Feature Reduction and Representation Learning for Visual Applications

Mengyang Yu

PhD

2016

Feature Reduction and Representation Learning for Visual Applications

Mengyang Yu

A thesis submitted in partial fulfilment of
the requirements of the
University of Northumbria at Newcastle
for the degree of
Doctor of Philosophy

Research undertaken in the
Faculty of Engineering and Environment

July 2016

Abstract

Computation on large-scale data spaces has been involved in many active problems in computer vision and pattern recognition. However, in realistic applications, most existing algorithms are heavily restricted by the large number of features, and tend to be inefficient and even infeasible. In this thesis, the solution to this problem is addressed in the following ways: (1) projecting features onto a lower-dimensional subspace; (2) embedding features into a Hamming space.

Firstly, a novel subspace learning algorithm called Local Feature Discriminant Projection (LFDP) is proposed for discriminant analysis of local features. LFDP is able to efficiently seek a subspace to improve the discriminability of local features for classification. Extensive experimental validation on three benchmark datasets demonstrates that the proposed LFDP outperforms other dimensionality reduction methods and achieves state-of-the-art performance for image classification. Secondly, for action recognition, a novel binary local representation for RGB-D video data fusion is presented. In this approach, a general local descriptor called Local Flux Feature (LFF) is obtained for both RGB and depth data by computing the local fluxes of the gradient fields of video data. Then the LFFs from RGB and depth channels are fused into a Hamming space via the Structure Preserving Projection (SP-P), which preserves not only the pairwise feature structure, but also a higher level connection between samples and classes. Comprehensive experimental results show the superiority of both LFF and SPP. Thirdly, in respect of unsupervised learning, SPP is extended to the Binary Set Embedding (BSE) for cross-modal retrieval. BSE outputs meaningful hash codes for local features from the image domain and word vectors from text domain. Extensive evaluation on two widely-used image-text datasets demonstrates the superior performance of BSE compared with state-of-the-art cross-modal hashing methods. Finally, a generalized multiview spectral embedding algorithm called Kernelized Multiview Projection (KMP) is proposed to fuse the multimedia data from multiple sources. Different features/views in the reproducing kernel Hilbert spaces are linearly fused together and then projected onto a low-dimensional subspace by KMP, whose performance is thoroughly evaluated on both image and video datasets compared with other multiview embedding methods.

To my parents and my wife

Declaration

I declare that the work contained in this thesis has not been submitted for any other award and that it is all my own work. I also confirm that this work fully acknowledges opinions, ideas and contributions from the work of others.

Any ethical clearance for the research presented in this thesis has been approved. Approval has been sought and granted by the Faculty Ethics Committee on 16/02/2016.

I declare that the Word Count of this Thesis is 39,472 words.

I declare that parts of the following papers have been included in this thesis.

1. **Mengyang Yu**, Ling Shao, Xiantong Zhen, and Xiaofei He. "Local Feature Discriminant Projection." Accepted by IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), vol. 38, no. 9, pp. 1908-1914, Sep. 2016. [Chapter 2]

2. **Mengyang Yu**, Li Liu and Ling Shao, "Structure-Preserving Binary Representations for RGB-D Action Recognition." IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), vol. 38, no. 8, pp. 1651-1664, Aug. 2016. [Chapter 3]

3. **Mengyang Yu**, Li Liu, and Ling Shao. "Binary Set Embedding for Cross-modal Retrieval." Accepted by IEEE Transactions on Neural Networks and Learning Systems (TNNLS), 2016, doi: 10.1109/TNNLS.2016.2609463. [Chapter 4]

4. Ling Shao, Li Liu, and **Mengyang Yu**. "Kernelized Multiview Projection for Robust Action Recognition." International Journal of Computer Vision (IJCV), vol. 118, no. 2, pp. 115-129, Jun. 2016. [Chapter 5]

5. **Mengyang Yu**, Li Liu, and Ling Shao. "Kernelized Multiview Projection." arXiv preprint arXiv:1508.00430. [Chapter 5]

6. Li Liu, **Mengyang Yu**, and Ling Shao. "Latent Structure Preserving Hashing." Accepted by International Journal of Computer Vision (IJCV), doi: 10.1007/s11263-016-0931-4, 2016.

7. Li Liu, **Mengyang Yu**, and Ling Shao. "Multiview Alignment Hashing for Image Search." IEEE Transactions on Image Processing (TIP), vol. 24, no. 3, pp. 956-966, Mar. 2015.

8. Li Liu, **Mengyang Yu**, and Ling Shao. "Unsupervised Local Feature Hashing for Image Similarity Search." IEEE Transactions on Cybernetics, vol. 46, no. 11, pp. 2548-2558, Nov. 2016.

9. Xiantong Zhen, Zhijie Wang, **Mengyang Yu**, and Shuo Li. "Supervised Descriptor Learning for Multi-Output Regression." IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

10. Li Liu, **Mengyang Yu**, and Ling Shao. "Projection Bank: From High-dimensional Data to Medium-length Binary Codes." IEEE International Conference on Computer Vision (ICCV), 2015.

11. Ziyun Cai, Li Liu, **Mengyang Yu**, and Ling Shao. "Latent Structure Preserving Hashing." British Machine Vision Conference (BMVC), 2015.

12. Li Liu, **Mengyang Yu**, and Ling Shao. "Local Feature Binary Coding for Approximate Nearest Neighbor Search." British Machine Vision Conference (BMVC), 2015.

13. Jie Qin, Li Liu, **Mengyang Yu**, Yunhong Wang and Ling Shao. "Fast Action Retrieval from Videos via Feature Disaggregation." British Machine Vision Conference (BMVC), 2015.

14. Fan Zhu, Ling Shao, and **Mengyang Yu**. "Cross-Modality Submodular Dictionary Learning for Information Retrieval." ACM International Conference on Information and Knowledge Management (CIKM), 2014.

Mengyang Yu
July 2016

Acknowledgements

First of all, I would like to thank my supervisor Prof. Ling Shao, who gave me the opportunity to complete my PhD in Northumbria University. Under his supervision, he showed me the path to the research area of computer vision and machine learning. He has always been providing me insightful advises and valuable ideas, and shaping me to be a qualified researcher.

I would like to thank my research fellows: Dr. Xiantong Zhen, Dr. Simon Jones, Dr. Ruomei Yan, Dr. Di Wu, Dr. Li Liu, Dr. Fan Zhu, Bo Dong, Peng Peng, Redzuan Bin Abdul Manap, Yawen Huang, Feng Zheng, Yang Long, Ziyun Cai, Yi Zhou, Bingzhang Hu, Yuming Shen, Daniel Organisciak and Dr. Lining Zhang for their helpful discussion. Xiantong helped me a lot with my study and life when I just joined group. It is very lucky to work with Li because of his wonderful brainstorm. Thank Fan for his excellent collaboration.

I would like to thank Dr. Hubert Shum, Dr. Richard Jiang and Dr. Jungong Han in our department for their kind support and suggestions.

I would like to thank Philip Kinghorn and Benjamin Fielding in our lab for their help and kindness.

I would also like to thank Prof. Xuelong Li and Prof. Xiaofei He for their kind academic guidance during my PhD study.

I specially thank the external examiner Prof. Tim Cootes and the internal examiner Dr. Fouad Khelifi for their invaluable suggestions and comments for improving the thesis. I would also like to thank the independent chair Dr. Paul Vickers for his work during the oral examination.

Finally, I would like to thank my parents for their unconditional support in the past 25 years and my wife Chuyue for her constant encouragement.

Contents

List of Figures

List of Tables

Nomenclature

Acronyms / Abbreviations

BSE	Binary Set Embedding
CCA	Canonical Correlation Analysis
DSDC	Differential Scatter Discriminant Criterion
DSE	Distributed Spectral Embedding
DTW	Dynamic Time Warping
FV	Fisher Vector
GMM	Gaussian Mixture Model
HOG	Histogram of Oriented Gradient
I2C Distance	Image-to-Class Distance
IFK	Improved Fisher Kernel
INBF	Incremental Naive Bayes Filter
KMP	Kernelized Multiview Projection
LDA	Linear Discriminant Analysis
LDE	Linear Discriminant Embedding
LDP	Linear Discriminant Projection
LFDP	Local Feature Discriminant Projection
LFF	Local Flux Feature

LPP	Locality Preserving Projections
LSH	Locality-Sensitive Hashing
MAP	Mean Average Precision
MKL	Multiple Kernel Learning
MSE	Multiview Spectral Embedding
NBNN	Naive Bayes Nearest Neighbor Classifier
NN-search	Nearest Neighbor Search
PCA	Principle Component Analysis
RKHS	Reproducing Kernel Hilbert Space
SIFT	Scale-Invariant Feature Transform
SPP	Structure Preserving Projection
SVM	Support Vector Machine

Chapter 1

Introduction and Literature Review

In the past decade, we have witnessed the explosion and the messiness of data across numerous fields of computer vision and machine learning. With the increasing amount of multimedia data and the use of advanced learning techniques such as deep learning, the performance of algorithms have been largely improved for various applications including image classification, action recognition, information retrieval, etc. However, in most situations, these data (at least million scale) usually have thousands or even hundreds of thousands of dimensions, which severely restricted the computational efficiency in realistic visual tasks and suffered from the curse of dimensionality. To address this problem, many subspace/manifold learning methods [7, 24, 41, 55, 59, 127, 152, 155, 181, 183] have been proposed to map high-dimensional data onto a lower-dimensional subspace wherein the embedded features have sufficiently discriminative ability. The high-dimensional data are simplified by the learned low-dimensional basis and some noise can be cleaned through embeddings as well. Subspace learning techniques are also able to learn an intrinsic low-dimensional manifold structure for high-dimensional data. For the instance of images taking from one face but with different viewpoints, it is obvious that powerful features extracted from these images should distribute on an one-dimensional manifold.

In addition to dimensionality reduction, another way to speedup the algorithms for visual applications is to reduce the data domain. Data are usually represented by the element of the real number field $\mathbb{R}$. If this domain is mapped onto a smaller field such as the smallest field: binary field $\{0,1\}$, the computation efficiency will be tremendously improved. Recent hashing techniques [43, 63, 76, 100, 110, 122, 148, 169] have attracted much attention in the computer vision community, which transform real-valued data points into binary codes. Researchers are dedicated to find the effective and efficient hash function $h : \mathbb{R}^D \to \{0,1\}^d$, where D and d is the original dimensionality of data and the reduced code length, respectively. Due to the binary representation and the indexing search mechanism, the computational efficiency is much improved and larger scale of data is available for the algorithms of almost every computer vision area. It also allows us to use short 0-1 representations for realistic applications with limited computing resources such as wearable or mobile devices. The research background of dimensionality reduction and hashing-based methods will be introduced respectively in the following sections.

1.1 Dimensionality Reduction

Principal Component Analysis (PCA) as a popular dimensionality reduction has been applied to many fields of data applications. PCA is able to find the direction which have the largest variations of a set of features and the orthogonal basis called principal components that minimizes the reconstruction error for the original data. Mathematically, given N data points $\mathbf{x}_1, \cdots, \mathbf{x}_N \in \mathbb{R}^D$, PCA aims to find an orthonormal basis $W = [\mathbf{w}_1, \cdots, \mathbf{w}_d] \in \mathbb{R}^{D \times d}$ of a d-dimensional subspace onto which the new projected representations of data, i.e., $W^T\mathbf{x}_1, \cdots, W^T\mathbf{x}_N$, gain the maximal variance. Let us denote $X = [\mathbf{x}_1, \cdots, \mathbf{x}_N]$ in matrix form. Then PCA needs to solve the following optimization problem:

$$\underset{W^TW=I}{\arg\max} \frac{1}{N} \sum_{i=1}^{N} \|W^T(\mathbf{x}_i - \mu)\|^2, \tag{1.1}$$

where $\mu = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i$. If we define the covariance matrix $S = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T$, the above optimization problem is transformed to:

$$\begin{aligned}\underset{W^TW=I}{\arg\max} \frac{1}{N} \sum_{i=1}^{N} \|W^T(\mathbf{x}_i - \mu)\|^2 &= \underset{W^TW=I}{\arg\max} \frac{1}{N} \sum_{i=1}^{N} (\mathbf{x}_i - \mu)^T W W^T (\mathbf{x}_i - \mu) \\ &= \underset{W^TW=I}{\arg\max} \frac{1}{N} \sum_{i=1}^{N} \mathrm{tr}(W^T(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T W) \\ &= \underset{W^TW=I}{\arg\max}\, \mathrm{tr}(W^T S W),\end{aligned} \tag{1.2}$$

which can be easily solved by the eigen-decomposition procedure. The column vectors of the optimal solution are the first d eigenvectors with the largest eigenvalues of S.

Another property of the projection learned by PCA is that the projected data can minimize the squared reconstruction error for the original data. Suppose $\mathbf{x}_1, \cdots, \mathbf{x}_N$ are zero-mean data, i.e., $\mu = \mathbf{0}$, then the orthonormal minimizer for the squared reconstruction error will be

$$\begin{aligned}
\arg\min_{W^TW=I} \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{x}_i - WW^T\mathbf{x}_i\|^2 &= \arg\min_{W^TW=I} \frac{1}{N}\sum_{i=1}^{N} (\mathbf{x}_i - WW^T\mathbf{x}_i)^T(\mathbf{x}_i - WW^T\mathbf{x}_i) \\
&= \arg\min_{W^TW=I} \frac{1}{N}\sum_{i=1}^{N} (\mathbf{x}_i^T - \mathbf{x}_i^T WW^T)(\mathbf{x}_i - WW^T\mathbf{x}_i) \\
&= \arg\min_{W^TW=I} \frac{1}{N}\sum_{i=1}^{N} (\mathbf{x}_i^T\mathbf{x}_i - \mathbf{x}_i^T WW^T\mathbf{x}_i - \mathbf{x}_i^T WW^T\mathbf{x}_i + \mathbf{x}_i^T WW^T WW^T\mathbf{x}_i) \\
&= \arg\min_{W^TW=I} \frac{1}{N}\sum_{i=1}^{N} (\mathbf{x}_i^T\mathbf{x}_i - \mathbf{x}_i^T WW^T\mathbf{x}_i - \mathbf{x}_i^T WW^T\mathbf{x}_i + \mathbf{x}_i^T WW^T\mathbf{x}_i) \\
&= \arg\min_{W^TW=I} \frac{1}{N}\sum_{i=1}^{N} (\mathbf{x}_i^T\mathbf{x}_i - \mathbf{x}_i^T WW^T\mathbf{x}_i) \\
&= \arg\min_{W^TW=I} \frac{1}{N}\sum_{i=1}^{N} (-\mathbf{x}_i^T WW^T\mathbf{x}_i) \\
&= \arg\max_{W^TW=I} \frac{1}{N}\sum_{i=1}^{N} \|W^T\mathbf{x}_i\|^2,
\end{aligned}$$

which is also the solution to PCA.

Like most unsupervised methods, PCA makes the reduced features less discriminative than supervised methods. Linear Discriminant Analysis (LDA), as a conventional supervised method based on the Fisher criterion, can successfully improve the results of the classification problem by using class labels. The projection of LDA is obtained by maximizing the between-class covariance while minimizing the within-class covariance. Specifically, suppose $\mathbf{x}_1, \cdots, \mathbf{x}_N$ are divided into C classes. For the i-th class, there are n_i samples $\mathbf{x}_{i1}, \cdots, \mathbf{x}_{in_i}$ with the mean $\mu_i = \frac{1}{n_i}\sum_{j=1}^{n_i}\mathbf{x}_{ij}$. Then we can define the between-class covariance matrix S_b and the within-class covariance matrix as follows:

$$\begin{aligned}
S_b &= \frac{1}{N}\sum_{i=1}^{C}(\mu_i - \mu)(\mu_i - \mu)^T, \\
S_w &= \frac{1}{N}\sum_{i=1}^{C}\sum_{j=1}^{n_i}(\mathbf{x}_{ij} - \mu_i)(\mathbf{x}_{ij} - \mu_i)^T.
\end{aligned}$$

Similar to the derivation of PCA, LDA aims to find the solution to the following trace ratio problem:

$$\underset{W^T W = I}{\arg\max} \frac{\mathrm{tr}(W^T S_b W)}{\mathrm{tr}(W^T S_w W)} \tag{1.3}$$

which can be solved by the generalized eigen-decomposition problem. The column vector of the optimal solution are the first d eigenvectors with the largest eigenvalues of $S_w^{-1} S_b$ if S_w is invertible. In the special case of $C = 2$, LDA is simplified as Fisher Discriminant Analysis [52].

As well as other linear methods such as Multidimensional scaling (MDS) [28], Factor analysis [52] and Independent Component Analysis (ICA) [61], PCA and LDA both assume that data points can be linearly represented by a potential basis. To overcome this limitation, their kernel extensions Kernel PCA [132] and Kernel Discriminant Analysis (KDA) [105] have been proposed respectively by the use of the kernel trick. In this kernel-based methods, a nonlinear feature map:

$$\begin{aligned} \phi : \mathbb{R}^D &\to \mathscr{H} \\ \mathbf{x} &\mapsto \phi(\mathbf{x}) \end{aligned}$$

has been used to map the original data into a high-dimensional feature space $\mathscr{H}$. This allows us to perform linear dimensionality reduction algorithms in a high-dimensional space wherein the data can be linearly represented. It is noticeable that the optimization procedures of PCA and LDA depend on the computation of covariance matrices which only calculate the inner product of data. Therefore, given the kernel function

$$k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j),$$

we can compute the kernel matrix

$$K = \Big(k(\mathbf{x}_i, \mathbf{x}_j) \Big)_{i,j=1,\cdots,N} = \phi(X)^T \phi(X)$$

without knowing the feature map $\phi(\cdot)$. The inner product of the covariance matrix is replaced by the kernel function of the kernel matrix, which is actually equivalent to the inner product in a higher-dimensional feature space.

