Facial Image Restoration and Retrieval Through Orthogonality

Dayong Tian

Faculty of Engineering and Information Technology

University of Technology Sydney

A thesis submitted for the degree of

Doctor of Philosophy

2017

To my loving parents
Jubao Tian and *Shuxia Song*
and my wife
Yiwen Wei

Certificate of Original Authorship

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Dayong Tian

Acknowledgements

I would like to express my special appreciation and thanks to my supervisor Professor Dacheng Tao. He accepted my application three years ago and gave me the chance to change my life. In last three years, he consistently taught me how to research. From reading literatures to writing my own papers, from implementing others' methods to devising my own methods, I learned a lot under his instructions.
I also wish to give special thanks to Chaoyue Wang, Guoliang Kang and Jiang Bian. Without their helps, my life in Sydney would not be so easy. I am grateful to Baosheng Yu and Zhe Chen. Our work stations are neighboring. Their opinions deeply impressed me. I would like to give my gratitude to Changxing Ding, Kede Ma, Maoying Qiao and Tongliang Liu for their helps on my research.
I would like to give thanks to my friends I met in Conversation@UTS, especially to Sakada Ko. Ko is an exchange student from Kyoto University of Foreign Studies. He shared lots of his experiences on learning English to me.
It is my fortune to met my friends, Archer, Berry, Beryl, Billy, Carrie, Celine, Chain, CK, Danny, David, Emma, Peyton, Rhonda, Vivi, Vivian, William, Wilton and Zoe. I cannot remember how may times Berry and Chain mixed drinks for us, how many times I was drunk with Rhonda and how many times Zoe drove me home. It is them who make my life diverse and give me completely different experiences. I will miss every minute spent with them.
Finally, I would like to express my gratitude to my family, my parents and my wife, for their encouragement and support.

Abstract

Orthogonality has different definitions in geometry, statistics and calculus. This thesis studies how to incorporate orthogonality to facial image restoration and retrieval tasks. A facial image restoration method and three retrieval methods were proposed.

Blur in facial images significantly impedes the efficiency of recognition approaches. However, most existing blind deconvolution methods cannot generate satisfactory results, due to their dependence on strong edges which are sufficient in natural images but not in facial images. A novel method is proposed in this report. Point spread functions (PSF) are represented by the linear combination of a set of pre-defined orthogonal PSFs and similarly, an estimated intrinsic sharp face image (EI) is represented by the linear combination of a set of pre-defined orthogonal face images. In doing so, PSF and EI estimation is simplified to discovering two sets of linear combination coefficients which are simultaneously found by the proposed coupled learning algorithm. To make the method robust to different kinds of blurry face images, several candidate PSFs and EIs are generated for a test image, and then a non-blind deconvolution method is adopted to generate more EIs by those candidate PSFs. Finally, a blind image quality assessment metric is deployed to automatically select the optimal EI.

On the other hand, the orthogonality is incorporated into the proposed Unimodal image retrieval method. Hashing methods have been widely investigated for fast approximate nearest neighbor searching in large datasets. Most existing methods use binary vectors in lower dimensional spaces to represent data points that are usually real vectors of higher dimensionality. The proposed method divides the hashing

process into two steps. Data points are first embedded in a low-dimensional space, and the Global Positioning System (GPS) method is subsequently introduced but modified for binary embedding. Data-independent and data-dependent methods are devised to distribute the satellites at appropriate locations. The proposed methods are based on finding the tradeoff between the information losses in these two steps. Experiments show that the data-dependent method outperforms other methods in different-sized datasets from 100K to 10M. By incorporating the orthogonality of the code matrix, both data-independent and data-dependent methods are particularly impressive in experiments on longer bits.

In social networks, heterogeneous multimedia data correlates to each other, such as videos and their corresponding tags in YouTube and image-text pairs in Facebook. Nearest neighbor retrieval across multiple modalities on large data sets becomes a hot yet challenging problem. Hashing is expected to be an efficient solution, since it represents data as binary codes. As the bit-wise XOR operations can be fast handled, the retrieval time is greatly reduced. Few existing multi-modal hashing methods consider the correlation among hashing bits. The correlation has negative impact on hashing codes. When the hashing code length becomes longer, the retrieval performance improvement becomes slower. The proposed method incorporates a so-called minimum correlation constraint which can be treated as a generalization of orthogonality constraint. Experiments show the superiority of the proposed method becomes greater as the code length increases.

Deep neural network is expected to be an efficient way for multi-modal hashing. We propose a hybrid neural network which consists of a convolutional neural network for facial images and a full-connected neural network for tags or labels. The minimum correlation regularization is imposed on the parameters of output layers. Experiments validates the superiority of the proposed hybrid neural network.

Contents

List of Figures

Chapter 1

Introduction

In this chapter, we discuss the definitions of orthogonality in various situations and how we use orthogonality in our facial image restoration and retrieval tasks.

1.1 Background

1.1.1 Orthogonality in Geometry

In geometry, orthogonality means a pair of mutually vertical vectors. A coordinate system in a d-dimensional space is comprised of d mutually vertical vectors. For $\mathbf{v}, \mathbf{w} \in \mathbb{R}^d$, their inner product is defined as:

$$\langle \mathbf{v}, \mathbf{w} \rangle = \sum_{i=1}^{d} v_i w_i \tag{1.1}$$

where v_i and w_i are the ith element of $\mathbf{v}$ and $\mathbf{w}$, respectively. $\mathbf{v}$ is vertical to $\mathbf{w}$ when their inner product, i.e. $\langle \mathbf{v}, \mathbf{w} \rangle = 0$.

A complete set of mutually vertical vectors in a d dimensional space consists of d vectors and it is the most efficient way to describe the location of points in the space. Fig. 1.1 shows two widely used coordinate systems. The Cartesian coordinate system is used for general purpose. A famous usage of polar system is aircraft navigation.

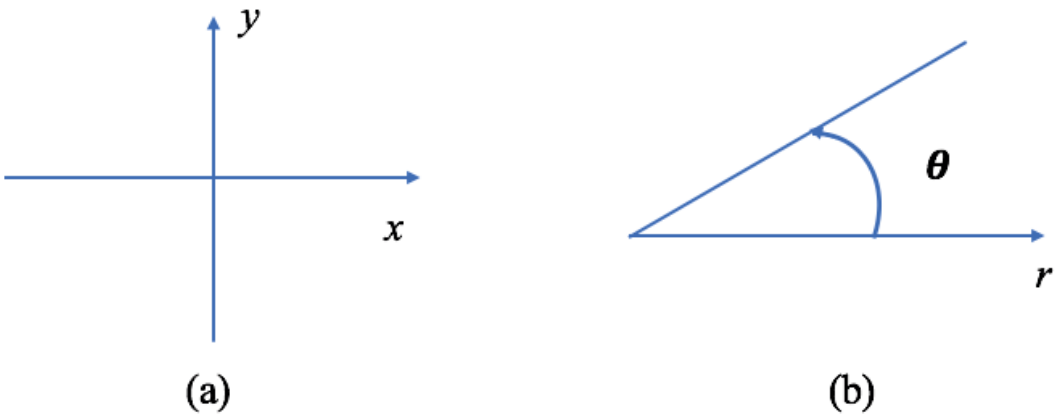


Figure 1.1: Illustration of three coordinate systems. (a) Cartesian coordinate system. (b) Polar coordinate system.

1.1.2 Orthogonality in Statistics

Principal component analysis [43] uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables. PCA is illustrated in Fig. 1.2. The first principal component has the largest possible variance and the succeeding component has the highest variance possible under the constraint that it is orthogonal to the preceding components. PCA is used in our facial image restoration and retrieval method. Consider a data matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$, where each row is a data point in d dimensional space. The columns of $\mathbf{X}$ are zero-centered. The first principal component is found by

$$\mathbf{w}_1 = \text{argmax} \left\{ \frac{\mathbf{w}^\top \mathbf{X}^\top \mathbf{X} \mathbf{w}}{\mathbf{w}^\top \mathbf{w}} \right\} \tag{1.2}$$

Further principal component can be found by subtracting the first $k-1$ principal components from $\mathbf{X}$:

$$\hat{\mathbf{X}}_k = \mathbf{X} - \sum_{s=1}^{k-1} \mathbf{X} \mathbf{w}_s \mathbf{w}_s^\top \tag{1.3}$$

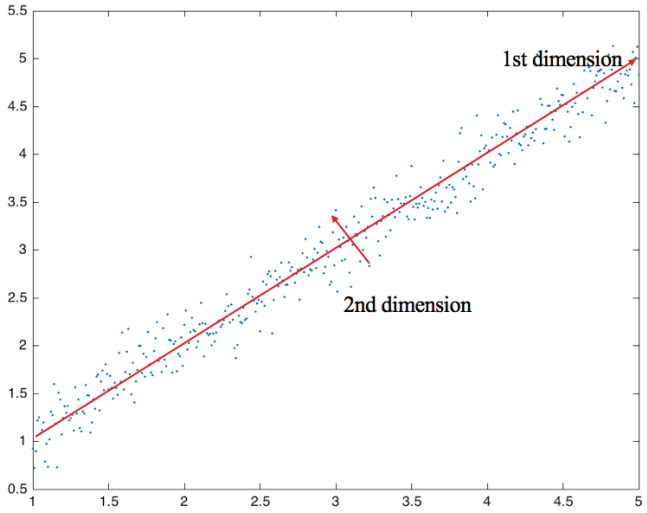


Figure 1.2: Illustration of principal component analysis on a 2-dimensional data set

and then maximize the variance of $\hat{\mathbf{X}}$

$$\mathbf{w}_k = \text{argmax} \left\{ \frac{\mathbf{w}^\top \hat{\mathbf{X}}_k^\top \hat{\mathbf{X}}_k \mathbf{w}}{\mathbf{w}^\top \mathbf{w}} \right\} \tag{1.4}$$

1.1.3 Orthogonality in Calculus

In calculus, if two functions are defined in the same interval. they are mutually orthogonal as long as the integral of the product of these two functions equals to 0, which can be formulated as:

$$\begin{aligned} \langle \phi_i, \phi_j \rangle = \int \phi_i(x)\phi_j(x)\mathrm{d}x = 0, \qquad & i \neq j \\ \langle \phi_i, \phi_j \rangle = 1, \qquad & i = j \end{aligned} \tag{1.5}$$

A piecewise continuous function can be represented by a linear combination of a complete set of orthogonal functions. Complete set contains n functions, $\phi_1, \ldots, \phi_n$, that satisfy following equation for any function f.

$$\lim_{n \to \text{inf}} \| f - (c_1\phi_1 + \ldots + c_n\phi_n) \| = 0 \tag{1.6}$$

1.1.4 Facial Image Restoration

Blur and additive noises in facial images significantly impedes the efficiency of recognition approaches. Facial blur is common in recorded face images. Examples include motion blur caused by the relative movement between the target and the camera and out-of-focus blur caused by misalignment between the target and the camera focus. In general, an observed facial image I can be modeled as

$$I = I^o * \phi + \eta \tag{1.7}$$

where I is the intrinsic sharp image, ϕ is the point spread function, η is the additive noises and $*$ is the convolution operation.
In our method, we first apply PCA on the training sharp image data set. We use a linear combination of principal components to represent the intrinsic sharp image I^o. We represent the point spread function ϕ by a linear combination of a set of 2-dimensional mutually orthogonal functions. After computing the coefficients of these two linear combinations, I^o and ϕ can be restored simultaneously.

1.1.5 Image Retrieval

Image retrieval is a kind of uni-modal retrieval. To improve the efficiency of nearest neighbor searching, hashing methods are proposed to map original input data points to binary hashing codes. As the bit-wise XOR operation can be handled fast, the retrieval speed can be greatly improved.
Generally, hashing methods embed high-dimensional real vectors to low-dimensional binary vectors. Hence, information are greatly missed during this procedure. To fully utilize each bit of the codes, researchers proposed various ways to generate an orthogonal code matrix of which each row is a hashing code and columns are orthogonal to each other. However, generating an exact orthogonal code matrix is an NP hard problem.
One of the most popular hashing method, ITerative Quantization (ITQ), circumvent this problem by formulating hashing as a minimization of quantization loss.

$$\underset{\mathbf{B},\mathbf{R}}{\operatorname{argmin}} \ |\mathbf{B} - \mathbf{XWR}|_F^2 \tag{1.8}$$

where $\mathbf{B}$ is the code matrix, $\mathbf{X}$ is the data matrix, $\mathbf{W}$ is comprised of the first c principal components, $R \in \mathbb{R}^{c \times c}$ is an orthogonal matrix and c is the code length. The major drawback of ITQ is that it cannot encode data matrix of which the dimensionality d is smaller than the code length c and it cannot generate a balanced code matrix which is considered as another constraint for good codes [115]. In our method, we modify the Global Positioning System to devise an efficient hashing method. We approximate the orthogonal code matrix by wisely distribute the satellites. Also, our method can generate a balanced code matrix.

1.1.6 Multi-modal Retrieval

In social networks, heterogeneous multimedia data correlates to each other, such as videos and their corresponding tags in YouTube and image-text pairs in Facebook. Nearest neighbor retrieval across multiple modalities on large data sets becomes a hot yet challenging problem.
We extend the ITQ to multi-modal version. As mentioned above, the ITQ does not work when $d < c$. In order to handle this problem, we proposed a minimum correlation constraint as a generalization of orthogonality constraint.

1.2 Summary of Contributions

The main aim of this thesis is incorporating orthogonality into facial image restoration and retrieval methods. A restoration method and three retrieval methods are proposed and briefly described in the following paragraphs.
In Chapter 2, we comprehensively review and discuss state-of-the-art deblur methods as well as facial deblur methods. Then propose our facial deblur method. Finally, the analysis and experimental results are given.
In Chapter 3, after a brief survey of current hashing methods, a data-independent and a data-dependent satellite distribution methods are proposed. The relation between our methods and existing methods are discussed, then. Experimental results are shown at the end.
In Chapter 4, a brief literature review is followed by the description of our method.

We compare our method to state-of-the-art multi-modal hashing methods and experimentally analyze the convergence property and parameter setting sensitivity of our method. In Chapter 5, a hybrid deep neural network is proposed to handle multimodal retrieval problem.

Chapter 2

Coupled Learning for Facial Deblur

Blur in facial images significantly impedes the efficiency of recognition approaches. However, most existing blind deconvolution methods cannot generate satisfactory results, due to their dependence on strong edges which are sufficient in natural images but not in facial images. In this chapter, we represent a point spread functions (PSF) by the linear combination of a set of pre-defined orthogonal PSFs and similarly, an estimated intrinsic sharp face image (EI) is represented by the linear combination of a set of pre-defined orthogonal face images. In doing so, PSF and EI estimation is simplified to discovering two sets of linear combination coefficients which are simultaneously found by our proposed coupled learning algorithm. To make our method robust to different kinds of blurry face images, we generate several candidate PSFs and EIs for a test image, and then a non-blind deconvolution method is adopted to generate more EIs by those candidate PSFs. Finally, we deploy a blind image quality assessment metric to automatically select the optimal EI. Thorough experiments on the The Facial Recognition Technology (FERET) Database[1], extended Yale Face Database B[2], CMU Pose, Illumination, and Expression (PIE) database[3] and Face Recognition Grand Challenge (FRGC)

[1] http://www.itl.nist.gov/iad/humanid/feret/feret_master.html

[2] http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html

[3] https://www.ri.cmu.edu/research_project_detail.html?project_id=418&menu_id=261

Database version 2.0[1] demonstrate that the proposed approach effectively restores intrinsic sharp face images and consequently improves the performance of face recognition.

2.1 Introduction

Facial blur is common in recorded face images. Examples include motion blur caused by the relative movement between the target and the camera, and out-of-focus blur caused by misalignment between the target and the camera focus. It remains challenging to improve the quality of an observed blurred face image (OB) for subsequent use in various applications, including face recognition and editing. A straightforward method of overcoming OB is to use blind deconvolution methods [90] [27] [122] [16] [118] [69] to obtain an estimated intrinsic face image (EI), and then to exploit the EI for subsequent recognition and analysis. The success of blind deconvolution methods designed for natural images relies on strong edges [61], which are relatively rare in most OBs. Therefore, this approach tends to perform poorly [127] and the obtained EIs do not significantly improve subsequent recognition.

Machine learning [35] [62] [34] [31] [71] [36] [120] [89] [21] [121] [64] is an effective tool for ill-pose problems. Recently, machine learning has been exploited to deconvolute OBs. Liao et al. [66] decomposed an intrinsic sharp face image into the eigen-face subspace and adopted a Gaussian prior to regularizing the EI. However, this approach assumed that the point spread function (PSF) has only one varying parameter, such as a Gaussian kernel with variable variance or a horizontal linear motion function of varying length, and fails to restore images blurred by sophisticated PSFs, e.g., a linear motion function with two varying parameters (direction and length).

Nishiyama et al. [83] proposed the FAcial DEblur INference (FADEIN) scheme, which models the PSF estimation procedure as a classification-like problem. By calculating the correlation matrix $\mathbf{A}_i$ that encodes the 2D Fourier transform features of all training images blurred by the i-th pre-defined PSF, the subspace ϕ_i, corresponding to the leading eigenvectors of $\mathbf{A}_i$, is used to model the i-th PSF.

[1] `http://www.nist.gov/itl/iad/ig/frgc.cfm`

The PSF corresponding to the subspace closest to the feature of the OB is then exploited to deconvolute the OB. In this way, FADEIN can only model a finite (and, in reality, small) discrete set of PSFs and fails to model PSFs that are not defined in the training stage.

Zhang et al. [127] proposed the Joint Restoration and Recognition (JRR)

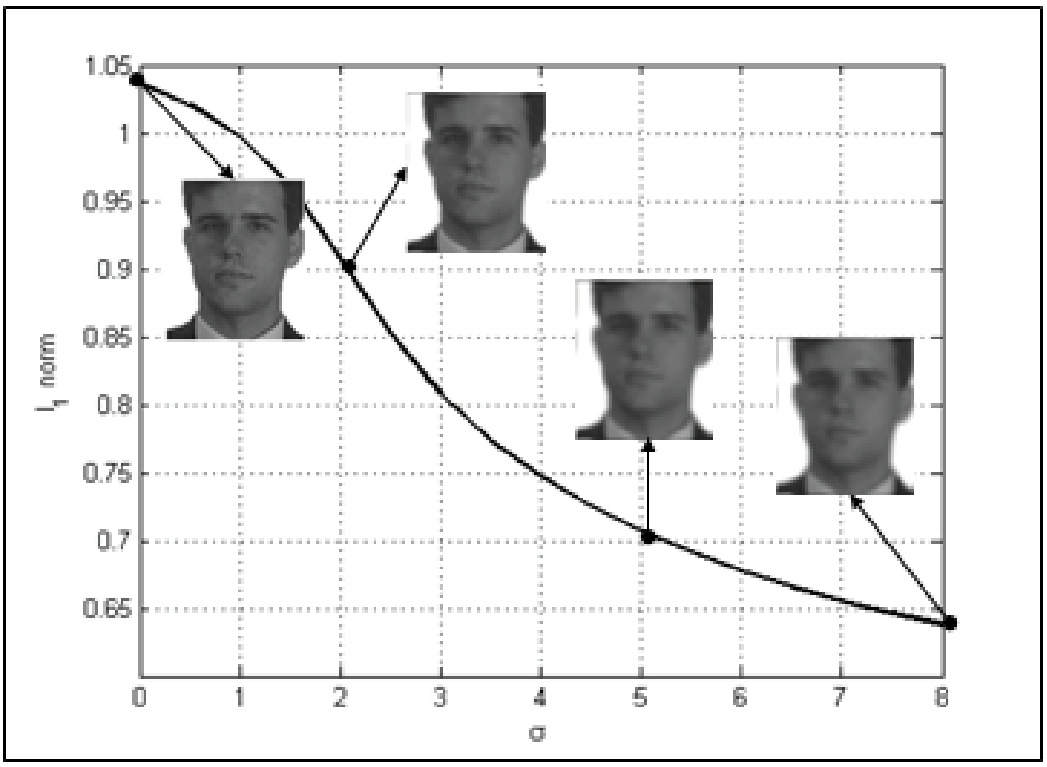


Figure 2.1: Illustration of the drawbacks of the sparse prior. The image is blurred by a Gaussian kernel of standard deviation 0-8. The l_1-norm of sparse coefficients monotonously decreases as the standard deviation increases. Hence, the sparse prior may lead to a blurry result.

scheme, which combines restoration and recognition within the framework of sparse learning. JRR assumes that the intrinsic sharp face image of an OB can be represented as a linear combination of face images in the training set, and the coefficients of the linear combination are assumed to be sparse. Although sparse prior has been proven to be effective in a wide range of applications including face recognition [105] [123] [117] [131] and image restoration [23] [24] [79] [111],

as shown in Figure 2.1, it is inappropriate for some deconvolution tasks, since the sparse prior may predispose to blurry images. In this situation, JRR may fail to deconvolute

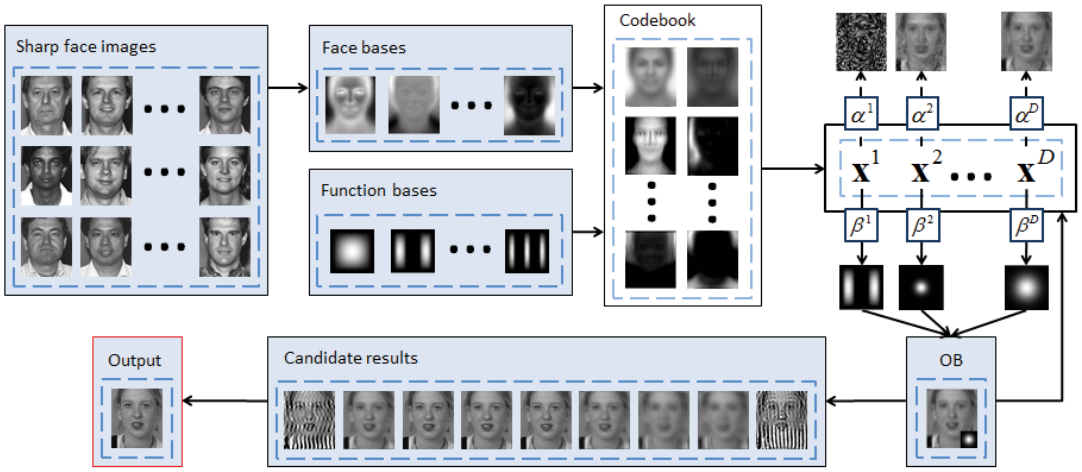


Figure 2.2: The framework of our method.

the OB. Furthermore, six parameters need to be empirically tuned, making JRR difficult to use in practice.

Recently, J. Pan et al. [86] proposed a face image deblurring method based on the contour of faces. The useless edges of a face, such as those around eyes and eyebrows, are removed firstly, because these edges have negative effects on the PSF estimation. Then, the PSF is estimated by finding a template for an OB from the training gallery. Rather than utilizing the information on edges intrinsically, like those unsupervised methods, J. Pan et al. try to utilize such information directly and smartly by filtering the edges. However, this work still depends on the edges.

To avoid the aforementioned problems and to conduct high performance face restoration, we cast the facial deblur procedure as a regression-like problem based on two mild assumptions: (1) any PSF can be represented by a linear combination of a set of orthogonal PSFs; and (2) the intrinsic sharp face image of an OB can be represented linearly by a set of orthogonalized sharp face images.

Under the above two assumptions, we develop a coupled learning algorithm (shown in Figure 2.2) to simultaneously calculate all possible PSFs and EIs by

discovering the coefficients of two associated linear combinations, in which each PSF corresponds to a particular EI. Empirically, all the EIs show far from satisfactory results, for the following two reasons. First, the dissimilarities between the training sharp face images and the intrinsic sharp face image of the OB can result in reconstruction errors. Second, the parameter space of EI is of tens of thousands of dimensions and thus to obtain a high quality EI requires a large number of training sharp images. By contrast, the parameter space of PSF is much smaller and can be estimated precisely given a small size training set. We therefore generate a sequence of PSFs and use a classical non-blind deconvolution method [12] to deblur the OB and generate candidate results. Lastly, a blind image quality assessment (BIQA) method [81] is adopted to automatically select the best EI which corresponds to a particular PSF.

In contrast to conventional face recognition scheme which consists of a face representation stage and a face matching stage [17], we propose a new recognition method based on our deblurring procedure. Intuitively, when the sharp face images have the same identity as the OB, the resulting EI is of high quality because these sharp face images are more similar to the intrinsic sharp face image of the OB. We therefore only need to deblur an OB using all the sets of sharp face images, where each set only contains sharp face images of one identity. The identity of the set that produces the best deblurring result is assigned to the OB. In this way, the proposed deblurring method simultaneously deblurs and recognizes the OB.

This chapter is organized as follows. The proposed deconvolution method is described in Section 2.2. In Section 2.3, we show how to reduce the computational costs for symmetric PSFs. In Section 3.5, we demonstrate the effectiveness of the proposed deconvolution and recognition methods. We conclude the chapter in Section 2.5.

2.2 On Combining Coupled Learning and BIQA for Facial Deblur

The proposed scheme for facial deblur, shown in Figure 2.2, is comprised of four major steps: (1) codebook construction; (2) coefficient $\mathbf{x}$ computation; (3) candidate PSF construction and candidate results generation; and (4) the BIQA-based best candidate result selection. This section shows the motivations and details of each step.

In general, the relationship between an OB I, its intrinsic sharp face image I^o, and the corresponding PSF ϕ is modeled as

$$I = I^o * \phi + \eta, \tag{2.1}$$

where η is the additive noise and $*$ is the convolution operation. We assume I^o can be represented by a linear combination of a set of bases $\{v_i\}, i = 1, \ldots, M$, and ϕ can be represented by a linear combination of a set of functions $\{\phi_j\}, j = 1, \ldots, N$. Hence, we have

$$I^o * \phi = \left(\sum_{i=1}^{M} \alpha_i v_i \right) * \left(\sum_{j=1}^{N} \beta_j \phi_j \right), \tag{2.2}$$

where α_is and β_js are coefficients of the linear combinations. Let $vec(\cdot)$ be the vectorization operation. Let the $(j+(i-1)N)$th column of matrix $\mathbf{A}$ be $vec(v_i * \phi_j)$ and the $(j + (i-1)N)$th element of vector $\mathbf{x}$ be $\alpha_i \beta_j$. Equation (2.1) can be rewritten as

$$\mathbf{I} = \mathbf{A}\mathbf{x} + vec(\eta). \tag{2.3}$$

where $\mathbf{I} = vec(I)$. In our approach, $\{v_i\}$ and $\{\phi_j\}$ are predefined. Hence, the remaining problems are calculating $\mathbf{x}$ in Equation (2.3) and calculating α_i and β_j from $\mathbf{x}$.

2.2.1 Construct A

Given a set of sharp face images $\{I_p\}, p = 1, \ldots, P$, we use the first M left singular vectors (i.e., those corresponding to the largest M singular values) of matrix $\mathbf{D}$

as $\mathbf{v}_i$, where $\mathbf{D}$ is defined as:

$$\mathbf{D} = [\mathbf{I}_1, \mathbf{I}_2, \ldots, \mathbf{I}_p] \tag{2.4}$$

To represent the PSFs, we use a set of orthogonal functions, called function bases

$$\begin{gathered} f(m,n,x,y) = sin\left(\frac{m\pi x}{a}\right)\sin\left(\frac{n\pi y}{b}\right) \\ m = 1,2,\ldots,L, \quad n = 1,2,\ldots,L \\ 0 \le x \le a, 0 \le y \le b \end{gathered} \tag{2.5}$$

If the first $K^2 (m = n = 1, \ldots, K)$ function bases are used, then we let $\phi_{m+(n-1)K} = f(m,n,x,y)$. Hence, we have

$$\begin{gathered} \mathbf{A} = [vec\left(I_1 * \phi_1\right), ..., vec\left(I_1 * \phi_{K^2}\right), ..., \\ vec\left(I_P * \phi_1\right), ..., vec\left(I_P * \phi_{K^2}\right)] \end{gathered} \tag{2.6}$$

2.2.2 Calculate x

An intuitive way to calculate $\mathbf{x}$ from Equation (2.3) is by minimizing

$$\underset{\mathbf{x}}{\text{argmin}} \ \| \mathbf{I} - \mathbf{A}\mathbf{x} \|^2 + R(\mathbf{x}), \tag{2.7}$$

where $R(\mathbf{x})$ is a regularization on $\mathbf{x}$. Here, we do not impose any regularization on $\mathbf{x}$, i.e., $R(\mathbf{x}) = 0$. Hence, minimizing Equation (2.7) is equivalent to solving

$$\mathbf{A}\mathbf{x} = \mathbf{I}. \tag{2.8}$$

Equation (2.8) has the closed form solution

$$\mathbf{x} = (\mathbf{A}^T\mathbf{A})^{-1}(\mathbf{A}^T\mathbf{I}), \tag{2.9}$$

where $\mathbf{A}^T$ is the transpose of $\mathbf{A}$. Alternatively, the conjugate gradient descent method [39] can be used to solve

$$\mathbf{x}^\mathbf{T}(\mathbf{A}^T\mathbf{A})\mathbf{x} - \mathbf{x}^T\mathbf{A}^T\mathbf{I} = 0. \tag{2.10}$$

Solving Equation (2.10) using the conjugate gradient descent method is much quicker than directly computing the inverse matrix of $\mathbf{A}^T\mathbf{A}$.

2.2.3 Calculate α and β

Once $\mathbf{x}$ is found, α and β can be calculated by solving the following nonlinear equation system:

$$\begin{aligned} x_{j+(i-1)N} &= \alpha_i \beta_j \\ i &= 1, \ldots, M \\ j &= 1, \ldots, N \end{aligned} \tag{2.11}$$

Equation (2.11) has $M+N$ unknown variables and MN equations, and is usually an over-determined system (i.e., $MN > M + N$). Hence, it can be solved by minimizing

$$\underset{\substack{\alpha_i, 1 \le i \le M \\ \beta_j, 1 \le j \le N}}{\operatorname{argmin}} \sum_{i=1}^{M} \sum_{j=1}^{N} (x_{j+(i-1)N} - \alpha_i \beta_j)^2. \tag{2.12}$$

Unlike linear over-determined equation systems, where the solution is unique when the rank of the coefficient matrix equals the number of columns, Equation (2.12) may have multiple solutions. For example, if $\{\alpha_i^*\}$ and $\{\beta_j^*\}$ are solutions of Equation (2.12), $\{\alpha_i^*/c\}$ and $\{c\beta_j^*\}$ will also be solutions, where c is a non-zero constant. Therefore, prior knowledge, regularizations, or constraints are required.

We observe that the projections of I^o on $\{v_i^*\}$ (i.e., α_is) are similar to those of I (Figure 2.3). Hence, the following item can be added into Equation (2.12):

$$(\langle I, v_1 \rangle - \alpha_1)^2, \tag{2.13}$$

where $\langle I, v_1 \rangle$ is the inner product, i.e., the projection of I on v_1.
It is desirable that the averages of the local windows of an OB should not be altered too much by an estimated PSF. Therefore, its integration should be equal to 1, that is,

$$\sum_{j=1}^{N} \beta_j \int_{\Omega} \phi_j d\Omega = 1, \tag{2.14}$$

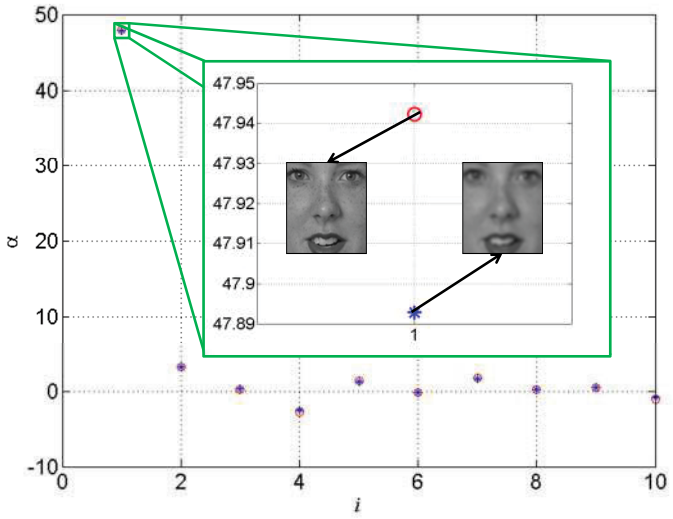


Figure 2.3: The difference between projections of an image and its blurred counterpart. 90% images in FERET dataset are used to construct $\mathbf{D}$ and the first 10 projections of a test image in the remaining 10% images and its blurred counterpart which is blurred by a Gaussian PSF with standard deviation 2 are shown here. It can be found that projections on the first left singular vector of these two images are very close to each other. The difference is approximately 0.1%.

where Ω is the domain in which the PSF is defined. The PSF should also be positive, that is,

$$[vec(\phi_1), vec(\phi_2), \ldots, vec(\phi_N)][\beta_1, \beta_2, \ldots, \beta_N]^T \geq [0, 0, \ldots, 0]^T. \tag{2.15}$$

In conclusion, the proposed minimization problem is

$$\begin{aligned} &\underset{\substack{\alpha_i, 1\le i\le M \\ \beta_j, 1\le j\le N}}{\operatorname{argmin}} \sum_{i=1}^{M}\sum_{j=1}^{N}(x_{j+(i-1)N} - \alpha_i\beta_j)^2 + (\langle I, v_1\rangle - \alpha_1)^2 \\ &s.t. \sum_{j=1}^{N}\beta_j \int_{\Omega}\phi_j d\Omega = 1 \\ &[vec(\phi_1), \ldots, vec(\phi_N)][\beta_1, \ldots, \beta_N]^T \ge [0, \ldots, 0]^T \end{aligned} \tag{2.16}$$

As SVD is adopted to generate face bases, we only use the projection on the first left singular vector which corresponds to the largest singular value as a regularization in the object function of our optimization problem. The problem (2.16) can be solved using the augmented Lagrange multiplier method [40].

2.2.4 Generate candidate results

Since reconstruction errors exist, the α_is calculated by the procedure above cannot usually reconstruct satisfactory results, especially when the identity of the OB is not included in the training set. However, as stated in the introduction, a PSF is much easier to estimate than its corresponding EI. Setting M as a certain value, one can get a PSF correspondingly. We therefore generate a sequence of PSFs by setting different M's and use a classical non-blind deconvolution method [12] to deblur the OB to generate M candidate results.

2.2.5 Assessing candidate results

BIQA across different images is a challenging task. However, blindly assessing the qualities of images distorted by different ways from an image is much easier. Among various BIQA methods, BRISQUE-L [81] [80] feature is used here to select the best candidate PSF, due to its robustness and computational efficiency.

Figure 2.4: Typical results and their corresponding scores. -2 denotes a failure, 0 is a blurry result, and 2 is a sharp image.

