm 0.63	27.52 \pm 0.72	28.95 \pm 0.46	33.42 \pm 0.88	30.05 \pm 0.71	29.21 \pm 0.98
MVLH	32.27 \pm 0.41	40.24 \pm 0.63	44.81 \pm 0.46	42.06 \pm 0.62	31.92 \pm 0.62	39.05 \pm 0.87	40.31 \pm 0.52	36.12 \pm 0.70
Ours	36.73\pm0.41	47.63\pm0.52	51.22\pm0.36	46.57\pm0.44	34.21\pm0.48	46.35\pm0.47	44.33\pm0.34	43.08\pm0.32

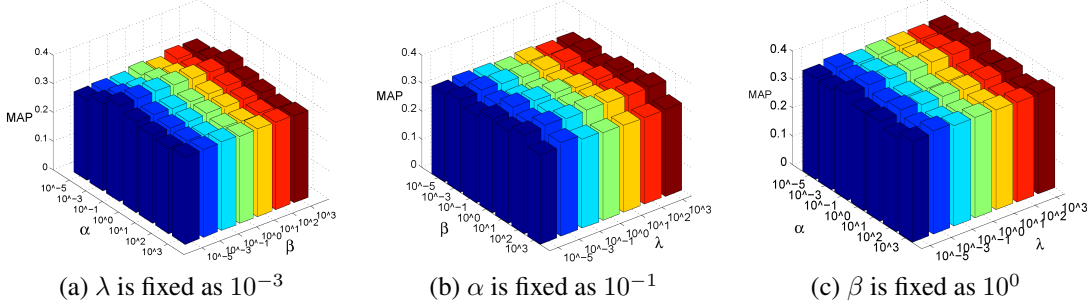


Figure 3: The MAP variations of different parameter settings on NUS-WIDE database.

Table 2: Training/test time comparison on different algorithms using 64 bits. All training/test time is recorded in second. The training size of two datasets are 30K and 100K, respectively.

Method	CIFAR-10		NUS-WIDE	
	Training	Test	Training	Test
MFH	32.8	6.4×10^{-5}	41.6	8.5×10^{-5}
CHMS	29.8	4.7×10^{-5}	37.2	7.8×10^{-5}
SSH	23.6	1.3×10^{-5}	31.7	2.4×10^{-5}
CKH	10.7	2.3×10^{-6}	15.3	3.2×10^{-6}
MVLH	20.4	2.2×10^{-6}	28.1	4.3×10^{-6}
Ours	14.1	2.6×10^{-6}	19.2	3.5×10^{-6}

Hamming ranking results, our method achieves the better performance than others and obviously increasing performance with less than 32 bits, which demonstrates that our approach with compact hashing codes can retrieve more semantically related images than all baselines in terms of hash lookup.

In Table 2, we report the comparison on training/test time over the two image benchmarks. **CKH** and our method are much more efficient by taking less than 15s and 20s respectively to train on CIFAR-10 and NUS-WIDE using 32 bits. The efficiency improvement comes from the usage of landmarks. While our method is slightly less efficient to **CKH** because of the low-rank kernelized similarity recovery, it is very comparable to **CKH** and consistently superior to **CKH** in other performance. **MVLH** is relatively costly due to its expensive matrix factorization in its kernel space. **MFH** and **CHMS** are time-consuming in training stage because they both involve the eigen-decomposition of a dense affinity matrix, which is not scalable to a large-scale setting. **SSH** has a gain in efficiency compared with **MFH** and **CHMS** on account of their approximation on the K-nearest graph construction [Kim *et al.*, 2012].

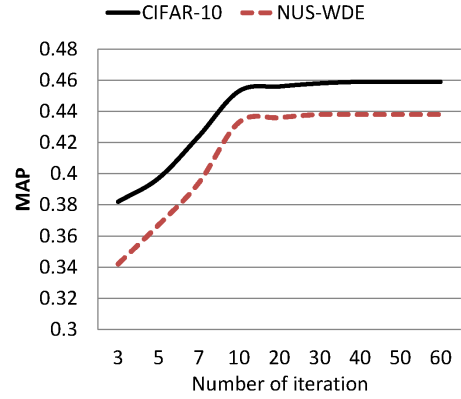


Figure 4: Convergence study over real-world datasets.