Different from classical dimensionality reduction approaches, recent manifold learning algorithms, e.g., Laplacian Eigenmap (LE) [7], Locally Linear Embedding (LLE) [127] and ISOMAP [155], were proposed to learn the nonlinear structure of the data manifold and preserve the manifold structure based on the nearest neighbor (NN) search in the original data space. In fact, these methods can be seen as special cases of Kernel PCA with different kernel matrices. However, all of these algorithms suffer from the out-of-sample problem [8]. Taking the example of LE, an adjacency graph G for $\mathbf{x}_i$ $(i = 1, \cdots, N)$ is firstly constructed based on the neighborhood structure. Two data points are defined as "connected" if one is among the k nearest neighbors of another one or within a sphere with a selected radius centered at another one. Then the each connected pair $(\mathbf{x}_i, \mathbf{x}_j)$ is assigned a weight as $W_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/t)$, where t is the smooth parameter. Let $Y = [\mathbf{y}_1, \cdots, \mathbf{y}_N] \in \mathbb{R}^{d \times N}$ be the low-dimensional embedding representations for X. With the specific norm constraint, the goal of LE is to minimize the following objective function:

$$\sum_{i,j} \|\mathbf{y}_i - \mathbf{y}_j\|^2 W_{ij}. \tag{1.4}$$

The above minimization problem could also be transformed to:

$$\min_{Y} \operatorname{tr}(Y^T L Y), \tag{1.5}$$

where $L = D - W$ is the Laplacian matrix and D is the diagonal matrix with $D_{ii} = \sum_j W_{ij}$. Nevertheless, the solution only contains the low-dimensional representations rather than the mapping function. Therefore, the model needs to be retrained when new incoming data is added into the dataset, which is not applicable for realistic situations.

Locality Preserving Projections (LPP) [55] and Neighborhood Preserving Embedding (NPE) [54] as the linearized versions of LE and LLE, respectively, were developed to solve the out-of-sample problem. LPP is the first linear manifold learning algorithm based on which later complicated subspace learning techniques [20, 22, 23, 25, 149] were proposed for various applications. The objective of LPP is to find an optimal linear solution to the eigenfunctions of the Laplace Beltrami operator on the data manifold while NPE aims to preserve the local information of each data point. Both of them output the projection matrix W such that $Y = W^T X$ is the linear approximation for the corresponding objective functions. In this way, the learned projection matrix can be directly applied to not only training data but also test data points. Recently, a matrix factorization method called Sparse Concept Coding (SCC) [19] was proposed to seek a sparse representation of the image space. Another interesting method called Discriminative Locality Alignment (DLA) [181] was also proposed to deal with the nonlinear structure of data and assign different weights to different samples according to its margin degree for classification.

The above dimensionality reduction techniques mainly focus on learning the manifold structure of global representations, e.g., GIST [114], VLAD [64] and Fisher Vector (FV) [62]. Another way to represent a sample is using a set of local feature descriptors such as Scale-invariant feature transform (SIFT) [103] rather than one holistic representation vector. As general methods, the above-mentioned dimensionality reduction methods can be applied to both global and local features. Compared to holistic representations which are very sensitive to partial occlusions and background variations, local features are considered as invariant and robust features for changing background, viewpoints, occlusions and scale. With the use of local features, we are able to characterize complex scenarios with multiple labels and retrieve similar objects from diverse scenes or backgrounds. However, the biggest drawback is that an image is usually represented by hundreds or even thousands of local features, which severely limits the efficiency of their applications. To alleviate the computational complexity, researchers also developed dimensionality reduction techniques for local features. The first attempt was PCA-SIFT provided by Ke et al. [69]. PCA was applied to project the gradient image vector of a patch to a more compact descriptor, which is significantly shorter than the standard SIFT descriptor but more robust to image deformations. Discriminative local feature reduction has been explored individually in [59] and [24], both of which use the same covariance matrices of pairwise matched and unmatched feature distances to find the linear projection. Recently, Simonyan et al. [141] proposed learning local feature descriptors using convex optimization. These methods were proposed for image matching and need extra ground truth with matched/unmatched pairs of local features for training.

Nevertheless, how to efficiently and effectively learn the discriminative structure of local features for the recognition task is still a crucial and challenging problem. On the perspective of the complexity issue, given N data points, all of the NN search based methods require at least $O(N^2)$ computational complexity due to the computation of pairwise similarity, which significantly restricts their application in large-scale data spaces. For the instance of $N = 1,000,000$ data points, the $O(N^2)$ training time would be at least 8 hours. On another perspective of the curse of dimensionality [52], the large number of local features is a kind of advantage for learning the manifold structure. The dimensionality of local features is usually in the range of $[100,1000]$. In contrast, global representations are usually of thousands or even hundreds of thousands of dimensionality, which makes them very sparse in the high-dimensional space. Consequently, to achieve the same learning effect as N 500-dimensional local features (N is usually at million scale), it at least needs N^{100} $50,000$-

dimensional global features for training, which is extremely unrealistic to collect such a enormous number of global features. Therefore, dimensionality reduction for local features provides an alternative way to analyze the sample relationship, since training them is much more effective than training high-dimensional global representations.

1.2 Hashing

Learning discriminative embedding has been a critical problem in many fields of information processing and analysis, such as object recognition [138, 187], image/video retrieval [48] and visual detection [47]. Among them, scalable retrieval of similar visual information is attractive, since with the advances of computer technologies and the development of the World Wide Web, a huge amount of digital data has been generated and applied. The most basic but essential scheme for similarity search is the NN-search: given a query image, to find an image that is most similar to it within a large database and assign the same label of the nearest neighbor to this query image. The NN-search is regarded as a linear search scheme, which is not scalable due to the large sample size in datasets of practical applications. Later, to overcome the computational complexity issue, some tree-based search schemes are proposed to partition the data space via various tree structures. Two representative methods are KD-tree and R-tree [40], which are successfully applied to index the data for fast query responses. However, these methods cannot operate with high-dimensional data and can not guarantee to gain a faster search time complexity than the linear scan. In practice, most vision-based tasks suffer from the curse of dimensionality. Thus, some hashing schemes are proposed to effectively embed data from a high-dimensional feature space into a similarity-preserving low-dimensional Hamming space. In this low-dimensional Hamming space, not only the original similarity between each data pair in the high-dimensional space is preserved, but also an approximate nearest neighbor of a given query can be found with sub-linear time complexity.

Different from the robust hashing algorithms for video/image copy detection which focus on uniqueness and discrimination of hash codes, the hashing for similarity search is proposed to speed up algorithms. Thus many hashing techniques are directly derived from dimensionality reduction algorithms. In other words, a projection matrix W is optimized by an objective of dimensionality reduction techniques. Then the hash function can be easily acquired by taking the sign function:

$$h(\mathbf{x}) = sgn(W^T\mathbf{x}). \tag{1.6}$$

One of the most well-known hashing techniques that preserve similarity information is Locality-Sensitive Hashing (LSH) [43]. LSH simply employs random linear projections (followed by random thresholding) to map data points close in a Euclidean space to similar codes. To gain a more complicated model, a kernel trick, which allows the use of a wide class of similarity functions, was combined with LSH to generalize locality-sensitive hashing with arbitrary kernel functions [77]. Spectral Hashing (SpH) [169] is a representative unsupervised hashing method, in which the Laplace-Beltrami eigenfunctions of manifolds are used to determine binary codes. In the SpH scheme, based on the idea of similarity-preserving of Laplacian Eigenmap, the binary codes $\mathbf{y}_1, \cdots, \mathbf{y}_N$ are obtained by optimizing the following problem:

$$\min \sum_{i,j} \|\mathbf{y}_i - \mathbf{y}_j\|^2 W_{ij}, \text{ s.t. } \mathbf{y}_i \in \{0,1\}^d$$

with some other bit constraints. Moreover, principled linear projections like PCA Hashing (PCAH) [165] has been suggested for better quantization rather than random projection hashing. Besides, another popular hashing approach, Anchor Graphs Hashing (AGH) [101], is proposed to learn compact binary codes via tractable low-rank adjacency matrices. AGH allows constant time hashing of a new data point by extrapolating graph Laplacian eigenvectors to eigenfunctions. Kernel Reconstructive Hashing (KRH) [176] was proposed to preserve the similarity defined by an arbitrary kernel using compact binary code. Compressed Hashing (CH) [89] has been effectively applied for large-scale data retrieval tasks as well.

To achieve better results, researchers have developed supervised hashing methods which could attain higher retrieval accuracy, since the label information is involved in the learning phase. A simple supervised hashing method is Linear Discriminant Analysis Hashing (LDAH) [148] which can tackle supervision via easy optimization but still lacks adequate performance due to the use of pairwise similarity of local descriptors without analyzing their discriminative information. Another recent supervised binary coding method called Kernel-Based Supervised Hashing (KSH) [100] has shown good discriminative ability of binary codes and outperformed other supervised methods such as Linear Discriminant Analysis

Hashing (LDAH) [148], Binary Reconstructive Embeddings (BRE) [76] and Minimal Loss Hashing (MLH) [112].

It also becomes a challenging problem when using millions of local descriptors with limited computational and storage resources in visual applications. Nowadays, it is more and more difficult to retrieve relative images and videos since they usually contain multiple objects, complex scenes and considerable semantic information. Therefore, using a group of local features to represent a sample is more effective than using a single holistic representation vector especially for matching and retrieval tasks. An early work of applying local features to image detection and retrieval was proposed in [70]. Based on LSH, Joly et al. [67] proposed a multi-probe locality sensitive hashing for approximate nearest-neighbor (ANN) search to improve the local feature based retrieval tasks [68]. Another ANN algorithm was introduced in [108] to speed up the searching algorithm and find the best algorithm configuration for various datasets. Although a hybrid hashing method for SIFT descriptors was proposed in [147], the relationships between local features are not included in the code learning phase. A main work for embedding local features to the Hamming space called Hamming Embedding (HE) [63] was proposed to map real-valued local features to binary codes. In this hashing scheme, the Hamming embedding and a weak geometric constraint were applied to improve the bag-of-words (BoW) model [143]. The above methods mainly focus on the feature-level analysis while most visual applications are image-oriented. It is necessary to develop a hashing method to effectively explore the relationship between images when each of them is represented by a set of local features.

1.3 Thesis Outline

The rest of this thesis is organized as follows. Chapter 2 presents a novel efficient supervised subspace learning algorithm for dimensionality reduction of local features, incorporating with a general orthogonalization method. Chapter 3 firstly presents a general descriptor for both RGB and depth video data and the proposed descriptors are then fused into binary representations by a novel structure-preserving local feature hashing. Chapter 4 extends the proposed binary coding method in Chapter 3 to an unsupervised scheme for cross-modal retrieval. Chapter 5 proposes a general subspace learning algorithm for multiview data. In Chapter 6, we conclude this thesis and discuss future works.

Chapter 2

Discriminant Analysis for Local Feature Reduction

2.1 Introduction

Recently, the use of local features has gained great popularity in computer vision. Based on local feature descriptors, e.g., SIFT [103], the sparse coding algorithm [85], dictionary learning [187], the naive Bayes nearest neighbor (NBNN) classifier [16], and Fisher kernels (FK) [62] have achieved state-of-the-art performance for image classification [99, 138]. Nevertheless, the increasingly large quantity of local feature descriptors makes local feature based algorithms severely restricted and even computationally intractable on large-scale data spaces. Dimensionality reduction algorithms [24, 41, 59, 183] are needed to reduce the computational complexity. However, due to the huge number N (up to 100M) of local feature descriptors, traditional algorithms [149, 168], e.g., manifold learning using nearest neighbor search (NN-search) with a computational complexity of at least $O(N^2)$, tend to be computationally prohabitive. Efficient algorithms are highly desirable to handle such huge amount of local feature descriptors for dimensionality reduction.

Furthermore, local feature descriptors, e.g., SIFT, are typically constructed in an unsupervised way, which would be less discriminative and contain redundant information. In contrast, supervised subspace learning [184] can not only reduce dimensions of local feature descriptors by removing redundant features but also improve the discriminability of local feature descriptors for classification. In fact, the label information could be used to achieve supervised dimensionality reduction of local feature descriptors, which however has not previously been investigated in the literature.

In this chapter, we propose a novel, efficient supervised subspace learning algorithm called Local Feature Discriminant Projection (LFDP) for dimensionality reduction of local features. Most dimensionality reduction methods are performed on the image representation level, while this paper focuses on the local feature level. LFDP offers an efficient discriminant analysis which can not only reduce the dimensionality but also enhance discriminative ability of local features. To achieve a supervised local feature reduction, we adopt the image-to-class (I2C) distance [16, 180, 183] which provides an effective measurement of distances between images and classes by incorporating class label information into local features. The discriminative analysis is established by adopting the Differential Scatter Discriminant Criterion (DSDC) [39, 152] into the I2C based image representations. The advantage of using DSDC is the avoidance of the matrix singularity problem [181], a shortcoming of LDA, which enables more accurate computation. Towards efficient computation of I2C distances, we use k-means clustering to reduce the range of NN-search into the centroids of local feature clusters in each class, which makes our algorithm computationally efficient without compromising the performance.

With the DSDC, we build our objective function to minimize the within-class variance while maximizing the between-class variance. However, the solution of our objective function is non-trivial due to its form of 4th order. We use the gradient descent algorithm on a sphere to solve this problem. In addition, an orthogonality constraint is imposed on the projections to make the subspace more compact while less redundant [59]. Unfortunately, existing orthogonalization methods [23, 35] cannot be straightforwardly applied to our scheme since they only orthogonalize the projections of the eigen-decomposition problem, which motivates us to propose a general orthogonalization on the projections via an induction method. The proposed generalized orthogonalization can also be widely applied to any other projection optimization problems. To summarize, the proposed LFDP possesses the following attractive merits:

- Unrestricted dimension: Unlike LDA, in which the reduced dimension is restricted by the number of classes, LFDP can project data onto any lower-dimensional space without suffering from the matrix singularity problem.

- $O(N)$ complexity: The time complexity of our algorithm is linear for N. In contrast to most manifold learning methods that need at least $O(N^2)$ time, our algorithm can be practically used for dimensionality reduction on large-scale data spaces.

- Generalized orthogonalization: The proposed orthogonalization method is more general and intuitive than previous methods [23, 35], and can also be applied to any other algorithms that need to compute projection matrices with the orthogonality constraints.

2.2 Related Work

Principal Component Analysis (PCA) is a popular dimensionality reduction method that can be directly applied to local features. Ke et al. [69] applied PCA to project the gradient image vector of a patch to a more compact descriptor, which is shorter than the standard SIFT descriptor but more robust to image deformations. Existing manifold learning algorithms, e.g., Laplacian Eigenmap (LE) [7], Locally Linear Embedding (LLE) [127] and ISOMAP [155], were proposed to learn the nonlinear structure of the data manifold. These algorithms suffer from the out-of-sample problem [8]. Locality Preserving Projections (LPP) [55] and Neighborhood Preserving Embedding (NPE) [54] as the linearized versions of LE and LLE, respectively, were developed to solve the out-of-sample problem. As unsupervised methods, they can be used for both global and local feature reduction. However, applying them to a large number of local features is computationally infeasible due to their high complexity. Moreover, similar to PCA, their discriminative ability is limited, as class label information is not used.

Linear Discriminant Analysis (LDA) is a conventional supervised method based on the Fisher criterion, which can also be imprudently employed for local feature reduction by using the class labels of the images from which local features are extracted. However, the large variability of local features will inevitably mislead the classifier since similar local features could be shared by images from different classes. Discriminative local descriptor learning has been explored individually in [59] and [24], both of which use the same covariance matrices of pair-wise matched and unmatched feature distances to find the linear projection. Recently, Simonyan et al. [141] proposed learning local feature descriptors using convex optimization. In fact, class labels of images are not used in the learning process, which makes the projections lose connection with classification and are therefore suboptimal. These discriminative methods [59, 141] need huge amount of ground truth with matched/unmatched pairs of local feature descriptors for training, which is not applicable in a realistic setting. Zhen et al. [183] proposed a supervised algorithm named I2C Distance Discriminative Embedding (I2CDDE) for dimensionality reduction of local features,

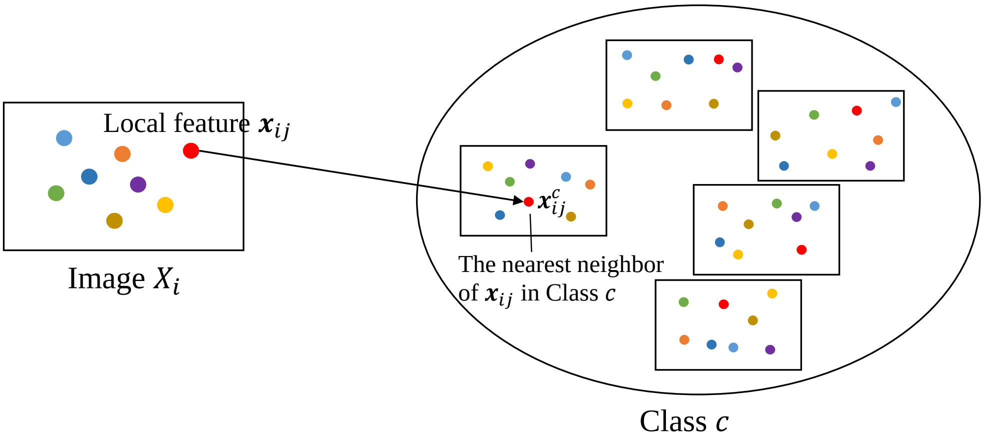

Local feature $\boldsymbol{x}_{ij}$
Image X_i
$\boldsymbol{x}_{ij}^c$
The nearest neighbor of $\boldsymbol{x}_{ij}$ in Class c
Class c

Fig. 2.1 The illustration of the I2C distance.

which is specifically designed for the NBNN classifier and also computationally expensive. Furthermore, these dimension reduction methods have at least $O(N^2)$ computational complexity, which severely limits their application in large-scale data spaces.

2.3 Local Feature Discriminant Projection

In this section, we introduce our Local Feature Discriminant Projection (LFDP) algorithm before which the I2C distance is revised. With image representations based on I2C distances, we build our objective function by incorporating the DSDC for discriminant analysis of local features. To solve the objective function, we present a gradient descent optimization algorithm with a novel, generalized orthogonalization procedure.