BRISQUE pre-processes an image by local mean removal, i.e.

$$\hat{I}(i,j) = \frac{I(i,j) - \mu(i,j)}{\lambda(i,j) + 1} \tag{2.17}$$

where $i \in \{1, 2, \ldots, M\}$, $j \in \{1, 2, \ldots, N\}$ are spatial indices, M and N are the image dimensions, and

$$\mu(i,j) = \sum_{k=-K}^{K} \sum_{l=-L}^{L} \omega_{k,l} I(i+k, j+l) \tag{2.18}$$

$$\lambda(i,j) = \sqrt{\sum_{k=-K}^{K} \sum_{l=-L}^{L} \omega_{k,l} \left[I(i+k, j+l) - \mu(i,j) \right]^2} \tag{2.19}$$

$\omega = \{\omega_{k,l} | k = -K, \ldots, K, l = -L, \ldots, L\}$ is a 2D circularly-symmetric Gaussian weighting function. K and L is set as 3 and ω is normalized to unit volume.

A Generalized Gaussian Model (GGD) distribution is used to model mean subtracted contrast normalized (MSCN) coefficients and how they change with distortion. The GGD with zero mean is:

$$f = \frac{\alpha}{2\beta\Gamma(1/\alpha)} \exp\left(- \left(\frac{|x|}{\beta} \right)^{\alpha} \right) \tag{2.20}$$

where

$$\beta = \sigma \sqrt{\frac{\Gamma(1/\alpha)}{\Gamma(3/\alpha)}} \tag{2.21}$$

and

$$\Gamma(a) = \int_0^\infty t^{a-1} e^{-t} dt \qquad a > 0 \tag{2.22}$$

is the gamma function. The parameters of the GGD is estimated by the moment-matching based approach [91].

A zero mode asymmetric generalized Gaussian distribution (AGGD) [60] is used to model the products of neighboring coefficients:

$$f(x; \gamma, \sigma_l^2, \sigma_r^2) = \frac{\gamma}{(\beta_l + \beta_r)\Gamma(1/\gamma)} \exp\left(-\left(\frac{|x|}{\beta_l}\right)^\gamma\right) \tag{2.23}$$

where

$$\beta_l = \sigma_k \sqrt{\frac{\Gamma\left(\frac{1}{\gamma}\right)}{\Gamma\left(\frac{3}{\gamma}\right)}} \tag{2.24}$$

$$\beta_r = \sigma_r \sqrt{\frac{\Gamma\left(\frac{1}{\gamma}\right)}{\Gamma\left(\frac{3}{\gamma}\right)}} \tag{2.25}$$

Mean of the distribution is also used as a feature:

$$\eta = (\beta_r - \beta_l) \frac{\Gamma\left(\frac{2}{\gamma}\right)}{\Gamma\left(\frac{1}{gamma}\right)}. \tag{2.26}$$

16 parameters are computed by estimating $(\gamma, \sigma_l^2, \sigma_r^2, \eta)$ along four orientations. Including two parameters of the GGD, i.e. (α, σ^2), an 18 dimensional feature vector can be generated. Filter and downsample the image by a factor of 2, another 18 dimension feature vector can be generated in the same way. Hence, a 36 dimensional feature vector is generated.

To robustify BRSIQUE, BRISQUE-L introduced L-moments which are closely related to L-estimators and extensively used in robust image filtering theory [10].

The L-moments of a sample $X_i, i = 1, \ldots, N$ use probability weighted moments [37] of the order statistics [9] [76] $X_{(i)}$ of the sample:

$$b_0 = \frac{\sum_{i=1}^N}{X_{(i)}} \tag{2.27}$$

$$b_r = \frac{\sum_{i=r+1}^{N} \frac{(i-1)(i-2)...(i-r)}{(n-1)(n-2)...(n-r)} X_{(i)}}{N} \tag{2.28}$$

The first four L-moments are given by:

$$l_1 = b_0 \tag{2.29}$$

$$l_2 = 2b_1 - b_0 \tag{2.30}$$

$$l_3 = 6b_2 - 6b_1 + b_0 \tag{2.31}$$

$$l_4 = 20b_3 - 30b_2 + 12b_1 - b_0 \tag{2.32}$$

Similar to BRISQUE, the first and fourth L-moments are computed from the pointwise statistics of MSCN coefficients. Along four orientations, the pairwise product of adjacent MSCN coefficients are computed, including the first and fourth L-moments, the second L-moment using negative products and the positive products. BRISQUE-L is also extracted on two different scales and totally 36-dimensional.

2.2.6 Simultaneous restoration and recognition

A subset of candidate results is manually evaluated at five levels, as shown in Figure 2.4. Finally, support vector regression (SVR) is trained to automatically evaluate the quality of the remaining candidate results.

Since a BRISQUE-L feature contains two parts (18 elements in each part), a multiple kernel learning (MKL) method [33] needs to be adopted. Of the various MKL algorithms, the SMO-MKL approach [107] has been shown to be efficient and effective across a wide range of applications, and its source code is available online[1]. We therefore use it here with five Gaussian kernels for each part and five Gaussian kernels for the whole feature, that is, 15 kernels in total.

[1] `http://research.microsoft.com/en-us/um/people/manik/code/SMO-MKL/download.html`

Let a BRISQUE-L feature be $\mathbf{w} = (w_1, \ldots, w_{36})^T$ and $\mathbf{w}(i:j)$ is a vector comprised of the ith to jth elements in $\mathbf{w}$. The standard deviation σ of Gaussian kernel is chosen from set $\{2^{-2}, 2^{-1}, 2^0, 2^1, 2^2\}$. Hence, the fifteen Gaussian kernels are:

$$\begin{aligned} G_1 &= \exp\left(-\frac{\mathbf{w}^T(1:18)\mathbf{w}(1:18)}{(2^{-2})^2}\right), \ldots, \\ G_5 &= \exp\left(-\frac{\mathbf{w}^T(1:18)\mathbf{w}(1:18)}{(2^{2})^2}\right), \\ G_6 &= \exp\left(-\frac{\mathbf{w}^T(19:36)\mathbf{w}(19:36)}{(2^{-2})^2}\right), \ldots, \\ G_{10} &= \exp\left(-\frac{\mathbf{w}^T(19:36)\mathbf{w}(19:36)}{(2^{2})^2}\right), \\ G_{11} &= \exp\left(-\frac{\mathbf{w}^T\mathbf{w}}{(2^{-2})^2}\right), \ldots, \\ G_{15} &= \exp\left(-\frac{\mathbf{w}^T\mathbf{w}}{(2^{2})^2}\right) \end{aligned} \tag{2.33}$$

The primal problem of multiple kernel learning for SVR is

$$\begin{aligned} &\min_{\mathbf{z},b,\xi^{\pm}\geq\mathbf{0},\mathbf{d}\geq\mathbf{0}} \frac{1}{2}\sum_k \mathbf{z}_k^T\mathbf{z}_k/d_k + C\sum_h(\xi_h^+ + \xi_h^-) + \frac{\lambda}{2}\left(\sum_k d_k^p\right)^p \\ &s.t. \pm\left(\sum_k \mathbf{z}_k^T G_k(\mathbf{w}_h) + b - s_h\right) \leq \epsilon + \xi_h^{\pm} \end{aligned} \tag{2.34}$$

where $\mathbf{z_k}$ is the support vector and d_k is the kernel weights of the linear combination of base kernels $\{G_k\}$. ξ_h^+ and ξ_h^- are slack variables allowing for errors around the regression function. C, λ, p and ϵ are positive constants set empirically. Introducing Lagrange Multipliers a_h^+ on constraints corresponding to ξ_h^+ and a_h^- on constraints corresponding to ξ_h^-, the dual problem of Eq. (2.34) is

$$\begin{aligned} &\max_{\substack{\mathbf{1}^T\mathbf{a}^+=\mathbf{1}^T\mathbf{a}^- \\ \mathbf{0}\leq\mathbf{a}^+,\mathbf{a}^-\leq C\mathbf{1}}} \mathbf{1}^T(S(\mathbf{a}^+ - \mathbf{a}^-) - \epsilon(\mathbf{a}^+ + \mathbf{a}^-)) - \\ &\frac{1}{8\lambda}\left(\sum_k \left((\mathbf{a}^+ - \mathbf{a}^-)^T G_k(\mathbf{a}^+ - \mathbf{a}^-)\right)^q\right)^{\frac{2}{q}} \end{aligned} \tag{2.35}$$

where S is a diagonal matrix whose elements are the scores of training samples. p and q satisfy

$$\frac{1}{p} + \frac{1}{q} = 1 \tag{2.36}$$

Sequential Minimal Optimization (SMO) [88] [26] [53] is used to solve Eq. (2.35).

The proposed recognition method is based on the SVR outputs. We construct $\mathbf{A}$ for each identity, which is then used to deblur each OB. Lastly, the identity of $\mathbf{A}$ that produces the best deblurring result is assigned to the OB. This simple manipulation results in the simultaneous production of deblurring and recognition results.

2.3 Efficient Implementation for Symmetric PSF

In the theoretical point view, our algorithm can inherently handle any kinds of PSF. It is based on the mathematical theory - any bounded function can be represented by a linear combination of a complete set of orthogonal functions.

We mainly focus on three types of blurring: out-of-focus blur (approximated by a Gaussian kernel), linear motion blur, and a combination of the two. Here, we show how to reduce computational costs by considering the symmetry of these three types of blur. The aim is to reduce the number of function bases, i.e., N.

To simplify our analysis, we assume that the PSFs of the linear motion blur and the combined blur only have four directions: $0, \pi/4, \pi/2$ and $3\pi/4$. Considering the symmetry of 0-PSFs (Figure 2.5), only the $f(m, n, x, y)$s that are symmetric about $x = a/2$ and $y = b/2$ (i.e., $m = 1, 3, 5, \ldots$ and $n = 1, 3, 5, \ldots$), can be used to represent such PSFs. Since both the symmetric axes of 0-PSFs and $\pi/2$-PSFs are $x = a/2$ and $y = b/2$, they can be represented by a common set of function bases, $\{\varphi_j\}$. By rotating this set of function bases by $\pi/4$ clockwise or counter-clockwise, the function bases set (denoted as $\{\psi_j\}$) can reconstruct $\pi/4$-PSFs and $3\pi/4$-PSFs.

Therefore, if the directions of the PSFs of OBs can be estimated, the OBs can be divided into two groups: Group I(0- & $\pi/2$-PSFs) and Group II ($\pi/4$ & $3\pi/4$-PSFs). By constructing two different $\mathbf{A}$s using $\{\varphi_j\}$ and $\{\psi_j\}$, respectively, an OB can be deconvoluted using the method proposed in Section 2. The only

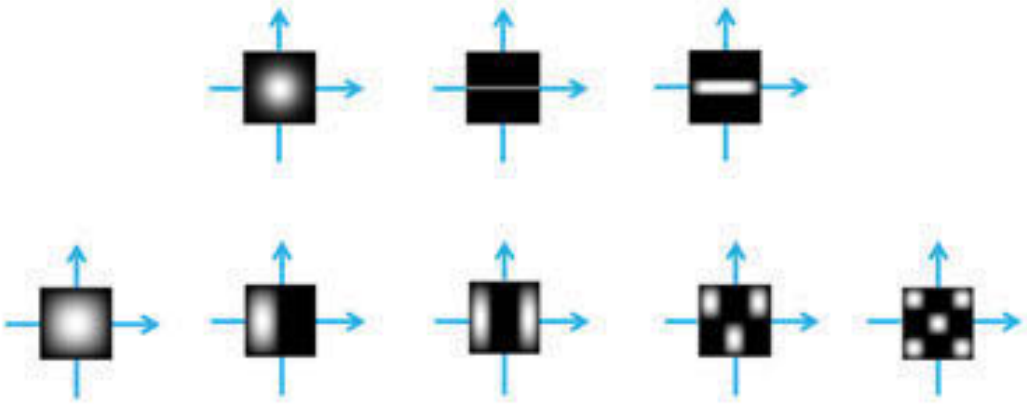


Figure 2.5: The symmetry of PSFs. The top row illustrates three PSFs: a Gaussian kernel, a linear motion kernel, and the combination of both. The bottom row illustrates five function bases, i.e.,$m = n = 1$, $m = 1\&n = 2$, $m = 1\&n = 3$, $m = 2\&n = 3$, and $m = n = 3$. The function bases of odd m and n are symmetric about the two axes and can be used to represent the three symmetric PSFs.

remaining problem is how to efficiently estimate the direction of the PSFs.

2.3.1 PSF direction estimation

Ignoring additive noise and taking the Fourier transform on both sides of Equation (2.1), we get

$$\mathcal{F}(I) = \mathcal{F}(I^o)\mathcal{F}(\phi), |\mathcal{F}(I)| = |\mathcal{F}(I^o)||\mathcal{F}(\phi)|, \tag{2.37}$$

where $\mathcal{F}(\cdot)$ denotes the Fourier transform. Given another PSF ϕ', it can be deduced that

$$\begin{aligned} 2|\mathcal{F}(I)||\mathcal{F}(\phi')| &= 2|\mathcal{F}(I^o)||\mathcal{F}(\phi)||\mathcal{F}(\phi')| \\ &\leq |\mathcal{F}(I^o)|(|\mathcal{F}(\phi)|^2 + |\mathcal{F}(\phi')|^2). \end{aligned} \tag{2.38}$$

In inequality (2.38), the equation holds, if and only if $|\mathcal{F}(\phi)| = |\mathcal{F}(\phi')|$. Inspired by the correlation-based shape alignment method [104], we try several ϕ's that are similar to ϕ and find the ϕ' whose $|\mathcal{F}(\phi')|$ maximizes $\int_\Omega 2|\mathcal{F}(I^o)||\mathcal{F}(\phi)||\mathcal{F}(\phi')|$

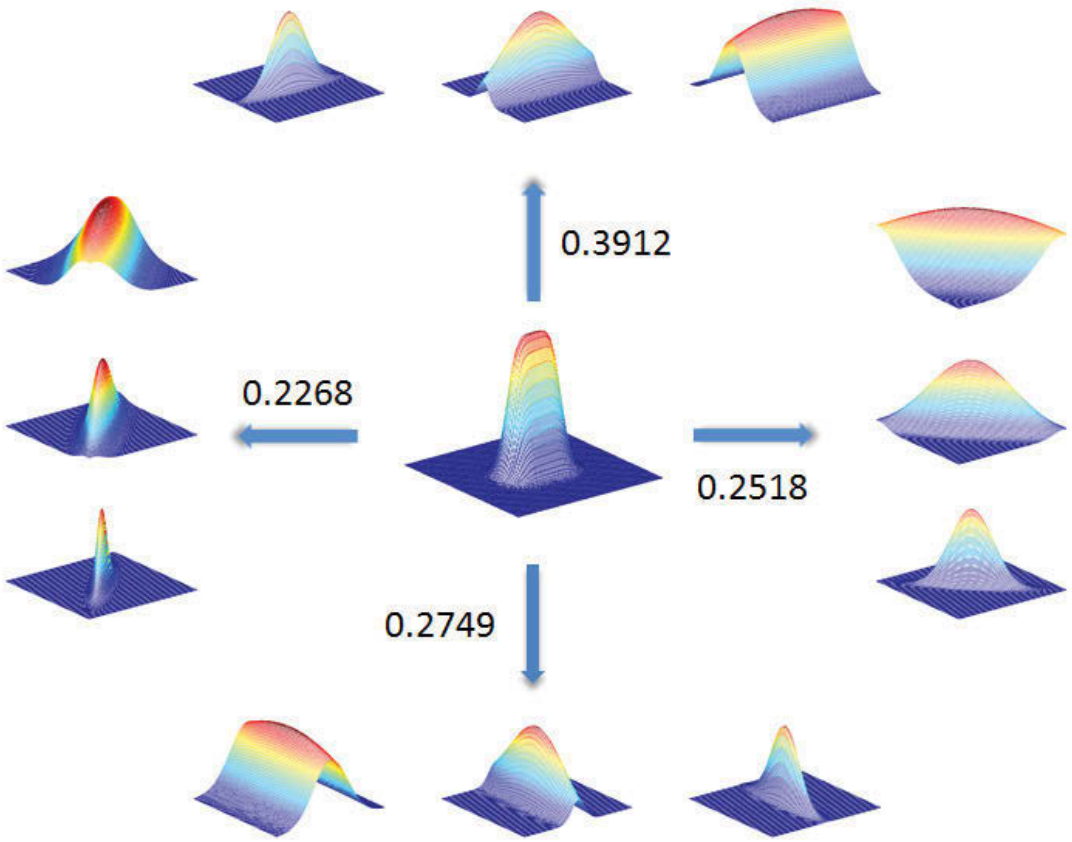


Figure 2.6: Illustration of proposed PSF direction estimation algorithm. A combination of linear motion and Gaussian PSF locates at the center of the graph. ϕ's with different σ locate around the PSF according to their direction. The numbers are the maximum values of $\int_\Omega |\mathcal{F}(\phi')||\mathcal{F}(\phi)|d\Omega$ on each direction.

(Fig. 2.6). Here, we use following function as ϕ':

$$\phi'(x, y, \gamma, \sigma, \theta) = \exp\left(\frac{x'^2 + \gamma^2 y'^2}{\sigma^2}\right), \quad \gamma \geq 1, \tag{2.39}$$

where $x' = x\cos\theta + y\sin\theta$ and $y' = y\cos\theta - x\sin\theta$.

The proposed direction estimation method is shown in **Algorithm 1**. Although we neglect the additive noises to develop our direction estimation algorithm, the algorithm turns out to be efficient when noises exist (Subsection 2.4.1). A reasonable explanation is that noises are random and directionless, and hence have little effect on direction estimation.

Algorithm 1 Direction Estimation Method

Input: I
for each $\theta \in \{0, \pi/4, \pi/2, 3\pi/4\}$ **do**
    for each $\sigma \in \{1, 2, \ldots, 9\}$ **do**
        if $V < |\mathcal{F}(I)||\mathcal{F}(\phi')|$ **then**
            $V = |\mathcal{F}(I)||\mathcal{F}(\phi')|$
            $L = \theta$
        end if
    end for
end for
Output: L

2.4 Experiments

The aim of the following experiments is three-fold: (1) to verify the accuracy of the proposed direction estimation method; (2) to validate the deconvolution method without recognition; and (3) to test the proposed recognition method.

We conducted experiments on FERET [87], CMU-PIE [96] and extended Yale B [32] datasets and compared our method with FADEIN [83], Krishnan *et al.* method [55], JRR [127]and Pan *et al* [86].

Dataset description. For FERET, we used the gallery (FA set) containing 1,196 images of 1,196 subjects. For CMU-PIE, there are 67 identities and we collected the images named as "27_**.jpg" under the "illum" folder of each identity. The file name "27_**.jpg" implies the photo was taken in frontal view. For the extended Yale B, we directly applied the images in "Cropped Yale" available at the official website. This subset contains 38 identities and 64 photos taken from the frontal view under different illumination conditions for each identity. For FRGC 2.0, we collect 2000 controlled still images and 500 uncontrolled still images of 200 subjects from both the training and validation partitions. Please note only placid faces were collected and a few uncontrolled still images contain no obvious blurs.

Evaluation method. There are three widely used indexes to evaluate image quality: Peak Signal-to-Noise Ratio (PSNR), Universal Image Quality Index (UIQI) [112] and Structural Similarity Index (SSIM) [112]. PSNR is based on the mean square error between two images, which is easy to calculate but less

efficient than indexes based on human visual system [4], such as UIQI and SSIM. Hence, SSIM which is an improvement of UIQI is adopted here to evaluate our deblur method.

The SSIM has two inputs, the sharp image and its synthetically blurred counterpart. To calculate SSIM, two means, two standard deviations and one covariance are computed on each $B \times B$ local window of two images as in Eq. (2.40).

$$\begin{gathered}
\mu_\zeta = \frac{1}{T}\sum_{q=1}^{T}\zeta_q \qquad \mu_\vartheta = \frac{1}{T}\sum_{q=1}^{T}\vartheta_q \\
\sigma_\zeta^2 = \frac{1}{T-1}\sum_{q=1}^{T}(\zeta_q - \bar{\zeta})^2 \quad \sigma_\vartheta = \frac{1}{T-1}\sum_{q=1}^{T}(\vartheta_q - \bar{\vartheta})^2 \\
\sigma_{\zeta\vartheta}^2 = \frac{1}{T-1}\sum_{q=1}^{T}(\zeta_q - \bar{\zeta})(\vartheta_q - \bar{\vartheta}) \\
SSIM(\zeta,\vartheta) = \frac{(2\mu_\zeta\mu_\vartheta + c_1)(2\sigma_{\zeta\vartheta} + c_2)}{(\mu_\zeta^2 + \mu_\vartheta^2 + c_1)(\sigma_\zeta^2 + \sigma_\vartheta^2 + c_2)}
\end{gathered} \tag{2.40}$$

where ζ and ϑ represent local windows on sharp images and blurred images, respectively. c_1 and c_2 are two constants. $T = B^2$ is the total number of a local window. The overall SSIM index is the mean of $SSIM(\zeta, \vartheta)$ of all local windows. In our experiments, the mean (μ) and standard deviation (σ) of SSIM are given on every dataset.

Global settings. Throughout our experiments, 30 dB additive white noise is synthesized into test images. We use the default setting of Chan *et al.*'s non-blind deconvolution method. Its code is available online[1].

2.4.1 PSF direction estimation

We randomly selected 1000 images from the three subsets of FERET, CMU-PIE and extended Yale B. The widths of the linear motion kernels were chosen as 3, 5, 7, and 9 and the standard deviations of the Gaussian kernels as 1, 3, 5, 7, and 9. Hence, $4(directions) \times 4(widths) \times 5(standard \quad deviations) \times 1000 = 80000$ images were generated for testing the proposed direction estimation method. For

[1] `http://videoprocessing.ucsd.edu/~stanleychan/deconvtv`

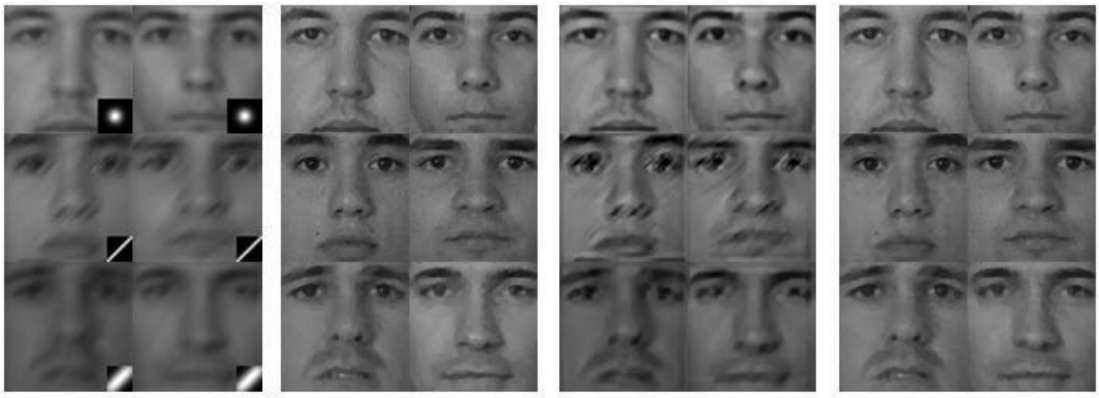


Figure 2.7: Results of experiments on the subset of FERET dataset. (a) Test images and their PSFs; (b) results of the proposed method; (c) results using the method proposed by Krishnan *et al.* [55]; (d) results of FADEIN [83] and (e) results using the method proposed by Pan *et al.* [86]

each combination of these two types of kernels, the rate of correctly estimating the PSF was recorded and shown to be 97%. Since the direction estimation is imperfect, some OBs will be miscategorized. Hence, OBs in which all candidate results have low scores, say less than 0, should be re-categorized into another group.

2.4.2 Facial deblur

Experiments in this subsection were conducted on FERET. We used 90% of the images to generate the matrix $\mathbf{A}$. The remaining 10% of images were treated as OBs and blurred by a Gaussian kernel of σ 2, linear motion of length 15 and direction $\pi/4$, or the combination of the two. The first 9 odd order orthogonal functions were used to construct $\mathbf{A}$. We generated nine candidate results for each OB by setting nine Ms, where the cumulative energy content for the Mth eigenvector occupied 60%, 70%, 80%, 90%, 91%, 92%, 93%, 94% and 95% of the total energy, respectively. Candidate results of 10 randomly-selected OBs were used to train SVR. FADEIN [83] and the Krishnan *et al.* method [55] were implemented and examined under the author-suggested settings to guarantee fair comparison. The final results are shown in Figure 2.7. The Krishnan *et al.*

Table 2.1: SSIM Index on FERET

	μ		
PSF	Combine	Gaussian	Motion
FADEIN [83]	0.8808	0.9290	0.9053
Krishnan *et al.* [55]	0.7730	0.8866	0.7177
Pan *et al.* [86]	0.8854	0.9211	0.9123
Ours	0.8939	0.9214	0.9158
	σ		
PSF	Combine	Gaussian	Motion
FADEIN [83]	0.0100	0.0110	0.0123
Krishnan *et al.* [55]	0.0083	0.0184	0.0324
Pan *et al.* [86]	0.0101	0.0103	0.0056
Ours	0.0100	0.0098	0.0043

method did not perform well on this dataset, because of its requirement of strong edges. Since FADEIN is a classification-like scheme, its deconvolution results are similar to ours if the PSFs of the OBs are pre-defined in the training set (the first row). When the PSFs are not pre-defined in the training set (the second and third rows), it tends to give the closest PSFs. Hence, the results are degraded somewhat.

2.4.3 Simultaneous facial deblur and recognition

The proposed method was compared with FADEIN [83] and JRR [127] on the subset of CMU-PIE and the subset of the extended Yale B. In contrast to FERET, the face images in these two subsets were taken under different illumination conditions.

For experiments on each subset, we adopted the first 50% of images of each identity in our method for training and the remaining 50% of images of each identity for testing. All test images were blurred by a Gaussian kernel of σ 3, a linear motion of length 15 and direction $\pi/4$ or the combination of both. All these blurred images were collected together as OBs. The first 9 odd order orthogonal functions were used to construct $\mathbf{A}$. The candidate results were generated by setting Ms in a similar way to those given in Subsection 4.2, but note that M was occasionally zero, because only tens of sharp face images were available to

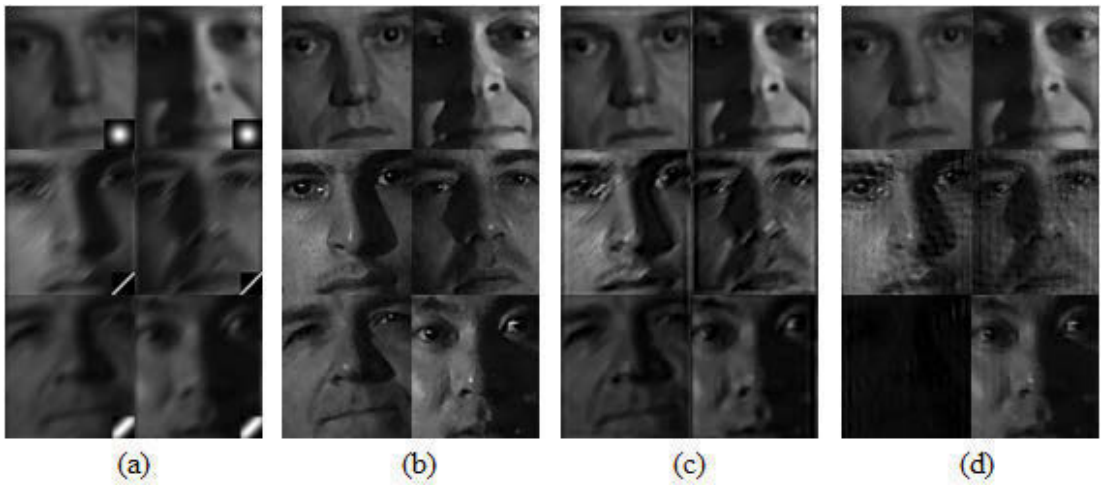


Figure 2.8: Results of experiments on the subset of CMU-PIE dataset. (a) Test images and their PSFs; (b) results of the proposed method; (c) results using the method proposed by Krishnan *et al.* [55]; (d) results of FADEIN [83] and (e) results using the method proposed by Pan *et al.* [86]

construct matrix **A** in each procedure of generating candidate results. We started the procedure with the smallest non-zero .

For fair comparison, we used exactly the same setting for our method on each dataset. According to [83], FADEIN adopted the local phase quantization (LPQ) [3] method for recognition. We carefully tuned the parameters of JRR to ensure it reached its best performance.

The deconvolution results on CMU-PIE are shown in Figure 2.8 and the corresponding recognition rates are listed in Table 2.4. The deconvolution and recognition results on extended Yale B are shown in Fig 2.9 and Table 2.5, respectively. The method of Krishnan *et al.* [55] failed to provide satisfactory deconvolution results. Due to the complex illumination conditions, FADEIN [83] cannot usually estimate the PSFs precisely and therefore performed poorly on recognition. The deconvolution results of JRR [127] are not shown here, because the authors explained that the deconvolution procedure in JRR was not designed for human visual perception [127]. It is unfair to compare the deconvolution results of JRR with other methods, but note that JRR performed poorly in terms of visual appearance. It can be concluded that the proposed method not only gives satisfactory deconvolution results, but also significantly boosts recognition per-

Figure 2.9: Results of experiments on the subset of extended Yale B dataset. (a) Test images and their PSFs; (b) results of the proposed method; (c) results using the method proposed by Krishnan *et al.* [55]; (d) results of FADEIN [83] and (e) results using the method proposed by Pan *et al.* [86]

formance.

Since recognition is based on image quality, our proposed method does not require compensation for illumination. However, due to the complex illumination conditions in some images, the proposed method may fail to deconvolute in some cases. Also, since Equation (2.16) has multiple local minima, this can result in erroneous recognition, because the correct $\{\beta_j\}$ can be given even when $\{\alpha_i\}$ is wrong. Even though the probability of this occurring is low, as shown by the high recognition rate, such mistakes are inevitable.

2.4.4 Experiments on camera-shaking blur

In this experiment, the restoration results of our method are compared with Krishnan *et al.* and its recognition results are compared with JRR on CMU-PIE and extended Yale B datasets. FADEIN is not available for this kind of blur, i.e., irregular and asymmetrical PSFs.

The experiment settings are same to those in Subsection 2.4.3, except follow-

Table 2.2: SSIM Index on CMU-PIE

	μ		
PSF	Combine	Gaussian	Motion
FADEIN [83]	0.7599	0.7909	0.8200
Krishnan *et al.* [55]	0.6398	0.7705	0.6579
Pan *et al.* [86]	0.7947	0.8448	0.8033
Ours	0.8068	0.8780	0.8386
	σ		
PSF	Combine	Gaussian	Motion
FADEIN [83]	0.0171	0.0174	0.0224
Krishnan *et al.* [55]	0.0212	0.0171	0.0463
Pan *et al.* [86]	0.0144	0.0189	0.0151
Ours	0.0131	0.0160	0.0063

ing three points: (1) all test images were blurred by two camera-shaking PSFs; (2) the first 36 order orthogonal functions were used; and (3) the facial images reconstructed by α's were also added into the gallery of candidate results.

BRISQUE which is widely used to evaluate the quality of natural images which are assumed to be Gaussian distributed. However, as mentioned in Section 2.1, the face images contain less strong edges, which means they may be not Gaussian distributed. The BRISQUE-L feature was reported to be relatively unaffected by small departures from model assumptions. Hence, we use BRISQUE-L feature for our task.

The restoration results on CMU-PIE and extended Yale B datasets are shown in Fig. 2.10 and Fig. 2.11, respectively. Our method significantly outperforms that of Krishnan *et al.* In terms of restoration, our method has a tiny flaw on restoring shadow areas (the bottom two images in Fig. 2.11). This phenomenon is caused by collecting reconstructed images as candidate results. In reality, the trained SVR tends to give higher scores to images with less illumination variance. For example, the red numbers in Fig. 2.11 are the scores of the original sharp images, while the blue numbers are the scores of the restoration results (in this case, the reconstructed images). The recognition results on both datasets are given in Table 2.8, which demonstrates the superiority of our method in the recognition of blurred facial images.

Table 2.3: SSIM Index on Extended Yale B

	μ		
PSF	Combine	Gaussian	Motion
FADEIN [83]	0.8458	0.8537	0.8537
Krishnan *et al.* [55]	0.6906	0.8822	0.6566
Pan *et al.* [86]	0.8521	0.8992	0.8841
Ours	0.8998	0.9167	0.9020
	σ		
PSF	Combine	Gaussian	Motion
FADEIN [83]	0.0349	0.0547	0.0547
Krishnan *et al.* [55]	0.0663	0.0297	0.0693
Pan *et al.* [86]	0.0334	0.0399	0.0372
Ours	0.0384	0.0516	0.0221

Table 2.4: Recognition rates on the subset of CMU-PIE dataset.

	Gaussian	Motion	Both
FADEIN+LPQ [83]	85.6	84.1	82.3
JRR [127]	92.7	92.3	91.7
Ours	95.1	95.0	95.0

2.4.5 Experiments on real blur

In this experiment, the restoration results of our method are compared with Krishnan *et al.* and Pan *et al.* and its recognition results are compared with JRR on FRGC 2.0 dataset. Again, FADEIN is not available for this case, because the PSF of real blur cannot be pre-defined in the training procedure.

All the sharp images of each identity and the first 36 order orthogonal functions were used for constructing $\mathbf{A}$. The candidate results were generated in a similar way to that in Subsection 2.4.4, so did the training of SVR.

The restoration results are shown in Fig. 2.12. In Fig. 2.12, our algorithm gave three linear combinations of face bases as the final restoration results which were marked by red rectangles. As there are no groudtruth sharp images for these

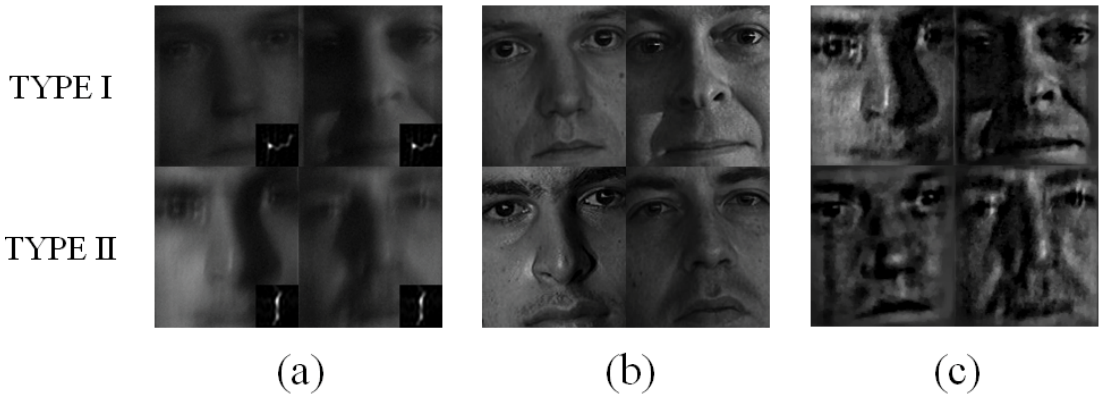


Figure 2.10: Results of experiments on the subset of CMU-PIE dataset. (a) Test images and their PSFs; (b) results of the proposed method; (c) results using the method proposed by Krishnan *et al.* [55] and (d) results using the method proposed by Pan *et al.* [86]

Table 2.5: Recognition rates on the subset of extended Yale B dataset.