6.3 Parameter Tuning

In this experiment, we test different parameter settings for our algorithm to study the performance sensitivity. We learn three parameters: α , λ , and β , corresponding to the term of requisite component, non-requisite decomposition, and hashing function learning in Eq.(8). For these parameters, we tune them from $\{10^{-5}, 10^{-3}, 10^{-1}, 10^0, 10^1, 10^2, 10^3\}$. We fix one of the parameters in α , λ , and β to report the MAP while the other two parameters are changing. The results are shown in Fig.3. In Fig.3 (a), by fixing $\lambda = 10^{-3}$, we show the performance variance on different pairs of α and β . We can observe that our algorithms achieves a relatively higher MAP when $\alpha = 0.1$, and $\beta = 10^0$. The similar performance can also be seen from Fig.3 (b) and Fig.3 (c). Thus, among different combinations, the method gains the best performance when $\alpha = 10^{-1}$, $\beta = 1$, and $\lambda = 10^{-3}$, while it is relatively insensitive to varied parameters setting. With optimal combi-

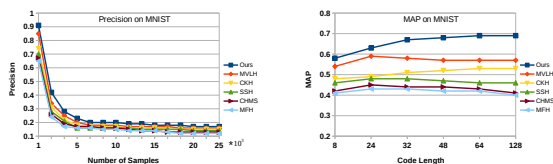


Figure 5: Performance comparison on MNIST database using base hash functions learn from CIFAR-10 dataset. Left: Precision with respect to number of returned sampled at 64 bits. Right: Mean average precision of Hamming ranking w.r.t. 8-128 bits.

nation of parameters, we study the issue of convergence. In Fig.4, we can observe that our algorithm becomes convergent in less than 40 iterations, demonstrating its fast convergence rate.

6.4 Out-of-Sample Case

In this experiment, we study the property of out-of-sample extension. We take the CIFAR-10 dataset as the base benchmark to train base embeddings. Another dataset MNIST is considered as the testing bed. The MNIST dataset [LeCun *et al.*, 1998] consists of 70K images, each of 784 dimensions, of handwritten digits from “0” to “9”. As in Fig.5, our method achieves the best results. On this dataset, we can clearly see that our method outperforms **MVLH** by a large margin, which increases as code length increases. This further demonstrates the advantage of kernelized low-rank embedding as a tool for hashing by embedding high dimensional data into a lower dimensional space. This dimensionality reduction procedure not only preserves the local neighborhood, but also reveals global structure.

7 Conclusion

In this paper, we motivate the problem of robust hashing for similarity search over multi-view data objects under a practical scenario that error corruptions for view-dependent feature representations are presented. Unlike existing multi-view hashing methods that take a two-phase scheme of constructing similarity matrices and learning hash functions separately, we propose a novel technique to jointly learn hash functions and a latent, low-rank, corruption-free kernelized similarity under multiple representations with potential noise corruptions. Extensive experiments conducted on real-world multi-view data sets demonstrate the superiority of our method in terms of efficacy.

References

[Bengio *et al.*, 2004] Yoshua Bengio, Olivier Delalleau, Nicolas Le Roux, Jean-Francois Paiement, Pascal Vincent, and Marie Ouimet. Learning eigenfunctions links spectral embedding and kernel pca. *Neural Comput.*, 16(10):2197–2219, 2004.

[Cai *et al.*, 2010] Jian-Feng Cai, Emmanuel J. Cands, and Zuowei Shen. A singular value thresholding algorithm

for matrix completion. *SIAM Journal on Optimization*, 20(4):1957–1982, 2010.

[Candes and Recht, 2009] Emmanuel J. Candes and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

[Chua *et al.*, 2009] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, and Zhiping Luo. Nus-wide: a real-world web image database from national university of singapore. In *ACM CIVR*, 2009.

[Datar *et al.*, 2004] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distribution. In *SOCG*, 2004.

[Deng *et al.*, 2013] Yue Deng, Qionghai Dai, Risheng Liu, Zengke Zhang, and Sanqing Hu. Low-rank structure learning via nonconvex heuristic recovery. *IEEE Transactions on Neural Networks and Learning Systems*, 24(3):383–396, 2013.

[Duchi *et al.*, 2008] John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the ℓ_1 -ball for learning in high dimensions. In *ICML*, 2008.

[Kim *et al.*, 2012] Saehoon Kim, Yoonseop Kang, and Seungjin Choi. Sequential spectral learning to hash with multiple representations. In *ECCV*, pages 538–551, 2012.

[Kulis and Grauman, 2009] Brian Kulis and Kristen Grauman. Kernelized locality-sensitive hashing for scalable image search. In *ICCV*, 2009.

[Kumar and III, 2011] Abhishek Kumar and Hal Daume III. A co-training approach for multi-view spectral clustering. In *ICML*, 2011.

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

[Kumar *et al.*, 2011] Abhishek Kumar, Piyush Rai, and Hal Daum. Co-regularized multi-view spectral clustering. In *NIPS*, 2011.