2.3.1 Notations

We are given n images $X_1, \cdots, X_n$ from C classes. For the c-th class, it contains n_c samples, $c = 1, \cdots, C$. Each image X_i is represented by a set of local feature descriptors $\{\mathbf{x}_{i1}, \cdots, \mathbf{x}_{im_i}\}$, where $\mathbf{x}_{ij} \in \mathbb{R}^D$ is the j-th local feature of the i-th image, $j = 1, \cdots, m_i$, $i = 1, \cdots, n$. We denote $N = \sum_{i=1}^{n} m_i$ as the total number of local feature descriptors from training images.

2.3.2 Image-to-Class Distance

The I2C distance introduced in the naive Bayes nearest neighbor (NBNN) classifier [16] represents the average of the sum of all distance squares from the local feature descriptors of an image to their corresponding nearest neighbors in each class. To be specific, the I2C distance from image X_i to class c is defined as

$$D_{X_i}^{c} = \frac{1}{m_i}\sum_{j=1}^{m_i} \|\mathbf{x}_{ij} - \mathbf{x}_{ij}^{c}\|^2,$$

where $\mathbf{x}_{ij}^{c}$ is the nearest neighbor of $\mathbf{x}_{ij}$ in class c and $\|\cdot\|$ is the L_2 norm. The illustration of I2C distance is shown in Fig. 2.1. However, in our scheme, to reduce the complexity of NN-search in the computation of I2C distances, we first employ the K-means clustering algorithm on the set of local feature descriptors of each class [98], i.e., $\bigcup_{X_i \in \text{class } c} X_i$, $c = 1, \cdots, C$. The search range is now reduced to the cluster centers, i.e., we let $\mathbf{x}^c \in$ Centroids of $\bigcup_{X_i \in \text{class } c} X_i$ for each c.

The I2C distance is a non-parametric approximation of the log-likelihood $\log p(X_i|c) = \log \prod_{j=1}^{m_i} p(\mathbf{x}_{ij}|c)$ [16]. When using Gaussian kernel density estimation, we have the following likelihood function:

$$p(\mathbf{x}|c) = \frac{1}{L_c}\sum_{k=1}^{L_c} \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x} - \mathbf{x}_k^{(c)}\|^2\right),$$

where $\mathbf{x}$ represents an arbitrary local feature descriptor and $\mathbf{x}_1^{(c)}, \cdots, \mathbf{x}_{L_c}^{(c)}$ are the local features extracted from all the images in class c. Note that with fixed centers, diagonal covariance matrices and equal weights, the density estimation turns out to be a special case of Gaussian mixture models (GMM) used in a state-of-the-art image representation called Fisher vectors [62, 119]. If we choose the centers, covariance matrices and weights of the GMM as, for instance, all of the training local features $\{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$, diagonal matrices and equal weights respectively, we have the likelihood function of the GMM

$$p(\mathbf{x}|\Theta) = \frac{1}{N}\sum_{i=1}^{N} \exp\left(-\frac{1}{2\sigma_i^2}\|\mathbf{x} - \mathbf{x}_i\|^2\right).$$

In this case, if the number of local features in each class (L_c) is the same, the log-likelihood of the GMM is positively related to the "average" of all the I2C distances and its gradients with respect to parameters construct a Fisher vector.

The Fisher vector is constructed as follows. We assume the local feature set $\mathscr{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$ can be modeled by a probability density function p_Θ (usually GMM). Then we can calculate the gradient vector

$$\nabla_\Theta \log p_\Theta(\mathscr{X}).$$

The above gradient vector of the log-likelihood describes the contribution of each parameter of p_Θ. Normally, a Fisher information matrix is defined as:

$$F_\Theta = E_{\mathscr{X}}[\nabla_\Theta \log p_\Theta(\mathscr{X}) \nabla_\Theta \log p_\Theta(\mathscr{X})^T].$$

Since F_Θ is a symmetric and positive definite matrix, by the Cholesky decomposition, we can have the following normalized gradient:

$$F_\Theta^{-\frac{1}{2}} \nabla_\Theta \log p_\Theta(\mathscr{X}),$$

which is usually referred as the Fisher vector of $\mathscr{X}$.

Based on I2C distances, we propose local feature discriminant projection (LFDP) by applying a discriminant analysis to local features for supervised dimensionality reduction. It is worthwhile to highlight that our LFDP is not restricted to the I2C distance. Other measurements, e.g., Kullback-Leibler divergence, the Hausdorff distance and the Bhattacharyya distance, could also be used to measure the relationship between images and classes. More importantly, our LFDP is a general supervised algorithm for dimension reduction which can be applied to any local feature descriptors including not only the handcrafted SIFT used in this paper, but also recent deep learning based representations [84, 97].

In addition, local features reduced by our LFDP can be fed to existing different representation methods, e.g., the bag-of-words model, sparse coding, NBNN and Fisher kernels. We use the Fisher kernels for the final image representations in order to achieve state-of-the-art performance.

2.3.3 Discriminant Analysis

Our goal is to seek a matrix $W \in \mathbb{R}^{D \times d}$ to project the original local features $\mathbf{x}_{ij}$ with dimension D to $W^T \mathbf{x}_{ij}$ in a lower-dimensional but more discriminative space $\mathbb{R}^d$. Note that after applying projection matrix W, the nearest neighbors may change. However, for the large-scale local feature space, we approximately adopt the sum of the distances from $W^T\mathbf{x}_{ij}$ to the projected nearest neighbor $W^T\mathbf{x}_{ij}^c$. Denote $\Delta X_{ic} = \frac{1}{\sqrt{m_i}}[(\mathbf{x}_{i1} - \mathbf{x}_{i1}^c), \cdots, (\mathbf{x}_{im_i} - \mathbf{x}_{im_i}^c)]^T \in \mathbb{R}^{m_i \times D}$. Then the projected I2C distance becomes

$$\begin{aligned}
\widehat{D}_{X_i}^c &= \frac{1}{m_i}\sum_{j=1}^{m_i} \|W^T\mathbf{x}_{ij} - (W^T\mathbf{x}_{ij})^c\|^2 \\
&\approx \frac{1}{m_i}\sum_{j=1}^{m_i} \|W^T\mathbf{x}_{ij} - W^T\mathbf{x}_{ij}^c\|^2 \\
&= \operatorname{tr}\left((\Delta X_{ic}W)(\Delta X_{ic}W)^T\right) \\
&= \operatorname{tr}\left((\Delta X_{ic}W)^T(\Delta X_{ic}W)\right) \\
&= \operatorname{tr}\left(W^T\Delta X_{ic}^T\Delta X_{ic}W\right).
\end{aligned}$$

Without loss of generality, we first consider the condition that W is a column vector $\mathbf{w}$ in the algorithm, i.e., $d = 1$. In fact, we will compute the column vectors of the projection matrix one by one. In this case, the projected I2C distances of an image will be

$$\mathbf{d}_i = (\widehat{D}_{X_i}^1, \cdots, \widehat{D}_{X_i}^C) = (\mathbf{w}^T\Delta X_{i1}^T\Delta X_{i1}\mathbf{w}, \cdots, \mathbf{w}^T\Delta X_{iC}^T\Delta X_{iC}\mathbf{w}), \tag{2.1}$$

which is called an *I2C vector*. In other words, for each image X_i, we have a corresponding vector $\mathbf{d}_i$ in linear space $\mathbb{R}^C$ which is called *I2C vector space*. Then we have the mean of the vectors in class i and the mean of all the vectors, denoted by μ_i and μ, respectively. Having the representations with I2C vectors, we incorporate the Differential Scatter Discriminant Criterion in the I2C vector space to obtain our objective function in the following form that needs to be maximized:

$$J = \underbrace{\sum_{c=1}^{C} n_c\|\mu_c - \mu\|^2}_{\text{Between class variance}} - \underbrace{\lambda\sum_{c=1}^{C}\sum_{\mathbf{d}_k\in\text{class }c}\|\mathbf{d}_k - \mu_c\|^2}_{\text{Within class variance}}, \tag{2.2}$$

where λ is a tuning parameter. μ_c and μ are computed by the following equations

$$\begin{aligned}
\mu_c &= \frac{1}{n_c}\sum_{\mathbf{d}_k\in\text{class }c}\mathbf{d}_k := (\mathbf{w}^TM_{c1}\mathbf{w}, \cdots, \mathbf{w}^TM_{cC}\mathbf{w}), \\
\mu &= \frac{1}{N}\sum_{k=1}^{N}\mathbf{d}_k := (\mathbf{w}^TM_1\mathbf{w}, \cdots, \mathbf{w}^TM_C\mathbf{w}),
\end{aligned}$$

where

$$M_{cj} = \frac{1}{n_c} \sum_{\mathbf{d}_k \in \text{class } c} \Delta X_{kj}^T \Delta X_{kj}, \; c, j = 1, \cdots, C,$$

and

$$M_j = \frac{1}{N} \sum_{i=1}^{N} \Delta X_{ij}^T \Delta X_{ij}, \; j = 1, \cdots, C.$$

Now we can formulate J as a function of $\mathbf{w}$ as follows:

$$J(\mathbf{w}) = \sum_{c=1}^{C} n_c \sum_{j=1}^{C} (\mathbf{w}^T \Delta M_{cj} \mathbf{w})^2 - \lambda \sum_{c=1}^{C} \sum_{\mathbf{d}_k \in \text{class } c} \sum_{j=1}^{C} (\mathbf{w}^T V_{kj}^c \mathbf{w})^2, \tag{2.3}$$

where $\Delta M_{cj} = M_{cj} - M_j$ and $V_{kj}^c = \Delta X_{kj}^T \Delta X_{kj} - M_{cj}$ for $\mathbf{d}_k \in$ class c, $c, j = 1, \cdots, C$.

2.3.4 Gradient Descent on Sphere

The classic eigen-decomposition of a matrix is not applicable to our problem due to the quartic form of the objective function. We adopt a procedure of gradient descent on a sphere to find the projection vector. Our goal is to find the optimal $\mathbf{w}$ by maximizing $J(\mathbf{w})$. To obtain the final orthonormal projection matrix, we set a norm constraint $\|\mathbf{w}\| = 1$ for each vector. However, the update rule of the traditional gradient descent for a maximization problem: $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + \gamma \nabla J(\mathbf{w}^{(t)})$ does not guarantee this constraint. Thus we amend the original algorithm and constrain it on the D-dimensional unit sphere.

Define two matrix-valued functions

$$M(\mathbf{w}) = \sum_{c=1}^{C} n_c \sum_{j=1}^{C} \mathbf{w}^T \Delta M_{cj} \mathbf{w} \cdot \Delta M_{cj} \tag{2.4}$$

and

$$V(\mathbf{w}) = \sum_{c=1}^{C} \sum_{\mathbf{d}_k \in \text{class } c} \sum_{j=1}^{C} \mathbf{w}^T V_{kj}^c \mathbf{w} \cdot V_{kj}^c. \tag{2.5}$$

We obtain the gradient of $J(\mathbf{w})$:

$$\nabla J(\mathbf{w}) = 2M(\mathbf{w})\mathbf{w} - 2\lambda V(\mathbf{w})\mathbf{w}. \tag{2.6}$$

We project $\nabla J(\mathbf{w})$ onto the tangent direction of $\mathbf{w}$ on the sphere as $\mathbf{p} = \nabla J(\mathbf{w}) - \langle \nabla J(\mathbf{w}), \mathbf{w} \rangle \mathbf{w}$

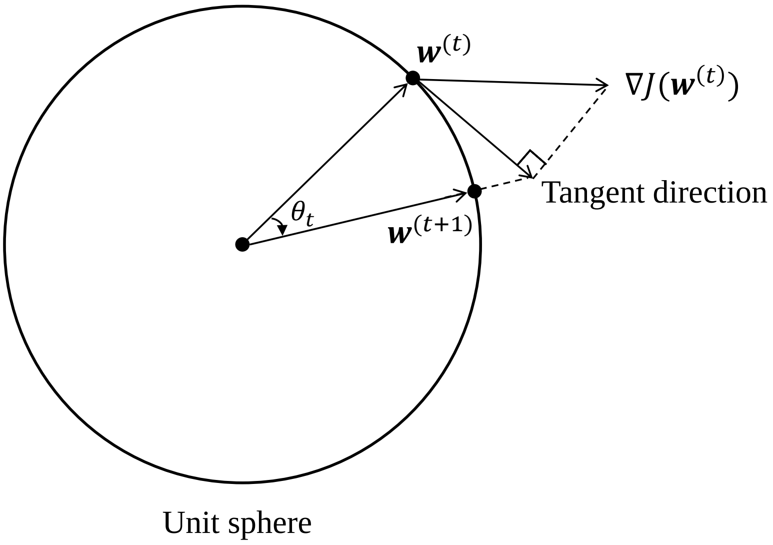


Fig. 2.2 The illustration of the gradient descent on sphere.

and normalize it as $\mathbf{p}_0 = \mathbf{p}/\|\mathbf{p}\|$. By using the first-order Taylor expansion, we know $\nabla J(\mathbf{w})$ is the steepest increasing direction. For direction $\mathbf{p}$, we have $\langle \mathbf{p}, \nabla J(\mathbf{w})\rangle = \langle \nabla J(\mathbf{w}), \nabla J(\mathbf{w})\rangle - \langle \nabla J(\mathbf{w}), \mathbf{w}\rangle^2 = \|\nabla J(\mathbf{w})\|^2 - \|\nabla J(\mathbf{w})\|^2 \cos^2\alpha \geq 0$, where α is the angle between $\nabla J(\mathbf{w})$ and $\mathbf{w}$. Thus $\mathbf{p}$ is still an increasing direction. Then for the t-th step, we have the following update rule:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} \cos\theta + \mathbf{p}_0^{(t)} \sin\theta, \tag{2.7}$$

where $\theta \in [0, \pi/2]$ is the step size. Since $\mathbf{w}$ and $\mathbf{p}_0$ are orthogonal, the norm of the updated variable remains of unit length. In addition, to accelerate the convergence, we also employ an adaptive step size θ_t, i.e., if $J(\mathbf{w}^{(t+1)}) \geq J(\mathbf{w}^{(t)})$, we set $\theta_{t+1} = \min(2\theta_t, \pi/2)$, otherwise, $\theta_{t+1} = \theta_t/2$. The iterative procedure is described in Algorithm 1 and the principle idea is illustrated in Fig. 2.2.

2.3.5 Orthogonality Constraints

Until now we have only computed the projection vector for the first dimension. In this section, we use the induction method to compute the remaining vectors successively and make them mutually orthogonal by using the matrix composed by previous output vectors. Previous works [35, 59] have highlighted the benefits of orthogonality constraints, for instance, avoidance of overfitting and redundancy in representing the subspace. With this

Algorithm 1 The Gradient Descent for Local Feature Discriminant Projection

```
Input: The local feature descriptors ${\mathbf{x}_{ij}}$ of each image and the parameter $K$ in K-means.
Output: The projection vector $\mathbf{w}$ in the first dimension.
  Employ K-means algorithm for the local feature set of each class;
  Find the nearest neighbor $\mathbf{x}_{ij}^c$ of $\{\mathbf{x}_{ij}\}$ in the centroids of each class;
  Compute matrix-valued functions $M(\mathbf{w})$ and $V(\mathbf{w})$ in Eqs. (2.4) and (2.5);
  Initialize step size $\theta_1 \in (0, \pi/2)$ and randomly initialize unit vector $\mathbf{w}^{(1)}$;
  repeat
    Compute the projection of $\nabla J(\mathbf{w}^{(t)})$ on the tangent direction of $\mathbf{w}^{(t)}$: $\mathbf{p}^{(t)} = \nabla J(\mathbf{w}^{(t)}) - \langle \nabla J(\mathbf{w}^{(t)}), \mathbf{w}^{(t)} \rangle \mathbf{w}^{(t)}$ and apply normalization $\mathbf{p}_0^{(t)} = \mathbf{p}^{(t)}/\|\mathbf{p}^{(t)}\|$;
    Compute $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} \cos\theta_t + \mathbf{p}_0^{(t)} \sin\theta_t$;
    while $J(\mathbf{w}^{(t+1)}) \geq J(\mathbf{w}^{(t)})$ do
      $\theta_t \leftarrow \theta_t/2$;
    end while
    Update $\theta_{t+1} = \min(2\theta_t, \pi/2)$;
  until convergence.
```

orthogonalization procedure, we can establish our whole algorithm.

Suppose we have obtained the first p $(p \geq 1)$ discriminant vectors $\mathbf{w}_1, \mathbf{w}_2, \cdots . \mathbf{w}_p$. We want to compute the next vector $\mathbf{w}_{p+1}$ to maximize $J(\mathbf{w})$ with the orthogonal constraints

$$\mathbf{w}_1^T \mathbf{w}_{p+1} = \mathbf{w}_2^T \mathbf{w}_{p+1} = \cdots = \mathbf{w}_p^T \mathbf{w}_{p+1} = 0, \tag{2.8}$$

and an additional norm constraint on $\mathbf{w}_{p+1}$, i.e., $\|\mathbf{w}_{p+1}\| = 1$. The method in [35] can not be applied in our scheme due to the high degree of Lagrangian in our case. We use an alternative but more general method by basis transformation to solve this issue. In other words, we compute the next discriminant vector in a special subspace in which the orthogonal constraints vanish.