	Gaussian	Motion	Both
FADEIN+LPQ [83]	77.3	75.8	70.1
JRR [127]	88.2	86.8	86.3
Ours	93.4	92.6	91.8

OBs, the SSIM index is not available here. However, it is easy to visually observe the superiority of our restoration results against compared ones'. The recognition rates of JRR and our method are 93.5% and 98.7%, respectively.

2.4.6 Computational efficiency

Our algorithm gives the deblurring and recognition results at the same time. There are four major parts that take relative longer time to compute: 1) SVD; 2) construct A; 3) train SVR and 4) minimize Eq. (2.16). For each dataset, 1), 2) and 3) are just computed once. The computation complexity of SVD of an $P_1 \times P_2$ matrix is $O(4P_1^2P_2 + 8P_1P_2^2 + 9P_2^3)$. In this case, P_1 is the number of image pixels and $P_2 = MN$ where M and N are the number of face and function bases, respectively. To construct $\mathbf{A}$, MN 2-D convolutions are needed. By using Fast Fourier Transformation (FFT), the computation complexity of 2-

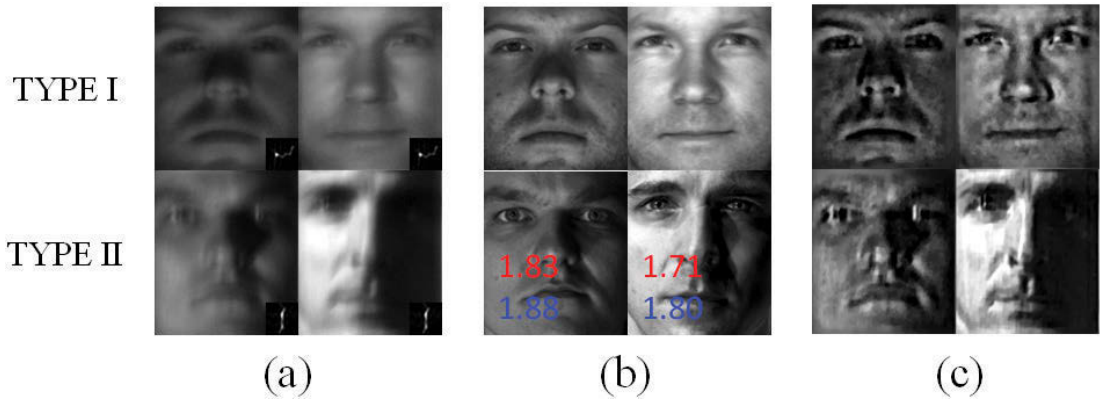


Figure 2.11: Results of experiments on the subset of extended Yale B dataset. (a) Test images and their PSFs; (b) results of the proposed method; (c) results using the method proposed by Krishnan *et al.* [55] and (d) results using the method proposed by Pan *et al.* [86]

D convolution is $O(P_1P_2\log(P_1P_2))$. In this case, P_1 and P_2 are the height and width of an image. 3) and 4) are two constrained optimization problem which are solved iteratively. The convergence rates are highly dependent on the datasets and parameter settings. For all our experiments which were done on MATLAB 2014a on a PC with Intel Core i5 3.2GHz and 8GB RAM, 3) takes less than 1 second and 4) takes 20 to 40 seconds for one test image.

2.5 Conclusions

In this chapter, we proposed a coupled learning method combined with blind image quality assessment (BIQA) for image deconvolution. The method is specifically designed for deblurring face images that have few strong edges, and can the-

Table 2.6: SSIM Index on CMU-PIE dataset

	μ		σ	
PSF	TYPE I	TYPE II	TYPE I	TYPE II
Krishnan *et al.* [55]	0.4973	0.5982	0.0321	0.0792
Pan *et al.* [86]	0.8149	0.7922	0.0930	0.0958
Ours	0.8566	0.8315	0.0749	0.1834

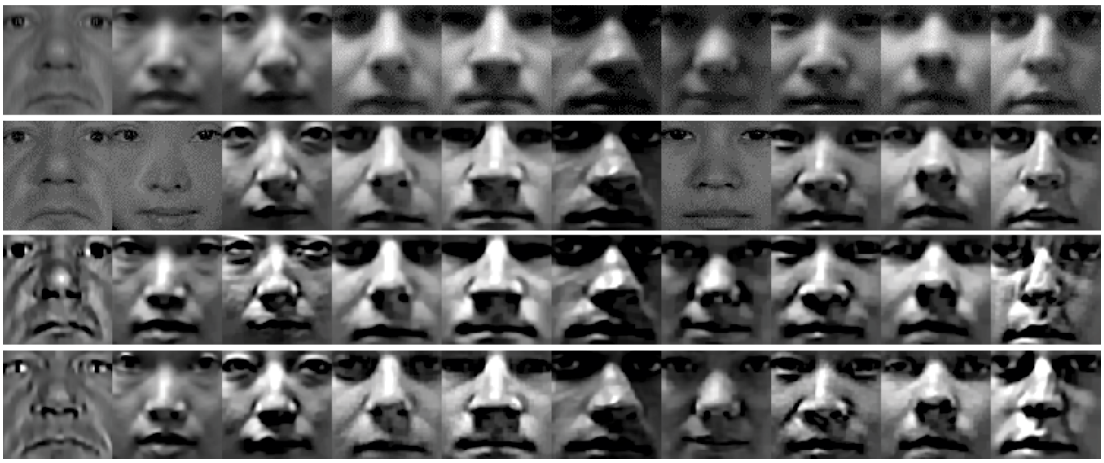


Figure 2.12: The restoration results on FRGC 2.0. From top to bottom, they are OBs, and results of our method, Krishnan *et al.* [55] and Pan *et al* [86], respectively. The results marked by red rectangles are generated by the linear combinations of the function bases.

oretically estimate any PSF due to the reasonable assumptions and the adopted priors. We illustrate how to reduce computational costs for three kinds of symmetric PSF that are common in real applications. To illustrate how subsequent recognition tasks can be improved, we propose a new method that simultaneously generates deconvolution and recognition results. Experimentally, our proposed deconvolution method is superior to representative methods and the recognition method produces high recognition rates for blurred face images. In future work, we will focus on extending our PSF estimation strategy to natural image deblurring and our recognition method to non-frontal face recognition problems [18].

Table 2.7: SSIM Index on Extended Yale B Dataset

	μ		σ	
PSF	TYPE I	TYPE II	TYPE I	TYPE II
Krishnan *et al.* [55]	0.5379	0.6428	0.0712	0.0924
Pan *et al.* [86]	0.8514	0.8112	0.0897	0.0926
Ours	0.8756	0.8531	0.1273	0.1857

Table 2.8: Recognition rates

	PIE		Yale B	
PSF	TYPE I	TYPE II	TYPE I	TYPE II
JRR [127]	90.0	85.1	82.8	77.4
Ours	94.9	91.9	90.3	83.5

Chapter 3

Global Hashing System for Fast Image Search

Hashing methods have been widely investigated for fast approximate nearest neighbor searching in large datasets. Most existing methods use binary vectors in lower dimensional spaces to represent data points that are usually real vectors of higher dimensionality. We divide the hashing process into two steps. Data points are first embedded in a low dimensional space, and the Global Positioning System (GPS) method is subsequently introduced but modified for binary embedding. We devise data-independent and data-dependent methods to distribute the satellites at appropriate locations. Our methods are based on finding the tradeoff between the information losses in these two steps. Experiments show that our data-dependent method outperforms other methods in different-sized datasets from 100K to 10M. By incorporating the orthogonality of the code matrix, both our data-independent and data-dependent methods are particularly impressive in experiments on longer bits.

3.1 Introduction

Hashing methods are efficient for approximate nearest neighbor (ANN) searching that is important in computer vision [15] [113] [128] [103] and machine learning [65] [74] [95] [114]. Hashing methods map original input data points to binary

hashing codes while preserving their mutual distances; that is, the binary strings of similar data points in the original feature space should have small Hamming distances. Hashing with short codes can substantially reduce storage requirements and boost the ANN searching speed.

Generally, hashing methods embed high-dimensional real vectors to low dimensional binary vectors. It can be divided into two steps: dimension reduction and binary embedding. We find that there is a tradeoff between the information losses in these two steps. For example, if the dimensionality is reduced to 1 by Principle Component Analysis (PCA), each data point can be represented by a real number. Although we can represent a real number by a 64-bit binary vector with neglectable information loss, obviously the real numbers achieved in this way cannot preserve the structure of original data points. On the other hand, if the dimensionality is reduced to 64, we can generate efficient hashing codes by some state-of-the-art hashing methods, such as Iterative Quantization (ITQ) [125]. In both cases, the original data points are embedded to 64-bit binary vectors. However, different settings lead to completely different situations. In this chapter, we devise our methods by considering this tradeoff.

We first reduce the dimensionality of the original data points, i.e., the descriptor vectors, by PCA. Dimensionality reduction is used to pre-process the data. Hence, any kinds of dimensionality reduction methods can be used. PCA is used here because of its simplicity and popularity. Next, the projections on the first d principal components are encoded by c-dimensional binary codes, where $c > d$. We need an over-determined system that can uniquely position every data point. This is similar to Global Positioning Systems (GPS) [42] that use dozens of satellites to position a receiver on the $2D$ Earth surface. Since our method is directly inspired by GPS, we name it the Global Hashing System (GHS). We tackle the major issue of how to distribute satellites and propose two methods: one data-dependent method and one data-independent method. Unlike most existing methods [125] [115] [50] that handle the degraded version of orthogonality of the code matrix in the continuous domain, both our methods approximate the orthogonal code matrix directly in the binary domain, which leads to better performance on long-bit experiments. Note that although spectral hashing (SH) can be regarded as assigning more bits to PCA directions along which the data

has greater ranges, it is somewhat heuristic [125].

After satellites are well distributed, distances from data points to each satellite are sorted separately (to simplify the following discussion, this distance is denoted as D2S hereafter). The nearest half is denoted as -1, while the other half is denoted as 1. Hence, our method can generate a balanced code matrix. Although a balanced code matrix is considered to be one of the two conditions for good codes [115], it is rarely considered because it usually results in an NP-hard problem.

3.2 Related Work

Popular hashing methods can be categorized into two groups according to their dependence on data. The most well-known data-independent hashing methods are Locality-Sensitive Hashing (LSH) [5] and its variances, *e.g.*, those adopting cosine similarity [13] and kernel similarity [58]. The main drawback of these methods is the demand of more bits per hashing table, due to randomized hashing [94].

Data-dependent methods have become popular in the machine learning community. Spectral Hashing (SH) [115], one of the most popular data-dependent methods, generates hashing codes by solving a relaxed mathematical problem to circumvent computation of pairwise distances in the whole dataset, i.e, the affinity matrix and the constraints that lead to an NP-hard problem. Anchor Graph Hashing (AGH) [73] optimizes the object function of SH by using anchor points to construct a highly sparse affinity matrix. Discrete Graph Hashing (DGH) [72] follows this idea and incorporates orthogonality of the hashing code matrix. There are also methods based on the linear projections of PCA [125] [54] [50] or Linear Discriminant Analysis [100] and those hashing in kernel space, such as binary reconstructive embeddings (BRE) [57], random maximum margin hashing (RMMH) [49] and kernel-based supervised hashing (KSH) [113]. Unlike ITQ [125] that rotates the projection matrix obtained by PCA to minimize the loss function, Neighborhood Discriminant Hashing (NDH) [102] incorporates computation of the projection matrix during the minimization procedure. In general,

linear dimensionality reduction techniques, such as PCA, are inferior to nonlinear manifold learning methods that are able to more effectively preserve the local structure of input data without assuming global linearity [101]. However, nonlinear manifold techniques may be intractable for large datasets because of their high computation costs. To address this problem, Inductive Manifold Hashing (IMH) [94] [92] learns the nonlinear manifold on a small subset and inductively inserts the remaining data. Hashing methods focusing on image representations have been developed recently. For example, Zhang *et al.* [130] unify feature extraction and hashing function learning. Zhang *et al.* [129] and Liu *et al* [70] develop their methods on multiple representations.

3.3 Methodology

Let us define the used notations. A set of n data points in a D-dimensional space is represented by $\{\mathbf{x}_1, ..., \mathbf{x}_n\}$, $\mathbf{x}_i \in \mathbb{R}^D$ which form the rows of the data matrix $\mathbf{X} \in \mathbb{R}^{n \times D}$. $\mathbf{W} \in \mathbb{R}^{D \times d}$ is obtained by the first d eigenvectors of the data covariance matrix $\mathbf{X}^\top \mathbf{X}$. $\mathbf{Y} = \mathbf{X}\mathbf{W}$ and $\mathbf{y}_i$ is the ith row vector of $\mathbf{Y}$. A binary code corresponding to $\mathbf{x}_i$ is defined by a row vector $\mathbf{b}_i = \{-1, +1\}^c$, where c is the length of the code and the code matrix $\mathbf{B} = \left[\mathbf{b}_1^\top, ..., \mathbf{b}_c^\top\right]^\top$.

3.3.1 Global Positioning/Hashing System

A satellite in a GPS (Fig. 3.1 (a)) has the ability to measure the distance between itself and a signal receiver on the Earth surface. This results in a circle on which every point has the same distance to this satellite as the receiver. Hence, at least three satellites are needed to determine the true position which is the unique intersection of three such circles. More generally, a d-dimensional point can be determined by its Euclidean distances to $d + 1$ other points in this space [1].

In our GHS, each satellite only has 1-bit to record the Euclidean distances. The receivers far from a satellite are denoted as 1 while the nearby ones are

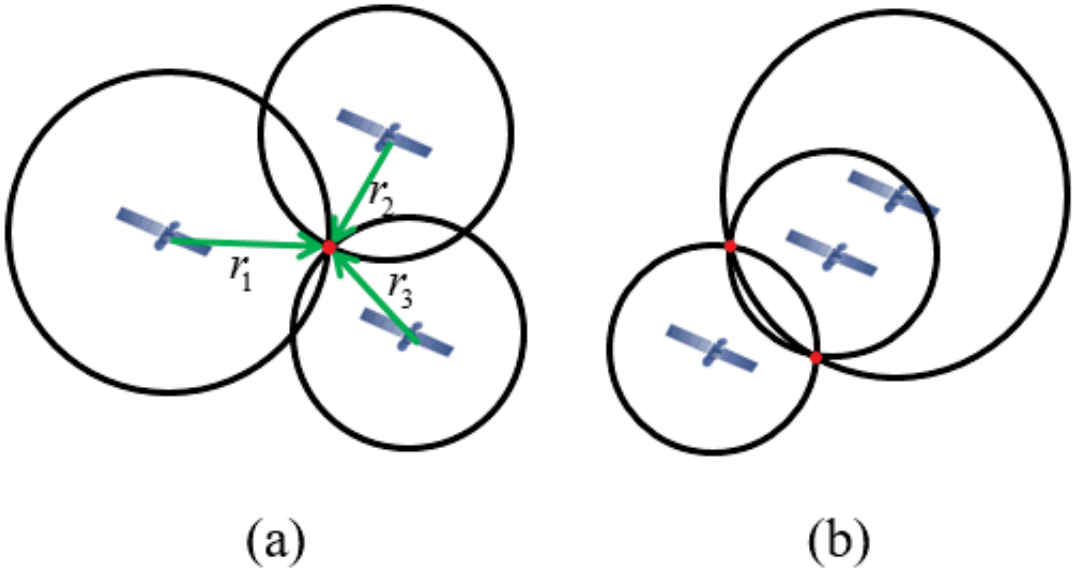


Figure 3.1: Illustration of a GPS. A satellite broadcasts its current time to the receiver (red spot). The distance is calculated by multiplying travel velocity of electromagnetic waves with the difference of the receivers' current time and the received satellite time. (a) The distances of a receiver to three satellites can uniquely determine its location on the Earth surface. (b) Such distributed satellites fail to uniquely determine the receiver's location.

denoted as -1. Hence, our hashing function can be defined as:

$$h\left(\mathbf{y}_i - \mathbf{s}_j\right) = \begin{cases} -1, & \left\|\mathbf{y}_i - \mathbf{s}_j\right\| \leq f\left(\left\|\mathbf{Y} - \mathbf{1}^{n\times 1}\mathbf{s}_j\right\|_c\right) \\ +1, & \left\|\mathbf{y}_i - \mathbf{s}_j\right\| > f\left(\left\|\mathbf{Y} - \mathbf{1}^{n\times 1}\mathbf{s}_j\right\|_c\right) \end{cases}, \tag{3.1}$$

where $\|\mathbf{A}\|_c$ computes the Frobenius norm of each row of $\mathbf{A}$ and f can be any proper functions that return a positive real number. Here $median()$ is adopted to generate a balanced code matrix. $\mathbf{s}_j$ is the coordinate of the jth satellite and it forms up the jth row of satellite matrix $\mathbf{S}$.

3.3.2 Data-dependent method (GHS-DD)

Since we are assuming the Euclidean distance in the feature space correlates with the Hamming distance in hashing code space, our hashing model can be described

as:

$$\arg\min_{\{\mathbf{s}_j\}} \sum_{i=1}^{n-1} \sum_{i'=i+1}^{n} e^{-\|\mathbf{y}_i - \mathbf{y}_{i'}\|^2} \left(\sum_{p=1}^{c} \|h(\mathbf{y}_i - \mathbf{s}_j) - h(\mathbf{y}_{i'} - \mathbf{s}_j)\| \right). \tag{3.2}$$

Randomly setting $\mathbf{s}_j$ does not produce satisfactory results. Furthermore, Eq. (4.10) requires the pairwise distance between each pair of data points, which leads to a heavy burden in storage and computation. Inspired by ITQ, we circumvent it by minimizing the quantization loss.

At first, let us consider the following quantization loss:

$$\arg\min_{B_{ij}\in\{-1,1\},\mathbf{s}_j} \sum_{i=1}^{n} \sum_{j=1}^{c} \left(\frac{B_{ij}+1}{2} - \|\mathbf{y}_i - \mathbf{s}_j\| \right)^2. \tag{3.3}$$

Because $\|\mathbf{y}_i - \mathbf{s}_j\|$ is always non-negative, we scale and shift B to $[0, 1]$. The underlying reasonability of Eq. (3.3) is similar to ITQ. To uniquely position a data point in d-dimensional space, at least $d+1$ satellites are required and the locations of these satellites should satisfy the following condition [1]:

$$\operatorname{rank}\left(\begin{bmatrix} \Gamma & \theta \end{bmatrix} \right) = d, \tag{3.4}$$

where $\Gamma = [\mathbf{s}_2; ...; \mathbf{s}_{d+1}]$ and $\theta = [\mathbf{s}_2 - \mathbf{s}_1; ...; \mathbf{s}_{d+1} - \mathbf{s}_1]$. Eq. (3.4) is called the existence and uniqueness condition for GPS solution [1]. Fig. 3.1 (b) shows an example that a GPS fails to get a unique solution. The condition can be satisfied by initializing an orthogonal Γ. We create g groups of satellites. Within each group, there are $d+1$ satellites, d of which are orthogonal to each other. We define $\rho := c/(d+1)$, a parameter discussed in Section 3.3.5. Note that no more than d mutual orthogonal vectors in a d-dimensional space can be found. Each group is rotated by an orthogonal matrix $\mathbf{R}_k$ to find the best location, which gives the following model:

$$\begin{aligned} \arg\min_{\substack{B_{ij}\in\{-1,1\} \\ \beta_j,\alpha_j,\mathbf{R}_k}} \; & E = \sum_{i=1}^{n} \sum_{j=1}^{c} \sum_{k=1}^{g} \delta_k(\mathbf{s}_j) \left(B_{ij} + \beta_j - \alpha_j \|\mathbf{y}_i - \mathbf{s}_j \mathbf{R}_k\| \right)^2 \\ & s.t. \qquad \mathbf{1B} = \mathbf{0},\ {\mathbf{R}_k}^\top \mathbf{R}_k = \mathbf{I}, \end{aligned} \tag{3.5}$$

where δ_k is an indicator function. $\delta_k(\mathbf{s}_j) = 1$, if $\mathbf{s}_j \in$ Group k and $\delta_k(\mathbf{s}_j) = 0$, if $\mathbf{s}_j \notin$ Group k. α_j and β_j are used to transform the values of D2S into a proper interval. Eq. (3.5) is minimized by iterative minimization.

3.3.3 Optimization

Initialization. In each group, Γ is initialized by the left singular vectors of a $d \times d$ random matrix, so does $\mathbf{R}_k$. Another random $1 \times d$ vector is added into each group.

Update B_{ij}. The jth column of $\mathbf{B}$ is calculated by Eq. (3.1).

Update α_j. Take the partial derivative with respect to α_j, resulting in

$$\alpha_j = \frac{\sum_{i=1}^{n}\sum_{k=1}^{g}\delta_k(\mathbf{s}_j)(B_{ij}+\beta_j)\|\mathbf{y}_i - \mathbf{s}_j\mathbf{R}_k\|}{\sum_{i=1}^{n}\sum_{k=1}^{g}\delta_k(\mathbf{s}_j)\|\mathbf{y}_i - \mathbf{s}_j\mathbf{R}_k\|^2}. \tag{3.6}$$

Update β_j. Similar to α_j,

$$\beta_j = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{g}\delta_k(\mathbf{s}_j)(\alpha_j\|\mathbf{y}_i - \mathbf{s}_j\mathbf{R}_k\| - B_{ij}). \tag{3.7}$$

Please note when we deduce Eq. (3.7), $\sum_{k=1}^{g}\delta_k(\mathbf{s}_j) = 1$ is applied.
Update $\mathbf{R}_k$. We divide this step to two sub-problems. First, $\mathbf{s}_j\mathbf{R}_k$ is substituted by $\mathbf{s}'_j$ to form up the following minimization problem:

$$\arg\min_{\mathbf{s}'_j}\sum_{i=1}^{n}\sum_{j=1}^{c}\left(B_{ij}+\beta_j-\alpha_j\left\|\mathbf{y}_i-\mathbf{s}'_j\right\|\right)^2, \tag{3.8}$$

which is equivalent to

$$\arg\min_{\mathbf{s}'_j}\sum_{i=1}^{n}\sum_{j=1}^{c}\left(B'_{ij}-\left\|\mathbf{y}_i-\mathbf{s}'_j\right\|\right)^2. \tag{3.9}$$

where $B'_{ij} = (B_{ij} + \beta_j)/\alpha_j$. If we treat $\mathbf{s}'_j$ as a receiver, $\mathbf{y}_i$ as the satellites and B'_{ij} as the D2S, the solution of Eq. (3.9) is the standard solution of GPS [6].

After $\mathbf{s}'_j$s are calculated, $\mathbf{R}_k$ is found by minimizing the following problem:

$$\arg\min_{\mathbf{R}_k} \sum_{j=1}^{c} \delta(\mathbf{s}_j) \left\| \mathbf{s}'_j - \mathbf{s}_j \mathbf{R}_k \right\|. \tag{3.10}$$

We construct the following two matrices for each $\mathbf{s}'_j$: $\overline{\mathbf{Y}} = \left[\mathbf{Y}, \mathbf{B}'_{\cdot j}\right]$ and $\mathbf{Z} = diag\left(\overline{\mathbf{Y}\mathbf{Y}}^{\top}\right)$, where $\mathbf{B}'_{\cdot j}$ represents the jth column of $\mathbf{B}'$ and $diag(\mathbf{A})$ returns a row vector which contains the diagonal elements of $\mathbf{A}$. Let $\overline{\mathbf{Y}}^{+} = \left(\overline{\mathbf{Y}}^{\top}\overline{\mathbf{Y}}\right)^{-1}\overline{\mathbf{Y}}^{\top}$. Then solve the following quadratic equation about Λ:

$$\Lambda^2\left(\overline{\mathbf{Y}}^{+}\mathbf{1}\right)^{\top}\left(\overline{\mathbf{Y}}^{+}\mathbf{1}\right) + 2\Lambda\left(\left(\overline{\mathbf{Y}}^{+}\mathbf{Z}^{\top}\right)^{\top}\left(\overline{\mathbf{Y}}^{+}\mathbf{1}\right) - 1\right) + \left(\overline{\mathbf{Y}}^{+}\mathbf{Z}^{\top}\right)^{\top}\left(\overline{\mathbf{Y}}^{+}\mathbf{Z}^{\top}\right) = 0. \tag{3.11}$$

Eq. (3.11) usually has two solutions Λ_1 and Λ_2, therefore two possible $\overline{\mathbf{s}'_j}$ can be found by $\overline{\mathbf{s}'_j} = \overline{\mathbf{Y}}^{+}\left(\mathbf{Z}^{\top} + \Lambda\mathbf{1}\right)$, where $\overline{\mathbf{s}'_j} = \left[\mathbf{s}'_j, \tau\right]$ and τ which is useless in our model is related to D2S. To automatically choose a suitable $\overline{\mathbf{s}'_j}$ from two solutions, we initialize $\mathbf{s}_j$ with $\|\mathbf{s}_j\| = r_s$, where r_s is a positive real constant. The $\overline{\mathbf{s}'_j}$ whose norm is closer to r_s is chosen for the following steps. r_s is also used in our data-independent satellite distribution algorithm and discussed in Section 3.3.5 along with parameter ρ.

Eq. (3.10) can be solved by singular value decomposition (SVD). Given $\mathbf{S}'_k$ and $\mathbf{S}_k$ which contain $\mathbf{s}'_j$ and $\mathbf{s}_j$ of Group k, respectively, through SVD, we can get $\mathbf{L}_1\mathbf{V}\mathbf{L}_2^{\top} = {\mathbf{S}'_k}^{\top}\mathbf{S}_k$ and $\mathbf{R}_k = \mathbf{L}_2\mathbf{L}_1^{\top}$.

Convergence. When $\left|E^{k-1} - E^k\right| < \varepsilon$ or the maximum iteration is reached, the algorithm is terminated, where ε is a small positive real constant.

Output. $\mathbf{S}$ and thresholds, *i.e.*, $g\left(\left\|\mathbf{Y} - \mathbf{1}^{n\times 1}\mathbf{s}_j\right\|_c\right)$ in Eq. (3.1).

Out-of-Sample Hashing. A new query is projected by $\mathbf{W}$ and then its distance to each satellite $\mathbf{s}_j$ is cut off by $g\left(\left\|\mathbf{Y} - \mathbf{1}^{n\times 1}\mathbf{s}_j\right\|_c\right)$.

3.3.4 Data-independent method (GHS-DI)

Another condition for a good code is uncorrelation [23], *i.e.*, $\mathbf{B}^\top\mathbf{B} = n\mathbf{I}$. A direct way to satisfy this condition is distributing the satellites such that only one is close to each receiver; that is, there is no intersection among all $(\mathbf{s}_j, r_j)$ spheres, where r_j is the minimum radius that includes the nearby data points of $\mathbf{s}_j$. However, in this situation, each receiver only has 1-bit 1. The Hamming distance between any pair of receivers is 0 or 2, which means the distance between two data points in the input space is not well preserved. If we strictly satisfy the balance condition as well as uncorrelation condition in this way, at most 2 satellites can be used.

An alternative way is minimizing the intersections of $(\mathbf{s}_j, r_j)$ sphere and $(\mathbf{s}_{j'}, r_{j'})$ sphere for any $j \neq j'$. That is, we set a tolerance for the values of non-diagonal elements of $\mathbf{B}^\top\mathbf{B}$. They are allowed to be non-zero numbers with small absolute values.

The intersection of two d-dimensional spheres is too difficult to compute; therefore, the pairwise distance between each pair of satellites is maximized. Without constraints, the resulting $\|\mathbf{s}_j\|$ may be $+\infty$. A reasonable constraint is distributing all satellites on the surface of $(\mathbf{0}, r_s)$ sphere. As there is no prior knowledge about the data, we assume data points are uniformly distributed in a $(\mathbf{0}, r)$ sphere. By $\|\mathbf{s}_1\| = ... = \|\mathbf{s}_c\| = r_s$, the D2S of each satellite will be comparable.

Under the abovementioned assumption, minimizing intersections can be achieved by maximizing the pairwise distance between each pair of satellites:

$$\arg\max_{\{\mathbf{s}_j\}} E := \sum_{j=1}^{c-1} \sum_{j'=j+1}^{c} \|\mathbf{s}_j - \mathbf{s}_{j'}\|^2 \quad s.t. \quad \|\mathbf{s}_j\|^2 = r_s^2, \forall j. \tag{3.12}$$

Eq. (3.12) can be maximized by the Gradient Projection Algorithm (GPA) [28]. The GPA iteratively updates $\mathbf{s}_j$ by moving $\mathbf{s}_j$ along the gradient direction of E and projects $\mathbf{s}_j$ to the boundary defined by the constraint (Algorithm 3). The

gradient of E with respect to $\mathbf{s}_j$ is

$$\frac{\partial E}{\partial \mathbf{s}_j} = (c-j)\,\mathbf{s}_j - \sum_{j'=j+1}^{c} \mathbf{s}_{j'}. \tag{3.13}$$

The projection step can be directly implemented by normalizing each $\mathbf{s}_j$. As the orthogonality of $\mathbf{B}$ is considered, our GHS-DI method usually produces the second best results on experiments of longer hash bits. Actually, the way that GHS-DD satisfies Eq. (3.4) intrinsically incorporates orthogonality. When $r_s \to +\infty$, the hyper-sphere surface that separates near and far data points can be treated as a hyper-plane. In this situation, with orthogonal $\{\mathbf{s}_j\}$ and the assumption of uniform distribution of data points, this property is easy to understand in $2D$ and $3D$ cases. More generally, we have the following theorem.

Theorem 1. *If (1) data points $\mathbf{y}_i \in \mathbb{R}^d$ are uniformly distributed in a $(\mathbf{0}, r)$ sphere, (2) $\mathbf{s}_j \perp \mathbf{s}_{j'}$ and (3) $r_s \to +\infty$, then $\mathbf{h}_j^\top \mathbf{h}_{j'} = 0 (j \neq j')$, where $\mathbf{h}_j$ and $\mathbf{h}_{j'}$ are column vectors whose elements are the binary hashing codes generated by Eq.* (3.1).

Proof. Since the data points are uniformly distributed in a $(\mathbf{0}, r)$ sphere, without losing generality, let us set $\mathbf{s}_j = r_s(1, 0, 0, ..., 0)^d$ and $\mathbf{s}_{j'} = r_s(0, 1, 0, ..., 0)^d$. In Eq. (1), if $\|\mathbf{y}_i - \mathbf{s}_j\| > r_s$, the ith element of $\mathbf{h}_j$ will be set to 1, otherwise it will be set to -1. For any two points $\mathbf{y}_i$ and $\mathbf{y}_j$ that satisfy $\|\mathbf{y}_i - \mathbf{s}_j\| = \|\mathbf{y}_j - \mathbf{s}_j\| = r_s$, we have $(\mathbf{y}_i - \mathbf{s}_j)(\mathbf{y}_j - \mathbf{s}_j)^\top / r_s^2 = 1$, when $r_s \to +\infty$. That is, $\cos\theta \to 1$ which implies $\theta \to 0$, where θ is the angle between two unit vectors along $\mathbf{y}_i - \mathbf{s}_j$ and $\mathbf{y}_j - \mathbf{s}_j$, respectively. Hence, $\mathbf{y}_i$ and $\mathbf{y}_j$ locate on a plane $\mathcal{P}$ whose distance to $\mathbf{s}_j$ is r_s.

Algorithm 2 Data-Independent Satellite Distribution Algorithm

Input: $\mathbf{S} \in R^{c \times d}$

1: **while** E not converged **do**
2: $\quad \mathbf{s}_j^{t+1/2} = \mathbf{s}_j^t + \Delta t \partial E / \partial \mathbf{s}_j^t$
3: $\quad \mathbf{s}_j^{t+1} = r_s \mathbf{s}_j^{t+1/2} / \left\| \mathbf{s}_j^{t+1/2} \right\|$
4: **end while**

Output: $\mathbf{S}$

To generate a balanced $\mathbf{h}_j$, $\mathcal{P}$ should cross the origin and perpendicular to $\mathbf{s}_j$. Since $\mathbf{s}_j \perp \mathbf{s}_{j'}$, $\mathcal{P}$ is also perpendicular to $\mathcal{Q}$ which corresponds to $\mathbf{s}_{j'}$. It is evident that $\mathcal{P}$ and $\mathcal{Q}$ separate the $(\mathbf{0}, r)$ sphere into four parts with equal volume:

$$\begin{cases} \{\mathbf{y}_i | \, \|\mathbf{y}_i - \mathbf{s}_j\| > r_s\} \cap \{\mathbf{y}_i | \, \|\mathbf{y}_i - \mathbf{s}_{j'}\| > r_s\} \\ \qquad \mathbf{h}_j(i) = 1, \mathbf{h}_{j'}(i) = 1 \\ \{\mathbf{y}_i | \, \|\mathbf{y}_i - \mathbf{s}_j\| > r_s\} \cap \{\mathbf{y}_i | \, \|\mathbf{y}_i - \mathbf{s}_{j'}\| < r_s\} \\ \qquad \mathbf{h}_j(i) = 1, \mathbf{h}_{j'}(i) = -1 \\ \{\mathbf{y}_i | \, \|\mathbf{y}_i - \mathbf{s}_j\| < r_s\} \cap \{\mathbf{y}_i | \, \|\mathbf{y}_i - \mathbf{s}_{j'}\| > r_s\} \\ \qquad \mathbf{h}_j(i) = -1, \mathbf{h}_{j'}(i) = 1 \\ \{\mathbf{y}_i | \, \|\mathbf{y}_i - \mathbf{s}_j\| < r_s\} \cap \{\mathbf{y}_i | \, \|\mathbf{y}_i - \mathbf{s}_{j'}\| < r_s\} \\ \qquad \mathbf{h}_j(i) = -1, \mathbf{h}_{j'}(i) = -1 \end{cases}. \tag{3.14}$$

Since there are equal number of data points in these four parts, it is easy to verify that $\mathbf{h}_j^\top \mathbf{h}_{j'} = 0$. □

In **Theorem** 1, condition (1) and (3) are impractical and therefore only the second sufficient condition can be satisfied by setting $c = d$; however, this contravenes and the existence and uniqueness condition for GPS solution. In Section 3.3.5, we will show $c = d$ usually cannot generate the best results. Although our methods cannot exactly fulfill these three conditions, its superiority of considering orthogonality was proven by its high F-measure in experiments on longer bits (Section 3.5).