[LeCun *et al.*, 1998] Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of IEEE*, 1998.

[Li *et al.*, 2015] Sheng Li, Ming Shao, and Yun Fu. Multi-view low-rank analysis for outlier detection. In *SIAM Data Mining*, pages 748–756, 2015.

[Lin *et al.*, 2010] Zhouchen Lin, Minming Chen, and Yi Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. In *arXiv:1009.5055*, 2010.

[Lin *et al.*, 2015] Zhouchen Lin, Risheng Liu, and Huan Li. Linearized alternating direction method with parallel splitting and adaptive penalty for separable convex programs in machine learning. *Machine Learning*, (2):287–325, 2015.

- [Liu *et al.*, 2010a] Guangcai Liu, Zhuochen Lin, and Yong Yu. Robust subspace segmentation by low-rank representation. In *ICML*, 2010.
- [Liu *et al.*, 2010b] Wei Liu, Jun Wang, and Shih-Fu Chang. Large graph construction for scalable semi-supervised learning. In *ICML*, 2010.
- [Liu *et al.*, 2011] Wei Liu, Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. Hashing with graphs. In *ICML*, 2011.
- [Liu *et al.*, 2012a] Wei Liu, Jun Wang, Rongrong Ji, Yugang Jiang, and Shih-Fu Chang. Supervised hashing with kernels. In *CVPR*, pages 2074 – 2081, 2012.
- [Liu *et al.*, 2012b] Xianglong Liu, Junfeng He, Di Liu, and Bo Lang. Compact kernel hashing with multiple features. In *ACM Multimedia*, pages 881–884, 2012.
- [Liu *et al.*, 2013] Guangcan Liu, Zhuochen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(1):171–184, 2013.
- [Lowe, 2004] David Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–110, 2004.
- [Masci *et al.*, 2014] Jonathan Masci, Michael M. Bronstein, Alexander M. Bronstein, and Jurgen Schmidhuber. Multimodal similarity-preserving hashing. *IEEE TPAMI*, 36(4):824–830, 2014.
- [Oliva and Torralba, 2001] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *IJCV*, 42(3):145–175, 2001.
- [Ou *et al.*, 2013] Mingdong Ou, Peng Cui, Fei Wang, Jun Wang, Wenwu Zhu, and Shiqiang Yang. Comparing apples to oranges: a scalable solution with heterogeneous hashing. In *ACM SIGKDD*, pages 230–238, 2013.
- [Shen *et al.*, 2013] Fumin Shen, Chunhua Shen, Qinfeng Shi, Anton van den Hengel, and Zhenmin Tang. Inductive hashing on manifolds. In *CVPR*, pages 1562 – 1569, 2013.
- [Shen *et al.*, 2015] Xiaobo Shen, Fumin Shen, Quan-Sen Sun, and Yun-Hao Yuan. Multi-view latent hashing for efficient multimedia search. In *ACM Multimedia*, pages 831–834, 2015.
- [Song *et al.*, 2011] Jingkuan Song, Yi Yang, Zi Huang, Heng-Tao Shen, and Richang Hong. Multiple feature hashing for real-time large scale near-duplicate video retrieval. In *ACM Multimedia*, pages 423–432, 2011.
- [Wang *et al.*, 2010] Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. Semi-supervised hashing for scalable image retrieval. In *CVPR*, pages 3424 – 3431, 2010.
- [Wang *et al.*, 2013] Yang Wang, Xuemin Lin, and Qing Zhang. Towards metric fusion on multi-view data: a cross-view based graph random walk approach. In *ACM CIKM*, pages 805–810, 2013.
- [Wang *et al.*, 2014] Yang Wang, Xuemin Lin, Lin Wu, Wenjie Zhang, and Qing Zhang. Exploiting correlation consensus: Towards subspace clustering for multi-modal data. In *ACM Multimedia*, pages 981–984, 2014.
- [Wang *et al.*, 2015a] Qifan Wang, Luo Si, and Bin Shen. Learning to hash on partial multi-modal data. In *IJCAI*, pages 3904–3910, 2015.
- [Wang *et al.*, 2015b] Yang Wang, Xuemin Lin, Lin Wu, and Wenjie Zhang. Effective multi-query expansions: Robust landmark retrieval. In *ACM Multimedia*, pages 79–88, 2015.
- [Wang *et al.*, 2015c] Yang Wang, Xuemin Lin, Lin Wu, Wenjie Zhang, and Qing Zhang. Lbmch: Learning bridging mapping for cross-modal hashing. In *ACM SIGIR*, 2015.
- [Wang *et al.*, 2015d] Yang Wang, Xuemin Lin, Lin Wu, Wenjie Zhang, Qing Zhang, and Xiaodi Huang. Robust subspace clustering for multi-view data by exploiting correlation consensus. *IEEE Transactions on Image Processing*, 24(11):3939–3949, 2015.
- [Wang *et al.*, 2015e] Yang Wang, Wenjie Zhang, Lin Wu, Xuemin Lin, and Xiang Zhao. Unsupervised metric fusion over multiview data by graph random walk-based cross-view diffusion. *IEEE Transactions on Neural Networks and Learning System*, 99:1–14, 2015.
- [Wang *et al.*, 2016a] Yang Wang, Xuemin Lin, Lin Wu, Qing Zhang, and Wenjie Zhang. Shifting multi-hypergraphs via collaborative probabilistic voting. *Knowledge and Information Systems*, 46(3):515–536, 2016.
- [Wang *et al.*, 2016b] Yang Wang, Wenjie Zhang, Lin Wu, Xuemin Lin, Meng Fang, and Shirui Pan. Iterative views agreement: An iterative low-rank based structured optimization method to multi-view spectral clustering. In *IJCAI*, 2016.
- [Wei *et al.*, 2014] Ying Wei, Yangqiu Song, Yi Zhen, Bo Liu, and Qiang Yang. Scalable heterogeneous translated hashing. In *ACM SIGKDD*, pages 791–800, 2014.
- [Weiss *et al.*, 2008] Yair Weiss, Antonio Torralba, and Rob Fergus. Spectral hashing. In *NIPS*, 2008.
- [Wright *et al.*, 2009a] John Wright, Yigang Peng, Yi Ma, Arvind Ganesh, and Shankar Rao. Robust principal component analysis: Exact recovery of corrupted low-rank matrices by convex optimization. In *NIPS*, 2009.
- [Wright *et al.*, 2009b] John Wright, Yigang Peng, Yi Ma, Arvind Ganesh, and Shankar Rao. Robust principal component analysis: exact recovery of corrupted low-rank matrices via convex optimization. In *NIPS*, 2009.
- [Wu *et al.*, 2013] Lin Wu, Yang Wang, and John Shepherd. Efficient image and tag co-ranking: a bregman divergence optimization method. In *ACM Multimedia*, 2013.
- [Wu *et al.*, 2016] Lin Wu, Yang Wang, and Shirui Pan. Exploiting attribute correlations: A novel trace lasso-based weakly supervised dictionary learning method. *IEEE Transactions on Cybernetics*, 2016.
- [Xia *et al.*, 2014] Rongkai Xia, Yan Pan, Lei Du, and Jian Yin. Robust multi-view spectral clustering via low-rank and sparse decomposition. In *AAAI*, pages 2149–2155, 2014.