According to the inductive assumption, vectors $\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_p$ should be an orthonormal basis of a subspace in $\mathbb{R}^D$. Let us denote $V_p = \text{span}(\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_p)$ and $W_p = [\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_p]$. Then V_p is a p-dimensional subspace and W_p is a $D \times p$ matrix. Recall that our primary goal is to seek an optimal $\mathbf{w}$ by maximizing $J(\mathbf{w})$:

$$\arg\max_{\mathbf{w} \in \mathbb{R}^D} J(\mathbf{w}). \tag{2.9}$$

Once we have obtained subspace V_p, $\mathbf{w}_{p+1}$ is required to be orthogonal to all the vectors in

V_p. Consequently, we need to compute the constrained optimization problem

$$\arg\max_{\mathbf{w}\in V_p^{\perp}} J(\mathbf{w}) \tag{2.10}$$

to find the solution of $\mathbf{w}_{p+1}$, where $V_p^{\perp}$ is the null space of V_p and $dimV_p^{\perp} = D-p$. Straightforwardly, the data can be projected onto subspace $V_p^{\perp}$ so that the computation process is completely performed in a $(D-p)$-dimensional linear subspace, i.e., the new coordinates are in $\mathbb{R}^{D-p}$. Then the output will be orthogonal to any vectors in V_p. For this reason, we need to find a basis $B_p = [\mathbf{b}_1, \cdots, \mathbf{b}_{D-p}] \in \mathbb{R}^{D\times(D-p)}$ of $V_p^{\perp}$. In fact, we need only to solve the linear equation $W_p^T X = 0$, which is commonly used in linear algebra. Furthermore, we make this basis orthonormal by following the Gram-Schmidt procedure.

Now with this orthonormal basis B_p, we project data from $\mathbb{R}^D$ onto subspace $V_p^{\perp}$. Specifically, for a local feature and an I2C vector, we have transformations $\mathbf{x}_{ij} \leftarrow B_p^T \mathbf{x}_{ij}$ and $\mathbf{d}_i \leftarrow (\mathbf{v}^T B_p^T \Delta X_{i1}^T \Delta X_{i1} B_p \mathbf{v}, \cdots, \mathbf{v}^T B_p^T \Delta X_{iC}^T \Delta X_{iC} B_p \mathbf{v})$, respectively, where $\mathbf{v}$ is a vector in $\mathbb{R}^{D-p}$. Then we only need to solve the unconstrained problem in a lower-dimensional space:

$$\arg\max_{\mathbf{v}\in\mathbb{R}^{D-p}} J_p(\mathbf{v}) = \arg\max_{\mathbf{v}\in\mathbb{R}^{D-p}} \left(\mathbf{v}^T M_p(\mathbf{v})\mathbf{v} - \lambda \mathbf{v}^T V_p(\mathbf{v})\mathbf{v}\right), \tag{2.11}$$

where $M_p(\cdot)$ and $V_p(\cdot)$ are the images of matrix-valued functions $M(\cdot)$ and $V(\cdot)$ after the projection, respectively, i.e., $\Delta M_{cj} \leftarrow B_p^T \Delta M_{cj} B_p$ and $V_{kj}^c \leftarrow B_p^T V_{kj}^c B_p$. Now it is an optimization problem where the constraints vanish and here we return to our first goal in the $(D-p)$-dimensional space.

Having the solution $\mathbf{v}^*$ for the optimization problem (2.11) in $\mathbb{R}^{D-p}$, we transform it to an element in $V_p^{\perp} \in \mathbb{R}^D$. Actually, $\mathbb{R}^{D-p}$ and $V_p^{\perp}$ are two isomorphic linear spaces and B_p can be regarded as a linear isomorphism between them. Through the representation of an orthonormal basis, for each $\mathbf{w} \in V_p^{\perp}$, we have $\mathbf{w} = \sum_{i=1}^{D-p} w_i \mathbf{b}_i$, where $w_i \in \mathbb{R}$, and the inner product of $\mathbf{w}$ and $\mathbf{b}_i$ will be $\langle \mathbf{w}, \mathbf{b}_i \rangle = w_i$, $\forall i$. Then $(w_1, \cdots, w_{D-p})^T = (\langle \mathbf{w}, \mathbf{b}_1 \rangle, \cdots, \langle \mathbf{w}, \mathbf{b}_{D-p} \rangle)^T = [\mathbf{b}_1, \cdots, \mathbf{b}_{D-p}]^T \mathbf{w} = B_p^T \mathbf{w}$, i.e., the result of multiplying the left side of $\mathbf{w}$ by B_p^T is the coefficient of the representation by B_p. Finally, we set $\mathbf{w}_{p+1} = B_p \cdot \mathbf{v}^* \in V_p^{\perp}$ as a linear combination of B_p. The whole LFDP algorithm is illustrated in Algorithm 4.

Remark. The proposed orthogonalization procedure is a more general way to comppute orthogonal projection matrices. Note that, in Algorithm 2, given the input of Algorithm 1, we need only Algorithm 1 to output a projection vector without need to know the computa-

Algorithm 2 Local Feature Discriminant Projection

```
Input: The input of Algorithm 1 and the target dimension $d$.
Output: The projection matrix $\mathbf{w}$.
  Initialization: $\mathbf{w} \leftarrow \emptyset$ and $B \leftarrow I$;
  for $i = 1$ to $d$ do
    Project training data onto the null space of $\mathrm{span}(\mathbf{w})$ by using the basis $B$;
    Call Algorithm 1 to compute the corresponding projection vector $\mathbf{w}_i$ in subspace $\mathrm{span}(\mathbf{w})^{\perp}$ and update $\mathbf{w}_i \leftarrow B\mathbf{w}_i$;
    Update $\mathbf{w} \leftarrow [\mathbf{w}, \mathbf{w}_i]$ and let $B$ be an orthonormal basis of $\mathrm{span}(\mathbf{w})^{\perp}$ by solving the corresponding linear equation and following the Gram-Schmidt procedure.
  end for
```

Table 2.1 Comparing the complexity of LFDP with other linear algorithms on N where K is the parameter of K-means and k is the parameter of the KNN algorithm.

Method	LFDP	PCA	LDA	I2CDDE [183]	LDE [59]	LDP [24]	LPP [55]	NPE [54]
Complexity	$O(KN)$	$O(N)$	$O(N)$	$O(N^2)$	$O(N^2)$	$O(N^2)$	$O(kN^2)$	$O(kN^2)$

tion process. Therefore, Algorithm 1 could be seen as a *black box* that is able to compute the projection vector (for those that output a matrix, we only need its first column). Now we have the following general proposition.

Proposition 1 *Given maximizing (minimizing) algorithm $\mathscr{A}$ which takes projected data $\mathbf{w}^T\mathbf{x}$ as input and outputs the optimal vector, and an orthonormal basis B_p of $(D-p)$-dimensional subspace $V_p^{\perp} \subseteq \mathbb{R}^D$, if $\mathbf{v}^*$ is the optimal solution of $\mathscr{A}(\mathbf{v}^T B_p^T \mathbf{x})$ in $\mathbb{R}^{D-p}$, $\mathbf{w}^* = B_p\mathbf{v}^*$ is the optimal solution of $\mathscr{A}(\mathbf{w}^T\mathbf{x})$ in $V_p^{\perp}$.*

2.3.6 Relations between Algorithm 2 and the ordinary eigen-decomposition

In fact, assuming that the optimization problem is simplified to the eigen-decomposition of a symmetric matrix $A \in \mathbb{R}^{D\times D}$ such as PCA, we prove that the proposed orthogonalization method finds the same eigenvectors with the eigen-decomposition by adopting mathematical induction. Suppose $A = \sum_{i=1}^{D} \lambda_i \mathbf{w}_i \mathbf{w}_i^T = W\Lambda W^T$ is the spectral decomposition of A and $\lambda_1 \geq \cdots \geq \lambda_D$, where $\Lambda = diag(\lambda_1, \cdots, \lambda_D)$ and $W = [\mathbf{w}_1, \cdots, \mathbf{w}_D]$. Then $\mathbf{w}_i^T\mathbf{w}_j = 0$ if $i \neq j$ and $\mathbf{w}_i^T\mathbf{w}_i = 1$ for $i = 1, \cdots, D$.

For the first vector, both Algorithm 2 and the eigen-decomposition output the eigenvector $\mathbf{w}_1$ corresponding to the largest eigenvalue of A. Assume Algorithm 2 has output the first k eigenvectors $\mathbf{w}_1, \cdots, \mathbf{w}_k$. For the $(k+1)$-th vector, $\mathbf{w}_{k+1}$ is the eigenvector corresponding to the eigenvalue λ_{k+1}. Algorithm 2 first finds an orthonomal basis $B \in \mathbb{R}^{D\times(D-k)}$ of $span(\mathbf{w}_1, \cdots, \mathbf{w}_k)^{\perp}$. Since W is an orthogonal matrix, we have $span(\mathbf{w}_1, \cdots, \mathbf{w}_k)^{\perp} = span(\mathbf{w}_{k+1}, \cdots, \mathbf{w}_D)$. Then there exists an orthogonal matrix $P \in \mathbb{R}^{(D-k)\times(D-k)}$ such that $B = W_{k+1}P$, where $W_{k+1} = [\mathbf{w}_{k+1}, \cdots, \mathbf{w}_D]$. In the $(k+1)$-th step of Algorithm 2, we eigen-decompose the matrix B^TAB to compute its largest eigenvalue. Note that

$$\begin{aligned} B^TAB &= P^TW_{k+1}^T\left(\sum_{i=1}^{D}\lambda_i\mathbf{w}_i\mathbf{w}_i^T\right)W_{k+1}P \\ &= P^TW_{k+1}^T\left(\sum_{i=k+1}^{D}\lambda_i\mathbf{w}_i\mathbf{w}_i^T\right)W_{k+1}P \\ &= P^TW_{k+1}^TW_{k+1}\Lambda_{k+1}W_{k+1}^TW_{k+1}P \\ &= P^T\Lambda_{k+1}P, \end{aligned}$$

where $\Lambda_{k+1} = diag(\lambda_{k+1}, \cdots, \lambda_D)$. Therefore, the largest eigenvalue of B^TAB is still λ_{k+1}, which indicates that the corresponding eigenvalues of the output vectors of Algorithm 2 are $\lambda_1, \cdots, \lambda_D$. Then the whole output set of Algorithm 2 is $\{\mathbf{w}_1, \cdots, \mathbf{w}_D\}$ up to sign.

2.3.7 Complexity Analysis

Our LFDP is computationally more efficient than most of the existing manifold learning methods. We provide a complexity analysis on the two procedures: gradient descent and orthogonalization of our LFDP in terms of time complexity and memory cost, since in the test phase, the complexity depends on the classifier and the time complexity will apparently be reduced after dimensionality reduction.

Gradient descent. During the iterative procedure of gradient descent, the main cost is induced by the computation of the I2C distances. The time complexity of a brute-force method of NN-search in K centroids with D-dimension is $O(KND)$. Computing $M(\mathbf{w})$ and $V(\mathbf{w})$ needs $O(D^2C^2)$ and $O(D^2Cn)$ time respectively, where n is the number of training images. Then the time complexity of the gradient descent with N_{iter} steps in a D-dimensional space is $O(N_{iter}(D^2C^2 + D^2Cn))$ and the time complexity of the whole procedure is at most $O(KND + N_{iter}D^2C^2)$. The memory cost of the iterative procedure is $O(D^2C^2 + D^2Cn)$.

Orthogonalization. We can observe that the main step in the orthogonalization procedure is the Gram-Schmidt procedure, which requires at most $O(nm^2)$ time and $O(nm + m^2)$ memory for computing on m n-dimensional vectors [44]. Notice that, in our algorithm, m varies from 1 to d and n varies from D to $D - d + 1$, where d is the dimension of the projected space.

In total, with the complexity $O(TKND)$ in the K-means, where T is the number of iterations in the K-means, our LFDP algorithm requires at most $O((T+1)KND + dN_{iter}(D^2C^2 + D^2Cn) + \frac{1}{6}d^3D)$ time complexity and $O(D^2C^2 + D^2Cn + \frac{1}{2}d^2D + \frac{1}{6}d^3)$ memory. Due to the large number of local feature descriptors, generally $N \gg D$, we show the computational complexity on N through comparing our algorithm with other dimensionality reduction methods in Table 2.1, where K is the parameter of K-means and k is the parameter of the k-nearest neighbor (KNN) algorithm. In fact, KNN-based algorithms highly rely on the neighborhood structure of each point, which will be changed by K-means clustering. In addition, K-means may also change the order of I2C distances where there are similar classes or noisy data points, and therefore, mislead the learning of I2CDDE leading to the failure of NBNN. In contrast, our discriminant analysis considers the relationships of intra-class and inter-class variations among I2C vectors, achieving a global optimization objective. Therefore, using K-means centroids can not only make our LFDP computationally more efficient but also tolerant to the fluctuation of I2C distances.

2.4 Experiments

We have extensively validated our LFDP algorithm on three widely used benchmark datasets, i.e., UIUC-Sports, Scene-15 and MIT Indoor. Experimental results show that our LFDP largely outperforms representative dimension reduction algorithms and achieves state-of-the-art performance in terms of the classification accuracy which is formulated as follows:

$$\text{accuracy} = \frac{\text{the number of correct predicted labels in the test set}}{\text{the total number of images in the test set}}.$$

2.4.1 Implementation details

The optimal tuning parameter λ for each dataset is selected from one of $\{0.1, 0.2, \cdots, 1\}$, which yields the best performance by 10-fold cross-validation on the training data. We fix $K = 300$ in K-means for all datasets and set the maximum number of the K-means iteration as 20. In addition, the K-means clustering for each class can be performed in a parallel way to speedup the algorithm. We take the Improved Fisher Kernel (IFK), which is an improved version of Fisher kernels [120], based on raw SIFT descriptors without dimension reduction as the baseline. We compare with PCA as a representative unsupervised algorithm which has shown competitive and even better performance than manifold learning algorithms including ISOMAP, LLE and LE on diverse tasks [158]. LDA is included for comparison as a supervised algorithm. The parameter k of the KNN algorithm in LPP and NPE is tuned by selecting from $\{5, 6, \cdots, 15\}$. By following the setting in [24], we randomly select 1.5×10^5 local features from all the training sets for training the projection of LDP. ISOMAP is not involved in the comparison due to the out-of-sample problem. All the experiments are implemented using Matlab 2013b on a workstation with an i7 processor and 32GB RAM.

2.4.2 Datasets

UIUC-Sports. The Sports event dataset was introduced in [88], consisting of 8 sports event categories. The number of images in each class ranges from 137 to 250. We follow the experimental setting in [88] to randomly select 70 and 60 images per class for training and testing respectively. The procedure is repeated five times and the average is reported as the final result. Differently, we use the original images rather than the resized ones.

Scene-15. The Scene-15 dataset [83] consists of 4485 images which are labeled in 15 distinct classes. The number of images in each class varies from 200 to 400. Following the experimental setting in [83], we randomly select 100 images in each class as training data and test the remaining images. The procedure is repeated five times and the average is reported as the final result.

MIT Indoor. The MIT Indoor scene dataset [123] contains 67 indoor scene categories for a total of 15620 images. The number of images in each class varies from 100 to 734. 80 and 20 images are selected in each category for training and testing respectively by following the experimental setting in [123]. The procedure is repeated five times and the average is reported as the final result.

2.4.3 Local Feature and Classifier

We use the software provided by Yang et al. [177] to compute the SIFT descriptors. In contrast to existing works which either use multi-scale SIFT descriptors [167], spatial pyramid representation [128] or multiple descriptors [16, 167], we simply use single-scale SIFT descriptors in patches of 16×16. In our experiments, the average numbers of local features extracted from each image in three datasets are all 1500. Then the total numbers (N) of the

Table 2.2 Resource requirements of different methods for the $900,000$ SIFT features from the UIUC-Sports dataset.

Method	Memory cost	Runtime
LFDP	1GB	20 mins
I2CDDE	1GB	8 hrs
LDE / LDP	1GB	8 hrs
LPP / NPE	900GB	16 hrs

training local features in the above three datasets are $900,000$, $2,000,000$ and $8,000,000$, respectively.

We employ a linear SVM classifier with IFK [120] and compute the Fisher vector for each image based on its local features by using 256 Gaussians in the GMM. As the settings in [120], we first use the power normalization by applying the following function:

$$f(z) = \mathrm{sgn}(z)|z|^{0.5}$$

to each entry of the Fisher vector, and then employ the L2 normalization by dividing the L2 norm of the power-normalized Fisher vector.

2.4.4 Resource Requirements

In Table 2.2, we list the resource requirements for training the projections by different dimensionality reduction methods. The nearest neighbor search and the computation for pairwise distances make $O(N^2)$ methods suffer from the high computational complexity. Note that the runtime for LPP and NPE is a theoretical value since it is infeasible to implement them with such large memory. Therefore, to use the largest possible number of features that can be handled by our workstation, a subset consisting 1.5×10^5 local features is randomly selected from the whole training set for evaluating these methods.

2.4.5 Results

The performance comparison of LFDP with other dimensionality reduction methods is shown in Fig. 2.3 (a), (b) and (c) for UIUC-Sports, Scene-15 and MIT Indoor, respectively. The baseline represents the performance of SVMs with IFK in the original 128-dimensional SIFT space without dimensionality reduction. The proposed method shows consistent ad-

Table 2.3 Performance (%) of linear SVMs with IFK after PCA, LDA and LFDP reduction on local features. The results listed in the table are their best accuracies. The baseline is the classification result of IFK without dimensionality reduction of local feature descriptors.