3.3.5 Parameters r_s and ρ

There are two key parameters in our methods - r_s and ρ. r_s should not be too small. Consider an extreme example where $r_s = 0$, then bits of points close to the origin will equal 0 and bits of other points will equal 1. Obviously, such codes are inefficient.

ρ should be moderate. If ρ is too large, the binary codes will gradually lose their ability to encode the values of projections which are real numbers. When ρ becomes small, fewer projections can be used, so data points reconstructed by these projections cannot approximate the original ones accurately enough.

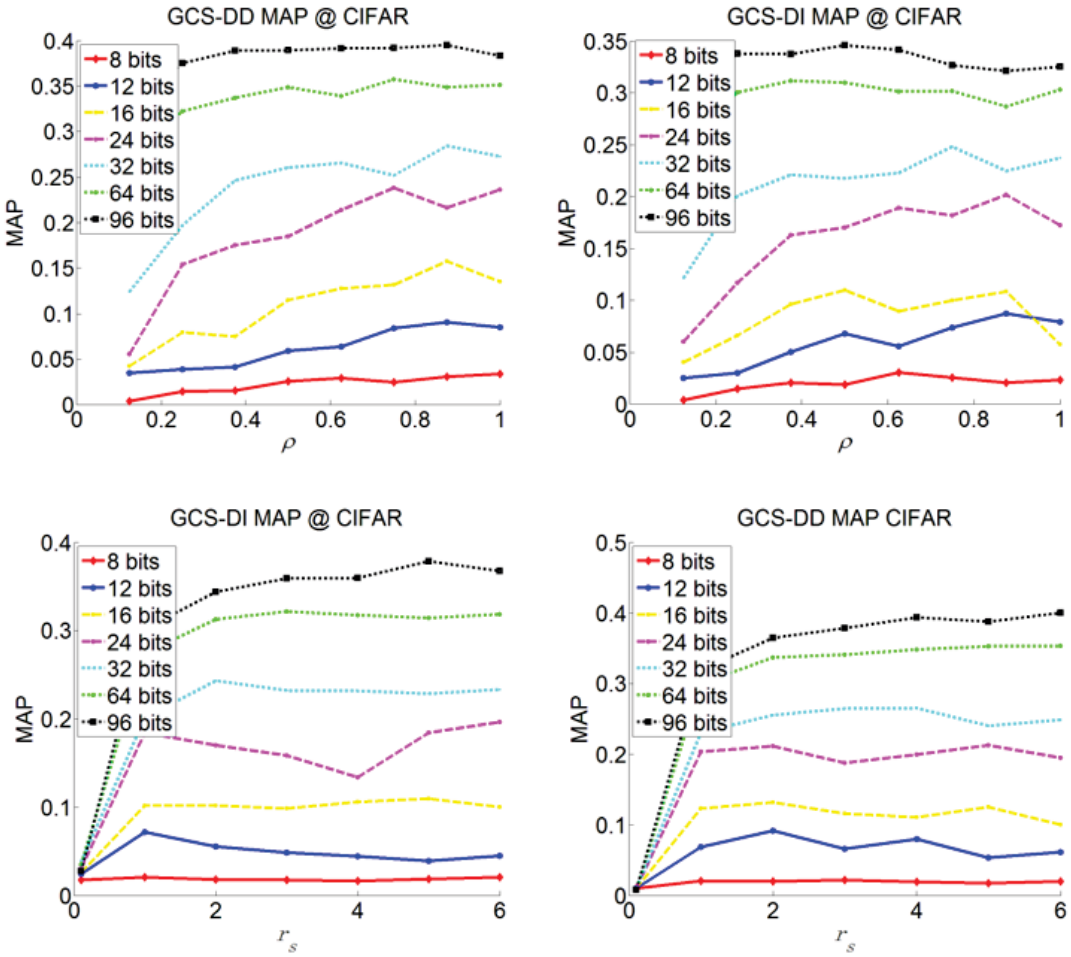


Figure 3.2: MAP on CIFAR-10 dataset for GHS-DI and GHS-DD. When r_s approximates 0, both methods fail to get satisfactory results. The performance of both methods become stable after r_s is larger than 1. On the other hand, GHS-DI gets its best results when ρ is in the interval $[0.5, 1]$, while it is $[0.7, 1]$ for GHS-DD. For $c < 16$, the best results appear when ρ approximates 1 because enough amounts of principal components should be selected.

The mean average precision (MAP) on CIFAR-10 dataset [56] with varying r_s and ρ is shown in Fig. 3.2. CIFAR-10 consists of 60K images from the 80 Million Tiny Image dataset [103] and we use a 1024-dimensional GIST descriptor to represent each image. Their PCA projections are normalized by the largest Euclidean norm of all projected data. When testing on different ρs , at most

one group containing less than $d+1$ satellites may exist. Based on the results in Fig. 3.2, we empirically set r_s as 2 for all experiments and set ρ as 1 for experiments whose $c \leq 16$, while 0.5 for others.

We also tested our two methods by setting $c = d$ (Table 3.1). The percentages shown in Table 3.1 denote the improvement by setting $c = d + 1$. Referring to Table 3.1, we observe that for $c > 16$, both methods perform $1\% - 8\%$ better with $c = d + 1$, suggesting that the existence and uniqueness condition for GPS solution is important. For experiment on $c \leq 16$, the situation is the opposite, because the number of PCA projections is too small and its effect dominates results. However, differences are slight in these cases (less than 1%), so we did not use parameter setting $c = d$ in experiments of Section 4.

Table 3.1: MAP @ CIFAR-10 for parameter setting $c = d + 1$ and $c = d$

		8	12	16	24
GHS-DD	$c = d$	0.1890	0.2232	0.2392	0.2761
	$c = d + 1$	0.1884	0.2214	0.2412	0.2806
		-0.32%	-0.81%	0.83%	1.60%
GHS-DI	$c = d$	0.1543	0.1838	0.2079	0.2581
	$c = d + 1$	0.1537	0.1861	0.2098	0.2688
		-0.39%	1.24%	0.91%	3.98%

		32	64	96	128
GHS-DD	$c = d$	0.3053	0.3816	0.4131	0.4352
	$c = d + 1$	0.3089	0.3972	0.4324	0.4506
		1.17%	3.93%	4.46%	3.54%
GHS-DI	$c = d$	0.2757	0.3474	0.4018	0.4223
	$c = d + 1$	0.3008	0.3653	0.4144	0.4410
		8.34%	4.90%	3.04%	4.43%

3.4 Relations to Existing Methods

During the past several years, many data-dependent hashing methods have been proposed. In this section, those related to our proposed methods are briefly reviewed.

3.4.1 Iterative Quantization (ITQ)

Gong *et al.* [125] formulated ITQ as a minimization problem:

$$\arg\min_{\mathbf{B},\mathbf{R}} \|\mathbf{B} - \mathbf{XWR}\|_F^2 . \tag{3.15}$$

Eq. (3.18) is minimized by iteratively updating $\mathbf{B}$ and $\mathbf{R}$. $\mathbf{R}$ is required to be orthogonal, which can be considered as a rotation to $\mathbf{W}$. IsoH [54] is directly derived from ITQ by finding a projection with equal variances for different dimensions. HH [119] rotates $\mathbf{W}$; however, unlike ITQ, it uses an auxiliary variable for the code matrix during iterative optimization and places an orthogonal constraint on it. Then, the auxiliary variable is thresholded to generate code matrix. ok-means [84] rotates and scales $\mathbf{B}$ to minimize the quantization loss. Our method rotates $\mathbf{S}$ and scales D2S. ITQ, IsoH and HH use principal components whose number is exactly equal to the bit length of hashing codes. That is, they cannot be used to produce hashing code that is longer than the data dimension. Our methods can produce hashing codes of arbitrary lengths.

3.4.2 Inductive Hashing on Manifolds (IMH)

IMH [94] first generates the Base matrix $\mathbf{C}$ by K-means clustering. Each column of $\mathbf{C}$ corresponds to a cluster center. Next, it embeds $\mathbf{B}$ into low dimensional space by manifold learning methods [106] [41]. The embedding methods affect the performance of IMH. Throughout this chapter, t-SNE [106] is used because it achieved the best results in the authors' experiments [94]. Finally, the embedding for the training data is calculated by

$$\mathbf{Y} = \overline{\mathbf{W}}_{\mathbf{XB}}\mathbf{Y}_{\mathbf{B}}, \tag{3.16}$$

where the element $\overline{\mathbf{W}}_{ij}$ in $\overline{\mathbf{W}}_{\mathbf{XB}}$ is defined as

$$\overline{\mathbf{W}}_{ij} = \frac{\exp\left(-\|\mathbf{x}_i - \mathbf{c}_j\|^2/\sigma^2\right)}{\sum_{i=1}^{m} \exp\left(-\|\mathbf{x}_i - \mathbf{c}_j\|^2/\sigma^2\right)}. \tag{3.17}$$

where $\mathbf{c_j}$ is the jth column of $\mathbf{C}$. Eq. (3.17) is quite similar to the membership in fuzzy c-means clustering [7]. The embedding for the training data is a linear combination of the embedding for $\mathbf{C}$. In our method, each satellite encodes 1-bit according to the distances from itself to the data points and we do not encode the satellites.

3.4.3 Spectral Hashing (SH)

Weiss *et al.* [115] formulated the SH as:

$$\begin{aligned} &\arg\min_{\mathbf{X}} \sum_{\mathbf{x}_i,\mathbf{x}_j\in\mathbf{X}} e^{-\|\mathbf{x}_i-\mathbf{x}_j\|^2/\sigma^2}\|\mathbf{b}_i-\mathbf{b}_j\|^2 \\ &s.t. \quad \mathbf{B}\in\{-1,1\}^{n\times c}, \quad \mathbf{B}^\top\mathbf{B}=n\mathbf{I}, \quad \mathbf{B}^\top\mathbf{1}=0. \end{aligned} \tag{3.18}$$

Eq. (4.10) is similar to Eq. (3.18). The graph affinity matrix $\mathbf{W}$ with $\mathbf{W}_{ij} = \exp\left(-\|\mathbf{x}_i-\mathbf{x}_j\|^2/\sigma^2\right)$ is intractable for large datasets. SH evaluates the c smallest eigenvalues for each PCA direction to create a list of cD eigenvalues, sorts this list to find the c smallest eigenvalues and then thresholds the corresponding eigenfunctions. However, this approach is somewhat heuristic [125]. AGH and DGH compute D2S to form a highly sparse affinity matrix to minimize the modified object function of SH. GHS-DD avoids computation and storage of pairwise distances of all data points by minimizing quantization loss. Our method generates a balanced code matrix but these methods cannot.

3.4.4 Spherical Hashing (SpH)

The final step of SpH [46] is the same as our method. SpH also generates a balanced code matrix. However, SpH searches locations of special points in the entire space, which makes it difficult to find a good solution. The authors claim that the distances between these points should be neither too large nor too small and hence an empirical point-finding procedure was devised that has less theoretical support. With more concrete theoretical analysis, our proposed methods

appear to outperform SpH.

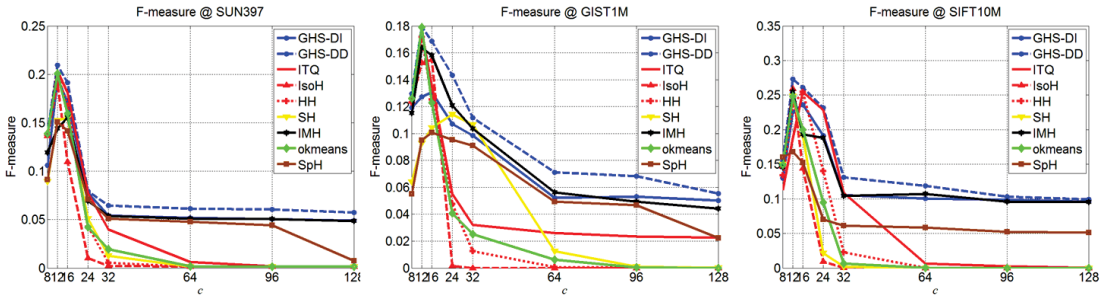


Figure 3.3: Mean F-measure of hash lookup with Hamming radius 2 for different methods on SUN397, GIST1M and SIFT10M.

3.5 Experiments

Our experiments were conducted on three datasets of three different scales: SUN397 [48], GIST1M [47] and SIFT10M. SUN397 contains approximately 108K images and we represent each image by a 512-dimensional GIST descriptor [85]. GIST1M consists of 1 million 960-dimensional GIST descriptors. SIFT10M is a 10 million subset of SIFT1B [47] dataset which consists of 1 billion 128-dimensional SIFT descriptors [77]. The 10 million data points are randomly chosen. 1K images are randomly selected from the whole SUN397 to form a separate test dataset. For GIST1M, there is a 1K test dataset available. For SIFT10M, we randomly selected 1K data points from its 10K test dataset. Groundtruth neighbors for a given query are defined as samples in the top 2% Euclidean distance.

3.5.1 Protocols and baselines

We evaluate our methods by comparison with seven hashing methods that include: Iterative Quantization (ITQ) [125], Isotropic Hashing (IsoH) [54], Harmonious Hashing (HH) [119], Spectral Hashing (SH) [115], Inductive Manifold Hashing (IMH) [94], Orthogonal K-means (ok-means) [84] and Spherical Hashing (SpH) [46]. Our data-dependent and data-independent methods are denoted as

Table 3.2: MAP on SUN397. c denotes the number of hash bits used in hashing methods.

	SUN397			
c	8	12	16	24
GHS-DI	0.1336	0.1744	0.2194	0.2290
GHS-DD	**0.1533**	**0.1945**	**0.2447**	**0.2746**
ITQ	0.1508	0.1859	0.2301	0.2619
IsoH	0.1420	0.1677	0.1881	0.1950
HH	0.1478	0.1866	0.2213	0.2554
SH	0.1219	0.1369	0.1475	0.1705
IMH	0.1296	0.1357	0.1533	0.2453
okmeans	0.1469	0.1852	0.2136	0.2524
SpH	0.0377	0.0359	0.0364	0.0365
	SUN397			
c	32	64	96	128
GHS-DI	0.2579	0.3167	0.3588	0.3860
GHS-DD	**0.2998**	**0.3492**	**0.3880**	**0.4096**
ITQ	0.2886	0.3317	0.3592	0.3750
IsoH	0.2278	0.2578	0.2873	0.2882
HH	0.2687	0.3253	0.3543	0.3739
SH	0.1758	0.1897	0.2180	0.2206
IMH	0.2689	0.2896	0.3077	0.3990
okmeans	0.2716	0.3248	0.3507	0.3658
SpH	0.0363	0.0599	0.0942	0.2578

GHS-DD and GHS-DI. We use publicly available codes of the compared methods and follow parameter settings suggested by corresponding publications. All data are zero-centered and their PCA projections are normalized by the largest Euclidean norm of all projected data in our methods. Two types of experiments - *Hamming ranking* and *hash lookup* were conducted. The performance of *Hamming ranking* is measured by MAP. F1 score denoted as F-measure is used for evaluating the performance of *hash lookup*, where the F1 score is defined as $2(precision \cdot recall)/(precision + recall)$.

Table 3.3: MAP on GIST1M. c denotes the number of hash bits used in hashing methods.

	GIST1M			
c	8	12	16	24
GHS-DI	0.1245	0.1552	0.1802	0.2052
GHS-DD	**0.1358**	**0.1682**	**0.1952**	**0.2211**
ITQ	0.1260	0.1593	0.1851	0.2098
IsoH	0.1121	0.1310	0.1844	0.1939
HH	0.1207	0.1603	0.1780	0.2019
SH	0.0871	0.0986	0.1033	0.1208
IMH	0.1248	0.1449	0.1748	0.1849
okmeans	0.1239	0.1610	0.1778	0.2070
SpH	0.0369	0.0349	0.0348	0.0359
	GIST1M			
c	32	64	96	128
GHS-DI	0.2191	0.2596	0.2790	0.2885
GHS-DD	**0.2438**	**0.2694**	**0.2854**	**0.2967**
ITQ	0.2269	0.2577	0.2703	0.2775
IsoH	0.2288	0.2579	0.2712	0.2854
HH	0.2247	0.2597	0.2745	0.2880
SH	0.1339	0.1682	0.1781	0.1781
IMH	0.1965	0.2161	0.2385	0.2638
okmeans	0.2201	0.2565	0.2741	0.2809
SpH	0.0356	0.0637	0.0788	0.1919

3.5.2 Quantitative evaluation

The mean average precision (MAP) values are given in Table 3.2-3.4. It can be observed that GHS-DD outperforms all the compared methods. The performance of GHS-DI is worse than ITQ, HH and SH except in 128-bit experiments. Benefitting from the tradeoff between information losses in two steps and a balanced code matrix, GHS-DD exceeds ITQ, IsoH and HH. Due to limitations on computation, SpH works on a small subset of the whole dataset and its empirical satellite distribution algorithm is demonstrated to be less efficient than ours. The F-measure is illustrated in Fig. 3.3. Again, GHS-DD exceeds others. It is worth noticing that GHS-DI generated the second best MAP and F-measure in

Table 3.4: MAP on SIFT10M. c denotes the number of hash bits used in hashing methods.

	SIFT10M			
c	8	12	16	24
GHS-DI	0.1738	0.2193	0.2674	0.3342
GHS-DD	**0.1864**	**0.2339**	**0.2769**	**0.3535**
ITQ	0.1666	0.2195	0.2655	0.3452
IsoH	0.1764	0.2224	0.2469	0.3326
HH	0.1701	0.2258	0.2516	0.3143
SH	0.1704	0.2170	0.2382	0.2708
IMH	0.1833	0.1888	0.2007	0.2254
okmeans	0.1814	0.2260	0.2699	0.3233
SpH	0.0440	0.0487	0.0400	0.0475
	SIFT10M			
c	32	64	96	128
GHS-DI	0.3837	0.5156	0.5569	0.5797
GHS-DD	**0.4098**	**0.5277**	**0.5692**	**0.5889**
ITQ	0.3906	0.5025	0.5522	0.5782
IsoH	0.3766	0.4653	0.5524	0.5695
HH	0.3524	0.4494	0.5163	0.5554
SH	0.2810	0.3148	0.3039	0.3157
IMH	0.2884	0.3052	0.3358	0.3634
okmeans	0.3605	0.4401	0.4538	0.4964
SpH	0.0381	0.0615	0.1721	0.1947

experiments on longer bits ($c > 96$) because GHS-DI considers orthogonality of the code matrix. The way that GHS-DD satisfies the condition of uniqueness and existence of GPS solution, *i.e.*, Eq. (3.4) and its data-dependent property make it work better than GHS-DI.

3.5.3 Computational efficiency

Training and testing time on 32-bit are given in Table 4.2. All experiments were performed on MATLAB R2013b installed on a PC with 2.85 GHz CPU and 128 GB RAM. The major computation cost of GHS-DI is the calculation of D2S at the final step, which is linearly related to the product of data dimension and the

size of the dataset. It takes the least time on GIST1M and SIFT10M. Because GHS-DD computes D2S in every iteration, its computation cost is moderate. When testing a new query, GHS-DI and GHS-DD computes D2S and hence their computation costs are approximate. Although the testing procedure of SpH is similar to ours, it computes D2S in the original input data space, the dimension of which is D; therefore, its testing time is longer.

Table 3.5: Training and average testing time in seconds

	SUN397		**GIST1M**		**SIFT10M**	
	Train	Test	Train	Test	Train	Test
GHS-DI	9.9	2.7×10^{-4}	130.4	3.5×10^{-4}	166.1	1.4×10^{-4}
GHS-DD	24.3	3.2×10^{-4}	212.3	3.5×10^{-4}	1005.1	1.4×10^{-4}
ITQ	14.8	3.1×10^{-5}	142.7	4.5×10^{-5}	322.0	1.3×10^{-5}
IsoH	**9.6**	$\mathbf{3.2 \times 10^{-5}}$	**136.5**	$\mathbf{6.1 \times 10^{-5}}$	**185.6**	$\mathbf{2.0 \times 10^{-5}}$
HH	26.8	2.1×10^{-5}	214.9	3.9×10^{-5}	1307.1	1.3×10^{-5}
SH	9.7	6.5×10^{-4}	119.3	9.2×10^{-4}	202.5	6.2×10^{-4}
IMH	97.4	2.3×10^{-4}	1024.4	2.8×10^{-4}	702.2	2.8×10^{-4}
okmeans	14.0	2.3×10^{-5}	144.5	5.5×10^{-5}	301.2	1.2×10^{-5}
SpH	28.2	3.3×10^{-4}	225.8	4.4×10^{-4}	190.7	2.7×10^{-4}

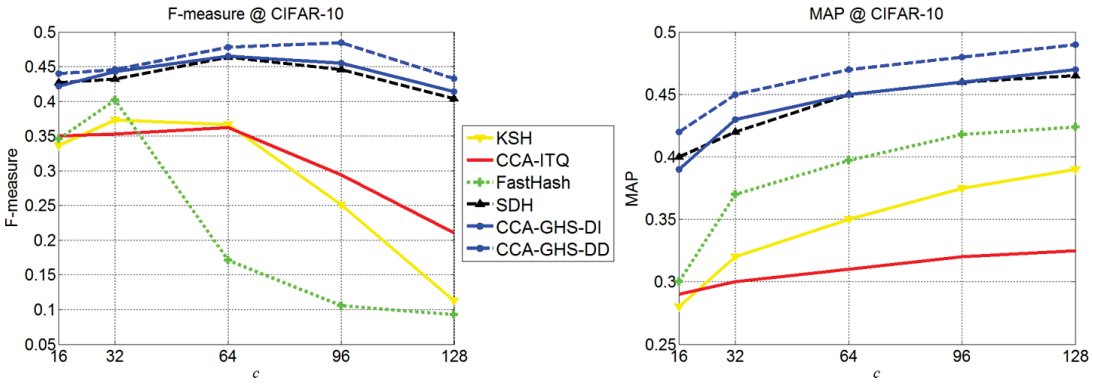


Figure 3.4: Mean F-measure of hash lookup with Hamming radius 2 and MAP for different methods on CIFAR-10.

Figure 3.5: The query images and the query results returned by compared methods with 32 hash bits.

3.5.4 Incorporating label information

To incorporate label information, a supervised dimensionality reduction method can be used to capture the semantic structure of the dataset. Among various supervised dimensionality reduction methods, Canonical Correlation Analysis (CCA) [44] has been proven to be efficient for extracting a common latent space from two views [29] and robust to noise [8].

Let $\mathbf{z_i} \in \{0,1\}^l$ be a label vector, where l is the total number of labels. If the ith image is associated with the corresponding label, $\mathbf{z_i} = 1$ and $\mathbf{z_i} = 0$ otherwise. $\mathbf{Z} \in \{0,1\}^{n \times l}$ is the matrix whose rows are comprised of the label vectors. The goal of CCA is to maximize the correlation between the projected data matrix $\mathbf{Y}$ and the label matrix $\mathbf{Z}$ by finding two projection directions $\mathbf{w_k}$ and $\mathbf{u_k}$. The correlation is defined as:

$$\begin{aligned} C\left(\mathbf{w_k}, \mathbf{u_k}\right) &= \frac{\mathbf{w_k}^\top \mathbf{X}^\top \mathbf{Y} \mathbf{u_k}}{\sqrt{\mathbf{w_k}^\top \mathbf{X}^\top \mathbf{X} \mathbf{w_k} \mathbf{u_k}^\top \mathbf{Y}^\top \mathbf{Y} \mathbf{u_k}}} \\ s.t.\, w_k^\top X^\top X w_k = 1,\, u_k^\top Y^\top Y u_k &= 1. \end{aligned} \tag{3.19}$$

$\mathbf{w_k}$ can be determined by solving the following generalized eigenvalue problem:

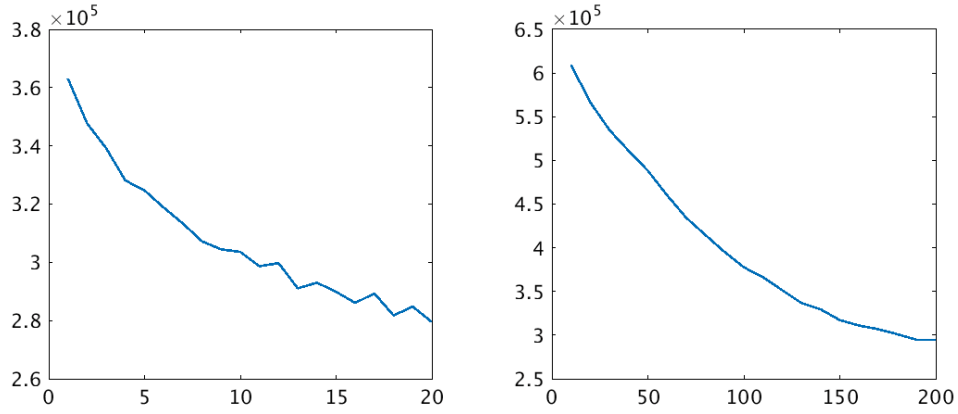
$$\mathbf{X}^\top \mathbf{Y} \left(\mathbf{Y}^\top \mathbf{Y} + \rho \mathbf{I}\right)^{-1} \mathbf{Y}^\top \mathbf{X} \mathbf{w_k} = \lambda_k^2 \left(\mathbf{X}^\top \mathbf{X} + \rho \mathbf{I}\right) \mathbf{w_k}, \tag{3.20}$$

where ρ is a small regularization constant and is set to be 0.0001 here. Just as in the case of PCA, the leading generalized eigenvectors $\mathbf{w_k}$ scaled their corresponding eigenvalues λ_k form up the rows of projection matrix $\widehat{\mathbf{W}} \in \mathbb{R}^{D \times d}$ and we obtain the embedded data matrix $\mathbf{Y} = \mathbf{X}\widehat{\mathbf{W}}$. Finally, both of our data-independent and data-dependent methods can be used to generate hashing codes.

CIFAR-10 dataset is used in this experiment. The 60K images in CIFAR-10 are labeled as 10 classes with 6,000 samples for each class. Again, each image is represented by a 1024 dimensional GIST feature. 1,000 samples are randomly chosen as queries and the remaining samples are used for training. Our proposed supervised hashing methods are denoted as CCA-GHS-DI and CCA-GHS-DD, respectively. The baseline methods are Supervised Discrete Hashing (SDH) [93],

KSH [113], FastHash [67] and CCA-ITQ [125].

The mean F-measure of hash lookup Hamming distance 2 and MAP scores of the compared methods are given in Fig. 3.4. CCA-GHS-DD achieves the best F-measures and MAPs for all code lengths, while CCA-GHS-DI is only a little inferior to SDH for 16-bit code length. In the hash lookup experiments, we found that setting Hamming distance as 2 is favorable for both of our proposed methods because two groups of satellites were used for experiments of $c > 16$. In Fig. 3.5, 5 queries with their corresponding results retrieved by compared methods using 32-bit hashing code are illustrated to qualitatively evaluate the performance. It can be observed that both CCA-GHS-DI and CCA-GHS-DD outperform the compared methods. Quantization loss is shown in Fig. 3.6. It



(a) (b)

Figure 3.6: Quantization loss of each iteration on CIFAR-10. (a) CCA-GHS-DD; (b) CCA-GHS-DI.

can be observed that CCA-GHS-DD converges much faster than CCA-GHS-DI because it incorporates the data information for distributing satellites.

3.5.5 Classification with hashing codes

In this subsection, the MNIST dateset is used for evaluating the performance of the learned hashing codes by compared methods. The MNIST dataset consists of 70, 000 images, each of which is 784-dimensional. These images are handwritten

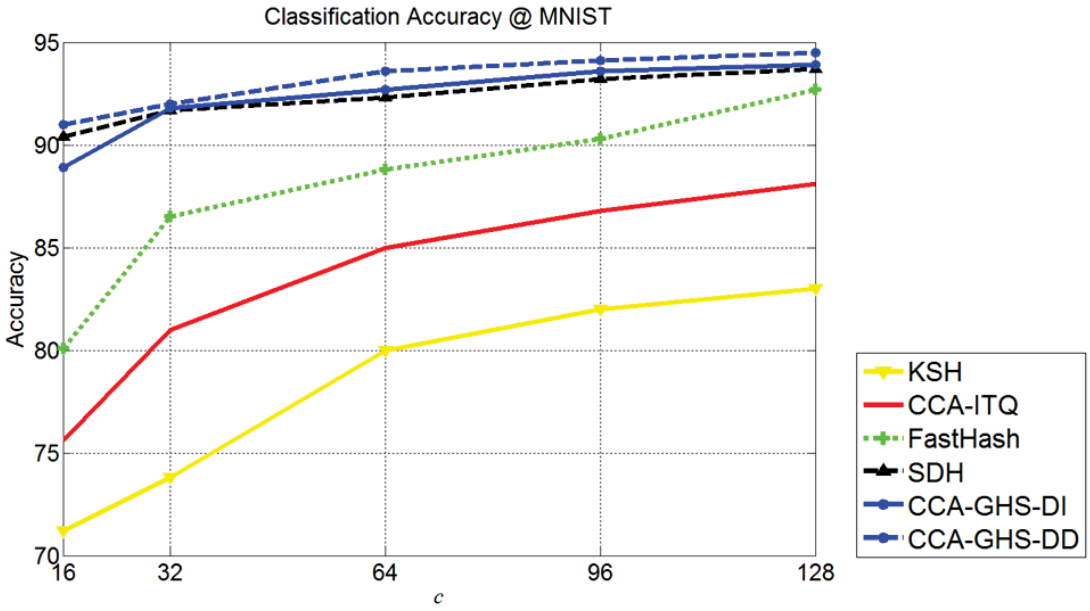


Figure 3.7: Classification accuracy (%) on MNIST.

digits from '0' to '9'. BRE, CCA-ITA, KSH, FastHash and SDH are used as baselines.

The ideal hashing codes are expected to preserve the intra-class distances of original features by Hamming distances. For binary vectors, the Hamming distances are equal to the square of the Euclidean distances; therefore, the ideal hashing codes should be linearly separable. Linear support vector machine (SVM) is applied to the hashing codes. The LIBLINEAR [25] solver is used to train the SVM. The classification results are given in Fig. 3.7. From Fig. 3.7, it can be observed that CCA-GHS-DD gets the highest classification accuracy over all hash bit length, while CCA-GHS-DI is the second best when $c > 16$ but trails SDH in experiments on 16-bit hashing codes.

3.5.6 Performance on facial images

Additional to FERET, FRGC, CMU-PIE and extended Yale B datasets used in Chapter 2, YouTube Faces [116] dataset is used to evaluate the retrieval performance of our method on facial images. YouTube Faces dataset contains 1,595 different people. Following [72], we choose 340 people such that each has at least

500 images to form a subset of 370,319 face images. Each image is represented by a 1,770 dimensional LBP feature vector [2]. The groundtruth neighbors for a query are those belonging to the same identity as the query.

The results can be found in Appendix A. Part of the results on YouTube Faces dataset is reported in Table 5.1 and Table 5.2. It can be seen that our GHS-DD method surpasses all compared methods.

Table 3.6: MAP results on YouTube Faces dataset

	YouTube Faces				
c	8	12	16	20	24
GHS-DI	0.1828	0.2303	0.2808	0.3158	0.3521
GHS-DD	**0.1960**	**0.2453**	**0.2912**	**0.3305**	**0.3709**
ITQ	0.1749	0.2307	0.2791	0.3202	0.3621
IsoH	0.1860	0.2335	0.2589	0.3044	0.3488
HH	0.1788	0.2379	0.2635	0.2977	0.3296
SH	0.1791	0.2275	0.2498	0.2674	0.2843
IMH	0.1925	0.1983	0.2111	0.2241	0.2368
okmeans	0.1908	0.2367	0.2834	0.3112	0.3395
SpH	0.0459	0.0518	0.0419	0.0461	0.0494

Table 3.7: F-measure results on YouTube Faces dataset

	YouTube Faces				
c	8	12	16	20	24
GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
GHS-DD	**0.1843**	**0.2296**	**0.2426**	**0.2362**	**0.2298**
ITQ	0.1654	0.1987	0.2159	0.1703	0.1247
IsoH	0.1648	0.1905	0.2210	0.1917	0.1624
HH	0.1615	0.1872	0.2136	0.1151	0.0166
SH	0.0817	0.1246	0.1498	0.1581	0.1663
IMH	0.1611	0.1698	0.1754	0.1710	0.1667
okmeans	0.1697	0.2136	0.2421	0.1475	0.0529
SpH	0.0799	0.1179	0.1357	0.1311	0.1265

3.6 Conclusion

We propose a novel hashing method based on the tradeoff between information losses in dimension reduction and binary embedding. To circumvent computation of pairwise distances between each pair of data points, we minimize the new formulation of quantization loss which is based on the Global Positioning System (GPS). Data-dependent and data-independent methods are proposed to distribute satellites. According to the experimental results on three scales of datasets, the data-dependent method (GHS-DD) was superior to other methods, and the data-independent method (GHS-DI) produced promising results in less training time. GHS-DD required a moderate length of time to train, and demand on RAM was limited by computation of the covariance matrix in PCA. By incorporating Canonical Correlation Analysis (CCA), the proposed methods can be used for supervised hashing. The performance of CCA-GHS-DI and CCA-GHS-DD are superior. Finally, the retained hashing codes are used for a classification problem to demonstrate the outstanding performance of the proposed methods. Future work will focus on improving computational efficiency and investigating methods to train the model using a few samples from the whole dataset to handle larger datasets, such as SIFT1B and Tiny 80M.

Chapter 4

Learning decorrelated hashing codes for multi-modal Retrieval

In Chapter 3, the labels of facial images are not incorporated in training stage. If labels are treated as another view or modality of the facial images, a multi-modal retrieval method can be used. With label information, a multi-modal retrieval method can achieve higher performance.
Multi-modal hashing embeds heterogeneous data to binary codes, which boosts the nearest-neighbor-search efficiency across multiple modalities. Most existing methods ignore the correlation of the code matrix and hence their performances on long code experiment are incrementally improved or even worse. In this paper, we model multi-modal hashing as a quantization error minimization problem. Then, the so-called minimum correlation regularization is introduced to decorrelate the code matrix in order to improve the retrieval performance on long codes. Experiments validate the effectiveness of the proposed method.