- [Xu *et al.*, 2013] Chang Xu, Dacheng Tao, and Chao Xu. A survey on multi-view learning. *arXiv:1304.5634*, 2013.
- [Ye *et al.*, 2012] Guangnan Ye, Dong Liu, I-Hong Jhuo, and Shih-Fu Chang. Robust late fusion with rank minimization. In *CVPR*, pages 3021–3028, 2012.
- [Zelnik-Manor and Perona, 2004] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *NIPS*, 2004.
- [Zhang *et al.*, 2011] Dan Zhang, Fei Wang, and Luo Si. Composite hashing with multiple information sources. In *ACM SIGIR*, pages 225–234, 2011.
- [Zhang *et al.*, 2014] Zhao Zhang, Shuicheng Yan, and Mingbo Zhao. Similarity preserving low-rank representation for enhanced data representation and effective subspace learning. *Neural Networks*, 53:81–94, 2014.
- [Zhang *et al.*, 2015] Zhao Zhang, Shuicheng Yan, Mingbo Zhao, and Fanzhang Li. Bilinear low-rank coding framework and extension for robust image recovery and feature representation. *Knowledge-Based Systems*, 86:143–157, 2015.
- [Zhang *et al.*, 2016] Zhao Zhang, Fanzhang Li, Mingbo Zhao, Li Zhang, and Shuicheng Yan. Joint low-rank and sparse principal feature coding for enhanced robust representation and visual classification. *IEEE Transactions on Image Processing*, 25(6):2429–2443, 2016.
- [Zheng *et al.*, 2015] Shuai Zheng, Xiao Cai, Chris Ding, Feiping Nie, and Heng Huang. A closed form solution to multi-view low-rank regression. In *AAAI*, pages 1973–1979, 2015.
- [Zhou *et al.*, 2013] Xiaowei Zhou, Can Yang, and Weichuan Yu. Moving object detection by detecting contiguous outliers in the low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(3):597–610, 2013.