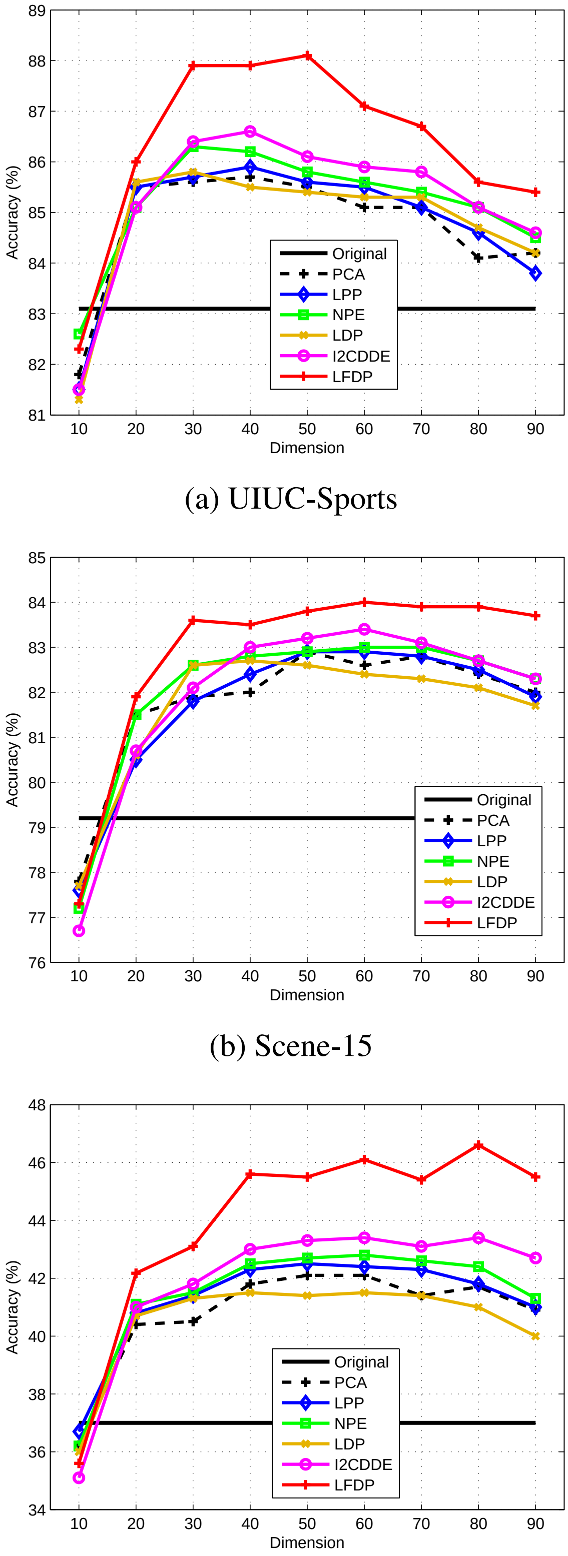
Method	UIUC-Sports	Scene-15	MIT Indoor
Baseline	83.1 ± 0.3	79.2 ± 0.2	37.0 ± 0.3
PCA	85.7 ± 0.2	82.9 ± 0.4	42.1 ± 0.4
LDA^1	81.2 ± 0.4	79.9 ± 0.4	38.6 ± 0.5
LDA^2	85.4 ± 0.4	83.0 ± 0.3	42.3 ± 0.4
LFDP^1	$\mathbf{88.1 \pm 0.5}$	$\mathbf{84.0 \pm 0.5}$	$\mathbf{46.6 \pm 0.4}$
LFDP^2	80.1 ± 0.4	78.3 ± 0.6	36.4 ± 0.5

LDA^1 is the LDA with the Fisher criterion. LDA^2 is the LDA with the DSDC. LFDP^1 is our algorithm with the orthogonality constraint and LFDP^2 is the LFDP without the orthogonality constraint.

vantages on all the three datasets. Our method improves the baseline phenomenally with a large margin. PCA usually reaches its highest accuracy around the dimension of 50 and remains stable with the increase of dimensionality. Other methods such as LPP, NPE, LDP and I2CDDE only sightly outperform PCA. In contrast with the above methods, we can observe that LFDP goes up rapidly with the increase of the dimension when the dimension is low and achieves the competitive results around the dimension of 40 (even at 30). With the reduced local feature descriptors by LFDP, the dimensionality of Fisher vectors is several times shorter than the original dimension, which reduces the computational cost for classification but strengthens the discriminative ability due to the supervised learning.

Furthermore, the advantage of our method has been also shown by comparing with L-DA. Note that LDA learns the projection matrix by directly labeling the local features with class labels of images they belong to. Since the performance of LDA is also restricted by the number of classes [153], the upper bound of reduced dimensionality of LDA is $C-1$, on which LDA reaches its best performance. We report the best results of PCA and LDA on different datasets for the comparison with the results of LFDP in Table 2.3. LDA with the Fisher criterion produces results below the baseline on the UIUC-Sports dataset since it contains only 8 classes so that the result is obtained by 7-dimensional local descriptors. To alleviate the dimension restriction of LDA with the Fisher criterion, we implement L-DA with the DSDC criterion using the parameter λ similar to Eq. (2.2). We tune λ in $\{0.1, 0.2, \cdots, 1\}$ and the best results are reported in Table 2.3. With the DSDC, the reduced dimension of LDA is not restricted by the number of classes and the results are significantly improved.

LFDP can efficiently find lower-dimensional but more discriminative feature space and



(a) UIUC-Sports

(b) Scene-15

(c) MIT Indoor

Fig. 2.3 Performance (%) of linear SVMs with IFK in different lower-dimensional subspaces on the UIUC-Sports, Scene-15 and MIT Indoor datasets. Note that we only use one type of local descriptor: SIFT in single-scale patches.

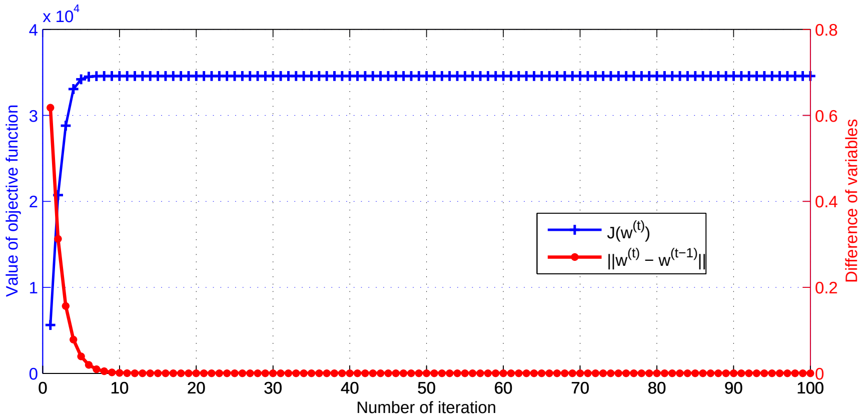


Fig. 2.4 The convergency of the objective function and the difference of variables with respect to the number of iteration.

achieves the state-of-the-art results [87, 90, 167], which reveals its capability in dimensionality reduction of ubiquitous local feature spaces in large scale.

2.4.6 Algorithm Analysis

We also evaluate the performance of Algorithm 1 in terms of convergency. We randomly initialize $\mathbf{w}$ 50 times on the UIUC-Sports dataset and the average value of the objective function in Eq. (4.15) and the average difference $||\mathbf{w}^{(t)} - \mathbf{w}^{(t-1)}||$ on the first dimension are reported in Fig. 2.4, where t is the number of iteration and λ is fixed at 0.1. We can observe that $\mathbf{w}$ converges within only 10 steps. Therefore, we always fix the maximum number of iteration at 10 in the experiments.

To show the effectiveness of the orthogonality constraint, we also compare the results of LFDP with/without the orthogonality constraint in Table 2.3. In the case without the orthogonality constraint, the objective function in Eq. (4.15) beomces $J(\mathbf{w}) = \sum_{c=1}^{C} n_c \sum_{j=1}^{C} \mathrm{tr}(\mathbf{w}^T \Delta M_{cj} \mathbf{w})^2 - \lambda \sum_{c=1}^{C} \sum_{\mathbf{d}_k \in \text{class } c} \sum_{j=1}^{C} \mathrm{tr}(\mathbf{w}^T V_{kj}^{c} \mathbf{w})^2$. By adopting the same gradient descent procedure in Section 2.3.4, the projection matrix $\mathbf{w} \in \mathbb{R}^{D \times D}$ can be acquired directly. As we can see from the results, LFDP performs much better than that without the orthogonality constraint, which indicates the effectiveness of the proposed orthogonal method.

In addition, LFDP achieves the best performance with a small value of K in K-means, which guarantees the computational efficiency. We have investigated the performance under different values of parameter K as shown in Table 2.4. On all the three datasets, our method yields the best results with $K = 300$ which is much smaller than the number of local features,

Table 2.4 Comparing the results (%) of LFDP with different K values. The best results while varying the target dimension are listed.

K Dataset	50	100	200	300	400	500
UIUC-Sports	76.5	83.2	86.7	**88.1**	88.0	88.0
Scene-15	75.3	82.7	83.6	**84.0**	83.8	84.0
MIT Indoor	36.7	40.3	44.8	**46.6**	46.4	46.4

which is up to $120,000$ in each class. This largely reduces the computational complexity.

2.5 Summary

A new subspace learning algorithm called Local Feature Discriminant Projection (LFDP) has been proposed for supervised dimensionality reduction of local features. The projections for reduction are obtained by optimizing an objective function constructed based on the Differential Scatter Discriminant Criterion and the I2C representations. A general orthogonalization method has been proposed to learn the projections which guarantees a more compact space with less redundancy. The proposed LFDP has a much lower complexity than popular manifold learning methods, providing an alternative way to efficiently analyze large-scale data. The experimental results on three widely used benchmarks for image classification have validated the effectiveness of LFDP and shown its advantages over traditional dimensionality reduction algorithms. In the next chapter, we will address the feature reduction for action recognition.

Chapter 3

Binary Structure-Preserving Representation Learning for RGB-D Video Data

3.1 Introduction

In the last chapter, we have discussed the discriminant analysis for image classification. This chapter mainly analyzes the feature reduction for action recognition. For this purpose, we consider the 3-dimensional coordinate of video data rather than the 2-dimensional image plane. The contribution of this chapter consists of two parts. First, we propose a new general descriptor called Local Flux Feature (LFF) for both RGB and depth video data. Then the proposed descriptors extracted from the RGB and depth channels are fused into the binary representations via the Structure Preserving Projection (SPP) to improve the efficiency and the accuracy of RGB-D action recognition.

RGB-D sensors such as Kinect receive increasing attention in the computer vision community [49]. They have been widely applied to many areas such as: human activity recognition [166], robot path planning [117], object detection [145], scene labeling [125], interactive gaming [29] and 3D mapping [56]. The combination of RGB and depth information enables enhanced capabilities of computer vision algorithms. It also provides an alternative way to learn features from video data for action recognition, especially through learning fused RGB-D representations.

To gain a more robust and accurate representation of samples, local feature descriptors such as: SIFT [103], HOG3D [73], HOG [30], HOF [82] and MBH [31] have been proposed and achieved notable success in classification and recognition. Based on these local features, the Bag-of-Words (BoW) model [143] and the Sparse Coding (SC) algorithm [85] have shown their effectiveness for both image classification and action recognition. During the last decade, extensive efforts have been put on the improvement of BoW and SC. However, in most situations, there are millions of local features with hundreds or even thousands of dimensions in vision-based tasks, which poses a severe restriction on the computational efficiency of similarity search in recognition algorithms. It is, therefore, highly desirable to find a compact and efficient but discriminative representation for local features.

The fast bitwise operations in Hamming space motivate us to propose a local binary representation for RGB-D video data. In this way, the similarity search is simply computing Hamming distances which are conducted by the XOR operation rather than computing Euclidean distances by the addition and multiplication in real numbers. Then the efficiency of classification and recognition algorithms will be significantly improved. Our proposed scheme is two-fold.

First, towards constructing a common representation applicable for both RGB and depth data, we view a video sequence in either RGB or depth as a scalar field in $\mathbb{R}^3$ with the frame coordinate (x,y) and the temporal axis t (for RGB data, we can use the three channels of red, green and blue to form three scalar fields in $\mathbb{R}^3$ separately. In the experiments, to alleviate the computational complexity, we only use the gray-scale information). To describe this scalar field, we compute the local flux of its gradient field and obtain a feature vector called Local Flux Feature (LFF) for each pixel. Generally speaking, the local flux $f_r(P)$ at point P is defined as the rate of the gradient field (flow) passing through a sphere surface with radius r centered at P. In other words, the local flux at point P captures the information of the orientation and the magnitude of the gradient field over a neighborhood of P, and $f_r(P)$, as a continuous function, represents an average quantity of the flow over this neighborhood. Many gradient-based features have been successfully applied to practical situations, since the gradient field represents the direction of the greatest change of a function. Theoretically, the Helmholtz theorem [4] in fluid mechanics states that we only need to know the divergence and curl of a twice continuously differentiable vector field to determine it. Given a C^2-smooth function $V(x,y,t):\mathbb{R}^3\to\mathbb{R}$, its gradient ∇V satisfies

$$\nabla \times \nabla V = (\nabla_{ty}V - \nabla_{yt}V, \nabla_{xt}V - \nabla_{tx}V, \nabla_{yx}V - \nabla_{xy}V) = \mathbf{0},$$

which means $\text{curl}(\nabla V) = \mathbf{0}$, showing that the divergence of ∇V provides the vital information for the gradient field. Fortunately, the divergence theorem converts computing the flux $f_r(P)$ through a closed sphere to computing the volume integral of the divergence inside the sphere. Obviously, computing $f_r(P)$ for every pixel is time-consuming and unnecessary. Thus we only calculate the local fluxes for the regions around the interest points or the points selected by dense sampling in RGB data and the corresponding pixels in depth data.

Second, we fuse the LFFs from RGB and depth channels of points into Hamming space. To make the above features more discriminative and meaningful in Hamming space, we propose a Structure Preserving Projection (SPP) method. Generally speaking, SPP preserves two levels of data structure. In terms of low-level features, we consider the relationship among local feature descriptors, i.e., their pairwise structure, which is maintained in the binary representation learning to embed high dimensional feature descriptors into a lower-dimensional structure-preserved Hamming space. In the learning phase, each pair of local features is given a weak label related to their Euclidean distance. Specifically, a *positive* pair is a pair of local features, if one feature of the pair is within the k nearest neighbors of the other; otherwise, it is a *negative* pair.

Considering the shape of the data distribution, the pairwise structure also includes the angles between each pair of local feature descriptors. Taking two negative pairs $(\mathbf{x}_1, \mathbf{x}_2)$ and $(\mathbf{x}_1, \mathbf{x}_3)$ as an example (since the majority of pairs are negative), they are encoded to the pairs which have large distances in the Hamming space. Nevertheless, an over-fitting condition is that pair $(\mathbf{x}_2, \mathbf{x}_3)$ is possibly mapped to the pair with a small distance as shown in Fig. 3.1. Therefore, preserving the angles can be regarded as a shape constraint for the structure of pairwise Euclidean distances. It ensures that the shape of data in the original space would not collapse in the Hamming space while pairwise distances are preserved.

Furthermore, in respect of high-level connection, we also want to establish links between samples and classes. The bipartite graph (a.k.a. bigraph) consisting of samples and classes, shows the relationship between samples and classes. To quantize the edges, we use the image-to-class (I2C) distance, which was first introduced in the naive Bayes nearest neighbor (NBNN) classifier [16] and was also proven to be an optimal distance for classification in [16]. It represents the sum of all distances from the local features of an image to their corresponding nearest neighbors in each class. Although it was proposed for image classification, it can be applied to any kind of samples represented by local feature descriptors. I2C distances can effectively avoid the quantization error in the bag-of-features model. Our algorithm shows that the performance can be enhanced by combining the sample-to-class structure (bigraph regularization) and the pairwise geometrical structure. It is worthwhile to

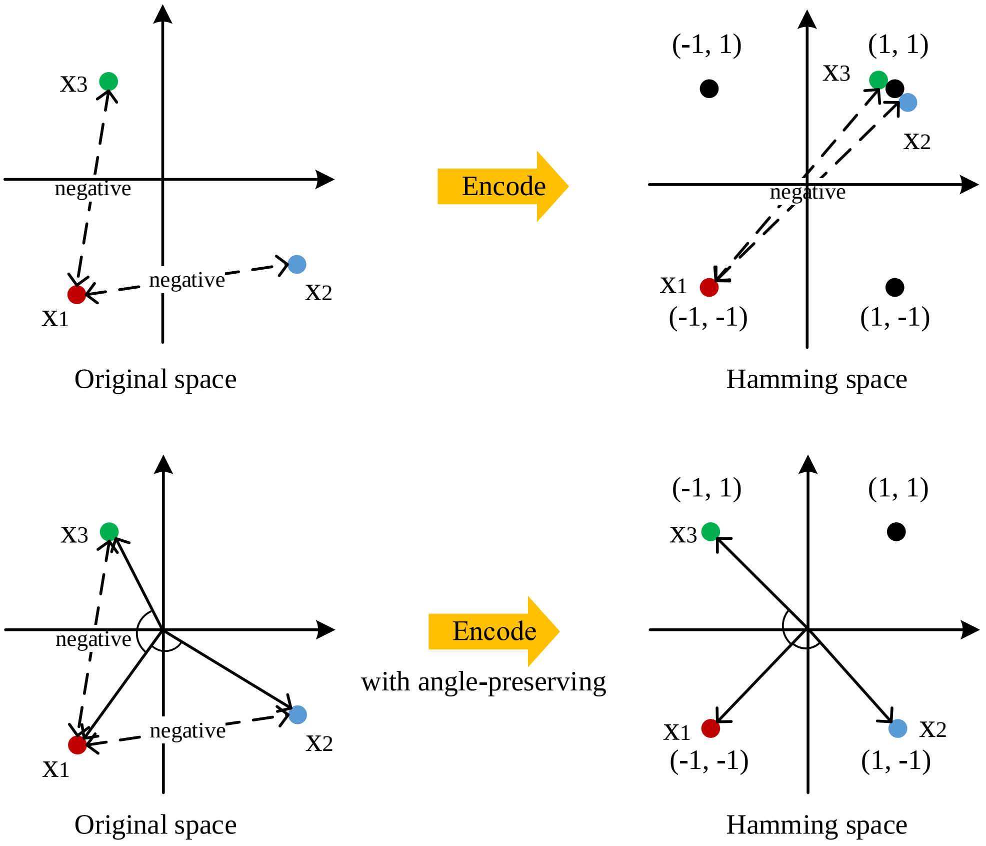


Fig. 3.1 Basic principle of the projection with angle-preserving in a two-dimensional example. The distances of two negative pairs $\|\mathbf{x}_1 - \mathbf{x}_2\|$ and $\|\mathbf{x}_1 - \mathbf{x}_3\|$ are expected to be maximized after the projection. The shape of $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ has collapsed in the Hamming space without angle-preserving, therefore, lost the discriminative ability.

highlight several properties of the proposed scheme:

- LFF is a continuous feature descriptor without loss of orientations and magnitudes of the gradient field, which makes it more suitable for the discretization of the final binary representation since every discretization will bring the deviation into results.
- SPP simultaneously preserves two independent aspects of geometrical structure: Euclidean distances and angles, which could balance each other and avoid over-fitting.
- SPP considers two levels of the relationship of data structure based on local feature descriptors. Preserving the local structure and the global structure in the original feature space makes local feature descriptors more discriminative in the lower-dimensional space.
- Our scheme fuses RGB and depth information. The fused local feature descriptors have learned the complementary nature of RGB and depth information.
- Our representation is linear and binary. This makes it extremely fast and useful for many practical applications.