4.1 Introduction

In social networks, heterogeneous multimedia data correlate to each other, such as videos and their corresponding tags in YouTube and image-text pairs in Facebook. Nearest neighbor retrieval across multiple modalities on large data sets becomes a hot yet challenging problem. Hashing is expected to be an efficient

solution, since it represents data as binary codes. As the bit-wise XOR operations can be fast handled, the retrieval time is greatly reduced.

Existing multi-modal hashing methods can be classified into supervised and unsupervised ones according to whether the label information is used. Unsupervised multi-modal hashing aims at preserving the Euclidean distances between each pair of data. Inter-media hashing (IMH) seeks a common Hamming space in which binary codes preserve inter-media consistency and intra-media consistency [97]. To avoid the large-scale graph which needs to compute and store the pairwise distances, linear cross-modal hashing (LCMH) [133] computes distances between each training data point and a small number of cluster centers. Collective matrix factorization hashing (CMFH) [22] uses collective matrix factorization on each modality to learn a unified hashing codes.

By incorporating label information, supervised hashing can preserve semantic information and achieve higher accuracy. Cross-modality similarity-sensitive hashing (CMSSH) [11] treats hashing as a binary classification problem. Cross-view hashing (CVH) [59] assumes the hashing codes be a linear embedding of the original data points. It substitutes the code matrix by this embedding. The objective function is a weighted summation of that of spectral hashing (SH) [115] on each modality. Multilatent binary embedding (MLBE) [132] treats hashing codes as the binary latent factors in the proposed probabilistic model and maps data points from multiple modalities to a common Hamming space. Semantics-preserving hashing (SePH) [68] learns the hashing codes by minimizing the KL-divergence of the probability distribution in Hamming space from that in semantic space. CMSSH, MLBE and SePH need to compute the affinities of all data points, which makes it intractable for large data set. Semantic correlation maximization (SCM) [126] circumvents this by learning only one bit each time and the explicit computation of affinity matrix is avoided through several mathematical manipulations. Multi-modal discriminative binary embedding (MDBE) models [108] hashing as a minimization problem. There are two main terms in its formulation. One term indicates different modalities and the labels can be embedded to the same latent space, while the other one indicates the embedded modalities can be further embedded as the labels. $l2$-norm is used to regularize the linear embedding matrix. SCM and MDBE discard the uncorrelation property of the code

matrix or embedding matrix, which makes their performance improve slowly as code length increases.

In this paper, we propose a new hashing method named Decorrelated Multimodal Hashing (DMH). First, a sigmoid function is applied on the linear transformations of original data points to map different modalities into a common code matrix. Then, we devise a minimum correlation regularization to improve the retrieval performance of long code length experiments. The rest of this paper is organized as follows. In Section 4.2, we, step by step, derive our model from a widely used uni-modal hashing method, Iterative Quantization (ITQ) [125]. The discussions on parameter settings and optimization algorithms are also given in Section 4.2. Experimental results are reported in Section 4.3. We conclude this paper in Section 4.4

4.2 Methodology

Terms "view" and "modality" are discriminated in some literatures [108]. Multiple views of data refers different type of features of one modality, e.g. SIFT [77] and GIST [85] features for images. However, we use these two words interchangeably since our method can be used in any situations as long as the data are represented by real matrices.

First, Let us define the used notations. Suppose that $\mathbf{X}^i$ is the i-th view matrix of the data and $\mathbf{X}^i = [\mathbf{x}_1^i, \ldots, \mathbf{x}_n^i]^\top$, where $\mathbf{x}_m^i \in \mathbb{R}^{d_i}$, n is the number of data points and $i = 1, \ldots, g$. A binary code corresponding to the m-th data is defined by a row vector $b_m = \{0, 1\}^c$, where c is the code length and the code matrix $\mathbf{B} = \left[\mathbf{b}_1^\top, \ldots, \mathbf{b}_n^\top\right]^\top$. $h^i(\mathbf{X}^i)$, the hashing function for the i-th view matrix, embeds $\mathbf{X}^i$ into a binary code matrix.

4.2.1 Problem Formulation

Iterative Quantization (ITQ) [125] is a successful hashing method for single view data. The formulation of ITQ is

$$\arg\min_{\mathbf{B},\mathbf{R}} E = \|\mathbf{B} - \mathbf{XWR}\|_F^2, \tag{4.1}$$

where $\mathbf{X} \in \mathbb{R}^{n\times d}$ is the data matrix, $\mathbf{W} \in \mathbb{R}^{d\times c}$ is obtained by principal component analysis (PCA) and $\mathbf{R} \in \mathbb{R}^{c\times c}$ is an orthogonal matrix. An intuitive multi-view extension of ITQ can be

$$\arg\min_{\mathbf{B},\mathbf{R}^i} E = \sum_i \alpha_i \left\|\mathbf{B} - \mathbf{X}^i\mathbf{W}^i\mathbf{R}^i\right\|_F^2, \tag{4.2}$$

where α_i is a positive real constant. As the maximum number of principal components pre-computed by PCA on the ith view matrix is d_i, Eq. (4.2) cannot be used when $c > d_i$. We remove $\mathbf{R}^i$ from Eq. (4.2). Then, we simultaneously calculate $\mathbf{W}^i \in \mathbb{R}^{d_i\times c}$ and $\mathbf{B}$ during the optimization process. This method can be modeled as

$$\arg\min_{\mathbf{B},\mathbf{W}^i} E = \sum_i \alpha_i \left\|\mathbf{B} - \mathbf{X}^i\mathbf{W}^i\right\|_F^2. \tag{4.3}$$

Because $\mathbf{B}$ is a binary matrix, $h^i(\mathbf{X}^i\mathbf{W}^i) = 1/(1+\exp(-(\beta_i * \mathbf{X}^i\mathbf{W}^i + \mathbf{1}\mathbf{v}^i)))$ is applied to transform the values of $\beta_i * \mathbf{X}^i\mathbf{W}^i + \mathbf{1}\mathbf{v}^i$ into interval $(0,1)$, where $\mathbf{1}$ is a n-dimensional column vector whose elements are equal to 1. β_i is a constant and $\mathbf{v}^i$ is a bias vector. Hence, Eq. (4.3) can be modified as following.

$$\arg\min_{\mathbf{B},\mathbf{W}^i,\mathbf{v}^i} E = \sum_i \alpha_i \left\|\mathbf{B} - \frac{1}{1+\exp\left(-(\beta_i\mathbf{X}^i\mathbf{W}^i + \mathbf{1}\mathbf{v}^i)\right)}\right\|_F^2. \tag{4.4}$$

The orthogonality condition for good codes [115] is approximated by orthogonal $\mathbf{W}$ in ITQ. However, when $c > d_i$, an orthogonal $\mathbf{W}^i$ does not exist. In this case, we use minimum correlation regularization to approximate the orthogonal condition. If ${\mathbf{W}^i}^\top\mathbf{W}^i = I$ where I is the identity matrix, $\mathbf{W}^i$ is an orthogonal

matrix. Thus, we define the minimum correlation regularization (MCR) as:

$$\text{MCR} = \left\| \mathbf{W}^{i\top}\mathbf{W}^{i} - \mathbf{I} \right\|_F \tag{4.5}$$

It is inessential to name Eq. (4.5) as "minimum correlation regularization" or "maximum uncorrelation regularization". Since Eq. (4.5) will be added into our hashing model which is formulated as a minimization problem, we use the former one to keep literal consistency. Unlike the orthogonal units proposed in [110] in which the square of Eq. (4.5) is used, Eq. (4.5) can results in a more sparse $\mathbf{W}^{i\top}\mathbf{W}^{i} - \mathbf{I}$. First, let us discuss some interesting properties of MCR. These properties are useful for understanding our method.

Proposition 1. When $c \leq d_i$, the $\mathbf{W}^i$ that minimizes Eq. (4.5) is an orthogonal matrix.

It is easy to prove **Proposition 1** by the definition of orthogonal matrix.

Proposition 2. Let the $\mathbf{W}^i$ that minimizes Eq. (4.5) consists of column vectors $\mathbf{w}_p^i$ where $p = 1, \ldots, c$. The angle between any pair of column vectors is equal to each other.

Proof. Let $\mathbf{V} = \mathbf{W}^{i\top}\mathbf{W}^{i}$ and let V_{pq} be the element in the pth row and qth column of $\mathbf{V}$. V_{pq} is the inner product of $\mathbf{w}_p^i$ and $\mathbf{w}_q^i$. When $\|\mathbf{w}_p^i\|_F^2 = 1$, the diagonal elements of MCR will be 0 and the angle between $\mathbf{w}_p^i$ and $\mathbf{w}_q^i$ will be $\arccos(\mathbf{w}_p^{i\top}\mathbf{w}_q^i)$. Eq. (4.5) can be written as:

$$\text{MCR} = \sum_{p,q} \mathbf{w}_p^{i\top}\mathbf{w}_q^i, \quad p \neq q \tag{4.6}$$

According to the inequality of arithmetic and geometric means, it can be deduced that

$$\frac{\sum_{p,q} \mathbf{w}_p^{i\top}\mathbf{w}_q^i}{c^2 - c} \geq \prod_{p,q} \sqrt[c^2-c]{\mathbf{w}_p^{i\top}\mathbf{w}_q^i} \tag{4.7}$$

The equality holds if and only if all $\mathbf{w}_p^{i\top}\mathbf{w}_q^i$ are equal. That is, the angle between any pair of column vectors is equal when $\mathbf{W}^i$ minimizes Eq. (4.5). □

Proposition 3. If $\mathbf{W}^i$ minimizes Eq. (4.5), the affine transformation of $\mathbf{W}^i$, i.e. $\mathbf{W}^i\mathbf{R}$ also minimizes Eq. (4.5) where $\mathbf{R}$ is an orthogonal matrix.

Proof. As $\mathbf{R}$ is orthogonal, we have

$$\left\|\mathbf{W}^{i\top}\mathbf{W}^{i}-\mathbf{I}\right\|_{F}=\left\|\mathbf{R}^{\top}\left(\mathbf{W}^{i\top}\mathbf{W}^{i}-\mathbf{I}\right)\mathbf{R}\right\|_{F} \tag{4.8}$$

Eq. (4.8) can be rewritten as

$$\left\|\mathbf{W}^{i\top}\mathbf{W}^{i}-\mathbf{I}\right\|_{F}=\left\|\mathbf{R}^{\top}\mathbf{W}^{i\top}\mathbf{W}^{i}\mathbf{R}-\mathbf{I}\right\|_{F} \tag{4.9}$$

Here, $\mathbf{R}^{\top}\mathbf{R}=I$ is used in the deduction. Hence, $\mathbf{W}^{i}\mathbf{R}$ also minimizes Eq. (4.5). □

In Fig. 4.1, we illustrate **Proposition 2** and **Proposition 3** in 2-dimensional case. Following the flowchart of ITQ, one can find c d-dimensional vectors distributed like those in Fig. 4.1 and then transform them by $\mathbf{R}$ to minimize Eq. (4.2). However, the complexity of theoretically finding such vectors increases dramatically in high dimensional spaces. Hence, we leave it to an optimization algorithm. That is, we add MCR to Eq. (4.4), which leads to the following model.

$$\underset{\mathbf{B},\mathbf{W}^{i},\mathbf{v}^{i}}{\arg\min}\,E=\sum_{i}\alpha_{i}\left(\left\|\mathbf{B}-\frac{1}{1+\mathbf{A}^{i}}\right\|_{F}^{2}+\gamma_{i}\left\|\mathbf{W}^{i\top}\mathbf{W}^{i}-\mathbf{I}\right\|_{F}\right), \tag{4.10}$$

where γ_i is a positive real constant, and

$$\mathbf{A}^{i}=\exp\left(-(\beta_{i}\mathbf{X}^{i}\mathbf{W}^{i}+\mathbf{1}\mathbf{v}^{i})\right). \tag{4.11}$$

4.2.2 Optimization

Eq. (4.10) is minimized by iterative minimization. Take the partial derivative with respect to $\mathbf{B}$, resulting in

$$\frac{\partial E}{\partial\mathbf{B}}=2\sum_{i}\alpha_{i}\mathbf{B}-2\sum_{i}\frac{\alpha_{i}}{1+\exp\left(-(\beta_{i}\mathbf{X}^{i}\mathbf{W}^{i}+\mathbf{1}\mathbf{v}^{i})\right)} \tag{4.12}$$

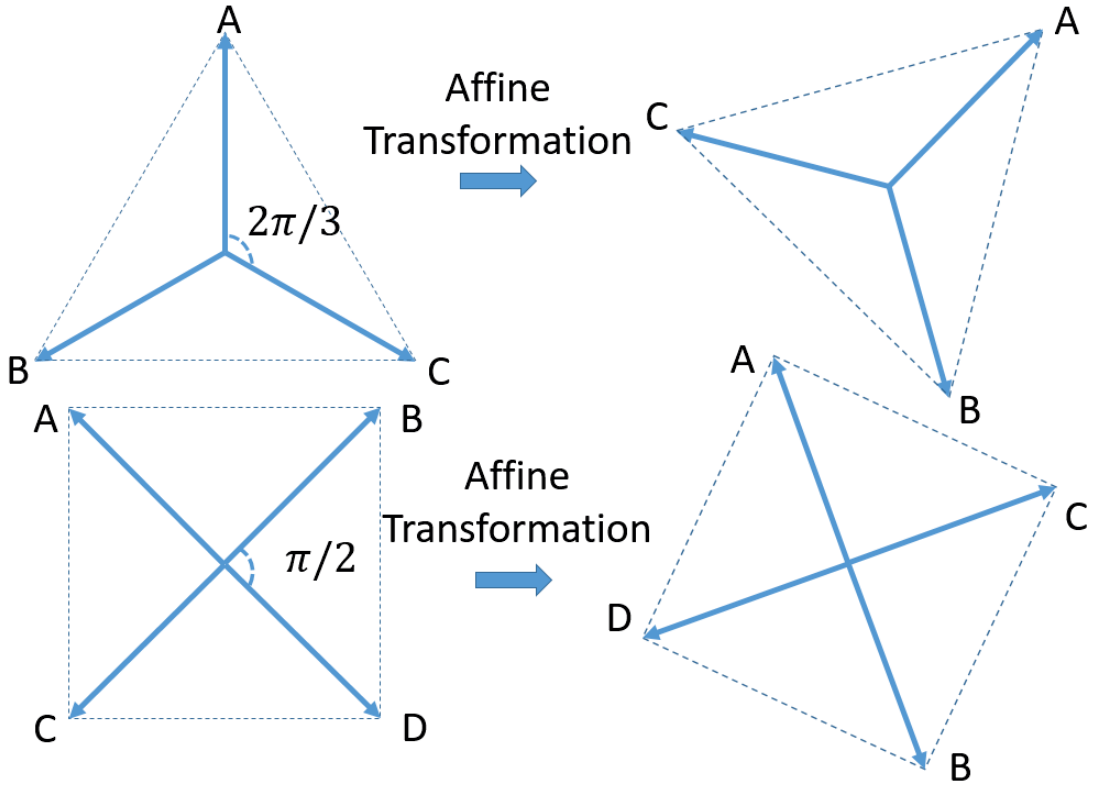


Figure 4.1: Illustration of **Proposition 2** and **Proposition 3**. If $\mathbf{W}^i \in \mathbb{R}^{2\times 3}$, its column vectors will align with the centerlines of an equilateral triangle. If $\mathbf{W}^i \in \mathbb{R}^{2\times 4}$, its column vectors will align with the diagonals of a square. The affine transformation will change the relative positions among vectors but the overall structure is kept. For example, in the equilateral triangle, point B is transformed to the clockwise direction of point A.

Setting Eq. (4.12) as 0, we can derive that

$$\mathbf{B} = \frac{1}{\sum_i \alpha_i} \sum_i \frac{\alpha_i}{1 + \exp\left(-(\beta_i \mathbf{X}^i \mathbf{W}^i + \mathbf{1}\mathbf{v}^i)\right)} \tag{4.13}$$

$\mathbf{B}$ is rounded in each iteration to ensure $\mathbf{B} \in \{0, 1\}^{n\times c}$.
Take the partial derivative with respect to $\mathbf{v}^i$, resulting in

$$\frac{\partial E}{\partial \mathbf{v}^i} = \frac{2\alpha_i}{n} \left(\frac{1}{1 + \mathbf{A}^i} - \mathbf{B} \right) \circ \left(\mathbf{A}^i \circ \frac{1}{(1 + \mathbf{A}^i)^2} \right) \tag{4.14}$$

In Eq. (4.14), "$\circ$" means element-wise multiplication. The division and square are also element-wise. The partial derivative with respect to $\mathbf{W}^i$ is

$$\frac{\partial E}{\partial \mathbf{W}^i} = 2\alpha_i\beta_i {\mathbf{X}^i}^\top \left(\frac{1}{1+\mathbf{A}^i} - \mathbf{B}\right) \circ \left(\mathbf{A}^i \circ \frac{1}{(1+\mathbf{A}^i)^2}\right) + \frac{\alpha_i\gamma_i \mathbf{W}^i \left({\mathbf{W}^i}^\top \mathbf{W}^i - \mathbf{I}\right)}{\left\|{\mathbf{W}^i}^\top \mathbf{W}^i - \mathbf{I}\right\|} \quad (4.15)$$

The prototype of the proposed training method is shown in **Algorithm 3**. In Subsection 4.2.3, the parameter settings and details for efficient implementation are discussed.

Algorithm 3 The Prototype of the Proposed Training Method

Input: α_i, β_i, $\triangle t$, $\mathbf{X}^i$

1: **while** E not converged **do**
2: Update $\mathbf{B}$ using Eq. (4.12).
3: $\mathbf{v}^i \leftarrow \mathbf{v}^i - \triangle t \cdot \partial E / \partial \mathbf{v}^i$
4: $\mathbf{W}^i \leftarrow \mathbf{W}^i - \triangle t \cdot \partial E / \partial \mathbf{W}^i$
5: **end while**

Output: $\mathbf{B}, \mathbf{W^i}, \mathbf{v^i}$

4.2.3 Implementation details

α_i is used to balance the value of the quantization loss of each view. Hence, it should be set according to d_i. A reasonable setting is

$$\alpha_1 : \ldots : \alpha_g = \frac{1}{\sqrt{d_i}} : \ldots : \frac{1}{\sqrt{d_g}}. \quad (4.16)$$

We first set $\alpha_i = 1/\sqrt{d_i}$ and then divide α_is by the minimum, i.e.,

$$\alpha_i = \frac{1/\sqrt{d_i}}{\min\left[1/\sqrt{d_1}, \ldots, 1/\sqrt{d_g}\right]}. \quad (4.17)$$

β_i is used to re-scale the view matrix. We empirically found that the proposed method achieves the best performance when the values of the re-scaled view ma-

trix are in the interval $[0, 255]$. For instance, in the NUS-WIDE dataset [14], images are represented by 500-dimensional bag-of-visual-words SIFT feature vectors whose values are in $[0, 255]$, texts are represented by 1000-dimensional index vectors whose values are 0 or 1 and labels are 10-dimensional index vectors. Hence, we set β as 1, 255 and 255 for image view matrix, text view matrix and label view matrix, respectively. To improve computation efficiency, β_i is multiplied with $\mathbf{X}_i$ before the iteration starts.

We set the maximum iteration times as K. $\triangle t$ linearly decreases from k_s to k_e by K iterations, i.e., in the k-th iteration, $\triangle t = k_s - (k_s - k_e)k/K$.

For large dataset, the first term in Eq. (4.10) is too large, which makes γ_i and $\triangle t$ difficult to be determined. We divide this term by its Frobenius norm so that we can fix γ_i and $\triangle t$ settings for all our experiments.

By considering all the above analysis, we modify the proposed method as following. With $Y^i = \beta_i \mathbf{X}^i$, Eq. (4.12) and Eq. (4.15) can be rewritten as:

$$\frac{\partial E}{\partial \mathbf{B}} = 2\sum_i \alpha_i \mathbf{B} - 2\sum_i \frac{\alpha_i}{1 + \exp\left(-(\mathbf{Y}^i \mathbf{W}^i + \mathbf{1}\mathbf{v}^i)\right)} \tag{4.18}$$

$$\frac{\partial E}{\partial \mathbf{W}^i} = 2\alpha_i \left(\frac{\mathbf{C}}{\|\mathbf{C}\|} + \frac{\gamma_i \mathbf{W}^i \left({\mathbf{W}^i}^\top \mathbf{W}^i - \mathbf{I} \right)}{\left\| {\mathbf{W}^i}^\top \mathbf{W}^i - \mathbf{I} \right\|} \right), \tag{4.19}$$

where

$$\mathbf{C} = {\mathbf{Y}^i}^\top \left(\frac{1}{1 + \mathbf{A}^i} - \mathbf{B} \right) \circ \left(\mathbf{A}^i \circ \frac{1}{(1 + \mathbf{A}^i)^2} \right) \tag{4.20}$$

and $\mathbf{A}^i = \exp\left(-(\mathbf{Y}^i \mathbf{W}^i + \mathbf{1}\mathbf{v}^i)\right)$. The efficient version of the proposed method is given in **Algorithm** 4

4.3 Experimental Results

In this section, we evaluate the retrieval performance and computational efficiency of the proposed method. First, we introduce the data sets, evaluation metrics and comparison methods. Then, two types of experiments - *Hamming ranking* and *hash lookup* were conducted. Finally, we analyze the convergence and computational efficiency.

Algorithm 4 The Proposed Training Method

Input: α_i, β_i, $\triangle t$, $\mathbf{X}^i$, k, k_s, k_e, K

while E not converged and $k < K$ **do**
    $\triangle t = k_s - (k_s - k_e)k/K$
    Update $\mathbf{B}$ using Eq. (4.18).
    $\mathbf{v}^i \leftarrow \mathbf{v}^i - \triangle t \cdot \partial E / \partial \mathbf{v}^i$
    $\mathbf{W}^i \leftarrow \mathbf{W}^i - \triangle t \cdot \partial E / \partial \mathbf{W}^i$
    $k \leftarrow k + 1$
end while

Output: $\mathbf{B}, \mathbf{W^i}, \mathbf{v^i}$

4.3.1 Data sets

MIRFlickr [45] contains 25,000 entries each of which consists of 1 image, several textual tags and labels. Following literature [68], we only keep those textural tags appearing at least 20 times and remove entries which have no label. Hence, 20,015 entries are left. For each entry, the image is represented by a 512-dimensional GIST [85] descriptors and the text is represented by a 500-dimensional feature vector derived from PCA on index vectors of the textural tags. 5% entries are randomly selected for testing and the remaining entries are used as training set. Ground-truth semantic neighbors for a test entry, i.e, a query, are defined as those sharing at least one label.

NUS-WIDE [14] is comprised of 269,648 images and over 5,000 textural tags collected from Flickr. Ground-truth of 81 concepts is provided for the entire data set. Following literatures [22] [68] [126], we select 10 most common concepts for labels and thus 186,577 entries are left. For each entry, the image is represented as a 500-dimensional bag-of-visual-words SIFT feature vector and text is represented as an index vector of the most frequent 1,000 tags. 1% entries are randomly selected for testing and the remaining are used for training. Ground-truth semantic neighbors for a test entry are defined as those sharing at least one label.

4.3.2 Evaluation Metrics

Hamming ranking and *hash lookup* are two widely used experiments for evaluating retrieval performance. In Hamming ranking experiment, all data points in the training set are ranked depending on their Hamming distances to a given query. The average precision (AP) is defined as

$$AP = \frac{1}{N} \sum_{r=1}^{R} P(r)\delta(r) \tag{4.21}$$

where N is the number of relevant instances in the retrieved set, $P(r)$ is the precision of the top r retrieved instances, and $\delta(r) = 1$ if the r-th retrieved instance is a true neighbor of the query, and otherwise $\delta(r) = 0$. Mean average precision (MAP) is the mean of APs of all the queries. For the ideal case that all retrieved instance are true neighbors of the queries, MAP is equal to 1, while MAP is equal to 0 for the worst case that all retrieved instance are not the true neighbors. Hence, the closer it is to 1, the better the performance.

In *hash lookup* experiment, the retrieved instances are those whose Hamming distances to a given query are not larger than 2. The performance are evaluated by F1-score which is defined as

$$F1 = 2\frac{precision \cdot recall}{precision + recall} \tag{4.22}$$

The F1-scores are averaged for all queries. Similar to MAP, F1 also varies in $[0, 1]$ and the closer it is to 1, the better the performance.

4.3.3 Baselines

The proposed method is compared with five state-of-the-art multimodal hashing methods CMSSH [11], CVH [59], MDBE [108], SCM [126] and SePH [68].

CMSSH and SePH requires too much computational cost. Following literatures [68] [126], 10,000 entries are randomly selected for training hashing functions and then we apply these functions to generate hashing codes. We use the codes provided by the authors except for MDBE. We re-implement MDBE and set pa-

rameters following the authors' suggestions. For our method, we use the following parameter settings, $k_s = 0.003$, $k_e = 0.0015$, $K = 400$ and $\gamma_i = 0.001$. α_i and β_i are set as discussed in Subsection 4.2.3.

Table 4.1: MAP results on MIRFlickr and NUS-WIDE data sets.

Task	Method	**MIRFlickr**				
		16 bits	32 bits	64 bits	96 bits	128 bits
Image-Text	CMSSH	0.5966	0.5674	0.5581	0.5692	0.5701
	CVH	0.6591	0.6145	0.6133	0.6091	0.6052
	SCM	0.6251	0.6361	0.6417	0.6446	0.6480
	SePH	0.6505	0.6447	0.6453	0.6497	0.6612
	MDBE	0.6784	0.7050	0.7083	0.7148	0.7156
	DMH	**0.7012**	**0.7057**	**0.7398**	**0.7424**	**0.7501**
Text-Image	CMSSH	0.6613	0.6510	0.6756	0.6643	0.6471
	CVH	0.6495	0.6213	0.6179	0.6050	0.5948
	SCM	0.6194	0.6302	0.6377	0.6377	0.6417
	SePH	0.6745	0.6824	0.6917	0.7059	0.7110
	MDBE	0.7521	0.7793	0.7894	0.7903	0.7919
	DMH	**0.7629**	**0.7817**	**0.7962**	**0.8301**	**0.8470**
Task	Method	**NUS-WIDE**				
		16 bits	32 bits	64 bits	96 bits	128 bits
Image-Text	CMSSH	0.4124	0.3533	0.3540	0.3578	0.3600
	CVH	0.4733	0.3505	0.2900	0.2812	0.2950
	SCM	0.5245	0.5394	0.5332	0.5376	0.5400
	SePH	0.5573	0.5481	0.5589	0.5572	0.5569
	MDBE	0.6281	0.6409	0.6617	0.6598	0.6644
	DMH	**0.6317**	**0.6506**	**0.6591**	**0.6740**	**0.6837**
Text-Image	CMSSH	0.4152	0.3515	0.3510	0.3555	0.3556
	CVH	0.4794	0.4195	0.3901	0.3552	0.3501
	SCM	0.5127	0.5214	0.5255	0.5302	0.5380
	SePH	0.7185	0.7258	0.7390	0.7455	0.7491
	MDBE	0.7623	0.7737	0.7953	0.7973	0.7987
	DMH	**0.7653**	**0.7827**	**0.8150**	**0.8192**	**0.8246**

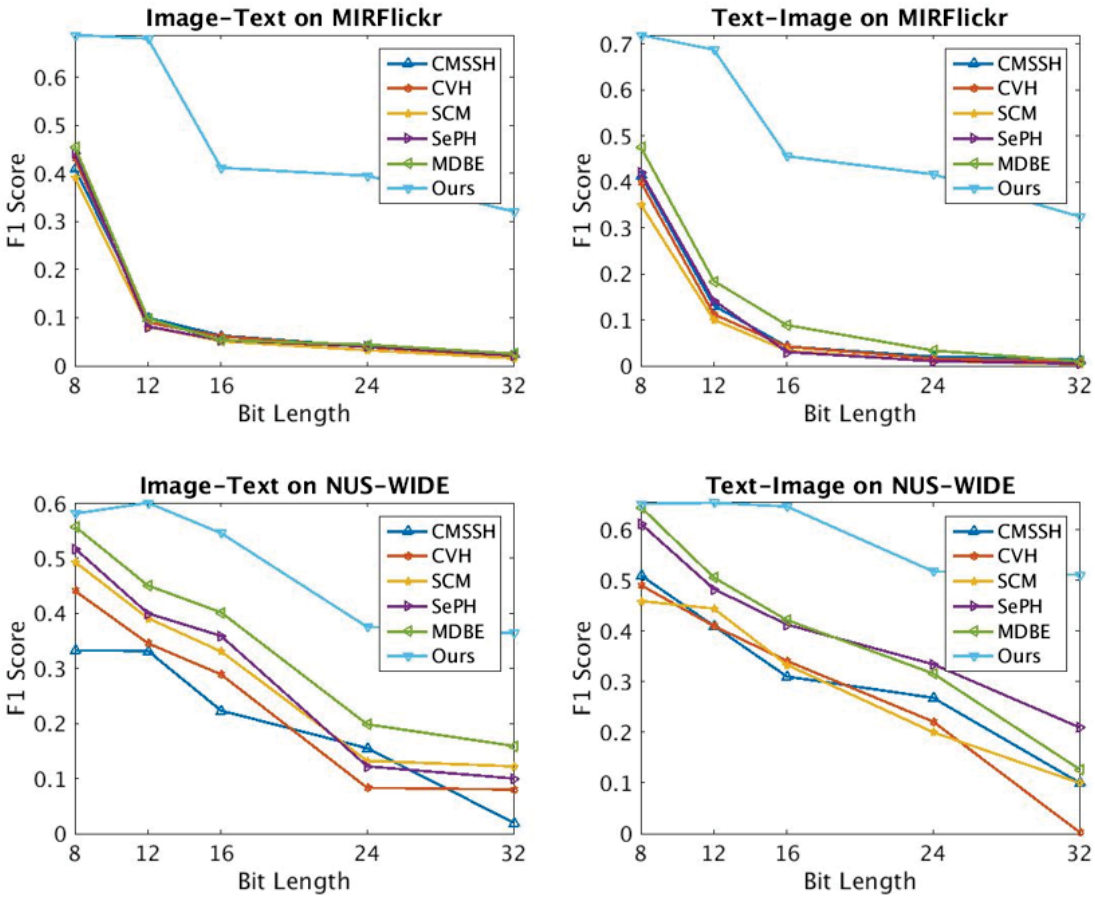


Figure 4.2: F1-score on MIRFlickr and NUS-WIDE data sets

4.3.4 Results

MAP results are shown in Table 4.1. In Table 4.1, "Image-Text" means using images to query texts, while "Text-Image" means using texts to query images. From Table 4.1, it can be observed that our method outperforms all compared methods. As the bit length increases, the performance of our method increases faster than baselines, which demonstrates the effectiveness of the proposed minimum correlation regularization. For example, in the "Image-Text" experiment on MIRFlickr, the performance improvement ranges from 3% to 5% as the bit length varies from 16 to 128, compared to the best baseline, i.e., MDBE.

F1-score results are shown in Figure 4.2. Similar to the MAP results, our method surpasses all baselines by a huge performance improvement, especially

on MIRFlickr. On MIRFlickr, the performance improvement ranges from 30% to 3,000%, compared to the best baseline. On NUS-WIDE, it is 5% to 200%. A reasonable explanation is that our method can precisely preserve the inter-class structure and therefore the lookup performance is significantly improved. Because the ranking performance depends on the preservation of the structure of the whole data set regardless of inter-class or intra-class structure, it is not as significant as that of the lookup experiment. The size of MIRFlickr is only about 1/10 of NUS-WIDE, so the simple non-linearity introduced in our method works much better on MIRFlickr. To achieve comparable performance improvement on NUS-WIDE data set, more sophisticated non-linear models are expected.

In both experiments, MDBE achieves the best performance among all the baselines. Actually, the main part of MDBE,

$$\|\mathbf{LU} - \mathbf{XW_x}\|_F^2 + \|\mathbf{LU} - \mathbf{YW_y}\|_F^2, \tag{4.23}$$

is equivalent to Eq. (4.3) which is an intuitive multi-modal extension of ITQ, where L is the label matrix, X is the image view matrix and Y is the text view matrix. W_x, W_y and U are variables. If we treat the label matrix as another view of the data and introduce an auxiliary variable B, it is easy to figure out that Eq. (4.23) and Eq. (4.3) are equivalent. By introducing non-linearity and minimum correlation regularization, our method performs much better than MDBE.

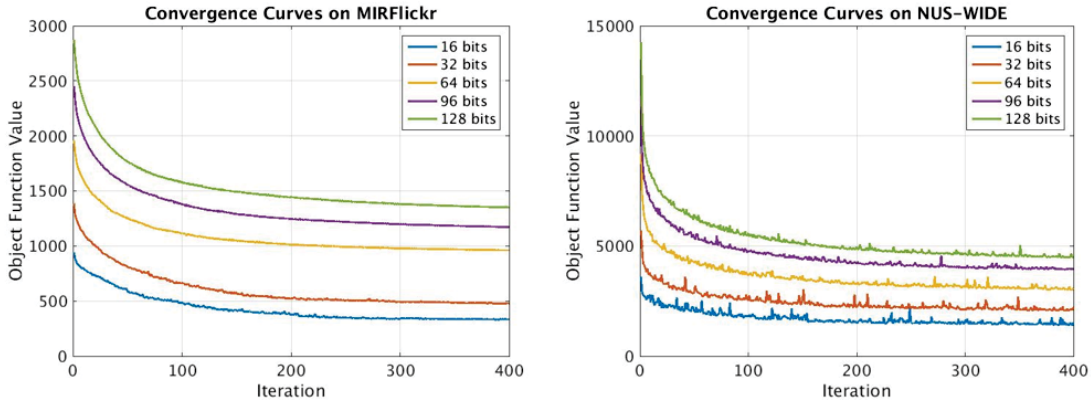


Figure 4.3: Convergence curves on MIRFlickr and NUS-WIDE data sets

4.3.5 Convergence Study

The objective function of our method is minimized by **Algorithm** 4. In **Algorithm** 4, we empirically amend the derivative of E with respect to $\mathbf{W}^i$ for easy parameter tuning. The convergence property is experimentally studied in this subsection. Fig. 4.3 shows the convergence curves. It can be seen that the objective function value decreases fast in the first 100 iterations and then slides relatively slowly. The convergence curves of experiments on MIRFlickr is smooth, while those of experiments on NUS-WIDE jitters because of more sophisticated data structure and therefore more saddle points across which the algorithm jumps.

4.3.6 Computation Efficiency

Training and testing time on 32-bit are given in Table 4.2. The training time is the mean time of 10 runs. The testing time is the average time cost for one query. All experiments were performed on MATLAB R2015b installed on a GNU/Linux Server with 2.30 GHz 16-core CPU and 768 GB RAM. From Table 4.2, it can be seen that the training time of our method is moderate among all methods. Its testing time is close to that of MDBE, because the encoding procedure for a new query of these two methods are similar.