3.2 Related Work

Feature extraction from RGB video data has been well explored [94, 95, 139, 187]. Detectors such as Spatio-Temporal Interest Points (STIP) [81] and Dollar's [34] are usually used to locate interest points before feature extraction. Many video descriptors are extended from their counterparts in the image domain [31, 38, 73, 103, 106]. As 3D versions of SURF [6], SIFT [103] and HOF [82], 3D speeded up robust features (SURF3D) [5], 3D scale invariant feature transforms (3D-SIFT) [134] and 3D motion features [46, 58] have been proposed for action recognition respectively. The Histogram of Oriented Gradients (HOG) is widely used in the above schemes, which discretizes the gradient orientations. In our work, however, discretization only performs in the pixel computation. Fathi et al. [37] developed a method to extract mid-level motion features by using the low-level optical flow for action recognition. Recently, the dense trajectories [162] gained high accuracies in most action recognition datasets. However, this method suffers from extremely high computational complexity. More feature extraction methods for action recognition could be found in a survey provided by Poppe [121].

Compared to the conventional RGB cameras, the depth cameras are relatively new. The existing features are specifically extracted for the depth information, since characteristics such as color and texture on depth data are far less than on the RGB data. Motion History Image (MHI) [15] is a typical template matching method for the analysis of depth information and the applications of human motion recognition [1]. Using the depth information only, Shotton et al. [140] proposed a method for human body joints analysis which is the core component of the Kinect gaming system. Nevertheless, more feature extraction methods are for the fusion with RGB information. Based on HOG, Spinello and Arras [145] proposed a method called Histogram of Oriented Depths (HOD) for depth description and probabilistically combined HOD and HOG into a Combo-HOD to detect people in urban environments. Methods in [146] and [79] simply optimize all available information in their algorithms for object detection and recognition respectively. Similarly, Ni et al. [111] designed two color-depth fusion schemes for human activity recognition. Using the depth and skeleton information of actions, Wang et al. [166] proposed a new feature called Local Occupancy Pattern (LOP) and an actionlet ensemble model which indicates a structure of features. Recently, the HON4D descriptor [115] was proposed to build the histogram of the normal unit vectors from the depth channel for activity recognition.

Apart from feature extraction, there are also many approaches to analyze actions with a temporal model. A typical one is dynamic time warping (DTW) [9], which was proposed for speech processing first. Due to the time-sequential property, DTW was also widely used as a measurement method in human action recognition for both depth data [135] and body joints of skeletons [109].

The above works are specifically designed for either RGB or depth data. In our work, LFF is a general descriptor which is suitable for both RGB and depth data. Besides, by calculating the local flux of the continuous gradient vector field, there are no bins and histograms in the computation of LFF, which can avoid the quantization error in most histogram-based methods. The Gradient Vector Flow (GVF) [175] has been successfully used in active contour alignments by solving the PDEs for an energy minimization problem. Engel et al. [36] calculated the flux flow on the GVF and adopted it for pedestrian detection. Based on the 3D vector field, a rotation invariant descriptor called 3D-Div [136] was proposed for 3D object recognition by computing the divergence of the vector field. Nonetheless, the point-wise divergence in [136] cannot capture the neighborhood information of each point. In our work, we focus on the discriminative ability of the local flux and its advantage in RGB-D action recognition.

In the aspect of hash/binary code learning, one classical method is Locality-Sensitive

Hashing (LSH) [43]. Another popular technique called Spectral Hashing (SpH) [169] was also proposed to preserve the locality information of data. Recently, a supervised method called Kernel-Based Supervised Hashing (KSH) [100] has shown good discriminative ability of binary codes and outperformed other supervised methods such as Linear Discriminant Analysis Hashing (LDAH) [148], Binary Reconstructive Embeddings (BRE) [76] and Minimal Loss Hashing (MLH) [112]. The above works mainly focus on preserving the pairwise distance, which is one part of SPP. To avoid overfitting as shown in Fig. 3.1, SPP also takes the pairwise angle into account. Towards local descriptors, Hamming Embedding (HE) [63] was proposed to map real-valued local features to binary codes. SPP contains a sample-to-class relationship when each sample is represented by a set of local descriptors, since most visual tasks are sample-oriented. Experimental results show that these three terms, i.e., the pairwise distance, the pairwise angle and the sample-to-class relationship, all contribute to the outstanding performance of the proposed method.

3.3 Local Flux Feature

Local features extracted from local regions in an image or a video sequence are used to describe the local structure of a sample. Usually, local regions are the neighborhoods of points which are determined by using an interest point detector or by dense sampling of the image plane or video volume. And then, a feature vector is computed for each local region by characterizing its properties. In our algorithm, we compute the new Local Flux Features (LFFs) from the RGB-D video data and then combine the local feature $\mathbf{x}_{RGB}$ from RGB information with the local feature $\mathbf{x}_{Depth}$ from depth information to obtain a concatenated feature vector $X \in \mathbb{R}^D$.

3.3.1 Flux Computation

The concept of flux has been studied deeply in applied physics, especially in fluid mechanics and electromagnetic theory. The flux of a vector field over a simply-connected closed district (a sphere in this chapter) is defined as the quantity of this vector field passing through the district. This quantity includes the information of the orientation and the magnitude of the vector field over the district. It is used for a description of the vector field. To describe a video sequence which is regarded as a scalar field, we consider its gradient field and compute the local flux of the gradient field.

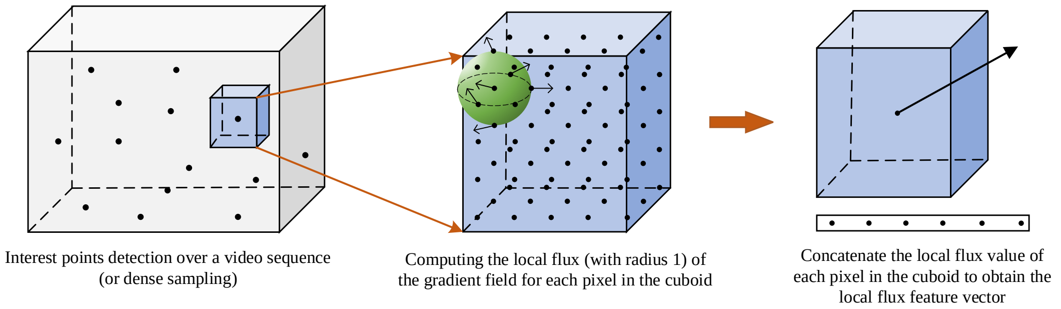

Interest points detection over a video sequence (or dense sampling)
Computing the local flux (with radius 1) of the gradient field for each pixel in the cuboid
Concatenate the local flux value of each pixel in the cuboid to obtain the local flux feature vector

Fig. 3.2 Illustration of the computation of local fluxes in the gradient field. The output LFF is regarded as a foundation for learning binary codes.

Given a video sequence $V(x,y,t)$ in either RGB[1] or depth, it can be seen as a function $V:\mathbb{R}^3\to\mathbb{R}$. We assume V is a C^2-smooth function, i.e., $V\in C^2(\Omega)$, where Ω is the district of the video sequence, usually an $L\times W\times H$ cuboid. In fact, in discrete condition, derivative computation can be regarded as an approximation by a convolution operation of matrices. Then for scalar field $V(x,y,t)$, we consider its gradient field $\nabla V(x,y,t)=(\nabla_x V,\nabla_y V,\nabla_t V)$. To describe the gradient field ∇V, we assign an $l\times w\times h$ cuboid centered at each candidate point (interest points or dense samples) and compute the local flux of every pixel (or lattice point if we regard the coordinates of a pixel as integers) in the cuboid. To be specific, denote $B_P(r)=\{(x',y',t')|(x'-x)^2+(y'-y)^2+(t'-t)^2\le r^2\}$ as the sphere with the center $P=(x,y,t)$ and radius r, the local flux at the point P over the sphere $\partial B_P(r)$ is calculated as

$$f_r(P)=\oint_{\partial B_P(r)}\nabla V\cdot \mathrm{d}\mathbf{S}, \tag{3.1}$$

where $\mathrm{d}\mathbf{S}$ represents the directed area unit of the boundary surface $\partial B_P(r)$. However, computing on the lattice points on the boundary $\partial B_P(r)$ is difficult and inaccurate. According to the divergence theorem, we have

$$\oint_{\partial B_P(r)}\nabla V\cdot \mathrm{d}\mathbf{S}=\int_{B_P(r)}\nabla\cdot\nabla V\ \mathrm{d}B_P(r), \tag{3.2}$$

i.e., we only need to compute for the points inside the sphere $B_P(r)$. Note that in the light of the Helmholtz theorem [4] in fluid mechanics, we only need to know the divergence and the curl of a twice continuously differentiable vector field to determine it. Hence, the fact that $curl(\nabla V)=\nabla\times\nabla V=\mathbf{0}$ implies that the divergence of ∇V provides the vital information,

[1] In fact, we only need the gray-scale information in our algorithm.

which is captured by the local flux $f_r(P)$. For realistic computation, we adopt the numerical approximation for the discrete condition of pixels:

$$f_r(P) = \int_{B_P(r)} \Delta V \; \mathrm{d}B_P(r) \approx \sum_{Q \in B_P(r) \cap \mathbb{Z}^3} \Delta V(Q), \tag{3.3}$$

where Δ is the Laplace operator. Suppose there are $D/2$ pixels in an $l \times w \times h$ cuboid, then we compute $D/2$ local fluxes in a specific order[2] and obtain an LFF vector $\mathbf{x} = (x_1, \cdots, x_{D/2}) \in \mathbb{R}^{D/2}$. Fig. 3.2 illustrates the outline of the computation of local fluxes. Having computed the LFF $\mathbf{x}_{RGB}$ from the RGB channel and $\mathbf{x}_{Depth}$ in the corresponding point from the depth channel, we concatenate their normalizations and obtain the new feature

$$X = \left[\frac{\mathbf{x}_{RGB}}{\|\mathbf{x}_{RGB}\|}, \frac{\mathbf{x}_{Depth}}{\|\mathbf{x}_{Depth}\|} \right]^T \in \mathbb{R}^D. \tag{3.4}$$

The combined LFF is regarded as the basic feature for the later learning of binary codes in our algorithm.

3.4 Structure Preserving Projection

In this section, we introduce our Structure Preserving Projection (SPP) algorithm. SPP simultaneously preserves the local structure and the integrated shape of local features. In addition, SPP also considers a higher level relationship among local features, i.e., the bipartite graph consisting of samples and classes. SPP aims to seek a specific matrix $\Theta \in \mathbb{R}^{D \times d}$ ($d < D$) to construct a binary function

$$H(X) = \mathrm{sgn}(\Theta^T X), \tag{3.5}$$

such that their discriminative ability for action recognition is improved. For computational convenience, we choose $\{-1, +1\}$ rather than $\{0, 1\}$ to represent binary codes in our algorithm.

[2] In the experiments, we obtain the LFF by listing the corresponding local flux values in the following pixel order: $(1,1,1), \cdots, (l,1,1), (1,2,1), \cdots, (l,2,1), \cdots, (l,w,1), \cdots, (l,w,h)$. In fact, the order has no effect on the final recognition results. The only requirement is the consistency of order in a vision task.

3.4.1 Pairwise Structure Preserving

We denote the set composed of all local features by $\mathscr{F} = \{X_1, \cdots, X_N\}$, where N is the number of local features in training data. As mentioned above, we aim to seek the binary representations with discriminative ability in the lower-dimensional space. We are concerned about the relationship between every two local features in the high-dimensional space, which should also be retained in the lower-dimensional space.

Pairwise Label

First, we assign a weak label for each pair of local features. With the pairwise labels, acquiring the class information of each local feature is unnecessary. Besides, similar local features with small Euclidean distances may appear in samples from many different classes. Motivated by the binary property of $H(X)$, we employ the pairwise label $\{-1, +1\}$ to represent the relationship between two local features based on the pairwise distance between them. Thus we have the pairwise label

$$\ell_{ij} = \begin{cases} +1, & X_i \in \mathrm{N}_k(X_j) \text{ or } X_j \in \mathrm{N}_k(X_i) \\ -1, & \text{otherwise} \end{cases},$$

where $\mathrm{N}_k(X)$ is the set of k nearest neighbors of X. To maintain the local structure, we make the product of each component in $H(X_i)$ and $H(X_j)$ consistent with their pairwise label ℓ_{ij}, i.e., $H(X_i)_m \cdot H(X_j)_m = \ell_{ij}$, $\forall m$. We denote $\mathscr{P} = \{(i,j) | X_i, X_j \in \mathscr{F}\}$. Therefore, we need to minimize the following function

$$\begin{aligned} & \sum_{(i,j)\in\mathscr{P}} \sum_{m=1}^{D} (\ell_{ij} - H(X_i)_m H(X_j)_m)^2 \\ = & \sum_{(i,j)\in\mathscr{P}} \sum_{m=1}^{D} \left(2 - 2\ell_{ij} H(X_i)_m H(X_j)_m\right) \\ = & \sum_{(i,j)\in\mathscr{P}} \left(2D - 2\ell_{ij} \sum_{m=1}^{D} H(X_i)_m H(X_j)_m\right) \\ = & \sum_{(i,j)\in\mathscr{P}} 2D - 2\ell_{ij} \langle H(X_i), H(X_j) \rangle . \end{aligned} \tag{3.6}$$

Then equivalently, we only need to maximize

$$\sum_{(i,j)\in\mathscr{P}} \ell_{ij}\langle H(X_i), H(X_j)\rangle. \tag{3.7}$$

The above function reaches its maximum value when $\ell_{ij}\text{sgn}(\Theta^T X_i)$ and $\text{sgn}(\Theta^T X_j)$ are similarly sorted due to the rearrangement inequality [51]. In other words, if $\ell_{ij} = 1$, X_i and X_j are then similarly encoded and vice versa.

Considering the effect of noise, we additionally assign a pairwise weight W_{ij}^P to the local feature pair (i, j) to avoid the disturbance:

$$W_{ij}^P = \exp\left(-l_{ij}\|X_i - X_j\|^2\right). \tag{3.8}$$

Then the objective function for pairwise labels becomes

$$\sum_{(i,j)\in\mathscr{P}} W_{ij}^P \ell_{ij}\langle H(X_i), H(X_j)\rangle. \tag{3.9}$$

Pairwise Angle

In addition to the distance factor, we are also concerned about the shape of the entire set of local features, which is regarded as a constraint for preserving the pairwise Euclidean distances. The shape constraint firms the data structure in the projected space and avoids some certain errors caused by the pairwise labels. We denote the angle between two local features X_i and X_j by θ_{ij}. Note that angle θ_{ij} is with the vertex at coordinate origin. Thus, the local features should be *centralized* before the further learning process. Orthogonal transformation ($d = D$ and $\Theta^T\Theta = \Theta\Theta^T = I$) preserves the lengths of local features and the angles between them since we have $\langle\Theta^T X_i, \Theta^T X_j\rangle = X_i^T\Theta\Theta^T X_j = X_i^T X_j = \langle X_i, X_j\rangle$, $\forall i, j$. When $d < D$, however, this property does not hold in orthogonal projection. We hope the angle $\widehat{\theta}_{ij}$ in the projected space[3] is (approximately) equal to θ_{ij}. Note that the distances are irrelevant with the angles, i.e., the pair of local features with a long distance can have a small angle and the pair with a short distance may have a large angle. Thus it is desirable to retain the angles of all pairs. We define our optimization problem for angle preserving in

[3] Since Hamming space is a discrete space, we first consider the angles in the linear subspace before taking the sign function.

the low dimensional space:

$$\arg\max_{\Theta} \sum_{(i,j)\in\mathscr{P}} \langle X_i, X_j\rangle \cdot \langle \Theta^T X_i, \Theta^T X_j\rangle. \tag{3.10}$$

Although it is the optimization for preserving the inner product, the following proposition shows that the optimal Θ^* preserves the pairwise angles.