Table 4.2: Training and Testing Time on MIRFlickr and NUS-WIDE data sets in seconds. The testing time is multiplied with 10^5

	MIRFlickr		**NUS-WIDE**	
Method	Training	Testing	Training	Testing
CMSSH	69.7	1.016	705.2	1.270
CVH	0.9	0.910	3.6	1.087
SCM	1.3	0.308	12.5	1.270
SePH	4711.2	4.244	5082.3	5.550
MDBE	25.0	0.431	241.8	0.572
DMH	29.8	0.432	398.0	0.572

4.3.7 Performance on Facial Images

Additional to FERET, FRGC, CMU-PIE and extended Yale B datasets used in Chapter 2, YouTube Faces [116] dataset is used to evaluate the retrieval performance of our method on facial images. YouTube Faces dataset contains 1,595 different people. Following [72], we choose 340 people such that each has at least 500 images to form a subset of 370,319 face images. Each image is represented by a 1,770 dimensional LBP feature vector [2]. The groundtruth neighbors for a query are those belonging to the same identity as the query.

The results can be found in Appendix A. Part of the results on YouTube Faces dataset is reported in Table 5.1 and Table 5.2. It can be seen that our GHS-DD method surpasses all compared methods.

Table 4.3: MAP results on YouTube Faces dataset

	YouTube Faces				
c	8	12	16	20	24
DMH	**0.2035**	**0.2529**	**0.3027**	**0.3386**	**0.3712**
MDBE	0.1954	0.2396	0.2909	0.3257	0.3603
CCA-GHS-DI	0.1852	0.2308	0.2831	0.3137	0.3496
CCA-GHS-DD	0.1960	0.2457	0.2919	0.3319	0.3698
SePH	0.1779	0.2302	0.2816	0.3192	0.3625
CMSSH	0.1862	0.2327	0.2614	0.3038	0.3466
CCA-ITQ	0.1798	0.2360	0.2633	0.2992	0.3298
SCM	0.1798	0.2259	0.2497	0.2688	0.2818
CVH	0.1916	0.1975	0.2128	0.2232	0.2362

4.4 Conclusion

This paper proposes an effective multi-modal hashing method which is modeled as a quantization error problem and the minimum correlation regularization is devised to improve the retrieval performance on long codes. Experiments on MIRFlickr and NUS-WIDE data sets show that the proposed method surpasses

Table 4.4: F-measure results on YouTube Faces dataset

	YouTube Faces				
c	8	12	16	20	24
DMH	**0.3406**	**0.5245**	**0.6703**	**0.7264**	**0.7825**
MDBE	0.2056	0.2545	0.2653	0.2539	0.2425
CCA-GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
CCA-GHS-DD	0.1843	0.2296	0.2426	0.2362	0.2298
SePH	0.1654	0.1987	0.2159	0.1703	0.1247
CMSSH	0.1648	0.1905	0.2210	0.1917	0.1624
CCA-ITQ	0.1615	0.1872	0.2136	0.1151	0.0166
SCM	0.0817	0.1246	0.1498	0.1581	0.1663
CVH	0.1611	0.1698	0.1754	0.1710	0.1667

the compared methods distinctively. Future works include testing more nonlinear embedding functions and refining optimization procedure for high computational efficiency.

Nonlinearity acts as a key role in the proposed method. In the following chapter, we will introduce deep neural network into multi-modal retrieval methods. Theoretically, a deep neural network can approximate any non-linear structures. Hence, the proposed deep neural network is expected to boost the retrieval accuracy.

Chapter 5

A Hybrid Neural Network for Multimodal Hashing

In Chapter 4, we found the nonlinearity could handle more sophisticated data structure. Deep neural network (DNN) is expected to be an efficient way for multi-modal hashing. In this chapter, we propose a hybrid neural network which consists of a convolutional neural network (CNN) for images and a full-connected neural network (FNN) for tags or labels.

5.1 Introduction

Deep neural networks have achieved outstanding success in machine learning [20] [51] [30] [124] and computer vision [78] [38] [19] [63] [75]. Recently, multi-modal deep learning becomes a hot topic [82] [99] [98]. However, few works focus on multi-modal hashing. To our best knowledge, there are two representative works [52] [109] in this area. However, these two works use the same type of neural network for each modality. In this chapter, we propose a hybrid neural network for multi-modal hashing. We choose the networks according to their performances on different types of data, such as CNN for images.

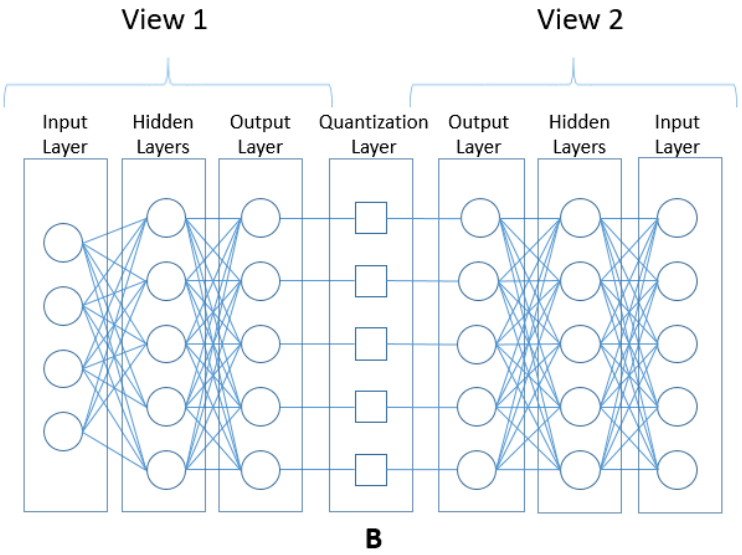


Figure 5.1: The network structure for double-view data

5.2 Methodology

The proposed network for two modalities is shown in Fig. 5.1. In the quantization layer, the quantization error is back-propagated to train the network. The output is used as the hashing codes. The loss function can be written as:

$$\underset{\mathbf{B},\theta_i}{\operatorname{argmin}} \ \|\mathbf{B} - N^i(\mathbf{X}^i;\theta^i)\| \tag{5.1}$$

where $\mathbf{B}$ is the hashing code matrix, $\mathbf{X}^i$ is the ith view of the data and $N^i(\cdot;\theta_i)$ is the ith neural network with parameter θ^i.
To incorporate orthogonality, we apply the minimum correlation regularization to the output layer. Hence, the overall loss function can be written as:

$$\underset{\mathbf{B},\theta_i}{\operatorname{argmin}} \ \|\mathbf{B} - N^i(\mathbf{X}^i;\theta_i)\| + \|{\mathbf{W}_o^i}^\top \mathbf{W}_o - \mathbf{I}\| \tag{5.2}$$

where $\mathbf{W}_o^i$ is the parameter of the output layer of the ith neural network.

5.3 Experimental Results

We evaluate our methods on the five facial datasets, i.e. CMU-PIE, Extended Yale B, FERET, FRGC and YouTube Faces. All results can be found in Appendix C. Part of MAP and F-measure results on YouTube dataset is given in Table 5.1 and Table 5.2. The CNN used for images is a three-convolutional-layer ($2c$,c,c) network following a full-connected network. As the label is an index vector which is a simple structure, we use a full-connected network with one hidden layer with c neurons, where c is the code length.

Table 5.1: MAP results on YouTube Faces dataset

	YouTube Faces				
c	8	12	16	20	24
HDNN	**0.2972**	**0.3630**	**0.4334**	**0.4783**	**0.5246**
DMH	0.2713	0.3369	0.4048	0.4483	0.4960
MDBE	0.2554	0.3191	0.3902	0.4340	0.4820
CCA-GHS-DI	0.2439	0.3029	0.3736	0.4224	0.4663
CCA-GHS-DD	0.2623	0.3273	0.3878	0.4414	0.4961
SePH	0.2325	0.3090	0.3734	0.4266	0.4841
CMSSH	0.2472	0.3073	0.3439	0.4091	0.4634
CCA-ITQ	0.2396	0.3174	0.3532	0.3951	0.4399
SCM	0.2397	0.3059	0.3343	0.3552	0.3811

Table 5.2: F-measure results on YouTube Faces dataset

	YouTube Faces				
c	8	12	16	20	24
HDNN	**0.3541**	**0.5393**	**0.6836**	**0.7397**	**0.7958**
DMH	0.3406	0.5245	0.6703	0.7264	0.7825
MDBE	0.2056	0.2545	0.2653	0.2539	0.2425
CCA-GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
CCA-GHS-DD	0.1843	0.2296	0.2426	0.2362	0.2298
SePH	0.1654	0.1987	0.2159	0.1703	0.1247
CMSSH	0.1648	0.1905	0.2210	0.1917	0.1624
CCA-ITQ	0.1615	0.1872	0.2136	0.1151	0.0166
SCM	0.0817	0.1246	0.1499	0.1581	0.1663

Chapter 6

Conclusions

In this thesis, we exploit the orthogonality to facial image restoration and retrieval.

First, in the proposed facial deblur method. The intrinsic sharp image is represented by a linear combination of principal components which are generated by principal component analysis (PCA) which is an orthogonal transformation widely used in statistics. The point spread function is represented as a linear combination of a set of orthogonal functions. By recovering the coefficients of these two linear combination, several candidate results can be found. Then, a trained support vector regression is used for automatically selecting the best candidate as the final output. The proposed method achieve better performance both in synthetic and real facial images.

Second, the global positioning system (GPS) is modified for hashing. The proposed method can generate a balanced code matrix which is an NP-hard problem and hence it is often circumvented in literatures. To satisfy the uniqueness and existence condition of GPS, the satellites in our data-dependent method are orthogonally distributed in each group. Such distributed satellites also approximately satisfy the orthogonality constraint on code matrix. Therefore, the proposed method surpasses the compared methods and as the code length increase, the superiority becomes more obvious.

Third, we generalize the orthogonality to $d \times c$ matrix, where $d < c$. Because it is impossible to find c mutually orthogonal d-dimensional vectors, the minimum correlation constraint is proposed. When $c = d$, the minimum correlation constraint

is equivalent to orthogonality constraint. The mimum correlation constraint is incorporated into the proposed multi-modal hashing method. The hashing ranking and lookup experiments validate the effectiveness of out method.

From the three methods proposed in this thesis, we can conclude that orthogonality is an effective way of representation and regularization. Hence, it can be used in other problem where effective representations are required, such as space-frequency transformation or where uncorrelation is preferred, such as image feature extraction.

The proposed methods inspire several possible directions for further improving the performance. For example, the nonlinear terms of the proposed multi-modal retrieval method can be treated as a single-layer neural network. One can use convolutional neural network for images and recurrent neural network for texts to substitute the nonlinear terms.

Appendix A

Results of Image Hashing Method

Table A.1: MAP results on CMU-PIE dataset

	CMU-PIE				
c	8	12	16	20	24
GHS-DI	0.1828	0.2303	0.2808	0.3158	0.3521
GHS-DD	**0.1960**	**0.2453**	**0.2912**	**0.3305**	**0.3709**
ITQ	0.1749	0.2307	0.2791	0.3202	0.3621
IsoH	0.1860	0.2335	0.2589	0.3044	0.3488
HH	0.1788	0.2379	0.2635	0.2977	0.3296
SH	0.1791	0.2275	0.2498	0.2674	0.2843
IMH	0.1925	0.1983	0.2111	0.2241	0.2368
okmeans	0.1908	0.2367	0.2834	0.3112	0.3395
SpH	0.0459	0.0518	0.0419	0.0461	0.0494
	CMU-PIE				
c	32	48	64	80	96
GHS-DI	0.4028	0.4724	0.5411	0.5630	0.5853
GHS-DD	**0.4299**	**0.4927**	**0.5538**	**0.5756**	**0.5980**
ITQ	0.4097	0.4688	0.5278	0.5540	0.5794
IsoH	0.3947	0.4422	0.4889	0.5342	0.5799
HH	0.3706	0.4214	0.4721	0.5058	0.5419
SH	0.2943	0.3127	0.3306	0.3256	0.3191
IMH	0.3037	0.3119	0.3209	0.3364	0.3523
okmeans	0.3788	0.4198	0.4620	0.4694	0.4762
SpH	0.0392	0.0522	0.0646	0.1225	0.1808
	CMU-PIE				
c	128	160	192	224	256
GHS-DI	0.5964	0.6037	0.6107	0.6184	0.6256
GHS-DD	**0.6100**	**0.6168**	**0.6246**	**0.6319**	**0.6395**
ITQ	0.5911	0.5981	0.6058	0.6131	0.6203
IsoH	0.5919	0.5990	0.6057	0.6132	0.6206
HH	0.5526	0.5598	0.5667	0.5729	0.5801
SH	0.3255	0.3297	0.3336	0.3373	0.3412
IMH	0.3602	0.3644	0.3692	0.3730	0.3778
okmeans	0.4862	0.4916	0.4981	0.5038	0.5101
SpH	0.1848	0.1867	0.1890	0.1917	0.1923

Table A.2: MAP results on Extended Yale B dataset

	Extended Yale B				
c	8	12	16	20	24
GHS-DI	0.1828	0.2303	0.2808	0.3158	0.3521
GHS-DD	**0.1960**	**0.2453**	**0.2912**	**0.3305**	**0.3709**
ITQ	0.1749	0.2307	0.2791	0.3202	0.3621
IsoH	0.1860	0.2335	0.2589	0.3044	0.3488
HH	0.1788	0.2379	0.2635	0.2977	0.3296
SH	0.1791	0.2275	0.2498	0.2674	0.2843
IMH	0.1925	0.1983	0.2111	0.2241	0.2368
okmeans	0.1908	0.2367	0.2834	0.3112	0.3395
SpH	0.0459	0.0518	0.0419	0.0461	0.0494
	Extended Yale B				
c	32	48	64	80	96
GHS-DI	0.4028	0.4724	0.5411	0.5630	0.5853
GHS-DD	**0.4299**	**0.4927**	**0.5538**	**0.5756**	**0.5980**
ITQ	0.4097	0.4688	0.5278	0.5540	0.5794
IsoH	0.3947	0.4422	0.4889	0.5342	0.5799
HH	0.3706	0.4214	0.4721	0.5058	0.5419
SH	0.2943	0.3127	0.3306	0.3256	0.3191
IMH	0.3037	0.3119	0.3209	0.3364	0.3523
okmeans	0.3788	0.4198	0.4620	0.4694	0.4762
SpH	0.0392	0.0522	0.0646	0.1225	0.1808
	Extended Yale B				
c	128	160	192	224	256
GHS-DI	0.5964	0.6037	0.6107	0.6184	0.6256
GHS-DD	**0.6100**	**0.6168**	**0.6246**	**0.6319**	**0.6395**
ITQ	0.5911	0.5981	0.6058	0.6131	0.6203
IsoH	0.5919	0.5990	0.6057	0.6132	0.6206
HH	0.5526	0.5598	0.5667	0.5729	0.5801
SH	0.3255	0.3297	0.3336	0.3373	0.3412
IMH	0.3602	0.3644	0.3692	0.3730	0.3778
okmeans	0.4862	0.4916	0.4981	0.5038	0.5101
SpH	0.1848	0.1867	0.1890	0.1917	0.1923

Table A.3: MAP results on FERET dataset

	FERET				
c	8	12	16	20	24
GHS-DI	0.1828	0.2303	0.2808	0.3158	0.3521
GHS-DD	**0.1960**	**0.2453**	**0.2912**	**0.3305**	**0.3709**
ITQ	0.1749	0.2307	0.2791	0.3202	0.3621
IsoH	0.1860	0.2335	0.2589	0.3044	0.3488
HH	0.1788	0.2379	0.2635	0.2977	0.3296
SH	0.1791	0.2275	0.2498	0.2674	0.2843
IMH	0.1925	0.1983	0.2111	0.2241	0.2368
okmeans	0.1908	0.2367	0.2834	0.3112	0.3395
SpH	0.0459	0.0518	0.0419	0.0461	0.0494
	FERET				
c	32	48	64	80	96
GHS-DI	0.4028	0.4724	0.5411	0.5630	0.5853
GHS-DD	**0.4299**	**0.4927**	**0.5538**	**0.5756**	**0.5980**
ITQ	0.4097	0.4688	0.5278	0.5540	0.5794
IsoH	0.3947	0.4422	0.4889	0.5342	0.5799
HH	0.3706	0.4214	0.4721	0.5058	0.5419
SH	0.2943	0.3127	0.3306	0.3256	0.3191
IMH	0.3037	0.3119	0.3209	0.3364	0.3523
okmeans	0.3788	0.4198	0.4620	0.4694	0.4762
SpH	0.0392	0.0522	0.0646	0.1225	0.1808
	FERET				
c	128	160	192	224	256
GHS-DI	0.5964	0.6037	0.6107	0.6184	0.6256
GHS-DD	**0.6100**	**0.6168**	**0.6246**	**0.6319**	**0.6395**
ITQ	0.5911	0.5981	0.6058	0.6131	0.6203
IsoH	0.5919	0.5990	0.6057	0.6132	0.6206
HH	0.5526	0.5598	0.5667	0.5729	0.5801
SH	0.3255	0.3297	0.3336	0.3373	0.3412
IMH	0.3602	0.3644	0.3692	0.3730	0.3778
okmeans	0.4862	0.4916	0.4981	0.5038	0.5101
SpH	0.1848	0.1867	0.1890	0.1917	0.1923

Table A.4: MAP results on FRGC dataset

	FRGC				
c	8	12	16	20	24
GHS-DI	0.1828	0.2303	0.2808	0.3158	0.3521
GHS-DD	**0.1960**	**0.2453**	**0.2912**	**0.3305**	**0.3709**
ITQ	0.1749	0.2307	0.2791	0.3202	0.3621
IsoH	0.1860	0.2335	0.2589	0.3044	0.3488
HH	0.1788	0.2379	0.2635	0.2977	0.3296
SH	0.1791	0.2275	0.2498	0.2674	0.2843
IMH	0.1925	0.1983	0.2111	0.2241	0.2368
okmeans	0.1908	0.2367	0.2834	0.3112	0.3395
SpH	0.0459	0.0518	0.0419	0.0461	0.0494
	FRGC				
c	32	48	64	80	96
GHS-DI	0.4028	0.4724	0.5411	0.5630	0.5853
GHS-DD	**0.4299**	**0.4927**	**0.5538**	**0.5756**	**0.5980**
ITQ	0.4097	0.4688	0.5278	0.5540	0.5794
IsoH	0.3947	0.4422	0.4889	0.5342	0.5799
HH	0.3706	0.4214	0.4721	0.5058	0.5419
SH	0.2943	0.3127	0.3306	0.3256	0.3191
IMH	0.3037	0.3119	0.3209	0.3364	0.3523
okmeans	0.3788	0.4198	0.4620	0.4694	0.4762
SpH	0.0392	0.0522	0.0646	0.1225	0.1808
	FRGC				
c	128	160	192	224	256
GHS-DI	0.5964	0.6037	0.6107	0.6184	0.6256
GHS-DD	**0.6100**	**0.6168**	**0.6246**	**0.6319**	**0.6395**
ITQ	0.5911	0.5981	0.6058	0.6131	0.6203
IsoH	0.5919	0.5990	0.6057	0.6132	0.6206
HH	0.5526	0.5598	0.5667	0.5729	0.5801
SH	0.3255	0.3297	0.3336	0.3373	0.3412
IMH	0.3602	0.3644	0.3692	0.3730	0.3778
okmeans	0.4862	0.4916	0.4981	0.5038	0.5101
SpH	0.1848	0.1867	0.1890	0.1917	0.1923

Table A.5: MAP results on YouTube Faces dataset

	YouTube Faces				
c	8	12	16	20	24
GHS-DI	0.1828	0.2303	0.2808	0.3158	0.3521
GHS-DD	**0.1960**	**0.2453**	**0.2912**	**0.3305**	**0.3709**
ITQ	0.1749	0.2307	0.2791	0.3202	0.3621
IsoH	0.1860	0.2335	0.2589	0.3044	0.3488
HH	0.1788	0.2379	0.2635	0.2977	0.3296
SH	0.1791	0.2275	0.2498	0.2674	0.2843
IMH	0.1925	0.1983	0.2111	0.2241	0.2368
okmeans	0.1908	0.2367	0.2834	0.3112	0.3395
SpH	0.0459	0.0518	0.0419	0.0461	0.0494
	YouTube Faces				
c	32	48	64	80	96
GHS-DI	0.4028	0.4724	0.5411	0.5630	0.5853
GHS-DD	**0.4299**	**0.4927**	**0.5538**	**0.5756**	**0.5980**
ITQ	0.4097	0.4688	0.5278	0.5540	0.5794
IsoH	0.3947	0.4422	0.4889	0.5342	0.5799
HH	0.3706	0.4214	0.4721	0.5058	0.5419
SH	0.2943	0.3127	0.3306	0.3256	0.3191
IMH	0.3037	0.3119	0.3209	0.3364	0.3523
okmeans	0.3788	0.4198	0.4620	0.4694	0.4762
SpH	0.0392	0.0522	0.0646	0.1225	0.1808
	YouTube Faces				
c	128	160	192	224	256
GHS-DI	0.5964	0.6037	0.6107	0.6184	0.6256
GHS-DD	**0.6100**	**0.6168**	**0.6246**	**0.6319**	**0.6395**
ITQ	0.5911	0.5981	0.6058	0.6131	0.6203
IsoH	0.5919	0.5990	0.6057	0.6132	0.6206
HH	0.5526	0.5598	0.5667	0.5729	0.5801
SH	0.3255	0.3297	0.3336	0.3373	0.3412
IMH	0.3602	0.3644	0.3692	0.3730	0.3778
okmeans	0.4862	0.4916	0.4981	0.5038	0.5101
SpH	0.1848	0.1867	0.1890	0.1917	0.1923

Table A.6: F-measure results on CMU-PIE dataset

	CMU-PIE				
c	8	12	16	20	24
GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
GHS-DD	**0.1843**	**0.2296**	**0.2426**	**0.2362**	**0.2298**
ITQ	0.1654	0.1987	0.2159	0.1703	0.1247
IsoH	0.1648	0.1905	0.2210	0.1917	0.1624
HH	0.1615	0.1872	0.2136	0.1151	0.0166
SH	0.0817	0.1246	0.1498	0.1581	0.1663
IMH	0.1611	0.1698	0.1754	0.1710	0.1667
okmeans	0.1697	0.2136	0.2421	0.1475	0.0529
SpH	0.0799	0.1179	0.1357	0.1311	0.1265
	CMU-PIE				
c	32	48	64	80	96
GHS-DI	0.1341	0.1053	0.0765	0.0763	0.0761
GHS-DD	**0.1538**	**0.1258**	**0.0977**	**0.0971**	**0.0965**
ITQ	0.0436	0.0433	0.0429	0.0424	0.0419
IsoH	0.0016	0.0018	0.0020	0.0021	0.0022
HH	0.0207	0.0147	0.0088	0.0078	0.0068
SH	0.1515	0.0873	0.0232	0.0179	0.0126
IMH	0.1517	0.1158	0.0799	0.0793	0.0787
okmeans	0.0342	0.0234	0.0127	0.0091	0.0055
SpH	0.1315	0.0992	0.0668	0.0659	0.0651
	CMU-PIE				
c	128	160	192	224	256
GHS-DI	0.0777	0.0786	0.0796	0.0805	0.0815
GHS-DD	**0.0985**	**0.0997**	**0.1009**	**0.1021**	**0.1033**
ITQ	0.0427	0.0432	0.0437	0.0443	0.0448
IsoH	0.0022	0.0022	0.0023	0.0023	0.0023
HH	0.0069	0.0070	0.0071	0.0071	0.0072
SH	0.0128	0.0130	0.0131	0.0133	0.0134
IMH	0.0803	0.0813	0.0822	0.0832	0.0842
okmeans	0.0056	0.0057	0.0058	0.0059	0.0059
SpH	0.0664	0.0672	0.0680	0.0688	0.0696

Table A.7: F-measure results on Extended Yale B dataset

	Extended Yale B				
c	8	12	16	20	24
GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
GHS-DD	**0.1843**	**0.2296**	**0.2426**	**0.2362**	**0.2298**
ITQ	0.1654	0.1987	0.2159	0.1703	0.1247
IsoH	0.1648	0.1905	0.2210	0.1917	0.1624
HH	0.1615	0.1872	0.2136	0.1151	0.0166
SH	0.0817	0.1246	0.1498	0.1581	0.1663
IMH	0.1611	0.1698	0.1754	0.1710	0.1667
okmeans	0.1697	0.2136	0.2421	0.1475	0.0529
SpH	0.0799	0.1179	0.1357	0.1311	0.1265
	Extended Yale B				
c	32	48	64	80	96
GHS-DI	0.1341	0.1053	0.0765	0.0763	0.0761
GHS-DD	**0.1538**	**0.1258**	**0.0977**	**0.0971**	**0.0965**
ITQ	0.0436	0.0433	0.0429	0.0424	0.0419
IsoH	0.0016	0.0018	0.0020	0.0021	0.0022
HH	0.0207	0.0147	0.0088	0.0078	0.0068
SH	0.1515	0.0873	0.0232	0.0179	0.0126
IMH	0.1517	0.1158	0.0799	0.0793	0.0787
okmeans	0.0342	0.0234	0.0127	0.0091	0.0055
SpH	0.1315	0.0992	0.0668	0.0659	0.0651
	Extended Yale B				
c	128	160	192	224	256
GHS-DI	0.0777	0.0786	0.0796	0.0805	0.0815
GHS-DD	**0.0985**	**0.0997**	**0.1009**	**0.1021**	**0.1033**
ITQ	0.0427	0.0432	0.0437	0.0443	0.0448
IsoH	0.0022	0.0022	0.0023	0.0023	0.0023
HH	0.0069	0.0070	0.0071	0.0071	0.0072
SH	0.0128	0.0130	0.0131	0.0133	0.0134
IMH	0.0803	0.0813	0.0822	0.0832	0.0842
okmeans	0.0056	0.0057	0.0058	0.0059	0.0059
SpH	0.0664	0.0672	0.0680	0.0688	0.0696

Table A.8: F-measure results on FERET dataset

	FERET				
c	8	12	16	20	24
GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
GHS-DD	**0.1843**	**0.2296**	**0.2426**	**0.2362**	**0.2298**
ITQ	0.1654	0.1987	0.2159	0.1703	0.1247
IsoH	0.1648	0.1905	0.2210	0.1917	0.1624
HH	0.1615	0.1872	0.2136	0.1151	0.0166
SH	0.0817	0.1246	0.1498	0.1581	0.1663
IMH	0.1611	0.1698	0.1754	0.1710	0.1667
okmeans	0.1697	0.2136	0.2421	0.1475	0.0529
SpH	0.0799	0.1179	0.1357	0.1311	0.1265
	FERET				
c	32	48	64	80	96
GHS-DI	0.1341	0.1053	0.0765	0.0763	0.0761
GHS-DD	**0.1538**	**0.1258**	**0.0977**	**0.0971**	**0.0965**
ITQ	0.0436	0.0433	0.0429	0.0424	0.0419
IsoH	0.0016	0.0018	0.0020	0.0021	0.0022
HH	0.0207	0.0147	0.0088	0.0078	0.0068
SH	0.1515	0.0873	0.0232	0.0179	0.0126
IMH	0.1517	0.1158	0.0799	0.0793	0.0787
okmeans	0.0342	0.0234	0.0127	0.0091	0.0055
SpH	0.1315	0.0992	0.0668	0.0659	0.0651
	FERET				
c	128	160	192	224	256
GHS-DI	0.0777	0.0786	0.0796	0.0805	0.0815
GHS-DD	**0.0985**	**0.0997**	**0.1009**	**0.1021**	**0.1033**
ITQ	0.0427	0.0432	0.0437	0.0443	0.0448
IsoH	0.0022	0.0022	0.0023	0.0023	0.0023
HH	0.0069	0.0070	0.0071	0.0071	0.0072
SH	0.0128	0.0130	0.0131	0.0133	0.0134
IMH	0.0803	0.0813	0.0822	0.0832	0.0842
okmeans	0.0056	0.0057	0.0058	0.0059	0.0059
SpH	0.0664	0.0672	0.0680	0.0688	0.0696

Table A.9: F-measure results on FRGC dataset

	FRGC				
c	8	12	16	20	24
GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
GHS-DD	**0.1843**	**0.2296**	**0.2426**	**0.2362**	**0.2298**
ITQ	0.1654	0.1987	0.2159	0.1703	0.1247
IsoH	0.1648	0.1905	0.2210	0.1917	0.1624
HH	0.1615	0.1872	0.2136	0.1151	0.0166
SH	0.0817	0.1246	0.1498	0.1581	0.1663
IMH	0.1611	0.1698	0.1754	0.1710	0.1667
okmeans	0.1697	0.2136	0.2421	0.1475	0.0529
SpH	0.0799	0.1179	0.1357	0.1311	0.1265
	FRGC				
c	32	48	64	80	96
GHS-DI	0.1341	0.1053	0.0765	0.0763	0.0761
GHS-DD	**0.1538**	**0.1258**	**0.0977**	**0.0971**	**0.0965**
ITQ	0.0436	0.0433	0.0429	0.0424	0.0419
IsoH	0.0016	0.0018	0.0020	0.0021	0.0022
HH	0.0207	0.0147	0.0088	0.0078	0.0068
SH	0.1515	0.0873	0.0232	0.0179	0.0126
IMH	0.1517	0.1158	0.0799	0.0793	0.0787
okmeans	0.0342	0.0234	0.0127	0.0091	0.0055
SpH	0.1315	0.0992	0.0668	0.0659	0.0651
	FRGC				
c	128	160	192	224	256
GHS-DI	0.0777	0.0786	0.0796	0.0805	0.0815
GHS-DD	**0.0985**	**0.0997**	**0.1009**	**0.1021**	**0.1033**
ITQ	0.0427	0.0432	0.0437	0.0443	0.0448
IsoH	0.0022	0.0022	0.0023	0.0023	0.0023
HH	0.0069	0.0070	0.0071	0.0071	0.0072
SH	0.0128	0.0130	0.0131	0.0133	0.0134
IMH	0.0803	0.0813	0.0822	0.0832	0.0842
okmeans	0.0056	0.0057	0.0058	0.0059	0.0059
SpH	0.0664	0.0672	0.0680	0.0688	0.0696

Table A.10: F-measure results on YouTube Faces dataset

	YouTube Faces				
c	8	12	16	20	24
GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
GHS-DD	**0.1843**	**0.2296**	**0.2426**	**0.2362**	**0.2298**
ITQ	0.1654	0.1987	0.2159	0.1703	0.1247
IsoH	0.1648	0.1905	0.2210	0.1917	0.1624
HH	0.1615	0.1872	0.2136	0.1151	0.0166
SH	0.0817	0.1246	0.1498	0.1581	0.1663
IMH	0.1611	0.1698	0.1754	0.1710	0.1667
okmeans	0.1697	0.2136	0.2421	0.1475	0.0529
SpH	0.0799	0.1179	0.1357	0.1311	0.1265
	YouTube Faces				
c	32	48	64	80	96
GHS-DI	0.1341	0.1053	0.0765	0.0763	0.0761
GHS-DD	**0.1538**	**0.1258**	**0.0977**	**0.0971**	**0.0965**
ITQ	0.0436	0.0433	0.0429	0.0424	0.0419
IsoH	0.0016	0.0018	0.0020	0.0021	0.0022
HH	0.0207	0.0147	0.0088	0.0078	0.0068
SH	0.1515	0.0873	0.0232	0.0179	0.0126
IMH	0.1517	0.1158	0.0799	0.0793	0.0787
okmeans	0.0342	0.0234	0.0127	0.0091	0.0055
SpH	0.1315	0.0992	0.0668	0.0659	0.0651
	YouTube Faces				
c	128	160	192	224	256
GHS-DI	0.0777	0.0786	0.0796	0.0805	0.0815
GHS-DD	**0.0985**	**0.0997**	**0.1009**	**0.1021**	**0.1033**
ITQ	0.0427	0.0432	0.0437	0.0443	0.0448
IsoH	0.0022	0.0022	0.0023	0.0023	0.0023
HH	0.0069	0.0070	0.0071	0.0071	0.0072
SH	0.0128	0.0130	0.0131	0.0133	0.0134
IMH	0.0803	0.0813	0.0822	0.0832	0.0842
okmeans	0.0056	0.0057	0.0058	0.0059	0.0059
SpH	0.0664	0.0672	0.0680	0.0688	0.0696

Appendix B

Results of Multimodal Retrieval Method

Table B.1: MAP results on CMU-PIE dataset

	CMU-PIE				
c	8	12	16	20	24
DMH	**0.2035**	**0.2529**	**0.3027**	**0.3386**	**0.3712**
MDBE	0.1954	0.2396	0.2909	0.3257	0.3603
CCA-GHS-DI	0.1852	0.2308	0.2831	0.3137	0.3496
CCA-GHS-DD	0.1960	0.2457	0.2919	0.3319	0.3698
SePH	0.1779	0.2302	0.2816	0.3192	0.3625
CMSSH	0.1862	0.2327	0.2614	0.3038	0.3466
CCA-ITQ	0.1798	0.2360	0.2633	0.2992	0.3298
SCM	0.1798	0.2259	0.2497	0.2688	0.2818
CVH	0.1916	0.1975	0.2128	0.2232	0.2362
	CMU-PIE				
c	32	48	64	80	96
DMH	**0.4242**	**0.4916**	**0.5617**	**0.5847**	**0.6042**
MDBE	0.4142	0.4826	0.5518	0.5723	0.5952
CCA-GHS-DI	0.4049	0.4747	0.5406	0.5618	0.5864
CCA-GHS-DD	0.4260	0.4917	0.5548	0.5755	0.5949
SePH	0.4087	0.4705	0.5294	0.5516	0.5786
CMSSH	0.3977	0.4428	0.4876	0.5374	0.5788
CCA-ITQ	0.3711	0.4232	0.4710	0.5078	0.5413
SCM	0.2954	0.3128	0.3321	0.3235	0.3188
CVH	0.3037	0.3115	0.3233	0.3365	0.3510
	CMU-PIE				
c	128	160	192	224	256
DMH	**0.6198**	**0.6252**	**0.6346**	**0.6422**	**0.6477**
MDBE	0.6053	0.6122	0.6194	0.6288	0.6382
CCA-GHS-DI	0.5971	0.6071	0.6109	0.6180	0.6259
CCA-GHS-DD	0.6065	0.6147	0.6251	0.6333	0.6387
SePH	0.5916	0.5984	0.6060	0.6156	0.6209
CMSSH	0.5918	0.5983	0.6052	0.6148	0.6215
CCA-ITQ	0.5551	0.5599	0.5636	0.5735	0.5771
SCM	0.3243	0.3271	0.3355	0.3383	0.3411
CVH	0.3606	0.3653	0.3702	0.3729	0.3806