Proposition 2 *Suppose Θ^* is the optimal solution of the optimization problem (3.10), then for any $1 \leq i, j \leq N$, the projection Θ^* preserves the angle between the local features X_i and X_j.*

Proof. According to the Cauchy-Schwarz inequality, we have

$$\sum_{(i,j)\in\mathscr{P}} \langle X_i, X_j\rangle \cdot \langle \Theta^T X_i, \Theta^T X_j\rangle \leq \left(\sum_{(i,j)\in\mathscr{P}} \langle X_i, X_j\rangle^2 \right)^{\frac{1}{2}} \left(\sum_{(i,j)\in\mathscr{P}} \langle \Theta^T X_i, \Theta^T X_j\rangle^2 \right)^{\frac{1}{2}},$$

and the equality holds if and only if $\langle X_i, X_j\rangle$ $((i,j) \in \mathscr{P})$ and $\langle \Theta^T X_i, \Theta^T X_j\rangle$ $((i,j) \in \mathscr{P})$ are collinear. We can first set a norm constraint $\sum_{(i,j)\in\mathscr{P}} \langle \Theta^T X_i, \Theta^T X_j\rangle^2 = 1$ for Θ. Then the objective function in Eq. (3.10) is smaller than a constant. If Θ^* is the optimal solution of the optimization problem (3.10), the left-hand-side of the above inequality reaches its maximum value at Θ^*. Then there exists a constant $\lambda \in \mathbb{R}$ such that

$$\frac{\langle (\Theta^*)^T X_i, (\Theta^*)^T X_j\rangle}{\langle X_i, X_j\rangle} = \lambda,\ \forall (i,j) \in \mathscr{P}.$$

Since for $i = j$, we have $\|(\Theta^*)^T X_i\| = \lambda \|X_i\|$, then $\lambda > 0$. Therefore, for the projected angle $\widehat{\theta}_{ij}$, it satisfies

$$\begin{aligned}
\cos\widehat{\theta}_{ij} &= \frac{\langle(\Theta^*)^T X_i, (\Theta^*)^T X_j\rangle}{\|(\Theta^*)^T X_i\|\|(\Theta^*)^T X_j\|} \\
&= \frac{\langle(\Theta^*)^T X_i, (\Theta^*)^T X_j\rangle}{\sqrt{\langle(\Theta^*)^T X_i, (\Theta^*)^T X_i\rangle}\sqrt{\langle(\Theta^*)^T X_j, (\Theta^*)^T X_j\rangle}} \\
&= \frac{\lambda\langle X_i, X_j\rangle}{\sqrt{\lambda\langle X_i, X_i\rangle}\sqrt{\lambda\langle X_j, X_j\rangle}} \\
&= \frac{\langle X_i, X_j\rangle}{\sqrt{\langle X_i, X_i\rangle}\sqrt{\langle X_j, X_j\rangle}} \\
&= \frac{\langle X_i, X_j\rangle}{\|X_i\|\|X_j\|} \\
&= \cos\theta_{ij},
\end{aligned}$$

which implies that the projection matrix Θ^* is an angle-preserving projection.

3.4.2 Bigraph Regularization

Not only the pairwise structure of local features, but also the connection between samples and classes, which is regarded as a higher level relationship among local features, is considered in our algorithm. We use the image-to-class (I2C) distance to measure the bipartite graph (a.k.a. bigraph) that consists of video samples and classes. Although the I2C distance was first introduced to measure the distances between images and classes, it can also be applied to all kinds of samples that are represented by local features. Our goal is to preserve the I2C distances in the lower-dimensional space. Given the set of local features of a sample $\mathscr{X}_i = \{X_{i1}, \cdots, X_{im_i}\}$, which contains all local features of sample i, the distance between sample i and class c is defined as

$$I^c_{\mathscr{X}_i} = \sum_{X \in \mathscr{X}_i} \|X - \mathrm{NN}^c(X)\|^2, \tag{3.11}$$

where $\mathrm{NN}^c(X)$ is the nearest neighbor (NN) of the local feature X in class c and $\|\cdot\|$ is the L_2-norm.

However, the complexity of NN-search linearly depends on the number of local features, which renders the nearest neighbor search in such a large-scale space of local features of each class will still cost much time. Hence, we first implement a K-means clustering algorithm for each class. In other words, we first find K centroids for each set $\bigcup_{C(\mathscr{X}_i)=c}\mathscr{X}_i$, $c=1,\cdots,C$, where C is the number of classes and $C(\cdot)\in\{1,\cdots,C\}$ is the label information function that represents the class label of the input. In this way, the searching range of nearest neighbors is reduced to the set of cluster centers, which has a much smaller size than the original space, i.e., for $c=1,\cdots,C$, we set

$$\mathrm{NN}^c(X)\in \text{Centroids }\{S_1,\cdots,S_K\}\text{ of }\bigcup_{C(\mathscr{X}_i)=c}\mathscr{X}_i.$$

Having obtained I2C distances, we build a bigraph $G=(V_1,V_2,E)$, where V_1 and V_2 are the node sets of samples and classes respectively. G is a complete and weighted bigraph. For each edge in E connecting sample i and class c, it has the weight W_{ic}^{D} determined by the I2C distance, named the I2C similarity. By heat kernel, we define the I2C similarity as follows:

$$W_{ic}^{I2C}=\exp\left(-(I_{\mathscr{X}_i}^{c})^2/\sigma\right),\ i=1,\cdots,n,\ c=1,\cdots,C, \tag{3.12}$$

where σ is the Gaussian smoothing parameter and n is the number of training samples. Correspondingly, we have the I2C distance in the objective Hamming space:

$$\widehat{I}_{\mathscr{X}_i}^{c}=\sum_{X\in\mathscr{X}_i}||\mathrm{H}(X)-\mathrm{NN}^c(\mathrm{H}(X))||^2. \tag{3.13}$$

With the above defined I2C similarity W_{ic}^{I2C} and the projected I2C distance $\widehat{I}_{\mathscr{X}_i}^{c}$, we can define the following optimization problem to quantize the bigraph regularization, i.e., I2C structure in the low dimensional space:

$$\arg\min_{\Theta}\sum_{i=1}^{n}\sum_{c=1}^{C}\widehat{I}_{\mathscr{X}_i}^{c}\cdot W_{ic}^{I2C}. \tag{3.14}$$

By minimizing the above equation, the sample which has a small I2C distance to class c in the high dimensional space is still close to class c in the low dimensional space. According to the rearrangement inequality [51], the above objective function reaches its minimum value if and only if $\{\widehat{I}_{\mathscr{X}_i}^{c}\}$ and $\{I_{\mathscr{X}_i}^{c}\}$ are similarly sorted, which means the projected I2C distances preserve the bigraph structure in the high dimensional space.

3.4.3 Objective Function and Optimization

In addition, to make the projected space more compact, we set the orthogonality constraint on the projection matrix, i.e., $\Theta^T\Theta = I$. Combining the objective functions for the pairwise structure and the bigraph regularizer, we obtain our final optimization problem for SPP:

$$\begin{aligned}\underset{\Theta^T\Theta=I}{\arg\max} \sum_{(i,j)\in\mathscr{P}} W_{ij}^P \ell_{ij}\langle H(X_i), H(X_j)\rangle + \sum_{(i,j)\in\mathscr{P}} \langle X_i, X_j\rangle \cdot \langle \Theta^T X_i, \Theta^T X_j\rangle \\ -\beta \sum_{i=1}^{n}\sum_{c=1}^{C} \widehat{I}_{\mathscr{X}_i}^c \cdot W_{ic}^{I2C},\end{aligned} \tag{3.15}$$

where β is the regularization parameter.

Optimization: Considering the discreteness of the binary function, we first use approximation $\mathrm{sgn}(x) \approx x$ to relax the objective function in the optimization problem (4.15) into a real-valued space. Then the objective function of the pairwise label part (see Eq. (3.9)) becomes

$$\begin{aligned}&\sum_{(i,j)\in\mathscr{P}} W_{ij}^P \ell_{ij}\langle H(X_i), H(X_j)\rangle \\ =& \sum_{(i,j)\in\mathscr{P}} W_{ij}^P \ell_{ij}\langle \mathrm{sgn}(\Theta^T X_i), \mathrm{sgn}(\Theta^T X_j)\rangle \\ \approx& \sum_{(i,j)\in\mathscr{P}} W_{ij}^P \ell_{ij}\langle \Theta^T X_i, \Theta^T X_j\rangle \\ =& \sum_{(i,j)\in\mathscr{P}} W_{ij}^P \ell_{ij}\,\mathrm{tr}(\Theta^T X_i(\Theta^T X_j)^T) \\ =& \sum_{(i,j)\in\mathscr{P}} W_{ij}^P \ell_{ij}\,\mathrm{tr}(\Theta^T X_i X_j^T \Theta).\end{aligned}$$

And for I2C distances, we denote $\mathrm{NN}^c(X) = X^c$. Note that after applying projection matrix Θ, the nearest neighbors may change. However, for the large-scale local feature space, we approximately adopt the sum of the distances from $\Theta^T X$ to the projected nearest neighbor

$\Theta^T X^c$. Then the projected I2C distance (see Eq. (3.13)) after applying matrix Θ becomes

$$\begin{aligned}
\widehat{I}^c_{\mathscr{X}_i} &\approx \sum_{X \in \mathscr{X}_i} \|\Theta^T X - \Theta^T X^c\|^2 \\
&= \sum_{X \in \mathscr{X}_i} \|\Theta^T (X - X^c)\|^2 \\
&= \sum_{k=1}^{m_i} \mathrm{tr}(\Theta^T (X_{ik} - X^c_{ik})(\Theta^T (X_{ik} - X^c_{ik}))^T) \\
&= \sum_{k=1}^{m_i} \mathrm{tr}(\Theta^T (X_{ik} - X^c_{ik})(X_{ik} - X^c_{ik})^T \Theta) \\
&=: \sum_{k=1}^{m_i} \mathrm{tr}(\Theta^T \Delta X^c_{ik} (\Delta X^c_{ik})^T \Theta),
\end{aligned}$$

where $\Delta X^c_{ik} = X_{ik} - X^c_{ik}$, $k = 1, \cdots, m_i$. Thus, by simple algebraic derivation, the optimization problem (4.15) is reduced to

$$\arg\max_{\Theta^T \Theta = I} \mathrm{tr}(\Theta^T M \Theta), \tag{3.16}$$

where

$$M = \sum_{(i,j) \in \mathscr{P}} (W^P_{ij} \ell_{ij} + \langle X_i, X_j \rangle) X_i X_j^T - \beta \sum_{i=1}^{n} \sum_{c=1}^{C} \sum_{j=1}^{m_i} W^{I2C}_{ic} \Delta X_{ij} \Delta X_{ij}^T. \tag{3.17}$$

Notice that $W^P_{ij} \ell_{ij} + \langle X_i, X_j \rangle = W^P_{ji} \ell_{ji} + \langle X_j, X_i \rangle$, $\forall i, j$, then we have

$$\begin{aligned}
M = &\sum_{1 \le i < j \le N} (W^P_{ij} \ell_{ij} + \langle X_i, X_j \rangle)(X_i X_j^T + X_j X_i^T) + \sum_{i=1}^{N} (W^P_{ii} \ell_{ii} + \langle X_i, X_i \rangle) X_i X_i^T \\
&- \beta \sum_{i=1}^{n} \sum_{c=1}^{C} \sum_{j=1}^{m_i} W^{I2C}_{ic} \Delta X_{ij} \Delta X_{ij}^T.
\end{aligned}$$

Thus M is a real-valued symmetric matrix. It is clear that the solution to the optimization problem (3.16) is the eigenvectors corresponding to the largest d eigenvalues of M. We summarize our algorithm in the following Algorithm 3.

3.4.4 Complexity Analysis

In this section, we provide a time complexity analysis of our algorithm. During the training phase, our algorithm mainly consists of three parts. The first part is the computation of LFFs. The derivative computation is actually the convolution of matrices which at most needs

Algorithm 3 Structure Preserving Projection for Local Flux Feature

```
Input: Training video sequences $V_1, \cdots, V_n$ in gray-scale and $V'_1, \cdots, V'_n$ in depth, the radius $r$ for the
    sphere $B_P(r)$, the parameter $k$ for pairwise structure preserving, the number of centroids $K$ in
    K-means, the label information function $C(\cdot) \in \{1, \cdots, C\}$, the regularization parameter $\beta$ and
    the objective dimension $d$.
Output: The projection matrix $\Theta$.
  Detect interest points (or densely sample) $\{P_1, \cdots, P_{m_i}\}$ from the $i$-th training video $V_i$, $i =
    1, \cdots, n$;
  Compute two LFFs for each point in gray-scale and depth respectively by Eq. (3.3) and combine
    them by Eq. (3.4) to obtain the local feature set of the $i$-th training video $\mathscr{X}_i = \{X_{i1}, \cdots, X_{im_i}\}$
    and the whole local feature set $\mathscr{F} = \bigcup \mathscr{X}_i = \{X_1, \cdots, X_N\}$;
  Centralize $X_i \leftarrow \frac{1}{N}\sum_{j=1}^{N} X_j$, $\forall i$;
  Construct local feature pairing set $\mathscr{P} = \{(i,j) | X_i, X_j \in \mathscr{F}\}$ and their corresponding pairwise
    labels $\ell_{ij} = \{-1, +1\}$, where $\ell_{ij} = +1$ if $X_i \in \mathrm{N}_k(X_j)$ or $X_j \in \mathrm{N}_k(X_i)$, and $\ell_{ij} = -1$ otherwise;
  Employ the K-means clustering algorithm on the set of local features of each class $\bigcup_{C(\mathscr{X}_i)=c} \mathscr{X}_i$,
    $c = 1, \cdots, C$;
  Compute pairwise weight $W_{ij}^{P}$ and I2C similarity $W_{ic}^{I2C}$ by Eqs. (3.8) and (3.12);
  Compute the matrix $M$ by Eq. (3.17);
  return the eigenvectors corresponding to the largest $d$ eigenvalues of $M$.
```

$O(3DL_m \log L_m)$ time [2], where $L_m = \max\{L, W, H\}$. The second part is the computation of pairwise structure preserving. The k-NN algorithm in the construction of pairwise labels and the computation of pairwise angles cost $O(kN^2)$ and $O(N^2)$ time respectively. The last part is the construction of the I2C similarity matrix $\left(W_{ic}^{I2C}\right)$. The time complexity of this part is $O(nCKDN)$. In total, the time complexity of the training phase is at most $O(3DL_m \log L_m) + O((k+1)N^2) + O(nCKDN)$.

In the test phase, binary codes can significantly reduce the runtime of the recognition algorithm since the distance computation in Hamming space is simply based on the XOR operation. Denote τ_m and τ_{XOR} as the time of one multiplication and one XOR operation respectively. Then the computational complexity of NBNN in the original space is $O(N_{train}N_{test}D)\tau_m$, where N_{train} and N_{test} are the numbers of local features in training and test sets respectively. With the binary local features, the time complexity is reduced to $O(N_{train}N_{test}d)\tau_{XOR}$. In general, we have $d \ll D$ and $\tau_{XOR} \ll \tau_m$. Thereby, when N_{train} and N_{test} are in the magnitude of millions or even greater, the hashing algorithm's effect is self-evident. We will list the run-time in the following section.

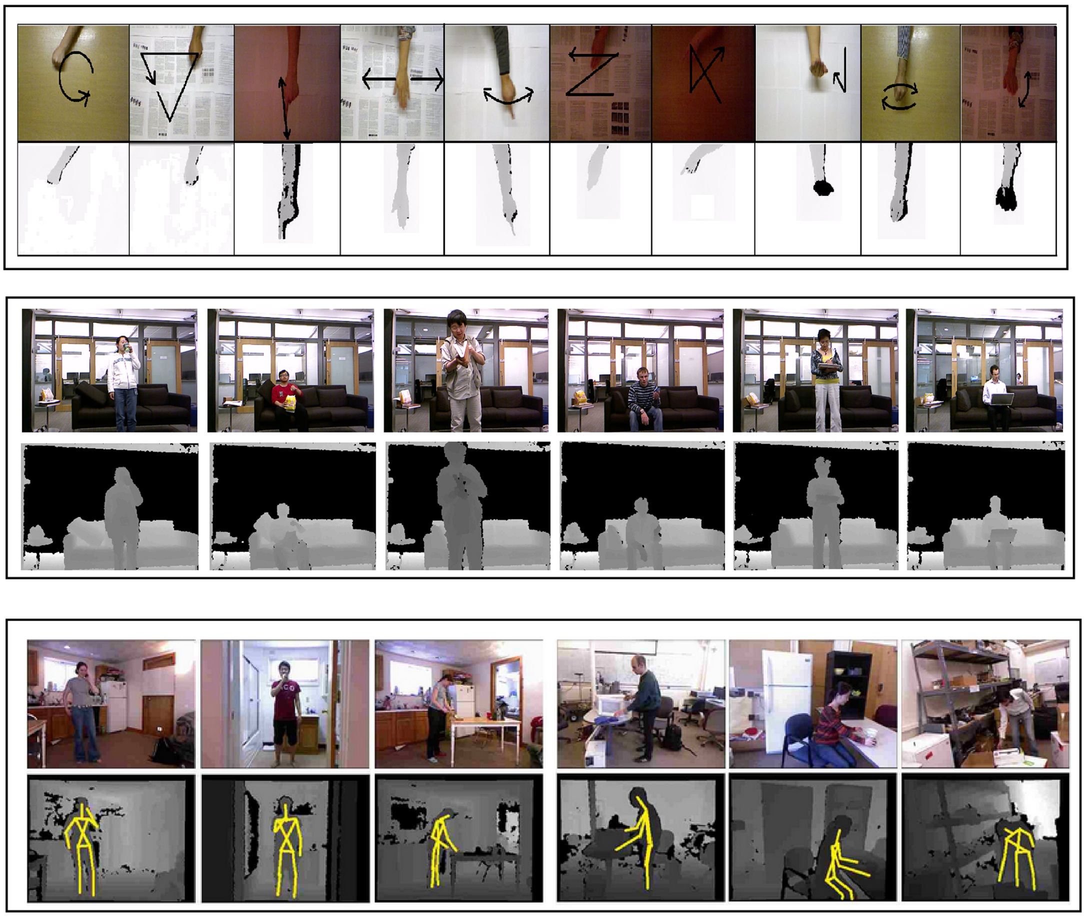


Fig. 3.3 Example frames of the three RGB-D datasets we used in the experiments. From top to bottom: SKIG, MSRDailyActivity3D and CAD-60.

3.5 Experiments and Results

In this section, we systematically evaluate our proposed method on three different RGB-D benchmarks: the SKIG hand gesture dataset [92], the MSRDailyActivity3D dataset [166] and the CAD-60 activity dataset [151]. Fig. 3.3 shows some example frames of these three datasets. Details of the datasets are introduced in the following subsection.