Table B.2: MAP results on Extended Yale B dataset

	Extended Yale B				
c	8	12	16	20	24
DMH	**0.2035**	**0.2529**	**0.3027**	**0.3386**	**0.3712**
MDBE	0.1954	0.2396	0.2909	0.3257	0.3603
CCA-GHS-DI	0.1852	0.2308	0.2831	0.3137	0.3496
CCA-GHS-DD	0.1960	0.2457	0.2919	0.3319	0.3698
SePH	0.1779	0.2302	0.2816	0.3192	0.3625
CMSSH	0.1862	0.2327	0.2614	0.3038	0.3466
CCA-ITQ	0.1798	0.2360	0.2633	0.2992	0.3298
SCM	0.1798	0.2259	0.2497	0.2688	0.2818
CVH	0.1916	0.1975	0.2128	0.2232	0.2362

	Extended Yale B				
c	32	48	64	80	96
DMH	**0.4242**	**0.4916**	**0.5617**	**0.5847**	**0.6042**
MDBE	0.4142	0.4826	0.5518	0.5723	0.5952
CCA-GHS-DI	0.4049	0.4747	0.5406	0.5618	0.5864
CCA-GHS-DD	0.4260	0.4917	0.5548	0.5755	0.5949
SePH	0.4087	0.4705	0.5294	0.5516	0.5786
CMSSH	0.3977	0.4428	0.4876	0.5374	0.5788
CCA-ITQ	0.3711	0.4232	0.4710	0.5078	0.5413
SCM	0.2954	0.3128	0.3321	0.3235	0.3188
CVH	0.3037	0.3115	0.3233	0.3365	0.3510

	Extended Yale B				
c	128	160	192	224	256
DMH	**0.6198**	**0.6252**	**0.6346**	**0.6422**	**0.6477**
MDBE	0.6053	0.6122	0.6194	0.6288	0.6382
CCA-GHS-DI	0.5971	0.6071	0.6109	0.6180	0.6259
CCA-GHS-DD	0.6065	0.6147	0.6251	0.6333	0.6387
SePH	0.5916	0.5984	0.6060	0.6156	0.6209
CMSSH	0.5918	0.5983	0.6052	0.6148	0.6215
CCA-ITQ	0.5551	0.5599	0.5636	0.5735	0.5771
SCM	0.3243	0.3271	0.3355	0.3383	0.3411
CVH	0.3606	0.3653	0.3702	0.3729	0.3806

Table B.3: MAP results on FERET dataset

	FERET				
c	8	12	16	20	24
DMH	**0.2035**	**0.2529**	**0.3027**	**0.3386**	**0.3712**
MDBE	0.1954	0.2396	0.2909	0.3257	0.3603
CCA-GHS-DI	0.1852	0.2308	0.2831	0.3137	0.3496
CCA-GHS-DD	0.1960	0.2457	0.2919	0.3319	0.3698
SePH	0.1779	0.2302	0.2816	0.3192	0.3625
CMSSH	0.1862	0.2327	0.2614	0.3038	0.3466
CCA-ITQ	0.1798	0.2360	0.2633	0.2992	0.3298
SCM	0.1798	0.2259	0.2497	0.2688	0.2818
CVH	0.1916	0.1975	0.2128	0.2232	0.2362
	FERET				
c	32	48	64	80	96
DMH	**0.4242**	**0.4916**	**0.5617**	**0.5847**	**0.6042**
MDBE	0.4142	0.4826	0.5518	0.5723	0.5952
CCA-GHS-DI	0.4049	0.4747	0.5406	0.5618	0.5864
CCA-GHS-DD	0.4260	0.4917	0.5548	0.5755	0.5949
SePH	0.4087	0.4705	0.5294	0.5516	0.5786
CMSSH	0.3977	0.4428	0.4876	0.5374	0.5788
CCA-ITQ	0.3711	0.4232	0.4710	0.5078	0.5413
SCM	0.2954	0.3128	0.3321	0.3235	0.3188
CVH	0.3037	0.3115	0.3233	0.3365	0.3510
	FERET				
c	128	160	192	224	256
DMH	**0.6198**	**0.6252**	**0.6346**	**0.6422**	**0.6477**
MDBE	0.6053	0.6122	0.6194	0.6288	0.6382
CCA-GHS-DI	0.5971	0.6071	0.6109	0.6180	0.6259
CCA-GHS-DD	0.6065	0.6147	0.6251	0.6333	0.6387
SePH	0.5916	0.5984	0.6060	0.6156	0.6209
CMSSH	0.5918	0.5983	0.6052	0.6148	0.6215
CCA-ITQ	0.5551	0.5599	0.5636	0.5735	0.5771
SCM	0.3243	0.3271	0.3355	0.3383	0.3411
CVH	0.3606	0.3653	0.3702	0.3729	0.3806

Table B.4: MAP results on FRGC dataset

	FRGC				
c	8	12	16	20	24
DMH	**0.2035**	**0.2529**	**0.3027**	**0.3386**	**0.3712**
MDBE	0.1954	0.2396	0.2909	0.3257	0.3603
CCA-GHS-DI	0.1852	0.2308	0.2831	0.3137	0.3496
CCA-GHS-DD	0.1960	0.2457	0.2919	0.3319	0.3698
SePH	0.1779	0.2302	0.2816	0.3192	0.3625
CMSSH	0.1862	0.2327	0.2614	0.3038	0.3466
CCA-ITQ	0.1798	0.2360	0.2633	0.2992	0.3298
SCM	0.1798	0.2259	0.2497	0.2688	0.2818
CVH	0.1916	0.1975	0.2128	0.2232	0.2362
	FRGC				
c	32	48	64	80	96
DMH	**0.4242**	**0.4916**	**0.5617**	**0.5847**	**0.6042**
MDBE	0.4142	0.4826	0.5518	0.5723	0.5952
CCA-GHS-DI	0.4049	0.4747	0.5406	0.5618	0.5864
CCA-GHS-DD	0.4260	0.4917	0.5548	0.5755	0.5949
SePH	0.4087	0.4705	0.5294	0.5516	0.5786
CMSSH	0.3977	0.4428	0.4876	0.5374	0.5788
CCA-ITQ	0.3711	0.4232	0.4710	0.5078	0.5413
SCM	0.2954	0.3128	0.3321	0.3235	0.3188
CVH	0.3037	0.3115	0.3233	0.3365	0.3510
	FRGC				
c	128	160	192	224	256
DMH	**0.6198**	**0.6252**	**0.6346**	**0.6422**	**0.6477**
MDBE	0.6053	0.6122	0.6194	0.6288	0.6382
CCA-GHS-DI	0.5971	0.6071	0.6109	0.6180	0.6259
CCA-GHS-DD	0.6065	0.6147	0.6251	0.6333	0.6387
SePH	0.5916	0.5984	0.6060	0.6156	0.6209
CMSSH	0.5918	0.5983	0.6052	0.6148	0.6215
CCA-ITQ	0.5551	0.5599	0.5636	0.5735	0.5771
SCM	0.3243	0.3271	0.3355	0.3383	0.3411
CVH	0.3606	0.3653	0.3702	0.3729	0.3806

Table B.5: MAP results on YouTube Faces dataset

	YouTube Faces				
c	8	12	16	20	24
DMH	**0.2035**	**0.2529**	**0.3027**	**0.3386**	**0.3712**
MDBE	0.1954	0.2396	0.2909	0.3257	0.3603
CCA-GHS-DI	0.1852	0.2308	0.2831	0.3137	0.3496
CCA-GHS-DD	0.1960	0.2457	0.2919	0.3319	0.3698
SePH	0.1779	0.2302	0.2816	0.3192	0.3625
CMSSH	0.1862	0.2327	0.2614	0.3038	0.3466
CCA-ITQ	0.1798	0.2360	0.2633	0.2992	0.3298
SCM	0.1798	0.2259	0.2497	0.2688	0.2818
CVH	0.1916	0.1975	0.2128	0.2232	0.2362
	YouTube Faces				
c	32	48	64	80	96
DMH	**0.4242**	**0.4916**	**0.5617**	**0.5847**	**0.6042**
MDBE	0.4142	0.4826	0.5518	0.5723	0.5952
CCA-GHS-DI	0.4049	0.4747	0.5406	0.5618	0.5864
CCA-GHS-DD	0.4260	0.4917	0.5548	0.5755	0.5949
SePH	0.4087	0.4705	0.5294	0.5516	0.5786
CMSSH	0.3977	0.4428	0.4876	0.5374	0.5788
CCA-ITQ	0.3711	0.4232	0.4710	0.5078	0.5413
SCM	0.2954	0.3128	0.3321	0.3235	0.3188
CVH	0.3037	0.3115	0.3233	0.3365	0.3510
	YouTube Faces				
c	128	160	192	224	256
DMH	**0.6198**	**0.6252**	**0.6346**	**0.6422**	**0.6477**
MDBE	0.6053	0.6122	0.6194	0.6288	0.6382
CCA-GHS-DI	0.5971	0.6071	0.6109	0.6180	0.6259
CCA-GHS-DD	0.6065	0.6147	0.6251	0.6333	0.6387
SePH	0.5916	0.5984	0.6060	0.6156	0.6209
CMSSH	0.5918	0.5983	0.6052	0.6148	0.6215
CCA-ITQ	0.5551	0.5599	0.5636	0.5735	0.5771
SCM	0.3243	0.3271	0.3355	0.3383	0.3411
CVH	0.3606	0.3653	0.3702	0.3729	0.3806

Table B.6: F-measure results on CMU-PIE dataset

	CMU-PIE				
c	8	12	16	20	24
DMH	**0.3406**	**0.5245**	**0.6703**	**0.7264**	**0.7825**
MDBE	0.2056	0.2545	0.2653	0.2539	0.2425
CCA-GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
CCA-GHS-DD	0.1843	0.2296	0.2426	0.2362	0.2298
SePH	0.1654	0.1987	0.2159	0.1703	0.1247
CMSSH	0.1648	0.1905	0.2210	0.1917	0.1624
CCA-ITQ	0.1615	0.1872	0.2136	0.1151	0.0166
SCM	0.0817	0.1246	0.1498	0.1581	0.1663
CVH	0.1611	0.1698	0.1754	0.1710	0.1667
	CMU-PIE				
c	32	48	64	80	96
DMH	**0.5661**	**0.5373**	**0.5085**	**0.3733**	**0.2381**
MDBE	0.1611	0.1323	0.1035	0.1033	0.1031
CCA-GHS-DI	0.1341	0.1053	0.0765	0.0763	0.0761
CCA-GHS-DD	0.1538	0.1258	0.0977	0.0971	0.0965
SePH	0.0436	0.0433	0.0429	0.0424	0.0418
CMSSH	0.0016	0.0018	0.0020	0.0021	0.0022
CCA-ITQ	0.0207	0.0147	0.0088	0.0078	0.0067
SCM	0.1515	0.0873	0.0232	0.0179	0.0126
CVH	0.1517	0.1158	0.0799	0.0793	0.0787
	CMU-PIE				
c	128	160	192	224	256
DMH	**0.2429**	**0.2459**	**0.2489**	**0.2518**	**0.2548**
MDBE	0.1052	0.1065	0.1078	0.1091	0.1104
CCA-GHS-DI	0.0777	0.0786	0.0796	0.0805	0.0815
CCA-GHS-DD	0.0985	0.0997	0.1009	0.1021	0.1033
SePH	0.0427	0.0432	0.0437	0.0443	0.0448
CMSSH	0.0022	0.0022	0.0023	0.0023	0.0023
CCA-ITQ	0.0069	0.0070	0.0071	0.0071	0.0072
SCM	0.0128	0.0130	0.0131	0.0133	0.0134
CVH	0.0803	0.0813	0.0822	0.0832	0.0842

Table B.7: F-measure results on Extended Yale B dataset

	Extended Yale B				
c	8	12	16	20	24
DMH	**0.3406**	**0.5245**	**0.6703**	**0.7264**	**0.7825**
MDBE	0.2056	0.2545	0.2653	0.2539	0.2425
CCA-GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
CCA-GHS-DD	0.1843	0.2296	0.2426	0.2362	0.2298
SePH	0.1654	0.1987	0.2159	0.1703	0.1247
CMSSH	0.1648	0.1905	0.2210	0.1917	0.1624
CCA-ITQ	0.1615	0.1872	0.2136	0.1151	0.0166
SCM	0.0817	0.1246	0.1498	0.1581	0.1663
CVH	0.1611	0.1698	0.1754	0.1710	0.1667

	Extended Yale B				
c	32	48	64	80	96
DMH	**0.5661**	**0.5373**	**0.5085**	**0.3733**	**0.2381**
MDBE	0.1611	0.1323	0.1035	0.1033	0.1031
CCA-GHS-DI	0.1341	0.1053	0.0765	0.0763	0.0761
CCA-GHS-DD	0.1538	0.1258	0.0977	0.0971	0.0965
SePH	0.0436	0.0433	0.0429	0.0424	0.0418
CMSSH	0.0016	0.0018	0.0020	0.0021	0.0022
CCA-ITQ	0.0207	0.0147	0.0088	0.0078	0.0067
SCM	0.1515	0.0873	0.0232	0.0179	0.0126
CVH	0.1517	0.1158	0.0799	0.0793	0.0787

	Extended Yale B				
c	128	160	192	224	256
DMH	**0.2429**	**0.2459**	**0.2489**	**0.2518**	**0.2548**
MDBE	0.1052	0.1065	0.1078	0.1091	0.1104
CCA-GHS-DI	0.0777	0.0786	0.0796	0.0805	0.0815
CCA-GHS-DD	0.0985	0.0997	0.1009	0.1021	0.1033
SePH	0.0427	0.0432	0.0437	0.0443	0.0448
CMSSH	0.0022	0.0022	0.0023	0.0023	0.0023
CCA-ITQ	0.0069	0.0070	0.0071	0.0071	0.0072
SCM	0.0128	0.0130	0.0131	0.0133	0.0134
CVH	0.0803	0.0813	0.0822	0.0832	0.0842

Table B.8: F-measure results on FERET dataset

	FERET				
c	8	12	16	20	24
DMH	**0.3406**	**0.5245**	**0.6703**	**0.7264**	**0.7825**
MDBE	0.2056	0.2545	0.2653	0.2539	0.2425
CCA-GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
CCA-GHS-DD	0.1843	0.2296	0.2426	0.2362	0.2298
SePH	0.1654	0.1987	0.2159	0.1703	0.1247
CMSSH	0.1648	0.1905	0.2210	0.1917	0.1624
CCA-ITQ	0.1615	0.1872	0.2136	0.1151	0.0166
SCM	0.0817	0.1246	0.1498	0.1581	0.1663
CVH	0.1611	0.1698	0.1754	0.1710	0.1667
	FERET				
c	32	48	64	80	96
DMH	**0.5661**	**0.5373**	**0.5085**	**0.3733**	**0.2381**
MDBE	0.1611	0.1323	0.1035	0.1033	0.1031
CCA-GHS-DI	0.1341	0.1053	0.0765	0.0763	0.0761
CCA-GHS-DD	0.1538	0.1258	0.0977	0.0971	0.0965
SePH	0.0436	0.0433	0.0429	0.0424	0.0418
CMSSH	0.0016	0.0018	0.0020	0.0021	0.0022
CCA-ITQ	0.0207	0.0147	0.0088	0.0078	0.0067
SCM	0.1515	0.0873	0.0232	0.0179	0.0126
CVH	0.1517	0.1158	0.0799	0.0793	0.0787
	FERET				
c	128	160	192	224	256
DMH	**0.2429**	**0.2459**	**0.2489**	**0.2518**	**0.2548**
MDBE	0.1052	0.1065	0.1078	0.1091	0.1104
CCA-GHS-DI	0.0777	0.0786	0.0796	0.0805	0.0815
CCA-GHS-DD	0.0985	0.0997	0.1009	0.1021	0.1033
SePH	0.0427	0.0432	0.0437	0.0443	0.0448
CMSSH	0.0022	0.0022	0.0023	0.0023	0.0023
CCA-ITQ	0.0069	0.0070	0.0071	0.0071	0.0072
SCM	0.0128	0.0130	0.0131	0.0133	0.0134
CVH	0.0803	0.0813	0.0822	0.0832	0.0842

Table B.9: F-measure results on FRGC dataset

	FRGC				
c	8	12	16	20	24
DMH	**0.3406**	**0.5245**	**0.6703**	**0.7264**	**0.7825**
MDBE	0.2056	0.2545	0.2653	0.2539	0.2425
CCA-GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
CCA-GHS-DD	0.1843	0.2296	0.2426	0.2362	0.2298
SePH	0.1654	0.1987	0.2159	0.1703	0.1247
CMSSH	0.1648	0.1905	0.2210	0.1917	0.1624
CCA-ITQ	0.1615	0.1872	0.2136	0.1151	0.0166
SCM	0.0817	0.1246	0.1498	0.1581	0.1663
CVH	0.1611	0.1698	0.1754	0.1710	0.1667
	FRGC				
c	32	48	64	80	96
DMH	**0.5661**	**0.5373**	**0.5085**	**0.3733**	**0.2381**
MDBE	0.1611	0.1323	0.1035	0.1033	0.1031
CCA-GHS-DI	0.1341	0.1053	0.0765	0.0763	0.0761
CCA-GHS-DD	0.1538	0.1258	0.0977	0.0971	0.0965
SePH	0.0436	0.0433	0.0429	0.0424	0.0418
CMSSH	0.0016	0.0018	0.0020	0.0021	0.0022
CCA-ITQ	0.0207	0.0147	0.0088	0.0078	0.0067
SCM	0.1515	0.0873	0.0232	0.0179	0.0126
CVH	0.1517	0.1158	0.0799	0.0793	0.0787
	FRGC				
c	128	160	192	224	256
DMH	**0.2429**	**0.2459**	**0.2489**	**0.2518**	**0.2548**
MDBE	0.1052	0.1065	0.1078	0.1091	0.1104
CCA-GHS-DI	0.0777	0.0786	0.0796	0.0805	0.0815
CCA-GHS-DD	0.0985	0.0997	0.1009	0.1021	0.1033
SePH	0.0427	0.0432	0.0437	0.0443	0.0448
CMSSH	0.0022	0.0022	0.0023	0.0023	0.0023
CCA-ITQ	0.0069	0.0070	0.0071	0.0071	0.0072
SCM	0.0128	0.0130	0.0131	0.0133	0.0134
CVH	0.0803	0.0813	0.0822	0.0832	0.0842

Table B.10: F-measure results on YouTube Faces dataset

	YouTube Faces				
c	8	12	16	20	24
DMH	**0.3406**	**0.5245**	**0.6703**	**0.7264**	**0.7825**
MDBE	0.2056	0.2545	0.2653	0.2539	0.2425
CCA-GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
CCA-GHS-DD	0.1843	0.2296	0.2426	0.2362	0.2298
SePH	0.1654	0.1987	0.2159	0.1703	0.1247
CMSSH	0.1648	0.1905	0.2210	0.1917	0.1624
CCA-ITQ	0.1615	0.1872	0.2136	0.1151	0.0166
SCM	0.0817	0.1246	0.1498	0.1581	0.1663
CVH	0.1611	0.1698	0.1754	0.1710	0.1667

	YouTube Faces				
c	32	48	64	80	96
DMH	**0.5661**	**0.5373**	**0.5085**	**0.3733**	**0.2381**
MDBE	0.1611	0.1323	0.1035	0.1033	0.1031
CCA-GHS-DI	0.1341	0.1053	0.0765	0.0763	0.0761
CCA-GHS-DD	0.1538	0.1258	0.0977	0.0971	0.0965
SePH	0.0436	0.0433	0.0429	0.0424	0.0418
CMSSH	0.0016	0.0018	0.0020	0.0021	0.0022
CCA-ITQ	0.0207	0.0147	0.0088	0.0078	0.0067
SCM	0.1515	0.0873	0.0232	0.0179	0.0126
CVH	0.1517	0.1158	0.0799	0.0793	0.0787

	YouTube Faces				
c	128	160	192	224	256
DMH	**0.2429**	**0.2459**	**0.2489**	**0.2518**	**0.2548**
MDBE	0.1052	0.1065	0.1078	0.1091	0.1104
CCA-GHS-DI	0.0777	0.0786	0.0796	0.0805	0.0815
CCA-GHS-DD	0.0985	0.0997	0.1009	0.1021	0.1033
SePH	0.0427	0.0432	0.0437	0.0443	0.0448
CMSSH	0.0022	0.0022	0.0023	0.0023	0.0023
CCA-ITQ	0.0069	0.0070	0.0071	0.0071	0.0072
SCM	0.0128	0.0130	0.0131	0.0133	0.0134
CVH	0.0803	0.0813	0.0822	0.0832	0.0842

Appendix C

Results of Hybrid Deep Neural Network

Table C.1: MAP results on CMU-PIE dataset

	CMU-PIE				
c	8	12	16	20	24
HDNN	**0.2972**	**0.3630**	**0.4334**	**0.4783**	**0.5246**
DMH	0.2713	0.3369	0.4048	0.4483	0.4960
MDBE	0.2554	0.3191	0.3902	0.4340	0.4820
CCA-GHS-DI	0.2439	0.3029	0.3736	0.4224	0.4663
CCA-GHS-DD	0.2623	0.3273	0.3878	0.4414	0.4961
SePH	0.2325	0.3090	0.3734	0.4266	0.4841
CMSSH	0.2472	0.3073	0.3439	0.4091	0.4634
CCA-ITQ	0.2396	0.3174	0.3532	0.3951	0.4399
SCM	0.2397	0.3059	0.3343	0.3552	0.3811
CVH	0.2567	0.2646	0.2836	0.2974	0.3154
	CMU-PIE				
c	32	48	64	80	96
HDNN	**0.5904**	**0.6822**	**0.7811**	**0.8021**	**0.8376**
DMH	0.5644	0.6575	0.7520	0.7796	0.8078
MDBE	0.5479	0.6445	0.7340	0.7663	0.7928
CCA-GHS-DI	0.5366	0.6309	0.7191	0.7516	0.7775
CCA-GHS-DD	0.5752	0.6574	0.7365	0.7658	0.7969
SePH	0.5476	0.6264	0.7047	0.7362	0.7741
CMSSH	0.5313	0.5912	0.6505	0.7138	0.7765
CCA-ITQ	0.4930	0.5593	0.6274	0.6749	0.7215
SCM	0.3946	0.4188	0.4404	0.4350	0.4251
CVH	0.4039	0.4126	0.4271	0.4468	0.4702
	CMU-PIE				
c	128	160	192	224	256
HDNN	**0.8494**	**0.8615**	**0.8695**	**0.8831**	**0.8932**
DMH	0.8239	0.8345	0.8440	0.8525	0.8649
MDBE	0.8082	0.8208	0.8271	0.8380	0.8512
CCA-GHS-DI	0.7937	0.8034	0.8134	0.8234	0.8354
CCA-GHS-DD	0.8118	0.8231	0.8342	0.8448	0.8517
SePH	0.7882	0.8004	0.8070	0.8189	0.8312
CMSSH	0.7932	0.7992	0.8070	0.8182	0.8266
CCA-ITQ	0.7364	0.7476	0.7578	0.7665	0.7741
SCM	0.4328	0.4392	0.4463	0.4483	0.4577
CVH	0.4783	0.4844	0.4920	0.4943	0.5056

Table C.2: MAP results on Extended Yale B dataset

	Extended Yale B				
c	8	12	16	20	24
HDNN	**0.2972**	**0.3630**	**0.4334**	**0.4783**	**0.5246**
DMH	0.2713	0.3369	0.4048	0.4483	0.4960
MDBE	0.2554	0.3191	0.3902	0.4340	0.4820
CCA-GHS-DI	0.2439	0.3029	0.3736	0.4224	0.4663
CCA-GHS-DD	0.2623	0.3273	0.3878	0.4414	0.4961
SePH	0.2325	0.3090	0.3734	0.4266	0.4841
CMSSH	0.2472	0.3073	0.3439	0.4091	0.4634
CCA-ITQ	0.2396	0.3174	0.3532	0.3951	0.4399
SCM	0.2397	0.3059	0.3343	0.3552	0.3811
CVH	0.2567	0.2646	0.2836	0.2974	0.3154
	Extended Yale B				
c	32	48	64	80	96
HDNN	**0.5904**	**0.6822**	**0.7811**	**0.8021**	**0.8376**
DMH	0.5644	0.6575	0.7520	0.7796	0.8078
MDBE	0.5479	0.6445	0.7340	0.7663	0.7928
CCA-GHS-DI	0.5366	0.6309	0.7191	0.7516	0.7775
CCA-GHS-DD	0.5752	0.6574	0.7365	0.7658	0.7969
SePH	0.5476	0.6264	0.7047	0.7362	0.7741
CMSSH	0.5313	0.5912	0.6505	0.7138	0.7765
CCA-ITQ	0.4930	0.5593	0.6274	0.6749	0.7215
SCM	0.3946	0.4188	0.4404	0.4350	0.4251
CVH	0.4039	0.4126	0.4271	0.4468	0.4702
	Extended Yale B				
c	128	160	192	224	256
HDNN	**0.8494**	**0.8615**	**0.8695**	**0.8831**	**0.8932**
DMH	0.8239	0.8345	0.8440	0.8525	0.8649
MDBE	0.8082	0.8208	0.8271	0.8380	0.8512
CCA-GHS-DI	0.7937	0.8034	0.8134	0.8234	0.8354
CCA-GHS-DD	0.8118	0.8231	0.8342	0.8448	0.8517
SePH	0.7882	0.8004	0.8070	0.8189	0.8312
CMSSH	0.7932	0.7992	0.8070	0.8182	0.8266
CCA-ITQ	0.7364	0.7476	0.7578	0.7665	0.7741
SCM	0.4328	0.4392	0.4463	0.4483	0.4577
CVH	0.4783	0.4844	0.4920	0.4943	0.5056

Table C.3: MAP results on FERET dataset

	FERET				
c	8	12	16	20	24
HDNN	**0.2972**	**0.3630**	**0.4334**	**0.4783**	**0.5246**
DMH	0.2713	0.3369	0.4048	0.4483	0.4960
MDBE	0.2554	0.3191	0.3902	0.4340	0.4820
CCA-GHS-DI	0.2439	0.3029	0.3736	0.4224	0.4663
CCA-GHS-DD	0.2623	0.3273	0.3878	0.4414	0.4961
SePH	0.2325	0.3090	0.3734	0.4266	0.4841
CMSSH	0.2472	0.3073	0.3439	0.4091	0.4634
CCA-ITQ	0.2396	0.3174	0.3532	0.3951	0.4399
SCM	0.2397	0.3059	0.3343	0.3552	0.3811
CVH	0.2567	0.2646	0.2836	0.2974	0.3154
	FERET				
c	32	48	64	80	96
HDNN	**0.5904**	**0.6822**	**0.7811**	**0.8021**	**0.8376**
DMH	0.5644	0.6575	0.7520	0.7796	0.8078
MDBE	0.5479	0.6445	0.7340	0.7663	0.7928
CCA-GHS-DI	0.5366	0.6309	0.7191	0.7516	0.7775
CCA-GHS-DD	0.5752	0.6574	0.7365	0.7658	0.7969
SePH	0.5476	0.6264	0.7047	0.7362	0.7741
CMSSH	0.5313	0.5912	0.6505	0.7138	0.7765
CCA-ITQ	0.4930	0.5593	0.6274	0.6749	0.7215
SCM	0.3946	0.4188	0.4404	0.4350	0.4251
CVH	0.4039	0.4126	0.4271	0.4468	0.4702
	FERET				
c	128	160	192	224	256
HDNN	**0.8494**	**0.8615**	**0.8695**	**0.8831**	**0.8932**
DMH	0.8239	0.8345	0.8440	0.8525	0.8649
MDBE	0.8082	0.8208	0.8271	0.8380	0.8512
CCA-GHS-DI	0.7937	0.8034	0.8134	0.8234	0.8354
CCA-GHS-DD	0.8118	0.8231	0.8342	0.8448	0.8517
SePH	0.7882	0.8004	0.8070	0.8189	0.8312
CMSSH	0.7932	0.7992	0.8070	0.8182	0.8266
CCA-ITQ	0.7364	0.7476	0.7578	0.7665	0.7741
SCM	0.4328	0.4392	0.4463	0.4483	0.4577
CVH	0.4783	0.4844	0.4920	0.4943	0.5056

Table C.4: MAP results on FRGC dataset

	FRGC				
c	8	12	16	20	24
HDNN	**0.2972**	**0.3630**	**0.4334**	**0.4783**	**0.5246**
DMH	0.2713	0.3369	0.4048	0.4483	0.4960
MDBE	0.2554	0.3191	0.3902	0.4340	0.4820
CCA-GHS-DI	0.2439	0.3029	0.3736	0.4224	0.4663
CCA-GHS-DD	0.2623	0.3273	0.3878	0.4414	0.4961
SePH	0.2325	0.3090	0.3734	0.4266	0.4841
CMSSH	0.2472	0.3073	0.3439	0.4091	0.4634
CCA-ITQ	0.2396	0.3174	0.3532	0.3951	0.4399
SCM	0.2397	0.3059	0.3343	0.3552	0.3811
CVH	0.2567	0.2646	0.2836	0.2974	0.3154
	FRGC				
c	32	48	64	80	96
HDNN	**0.5904**	**0.6822**	**0.7811**	**0.8021**	**0.8376**
DMH	0.5644	0.6575	0.7520	0.7796	0.8078
MDBE	0.5479	0.6445	0.7340	0.7663	0.7928
CCA-GHS-DI	0.5366	0.6309	0.7191	0.7516	0.7775
CCA-GHS-DD	0.5752	0.6574	0.7365	0.7658	0.7969
SePH	0.5476	0.6264	0.7047	0.7362	0.7741
CMSSH	0.5313	0.5912	0.6505	0.7138	0.7765
CCA-ITQ	0.4930	0.5593	0.6274	0.6749	0.7215
SCM	0.3946	0.4188	0.4404	0.4350	0.4251
CVH	0.4039	0.4126	0.4271	0.4468	0.4702
	FRGC				
c	128	160	192	224	256
HDNN	**0.8494**	**0.8615**	**0.8695**	**0.8831**	**0.8932**
DMH	0.8239	0.8345	0.8440	0.8525	0.8649
MDBE	0.8082	0.8208	0.8271	0.8380	0.8512
CCA-GHS-DI	0.7937	0.8034	0.8134	0.8234	0.8354
CCA-GHS-DD	0.8118	0.8231	0.8342	0.8448	0.8517
SePH	0.7882	0.8004	0.8070	0.8189	0.8312
CMSSH	0.7932	0.7992	0.8070	0.8182	0.8266
CCA-ITQ	0.7364	0.7476	0.7578	0.7665	0.7741
SCM	0.4328	0.4392	0.4463	0.4483	0.4577
CVH	0.4783	0.4844	0.4920	0.4943	0.5056

Table C.5: MAP results on YouTube Faces dataset

	YouTube Faces				
c	8	12	16	20	24
HDNN	**0.2972**	**0.3630**	**0.4334**	**0.4783**	**0.5246**
DMH	0.2713	0.3369	0.4048	0.4483	0.4960
MDBE	0.2554	0.3191	0.3902	0.4340	0.4820
CCA-GHS-DI	0.2439	0.3029	0.3736	0.4224	0.4663
CCA-GHS-DD	0.2623	0.3273	0.3878	0.4414	0.4961
SePH	0.2325	0.3090	0.3734	0.4266	0.4841
CMSSH	0.2472	0.3073	0.3439	0.4091	0.4634
CCA-ITQ	0.2396	0.3174	0.3532	0.3951	0.4399
SCM	0.2397	0.3059	0.3343	0.3552	0.3811
CVH	0.2567	0.2646	0.2836	0.2974	0.3154
	YouTube Faces				
c	32	48	64	80	96
HDNN	**0.5904**	**0.6822**	**0.7811**	**0.8021**	**0.8376**
DMH	0.5644	0.6575	0.7520	0.7796	0.8078
MDBE	0.5479	0.6445	0.7340	0.7663	0.7928
CCA-GHS-DI	0.5366	0.6309	0.7191	0.7516	0.7775
CCA-GHS-DD	0.5752	0.6574	0.7365	0.7658	0.7969
SePH	0.5476	0.6264	0.7047	0.7362	0.7741
CMSSH	0.5313	0.5912	0.6505	0.7138	0.7765
CCA-ITQ	0.4930	0.5593	0.6274	0.6749	0.7215
SCM	0.3946	0.4188	0.4404	0.4350	0.4251
CVH	0.4039	0.4126	0.4271	0.4468	0.4702
	YouTube Faces				
c	128	160	192	224	256
HDNN	**0.8494**	**0.8615**	**0.8695**	**0.8831**	**0.8932**
DMH	0.8239	0.8345	0.8440	0.8525	0.8649
MDBE	0.8082	0.8208	0.8271	0.8380	0.8512
CCA-GHS-DI	0.7937	0.8034	0.8134	0.8234	0.8354
CCA-GHS-DD	0.8118	0.8231	0.8342	0.8448	0.8517
SePH	0.7882	0.8004	0.8070	0.8189	0.8312
CMSSH	0.7932	0.7992	0.8070	0.8182	0.8266
CCA-ITQ	0.7364	0.7476	0.7578	0.7665	0.7741
SCM	0.4328	0.4392	0.4463	0.4483	0.4577
CVH	0.4783	0.4844	0.4920	0.4943	0.5056

Table C.6: F-measure results on CMU-PIE dataset

	CMU-PIE				
c	8	12	16	20	24
HDNN	**0.3541**	**0.5393**	**0.6836**	**0.7397**	**0.7958**
DMH	0.3406	0.5245	0.6703	0.7264	0.7825
MDBE	0.2056	0.2545	0.2653	0.2539	0.2425
CCA-GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
CCA-GHS-DD	0.1843	0.2296	0.2426	0.2362	0.2298
SePH	0.1654	0.1987	0.2159	0.1703	0.1247
CMSSH	0.1648	0.1905	0.2210	0.1917	0.1624
CCA-ITQ	0.1615	0.1872	0.2136	0.1151	0.0166
SCM	0.0817	0.1246	0.1499	0.1581	0.1663
CVH	0.1611	0.1698	0.1754	0.1710	0.1667
	CMU-PIE				
c	32	48	64	80	96
HDNN	**0.5796**	**0.5508**	**0.5220**	**0.3868**	**0.2516**
DMH	0.5661	0.5373	0.5085	0.3733	0.2381
MDBE	0.1611	0.1323	0.1035	0.1033	0.1031
CCA-GHS-DI	0.1341	0.1053	0.0765	0.0763	0.0761
CCA-GHS-DD	0.1538	0.1258	0.0977	0.0971	0.0965
SePH	0.0436	0.0433	0.0429	0.0424	0.0419
CMSSH	0.0016	0.0018	0.0020	0.0021	0.0022
CCA-ITQ	0.0207	0.0147	0.0088	0.0078	0.0067
SCM	0.1515	0.0873	0.0232	0.0179	0.0126
CVH	0.1517	0.1158	0.0799	0.0793	0.0787
	CMU-PIE				
c	128	160	192	224	256
HDNN	**0.2567**	**0.2598**	**0.2630**	**0.2661**	**0.2693**
DMH	0.2429	0.2459	0.2489	0.2518	0.2548
MDBE	0.1052	0.1065	0.1078	0.1091	0.1104
CCA-GHS-DI	0.0777	0.0786	0.0796	0.0805	0.0815
CCA-GHS-DD	0.0985	0.0997	0.1009	0.1021	0.1033
SePH	0.0427	0.0432	0.0437	0.0443	0.0448
CMSSH	0.0022	0.0022	0.0023	0.0023	0.0023
CCA-ITQ	0.0069	0.0070	0.0071	0.0071	0.0072
SCM	0.0128	0.0130	0.0131	0.0133	0.0134
CVH	0.0803	0.0813	0.0822	0.0832	0.0842

Table C.7: F-measure results on Extended Yale B dataset

	Extended Yale B				
c	8	12	16	20	24
HDNN	**0.3541**	**0.5393**	**0.6836**	**0.7397**	**0.7958**
DMH	0.3406	0.5245	0.6703	0.7264	0.7825
MDBE	0.2056	0.2545	0.2653	0.2539	0.2425
CCA-GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
CCA-GHS-DD	0.1843	0.2296	0.2426	0.2362	0.2298
SePH	0.1654	0.1987	0.2159	0.1703	0.1247
CMSSH	0.1648	0.1905	0.2210	0.1917	0.1624
CCA-ITQ	0.1615	0.1872	0.2136	0.1151	0.0166
SCM	0.0817	0.1246	0.1499	0.1581	0.1663
CVH	0.1611	0.1698	0.1754	0.1710	0.1667
	Extended Yale B				
c	32	48	64	80	96
HDNN	**0.5796**	**0.5508**	**0.5220**	**0.3868**	**0.2516**
DMH	0.5661	0.5373	0.5085	0.3733	0.2381
MDBE	0.1611	0.1323	0.1035	0.1033	0.1031
CCA-GHS-DI	0.1341	0.1053	0.0765	0.0763	0.0761
CCA-GHS-DD	0.1538	0.1258	0.0977	0.0971	0.0965
SePH	0.0436	0.0433	0.0429	0.0424	0.0419
CMSSH	0.0016	0.0018	0.0020	0.0021	0.0022
CCA-ITQ	0.0207	0.0147	0.0088	0.0078	0.0067
SCM	0.1515	0.0873	0.0232	0.0179	0.0126
CVH	0.1517	0.1158	0.0799	0.0793	0.0787
	Extended Yale B				
c	128	160	192	224	256
HDNN	**0.2567**	**0.2598**	**0.2630**	**0.2661**	**0.2693**
DMH	0.2429	0.2459	0.2489	0.2518	0.2548
MDBE	0.1052	0.1065	0.1078	0.1091	0.1104
CCA-GHS-DI	0.0777	0.0786	0.0796	0.0805	0.0815
CCA-GHS-DD	0.0985	0.0997	0.1009	0.1021	0.1033
SePH	0.0427	0.0432	0.0437	0.0443	0.0448
CMSSH	0.0022	0.0022	0.0023	0.0023	0.0023
CCA-ITQ	0.0069	0.0070	0.0071	0.0071	0.0072
SCM	0.0128	0.0130	0.0131	0.0133	0.0134
CVH	0.0803	0.0813	0.0822	0.0832	0.0842

Table C.8: F-measure results on FERET dataset

	FERET				
c	8	12	16	20	24
HDNN	**0.3541**	**0.5393**	**0.6836**	**0.7397**	**0.7958**
DMH	0.3406	0.5245	0.6703	0.7264	0.7825
MDBE	0.2056	0.2545	0.2653	0.2539	0.2425
CCA-GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
CCA-GHS-DD	0.1843	0.2296	0.2426	0.2362	0.2298
SePH	0.1654	0.1987	0.2159	0.1703	0.1247
CMSSH	0.1648	0.1905	0.2210	0.1917	0.1624
CCA-ITQ	0.1615	0.1872	0.2136	0.1151	0.0166
SCM	0.0817	0.1246	0.1499	0.1581	0.1663
CVH	0.1611	0.1698	0.1754	0.1710	0.1667
	FERET				
c	32	48	64	80	96
HDNN	**0.5796**	**0.5508**	**0.5220**	**0.3868**	**0.2516**
DMH	0.5661	0.5373	0.5085	0.3733	0.2381
MDBE	0.1611	0.1323	0.1035	0.1033	0.1031
CCA-GHS-DI	0.1341	0.1053	0.0765	0.0763	0.0761
CCA-GHS-DD	0.1538	0.1258	0.0977	0.0971	0.0965
SePH	0.0436	0.0433	0.0429	0.0424	0.0419
CMSSH	0.0016	0.0018	0.0020	0.0021	0.0022
CCA-ITQ	0.0207	0.0147	0.0088	0.0078	0.0067
SCM	0.1515	0.0873	0.0232	0.0179	0.0126
CVH	0.1517	0.1158	0.0799	0.0793	0.0787
	FERET				
c	128	160	192	224	256
HDNN	**0.2567**	**0.2598**	**0.2630**	**0.2661**	**0.2693**
DMH	0.2429	0.2459	0.2489	0.2518	0.2548
MDBE	0.1052	0.1065	0.1078	0.1091	0.1104
CCA-GHS-DI	0.0777	0.0786	0.0796	0.0805	0.0815
CCA-GHS-DD	0.0985	0.0997	0.1009	0.1021	0.1033
SePH	0.0427	0.0432	0.0437	0.0443	0.0448
CMSSH	0.0022	0.0022	0.0023	0.0023	0.0023
CCA-ITQ	0.0069	0.0070	0.0071	0.0071	0.0072
SCM	0.0128	0.0130	0.0131	0.0133	0.0134
CVH	0.0803	0.0813	0.0822	0.0832	0.0842

Table C.9: F-measure results on FRGC dataset

	FRGC				
c	8	12	16	20	24
HDNN	**0.3541**	**0.5393**	**0.6836**	**0.7397**	**0.7958**
DMH	0.3406	0.5245	0.6703	0.7264	0.7825
MDBE	0.2056	0.2545	0.2653	0.2539	0.2425
CCA-GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
CCA-GHS-DD	0.1843	0.2296	0.2426	0.2362	0.2298
SePH	0.1654	0.1987	0.2159	0.1703	0.1247
CMSSH	0.1648	0.1905	0.2210	0.1917	0.1624
CCA-ITQ	0.1615	0.1872	0.2136	0.1151	0.0166
SCM	0.0817	0.1246	0.1499	0.1581	0.1663
CVH	0.1611	0.1698	0.1754	0.1710	0.1667
	FRGC				
c	32	48	64	80	96
HDNN	**0.5796**	**0.5508**	**0.5220**	**0.3868**	**0.2516**
DMH	0.5661	0.5373	0.5085	0.3733	0.2381
MDBE	0.1611	0.1323	0.1035	0.1033	0.1031
CCA-GHS-DI	0.1341	0.1053	0.0765	0.0763	0.0761
CCA-GHS-DD	0.1538	0.1258	0.0977	0.0971	0.0965
SePH	0.0436	0.0433	0.0429	0.0424	0.0419
CMSSH	0.0016	0.0018	0.0020	0.0021	0.0022
CCA-ITQ	0.0207	0.0147	0.0088	0.0078	0.0067
SCM	0.1515	0.0873	0.0232	0.0179	0.0126
CVH	0.1517	0.1158	0.0799	0.0793	0.0787
	FRGC				
c	128	160	192	224	256
HDNN	**0.2567**	**0.2598**	**0.2630**	**0.2661**	**0.2693**
DMH	0.2429	0.2459	0.2489	0.2518	0.2548
MDBE	0.1052	0.1065	0.1078	0.1091	0.1104
CCA-GHS-DI	0.0777	0.0786	0.0796	0.0805	0.0815
CCA-GHS-DD	0.0985	0.0997	0.1009	0.1021	0.1033
SePH	0.0427	0.0432	0.0437	0.0443	0.0448
CMSSH	0.0022	0.0022	0.0023	0.0023	0.0023
CCA-ITQ	0.0069	0.0070	0.0071	0.0071	0.0072
SCM	0.0128	0.0130	0.0131	0.0133	0.0134
CVH	0.0803	0.0813	0.0822	0.0832	0.0842

Table C.10: F-measure results on YouTube Faces dataset

	YouTube Faces				
c	8	12	16	20	24
HDNN	**0.3541**	**0.5393**	**0.6836**	**0.7397**	**0.7958**
DMH	0.3406	0.5245	0.6703	0.7264	0.7825
MDBE	0.2056	0.2545	0.2653	0.2539	0.2425
CCA-GHS-DI	0.1786	0.2275	0.2383	0.2269	0.2155
CCA-GHS-DD	0.1843	0.2296	0.2426	0.2362	0.2298
SePH	0.1654	0.1987	0.2159	0.1703	0.1247
CMSSH	0.1648	0.1905	0.2210	0.1917	0.1624
CCA-ITQ	0.1615	0.1872	0.2136	0.1151	0.0166
SCM	0.0817	0.1246	0.1499	0.1581	0.1663
CVH	0.1611	0.1698	0.1754	0.1710	0.1667
	YouTube Faces				
c	32	48	64	80	96
HDNN	**0.5796**	**0.5508**	**0.5220**	**0.3868**	**0.2516**
DMH	0.5661	0.5373	0.5085	0.3733	0.2381
MDBE	0.1611	0.1323	0.1035	0.1033	0.1031
CCA-GHS-DI	0.1341	0.1053	0.0765	0.0763	0.0761
CCA-GHS-DD	0.1538	0.1258	0.0977	0.0971	0.0965
SePH	0.0436	0.0433	0.0429	0.0424	0.0419
CMSSH	0.0016	0.0018	0.0020	0.0021	0.0022
CCA-ITQ	0.0207	0.0147	0.0088	0.0078	0.0067
SCM	0.1515	0.0873	0.0232	0.0179	0.0126
CVH	0.1517	0.1158	0.0799	0.0793	0.0787
	YouTube Faces				
c	128	160	192	224	256
HDNN	**0.2567**	**0.2598**	**0.2630**	**0.2661**	**0.2693**
DMH	0.2429	0.2459	0.2489	0.2518	0.2548
MDBE	0.1052	0.1065	0.1078	0.1091	0.1104
CCA-GHS-DI	0.0777	0.0786	0.0796	0.0805	0.0815
CCA-GHS-DD	0.0985	0.0997	0.1009	0.1021	0.1033
SePH	0.0427	0.0432	0.0437	0.0443	0.0448
CMSSH	0.0022	0.0022	0.0023	0.0023	0.0023
CCA-ITQ	0.0069	0.0070	0.0071	0.0071	0.0072
SCM	0.0128	0.0130	0.0131	0.0133	0.0134
CVH	0.0803	0.0813	0.0822	0.0832	0.0842

References

[1] J. S. Abel and J. W. Chaffee, "Existence and uniqueness of GPS solutions," *IEEE Transactions on Aerospace and Electronic System*, vol. 27, no. 6, pp. 952–956, Nov. 1991. 39, 41

[2] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 2037–2041, Dec 2006. 60, 77

[3] T. Ahonen, E. Rahtu, V. Ojansivu, and J. Heikkila, "Recognition of blurred faces using local phase quantization," in *Proceedings of International Conference on Pattern Recognition*, 2008, pp. 1–4. 28

[4] Y. A. Y. AI-Najjar and D. C. Soong, "Comparison of image quality assessment: PSNR, HVS, SSIM, UIQI," *International Journal of Scientific & Engineering Research*, vol. 3, no. 8, Aug 2012. 25

[5] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," *Communications of the ACM*, vol. 51, no. 1, pp. 117–122, Jan. 2008. 38

[6] S. Bancroft, "An algebraic solution of the GPS equations," *IEEE Transactions on Aerospace and Electronic System*, vol. 21, pp. 56–59, Jan. 1985. 43

[7] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, 1981. 50

[8] M. B. Blaschko and C. H. Lampert, "Correlational spectral clustering," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8. 57

[9] A. Bovik, T. Huang, and D. Munson, "A generalization of median filtering using linear combinations of order statistics," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 31, no. 6, pp. 1342–1350, Dec 1983. 18

[10] A. Bovik, T. Huang, and J. Munson, D.C., "A generalization of median filtering using linear combinations of order statistics," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 31, no. 6, pp. 1342–1350, Dec 1983. 18

[11] M. M. Bronstein, A. M. Bronstein, F. Michel, and N. Paragios, "Data fusion through cross-modality metric learning using similarity-sensitive hashing," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2010, pp. 3594–3601. 63, 72

[12] S. Chan, R. Khoshabeh, K. Gibson, P. Gill, and T. Nguyen, "An augmented lagrangian method for total variation video restoration," *Image Processing, IEEE Transactions on*, vol. 20, no. 11, pp. 3097–3111, Nov 2011. 11, 16

[13] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *ACM Symposium on Theory of Computing*, 2002, pp. 380–388. 38

[14] 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 *Proc. of ACM Conf. on Image and Video Retrieval (CIVR'09)*, Santorini, Greece., 2009. 70, 71

[15] T. Dean, M. A. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, and J. Yagnik, "Fast, accurate detection of 100,000 object classes on a single machine," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1814–1821. 36

[16] M. Delbracio and G. Sapiro, "Removing camera shake via weighted fourier burst accumulation," *Image Processing, IEEE Transactions on*, vol. 24, no. 11, pp. 3293–3307, Nov 2015. 8

[17] C. Ding, J. Choi, D. Tao, and L. S. Davis, "Multi-directional multi-level dual-cross patterns for robust face recognition," *IEEE Trans. Pattern Anal. Mach. Intell., doi: 10.1109/TPAMI.2015.2462338*, 2015. 11

[18] C. Ding and D. Tao, "A comprehensive survey on pose-invariant face recognition," *ACM Trans. Intell. Syst. Technol.*, 2015. 34

[19] ——, "Face recognition via multimodal deep face representation," *CoRR*, vol. abs/1509.00244, 2015. 79

[20] ——, "Trunk-branch ensemble convolutional neural networks for video-based face recognition," *CoRR*, vol. abs/1607.05427, 2016. 79

[21] ——, "Pose-invariant face recognition with homography-based normalization," *Pattern Recognition*, vol. 66, pp. 144–152, 2017. 8

[22] G. Ding, Y. Guo, and J. Zhou, "Collective matrix factorization hashing for multimodal data," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 2083–2090. 63, 71

[23] W. Dong, D. Zhang, G. Shi, and X. Wu, "Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization," *Image Processing, IEEE Transactions on*, vol. 20, no. 7, pp. 1838–1857, Jul 2011. 9

[24] W. Dong, L. Zhang, G. Shi, and X. Li, "Nonlocally centralized sparse representation for image restoration," *Image Processing, IEEE Transactions on*, vol. 22, no. 4, pp. 1620–1630, Apr 2013. 9

[25] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, Nov. 2008. 59

[26] R.-E. Fan, P.-H. Chen, and C.-J. Lin, "Working set selection using second order information for training support vector machines," *Journal of Machine Learning Research*, vol. 6, pp. 1889–1918, 2005. 21

[27] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman, "Removing camera shake from a single photograph," in *SIGGRAPH*, 2006, pp. 787–794. 8

[28] M. A. T. Figueiredo, R. D. Nowak, and S. J. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," *IEEE Journal of Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 586–597, Jan. 2007. 44

[29] D. P. Foster, S. M. Kakade, and T. Zhang, "Multi-view dimensionality reduction via canonical correlation analysis," Tech. Rep., 2008. 57

[30] H. Fu, C. Wang, D. Tao, and M. J. Black, "Occlusion boundary detection via deep exploration of context," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, 2016, pp. 241–250. 79

[31] F. Gao, D. Tao, X. Gao, and X. Li, "Learning to rank for blind image quality assessment," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 26, no. 10, pp. 2275–2290, 2015. 8

[32] A. Georghiades, P. Belhumeur, and D. Kriegman, "From few to many: illumination cone models for face recognition under variable lighting and pose," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 23, no. 6, pp. 643–660, Jun 2001. 24

[33] M. Gonen and E. Alpaydm, "Multiple kernel learning algorithms," *Journal of Machine Learning Research*, vol. 12, pp. 2211–2268, 2011. 19

[34] C. Gong, T. Liu, D. Tao, K. Fu, E. Tu, and J. Yang, "Deformed graph laplacian for semisupervised learning," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 26, no. 10, pp. 2261–2274, 2015. 8

[35] C. Gong, D. Tao, K. Fu, and J. Yang, "Fick's law assisted propagation for semisupervised learning," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 26, no. 9, pp. 2148–2162, 2015. 8

[36] C. Gong, D. Tao, W. Liu, S. J. Maybank, M. Fang, K. Fu, and J. Yang, "Saliency propagation from simple to difficult," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, 2015, pp. 2531–2539. 8

[37] J. A. Greenwood, J. M. Landwehr, N. C. Matalas, and J. R. Wallis, "Probability weighted moments: Definition and relation to parameters of several distributions expressable in inverse form," *Water Resources Research*, vol. 15, no. 5, pp. 1049–1054, 1979. 18

[38] Y. Guo, D. Tao, J. Yu, H. Xiong, Y. Li, and D. Tao, "Deep neural networks with relativity learning for facial expression recognition," in *2016 IEEE International Conference on Multimedia & Expo Workshops, ICME Workshops 2016, Seattle, WA, USA, July 11-15, 2016*, 2016, pp. 1–6. 79

[39] M. R. Hestenes and E. Stiefel, "Methods of conjugate gradients for solving linear systems," *Journal of Research of National Bureau of Standards*, vol. 49, no. 6, pp. 409–436, 1952. 13

[40] M. R. Hestenes, "Multiplier and gradient methods," *Journal of Optimization Theory and Applications*, vol. 4, pp. 303–320, 1969. 16

[41] G. Hinton and S. Roweis, "Stochastic neighbor embedding," in *Advances in Neural Information Processing Systems*, 2002, pp. 833–840. 49

[42] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins, *Global Positioning System: Theory and Practice*. Springer-Verlag, 1997. 37

[43] H. Hotelling, "Analysis of a complex of statistical variables into principal components," *Journal of Educational Psychology*, vol. 24, no. 6, pp. 417–441, Sep 1933. 2

[44] ——, "Relations between two sets of variables," *Biometrika*, vol. 28, pp. 321–377, Dec. 1936. 57

[45] M. J. Huiskes and M. S. Lew, "The mir flickr retrieval evaluation," in *MIR '08: Proceedings of the 2008 ACM International Conference on Multimedia Information Retrieval*. New York, NY, USA: ACM, 2008. 71

[46] H. Jae-Pil, L. Youngwoon, H. Junfeng, C. Shih-Fu, and Y. Sung-Eui, "Spherical hashing," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 2957–2964. 50, 51

[47] H. Jegou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 117–128, Mar. 2011. 51

[48] X. Jianxiong, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 3485–3492. 51

[49] A. Joly and O. Buisson, "Random maximum margin hashing," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 873–880. 38

[50] W. Jun, S. Kumar, and C. Shih-Fu, "Semi-supervised hashing for large-scale search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 12, pp. 2393–2406, Sep. 2012. 37, 38

[51] G. Kang, J. Li, and D. Tao, "Shakeout: A new regularized deep neural network training scheme," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, 2016, pp. 1751–1757. 79

[52] Y. Kang, S. Kim, and S. Choi, "Deep learning to hash with multiple representations," in *2012 IEEE 12th International Conference on Data Mining*, Dec 2012, pp. 930–935. 79

[53] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy, "Improvements to platt's smo algorithm for svm classifier design," *Neural Computation*, vol. 13, no. 3, pp. 637–649, Mar 2001. 21

[54] W. Kong and W.-J. Li, "Isotropic hashing," in *Advances in Neural Information Processing Systems*, 2012, pp. 1646–1654. 38, 49, 51

[55] D. Krishnan, T. Tay, and R. Fergus, "Blind deconvolution using a normalized sparsity measure," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 233–240. xiii, xiv, 24, 26, 27, 28, 29, 30, 31, 32, 33, 34

[56] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009. 47

[57] B. Kulis and T. Darrell, "Learning to hash with binary reconstructive embeddings," in *Advances in Neural Information Processing Systems*, 2009, pp. 1042–1050. 38

[58] B. Kulis and K. Grauman, "Kernelized locality-sensitive hashing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 6, pp. 1092–1104, Nov. 2012. 38

[59] S. Kumar and R. Udupa, "Learning hash functions for cross-view similarity search," in *IJCAI-11*, July 2011. 63, 72

[60] N. E. Lasmar, Y. Stitou, and Y. Berthoumieu, "Multiscale skewed heavy tailed model for texture analysis," in *2009 16th IEEE International Conference on Image Processing (ICIP)*, Nov 2009, pp. 2281–2284. 18

[61] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, "Understanding blind deconvolution algorithms," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 12, pp. 2354–2357, 2011. 8

[62] J. Li, X. Lin, X. Rui, Y. Rui, and D. Tao, "A distributed approach toward discriminative distance metric learning," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 26, no. 9, pp. 2111–2122, 2015. 8

[63] J. Li, X. Mei, D. V. Prokhorov, and D. Tao, "Deep neural network for structural prediction and lane detection in traffic scene," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 28, no. 3, pp. 690–703, 2017. 79

[64] J. Li, C. Xu, W. Yang, C. Sun, and D. Tao, "Discriminative multi-view interactive image re-ranking," *IEEE Trans. Image Processing*, vol. 26, no. 7, pp. 3113–3127, 2017. 8

[65] P. Li, A. Shrivastava, J. Moore, and A. C. Konig, "Hashing algorithms for large-scale learning," in *Advances in Neural Information Processing System*, 2011. 36

[66] Y. Liao and X. Lin, "Blind image restoration with eigen-face subspace," *Image Processing, IEEE Transactions on*, vol. 14, no. 11, pp. 1766–1772, 2005. 8

[67] G. Lin, C. Shen, Q. Shi, A. van den Hengel, and D. Suter, "Fast supervised hashing with decision trees for high-dimensional data," in *The IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1971–1978. 58

[68] Z. Lin, G. Ding, M. Hu, and J. Wang, "Semantics-preserving hashing for cross-view retrieval," in *2015 IEEE Conference on Computer Vision and Pattern Recognition*, June 2015, pp. 3864–3872. 63, 71, 72

[69] H. Liu, X. Sun, L. Fang, and F. Wu, "Deblurring saturated night image with function-form kernel," *Image Processing, IEEE Transactions on*, vol. 24, no. 11, pp. 4637–4650, Nov 2015. 8

[70] L. Liu, M. Yu, and L. Shao, "Multiview alignment hashing for efficient image search," *IEEE Transactions on Image Processing*, vol. 24, no. 3, pp. 956–966, March 2015. 39

[71] M. Liu, Y. Luo, D. Tao, C. Xu, and Y. Wen, "Low-rank multi-view learning in matrix completion for multi-label image classification," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, 2015, pp. 2778–2784. 8

[72] W. Liu, C. Mu, S. Kumar, and S.-F. Chang, "Discrete graph hashing," in *Advances in Neural Information Processing Systems*, 2014. 38, 59, 77

[73] W. Liu, J. Wang, and S.-f. Chang, "Hashing with graphs," in *International Conference on Machine Learning*, 2011. 38

[74] W. Liu, J. Wang, Y. Mu, S. Kumar, and S.-F. Chang, "Compact hyperplane hashing with bilinear functions," in *International Conference on Machine Learning*, 2012. 36

[75] X. Liu, D. Tao, M. Song, L. Zhang, J. Bu, and C. Chen, "Learning to track multiple targets," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 26, no. 5, pp. 1060–1073, 2015. 79

[76] H. G. Longbotham and A. C. Bovik, "Theory of order statistic filters and their relationship to linear fir filters," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 2, pp. 275–287, Feb 1989. 18

[77] D. G. Lowe, "Object recognition from local scale-invariant features," in *IEEE International Conference on Computer Vision*, 1999, pp. 1150–1157. 51, 64

[78] K. Ma, H. Fu, T. Liu, Z. Wang, and D. Tao, "Local blur mapping: Exploiting high-level semantics by deep neural networks," *CoRR*, vol. abs/1612.01227, 2016. 79

[79] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration," *Image Processing, IEEE Transactions on*, vol. 17, no. 1, pp. 53–69, Jan 2008. 9

[80] A. Mittal, A. Moorthy, and A. Bovik, "No-reference image quality assessment in the spatial domain," *Image Processing, IEEE Transactions on*, vol. 21, no. 12, pp. 4695–4708, Dec 2012. 16

[81] ——, "Making image quality assessment robust," in *IEEE Conference Record of the Forty Sixth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, 2012, pp. 1718–1722. 11, 16

[82] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal Deep Learning," in *International Conference on Machine Learning*, 2011, pp. 689–696. 79

[83] M. Nishiyama, A. Hadid, H. Takeshima, J. Shotton, T. Kozakaya, and O. Yamaguchi, "Facial deblur inference using subspace analysis for recognition of blurred faces," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 4, pp. 838–845, 2011. xiii, 8, 24, 26, 27, 28, 29, 30, 31, 32

[84] M. Norouzi and D. J. Fleet, "Cartesian k-means," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3017–3024. 49, 51

[85] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, May 2001. 51, 64, 71

[86] J. Pan, Z. Hu, Z. Su, and M.-H. Yang, "Deblurring face images with exemplars," in *European Conference on Computer Vision*, 2014, pp. 47–62. xiii, xiv, 10, 24, 26, 27, 28, 29, 30, 31, 32, 33, 34

[87] P. Phillips, H. Moon, S. Rizvi, and P. Rauss, "The feret evaluation methodology for face-recognition algorithms," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 10, pp. 1090–1104, Oct 2000. 24

[88] J. C. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods - Support Vector Learning*. MIT Press, Jan 1998. 21

[89] M. Qiao, L. Liu, J. Yu, C. Xu, and D. Tao, "Diversified dictionaries for multi-instance learning," *Pattern Recognition*, vol. 64, pp. 407–416, 2017. 8

[90] Q. Shan, J. Jia, and A. Agarwala, "High-quality motion deblurring from a single image," in *SIGGRAPH ASIA*, 2008, pp. 73:1–73:10. 8

[91] K. Sharifi and A. Leon-Garcia, "Estimation of shape parameter for generalized gaussian distributions in subband decompositions of video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 1, pp. 52–56, Feb 1995. 18

[92] F. Shen, C. Shen, Q. Shi, A. van den Hengel, Z. Tang, and H. T. Shen, "Hashing on nonlinear manifolds," *IEEE Transactions on Image Processing*, vol. 24, no. 6, pp. 1839–1851, 2015. 39

[93] F. Shen, C. Shen, W. Liu, and H. Tao Shen, "Supervised discrete hashing," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 37–45. 57

[94] F. Shen, C. Shen, Q. Shi, A. v. d. Hengel, and Z. Tang, "Inductive hashing on manifolds," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1562–1569. 38, 39, 49, 51

[95] Q. Shi, J. Petterson, G. Dror, J. Langford, A. Smola, and S. Vishwanathan, "Hash kernels for structured data," *Journal of Machine Learning Research*, vol. 10, pp. 2615–2637, Nov. 2009. 36

[96] T. Sim, S. Baker, and M. Bsat, "The cmu pose, illumination, and expression database," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25, no. 12, pp. 1615–1618, Dec 2003. 24

[97] J. Song, Y. Yang, Y. Yang, Z. Huang, and H. T. Shen, "Inter-media hashing for large-scale retrieval from heterogeneous data sources," in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '13. New York, NY, USA: ACM, 2013, pp. 785–796. 63

[98] N. Srivastava, "Learning representations for multimodal data with deep belief nets," in *International Conference on Machine Learning Workshop*, 2012. 79

[99] N. Srivastava and R. R. Salakhutdinov, "Multimodal learning with deep boltzmann machines," in *Advances in Neural Information Processing Systems 25*, 2012, pp. 2222–2230. 79

[100] C. Strecha, A. M. Bronstein, M. M. Bronstein, and P. Fua, "Ldahash: Improved matching with smaller descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 1, pp. 66–78, May 2012. 38

[101] A. Talwalkar, S. Kumar, and H. Rowley, "Large-scale manifold learning," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 2008, pp. 1–8. 39

[102] J. Tang, Z. Li, M. Wang, and R. Zhao, "Neighborhood discriminant hashing for large-scale image retrieval," *IEEE Transactions on Image Processing*, vol. 24, no. 9, pp. 2827–2840, Sept 2015. 38

[103] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large data set for nonparametric object and scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 11, pp. 1958–1970, May 2008. 36, 47

[104] Y. Tsin and T. Kannade, "A correlation-based approach to robust point set registration," in *Proceedings of European Conference on Computer Vision*, 2004, pp. 558–569. 22

[105] G. Tzimiropoulos, S. Zafeiriou, and M. Pantic, "Sparse representations of image gradient orientations for visual recognition and tracking," in *Proceedings of IEEE Computer Vision and Pattern Recognition, Workshop on CVPR for Human Behaviour Analysis*, 2011, pp. 26–33. 9

[106] L. van der Maaten and G. Hinton, "Visualizing data using t-sne." 49

[107] S. Vishwanathan, Z. Sun, N. Theera-Ampornpunt, and M. Varma, "Multiple kernel learning and the SMO algorithm," in *Advances in Neural Informatin Processing Systems*, Dec 2010, pp. 2361–2369. 19

[108] D. Wang, X. Gao, X. Wang, L. He, and B. Yuan, "Multimodal discriminative binary embedding for large-scale cross-modal retrieval," *IEEE Transactions on Image Processing*, vol. 25, no. 10, pp. 4540–4554, Oct 2016. 63, 64, 72

[109] D. Wang, P. Cui, M. Ou, and W. Zhu, "Deep multimodal hashing with orthogonal regularization," in *Proceedings of the 24th International Conference on Artificial Intelligence*. 79

[110] ——, "Deep multimodal hashing with orthogonal regularization," in *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)*, 2015, pp. 2291– 2297. 66

[111] R. Wang and D. Tao, "Recent progress in image deblurring," *arXiv:1409.6838v1*, 2014. 9

[112] Z. Wang and A. Bovik, "A universal image quality index," *Signal Processing Letters, IEEE*, vol. 9, no. 3, pp. 81–84, Mar 2002. 24

[113] L. Wei, W. Jun, J. Rongrong, J. Yu-Gang, and C. Shih-Fu, "Supervised hashing with kernels," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 2074–2081. 36, 38, 58

[114] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg, "Feature hashing for large scale multitask learning," in *International Conference on Machine Learning*, 2009, pp. 1113–1120. 36

[115] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Advances in Neural Information Processing Systems*, 2008, pp. 1753–1760. 5, 37, 38, 50, 51, 63, 65

[116] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *CVPR 2011*, June 2011, pp. 529–534. 59, 77

[117] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 2, pp. 210–227, Feb 2009. 9

[118] L. Xiao, J. Gregson, F. Heide, and W. Heidrich, "Stochastic blind motion deblurring," *Image Processing, IEEE Transactions on*, vol. 24, no. 10, pp. 3071–3085, Oct 2015. 8

[119] B. Xu, J. Bu, Y. Lin, C. Chen, X. He, and D. Cai, "Harmonious hashing," in *International Joint Conference on Artificial Intelligence*, 2013, pp. 1820–1826. 49, 51

[120] Z. Xu, S. Huang, Y. Zhang, and D. Tao, "Augmenting strong supervision using web data for fine-grained categorization," in *2015 IEEE International Conference on Computer Vision, ICCV*, 2015, pp. 2524–2532. 8

[121] Z. Xu, D. Tao, S. Huang, and Y. Zhang, "Friend or foe: Fine-grained categorization with weak supervision," *IEEE Trans. Image Processing*, vol. 26, no. 1, pp. 135–146, 2017. 8

[122] F. Xue and T. Blu, "A novel sure-based criterion for parametric psf estimation," *Image Processing, IEEE Transactions on*, vol. 24, no. 2, pp. 595–607, Feb 2015. 8

[123] A. Yang, Z. Zhou, A. Balasubramanian, S. Sastry, and Y. Ma, "Fast l_1 minimization algorithms for robust face recognition," *Image Processing, IEEE Transactions on*, vol. 22, no. 8, pp. 3234–3246, Aug 2013. 9

[124] B. Yu, M. Fang, and D. Tao, "Per-round knapsack-constrained linear submodular bandits," *Neural Computation*, vol. 28, no. 12, pp. 2757–2789, 2016. 79

[125] G. Yunchao and S. Lazebnik, "Iterative quantization: A procrustean approach to learning binary codes," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 817–824. 37, 38, 49, 50, 51, 58, 64, 65

[126] D. Zhang and W.-J. Li, "Large-scale supervised multimodal hashing with semantic correlation maximization," in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, ser. AAAI'14. AAAI Press, 2014, pp. 2177–2183. 63, 71, 72

[127] H. Zhang, J. Yang, Y. Zhang, and N. M. N. adn Thomas S. Huang, "Close the loop: Joint blind image restoration and recognition with sparse representation prior," in *Proceedings of International Conference on Computer Vision*, 2011, pp. 770–777. 8, 9, 24, 27, 28, 31, 32, 35

[128] L. Zhang, H. Lu, D. Du, and L. Liu, "Sparse hashing tracking," *IEEE Transactions on Image Processing*, vol. 25, no. 2, pp. 840–849, Feb 2016. 36

[129] L. Zhang, Y. Zhang, R. Hong, and Q. Tian, "Full-space local topology extraction for cross-modal retrieval," *IEEE Transactions on Image Processing*, vol. 24, no. 7, pp. 2212–2224, July 2015. 39

[130] R. Zhang, L. Lin, R. Zhang, W. Zuo, and L. Zhang, "Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 4766–4779, Dec 2015. 39

[131] X. Zhao, X. Chai, Z. Niu, H. C. Keng, and S. Shan, "Sparsely encoded local descriptor for face recognition," in *International Conference on Automatic Face and Gesture Recognition*, 2011, pp. 149–154. 9

[132] Y. Zhen and D.-Y. Yeung, "A probabilistic model for multimodal hash function learning." in *KDD*. ACM, 2012, pp. 940–948. 63

[133] X. Zhu, Z. Huang, H. T. Shen, and X. Zhao, "Linear cross-modal hashing for efficient multimedia search," in *Proceedings of the 21st ACM International Conference on Multimedia*, ser. MM '13. New York, NY, USA: ACM, 2013, pp. 143–152. [Online]. Available: http://doi.acm.org/10.1145/2502081.2502107 63