3.5.1 Datasets and Settings

The **SKIG** dataset has 2160 hand gesture sequences (1080 RGB sequences and 1080 depth sequences) collected from 6 subjects. All these sequences are synchronously captured with a Kinect sensor (including a RGB camera and a depth camera). This dataset collects 10 categories of hand gestures in total: *circle (clockwise), triangle (anti-clockwise), up-down, right-left, wave, "Z", cross, comehere, turnaround and pat.* In the collection process, all these ten categories are performed with three hand postures: fist, index and flat. To increase the diversity, the sequences are recorded under 3 different backgrounds (i.e., wooden board, white plain paper and paper with characters) and 2 illumination conditions (i.e., strong light and poor light). Consequently, for each subject, there are $10(categories) \times 3(poses) \times 3(backgrounds) \times 2(illumination) \times 2(RGB\ and\ depth) = 360$ gesture sequences. The training size for each category is varied as one of $\{10, 20, 35, 45, 60, 70\}$ and the rest of the sequences are used for testing.

The **MSRDailyActivity3D** dataset is a human activity dataset captured with the RGB channel and the depth channel using the Kinect sensor. The total sequence number is 640 (i.e., 320 sequences for each channel) with 16 activities: *drink, eat, read book, call cellphone, write on a paper, use laptop, use vacuum cleaner, cheer up, sit still, toss paper, play game, lie down on sofa, walk, play guitar, stand up, sit down.* There are 10 subjects in the dataset and each subject performs each activity twice, once in standing position, and once in sitting position. The training size for each subject is chosen as one of $\{5, 10, 15, 20, 25\}$ and the rest is used for testing.

The **Cornell Activity** dataset (CAD-60) contains 60 RGB-depth sequences acted by four subjects and captured with a Kinect camera. The actions in this dataset are categorized into five different environments: office, kitchen, bedroom, bathroom, and living room. Three or four common activities were identified for each environment, giving a total of twelve unique actions: rinsing mouth, brushing teeth, wearing contact lens, drinking water, opening pill container, cooking (chopping), cooking (stirring), talking on couch, relaxing on couch,

Table 3.1 Performance comparison (%) of NBNN with the LFFs computed on detected points with different radii. The training sizes are 70, 25 and 4 in each class for SKIG, MSRDailyActivity3D and CAD-60, respectively. All the code lengths are 96-bit.

	$r=1$	$r=2$	$r=3$	$r=4$	$r=5$	$r=6$	$r=7$	$r=8$	$r=9$	$r=10$
SKIG	88.5	90.3	92.4	**93.7**	93.1	92.5	91.6	91.2	90.7	88.4
MSRDailyActivity3D	85.7	86.2	88.7	**89.8**	88.9	88.1	87.6	87.5	86.7	85.2
CAD-60	93.2	94.1	94.9	95.2	**95.7**	94.8	94.1	93.5	92.2	90.8

talking on the phone, writing on whiteboard, working on computer. The training size for each action is assigned as one of $\{1,2,3,4\}$ and remaining sequences are adopted for testing.

All the training samples are selected randomly from every class in each dataset and all the procedures are repeated five times. We report the averages as the final results.

For the experimental settings, we fix the size of the cuboid $l \times w \times h$ in the computation of LFF as $7 \times 7 \times 9$. We set $r = 4,4,5$ in each dataset respectively due to the comparison results with different radii r in Table 3.1. If the radius r is too small, the LFF degenerates to the second order derivative, and if r is too big, LFFs are almost the same for adjacent pixels, which tends to be less discriminative. In addition, the computational cost of LFF is proportional to r^3 because of the volume integral in Eq. (3.3). Thus, the selection of r should also be in the range of small numbers. We always set $k = 15$ for the pairwise data structure. Actually, we utilize the training data as the cross-validation set in SPP. The parameter K of the K-means is selected from one of $\{100,200,\cdots,1000\}$ with the step of 100 , which yields the best performance by 10-fold cross-validation. The optimal parameter β is selected from $\{0.1,0.2,\cdots,1.0\}$ with the step of 0.1 by 10-fold cross-validation on the cross-validation set, as well. In particular, the nested cross-validation strategy is applied to these two parameters, i.e., K and β. We always first fix the value of K as one of $\{100,200,\cdots,1000\}$ and select the best parameter β from $\{0.1,0.2,\cdots,1.0\}$, and then assign another value to K and select the best parameter β from $\{0.1,0.2,\cdots,1.0\}$ again. In this way, the optimal pair of parameters K and β can be obtained under the nested cross-validation strategy.

Since the acceleration of NBNN is quite conspicuous using the Hamming distance instead of the L_2-norm in the NN-search and NBNN classifier always outperforms the BoW model, we mainly use NBNN to evaluate our recognition precision.

3.5.2 Compared Results

First of all, we illustrate the effectiveness of all the three terms used in SPP, i.e., the pairwise label preserving term, the pairwise angle preserving term and the bigraph regularization. We

Table 3.2 Performance comparison (%) of different variants of LFF+SPP to prove the effectiveness of the improvement on RGB-D fusion. All the code lengths are 96-bit. The bold numbers represent the best performance for each dataset.

label preserving	angle preserving	bigraph regularization	Datasets Methods	**SKIG**	**MSRDaily Activity3D**	**CAD-60**
✓	✓		LFF+SPP^1	85.1	82.4	90.4
	✓	✓	LFF+SPP^2	89.6	83.1	93.5
✓		✓	LFF+SPP^3	91.2	85.8	94.2
✓	✓	✓	**LFF+SPP**	**93.7**	**89.8**	**95.7**

(SPP^1 is the original SPP without the bigraph regularization; SPP^2 denotes the original SPP without the pairwise label preserving term; SPP^3 represents the original SPP without the pairwise angle preserving term.)

remove one of them and keep the other two terms, and optimize the problem in (4.15). The results are listed in Table 3.2, from which we can observe that the bigraph regularization contributes the most to the accuracies.

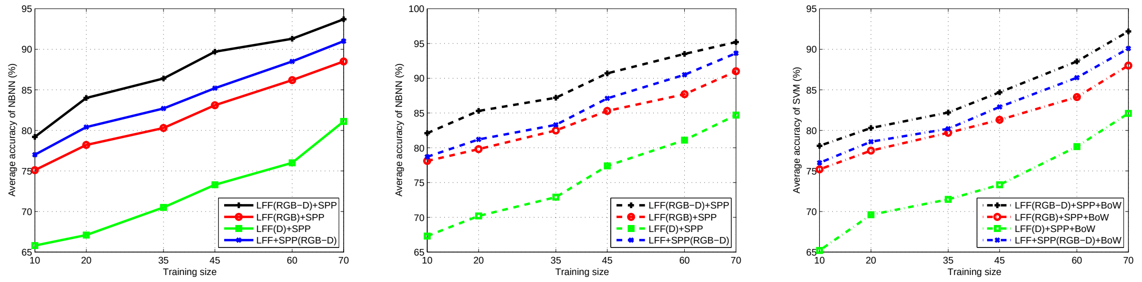
Next, for all three datasets, we apply three different schemes to achieve RGB-D video classification: (1) Detected interest points[4] + LFF + SPP; (2) Dense sampling[5] + LFF + SPP; (3) Detected interest points + LFF + SPP + Bag-of-Words. For (1) and (2), we adopt NBNN as the classifier and the linear SVM is applied for the third scheme for classification. The codebook lengths of BoW for each dataset are chosen as one of $\{500, 1000, 1500, 2000\}$ and the best results are reported.

For each scheme, we apply SPP on LFFs from RGB and depth information. According to all the possible combinations, we evaluate four different kind of local binary codes on three datasets: LFF(RGB-D)+SPP denotes our full algorithm; LFF(RGB)+SPP only uses RGB information to compute LFFs and then apply SPP; LFF(D)+SPP only uses depth information to compute LFFs and then apply SPP; LFF+SPP(RGB-D) concatenates LFF(RGB)+SPP and LFF(D)+SPP.

From Figs. 3.4–3.6, we can observe that the performance of our full algorithm is consistently higher than that of other versions on the three datasets. And dense sampling generally outperforms interest points detection due to the large amount of local feature descriptors. Another observation is that LFF(RGB-D)+SPP always outperforms LFF+SPP(RGB-D), since the former outputs the fused binary representation with the consideration of the structures of RGB-D features. In contrast, LFF+SPP(RGB-D) outputs binary codes separately for RGB and depth features, therefore, loses the connection between them.

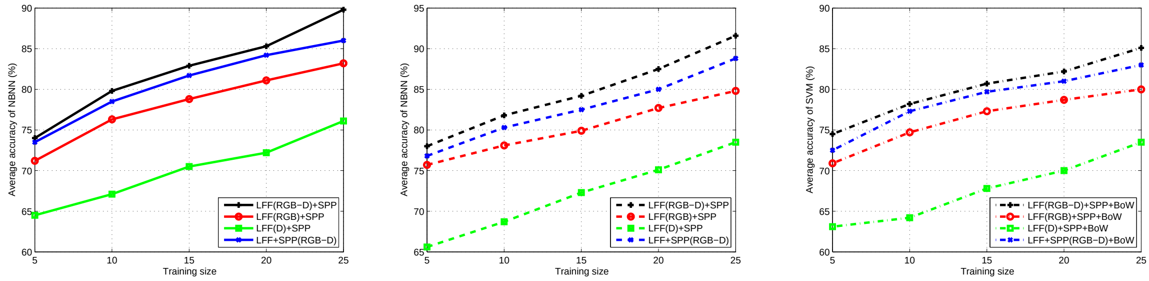
[4] Dollar's interest points detector [34] is used in our experiments. We only detect the interest points on the RGB data and find the corresponding locations on the depth video as the detected points for depth data.

[5] We set the distance between adjacent pixels as 5.



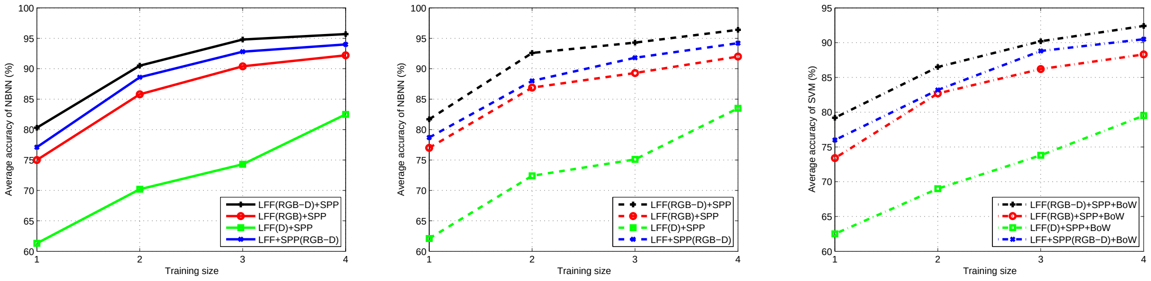
(a) Dollar's detector (b) Dense sampling (c) SVM with BoW

Fig. 3.4 Performance comparison with different training sizes in each category and different versions of LFFs on the SKIG dataset at 96-bit.



(a) Dollar's detector (b) Dense sampling (c) SVM with the BoW model

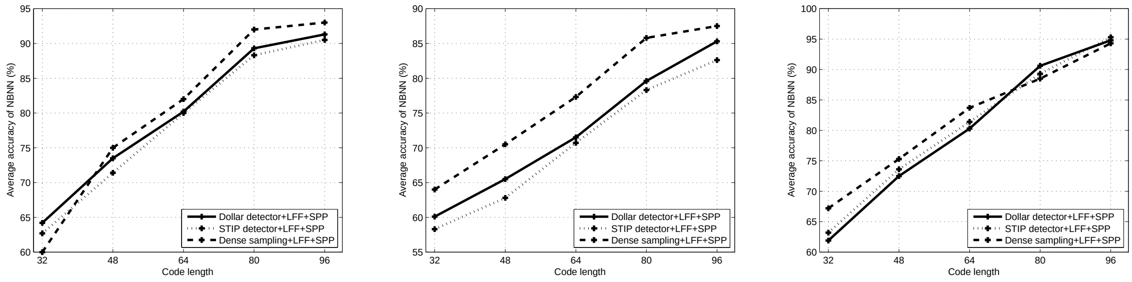
Fig. 3.5 Performance comparison with different training sizes for each subject and different versions of LFFs on the MSRDailyActivity3D dataset at 96-bit.



(a) Dollar's detector (b) Dense sampling (c) SVM with the BoW model

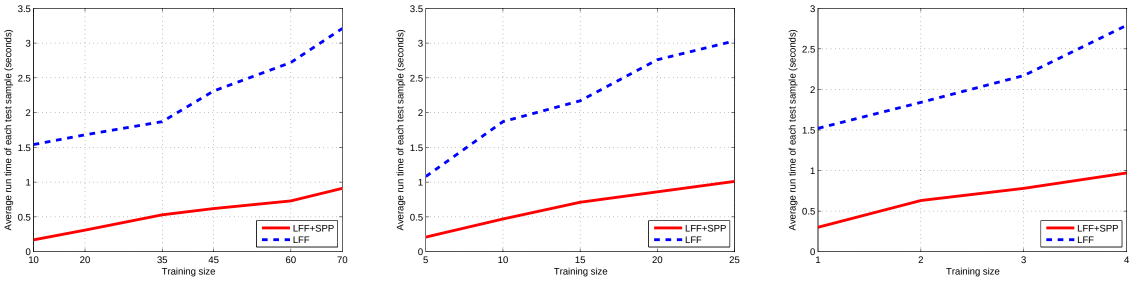
Fig. 3.6 Performance comparison with different training sizes in each action and different versions of LFFs on the CAD-60 dataset at 96-bit.

In Fig. 3.7, we also compare the performance of our algorithm with different code lengths by using different point selection methods, i.e., interest points detection (Dollar's detector and STIP) and dense sampling, on the three datasets. It is noticeable that, on the CAD-60 dataset, the accuracy of dense sampling is slightly lower than that of interest points



(a) SKIG (training size 60) (b) MSRDailyActivity3D (training size 20) (c) CAD-60 (training size 3)

Fig. 3.7 Performance comparison of NBNN with different point selection methods on three datasets.



(a) SKIG (b) MSRDailyActivity3D (c) CAD-60

Fig. 3.8 Average runtime of one test sample of NBNN by using 96-bit binary codes after SPP and the original 882-dimensional LFF with different training sizes.

detection because the noise of the background has a negative effect on the dense sampling when the code length increases. In this situation, the detection method is more effective than dense sampling.

Finally, Fig. 5.6 shows the average runtime comparison. Our learned binary codes show a significant advantage compared to the original LFF consisting of real numbers since NBNN largely depends on NN-search. All the experiments are conducted using Matlab 2013a on a server configured with a 12-core processor and 128G of RAM running the Linux OS.

3.5.3 Comparison with Other Methods

In Table 3.3, we first compare the proposed LFF descriptor with state-of-the-art video descriptors (i.e., HOG, HOF, MBH, HON4D and HOG3D) for RGB-D action recognition. All the methods are computed on the interest points from the RGB channel detected by Dollar's

Table 3.3 Performance comparison (%) of our algorithm and other coding methods on three datasets. In the RGB-D fusion scheme, we first concatenate features in RGB and depth, then apply hashing methods. In the RGB-D concatenation (Cat) scheme, we first apply hashing methods to features in RGB and depth separately, then concatenate them. The bold numbers represent the best performance for each dataset.

Methods	**SKIG**				**MSRDailyActivity3D**				**CAD-60**			
	RGB Channel	**Depth Channel**	**RGB-D Cat**	**RGB-D Fusion**	**RGB Channel**	**Depth Channel**	**RGB-D Cat**	**RGB-D Fusion**	**RGB Channel**	**Depth Channel**	**RGB-D Cat**	**RGB-D Fusion**
HOG	81.4	72.7	82.9	-	76.4	62.3	79.2	-	78.4	60.3	79.6	-
HOF	79.0	71.2	80.6	-	75.6	62.2	78.9	-	77.0	58.5	77.8	-
MBH	82.1	74.7	83.2	-	76.7	63.1	80.1	-	79.5	61.2	81.8	-
HON4D	-	80.1	-	-	-	78.4	-	-	-	69.2	-	-
HOG3D	81.8	73.4	83.1	-	77.2	62.4	79.5	-	78.5	60.4	80.5	-
LFF	84.0	76.2	85.4	-	80.6	72.8	81.6	-	81.0	63.6	83.2	-
Action ensemble*	-	-	-	-	-	87.6	-	-	-	91.8	-	-
HOG3D+IFV	86.9	79.8	89.7	92.1	83.1	75.1	85.6	89.5	91.0	80.8	92.4	94.8
LFF+IFV	88.7	80.5	91.5	93.2	84.8	76.0	87.4	**91.1**	91.4	82.0	93.4	95.1
HOG3D+SPP	86.3	78.6	88.2	91.4	84.3	71.5	85.2	87.4	88.1	67.4	92.1	93.0
LFF+SPP	88.5	81.1	91.0	**93.7**	83.2	76.1	86.0	89.8	92.2	82.5	94.0	**95.7**
LFF+KSH	81.7	67.9	82.4	80.1	80.1	72.1	82.5	81.0	76.0	52.5	77.2	76.8
LFF+BRE	79.8	63.4	80.2	80.8	78.1	68.1	81.3	79.8	75.5	56.7	76.0	76.1
LFF+MLH	77.5	63.8	78.4	78.8	74.2	69.3	75.0	76.2	75.3	48.6	75.8	74.7
LFF+LSH	69.4	54.2	71.4	68.2	60.5	41.1	62.3	58.4	61.4	30.7	62.5	60.2
LFF+SpH	77.5	68.1	78.5	79.0	76.2	60.7	78.3	78.2	70.7	50.4	71.3	73.1
LFF+AGH	74.2	70.5	77.4	78.3	77.5	63.2	78.4	79.5	73.6	48.2	74.7	74.0
LFF+PCAH	68.0	60.2	68.3	60.4	61.3	48.7	63.0	62.1	65.3	41.0	67.9	60.1
LFF+BSSC	77.8	55.4	80.3	81.3	76.9	65.3	76.7	78.0	74.2	48.2	76.8	77.2
LFF+RBM	78.5	67.6	79.5	79.7	77.2	60.0	78.3	78.5	77.4	58.3	79.7	78.8

$*$ The action ensemble method adopted the depth and skeleton information with real-valued features. The skeleton information is only available in MSRDailyActivity3D and CAD-60.

All the results (except action ensemble, LFF+IFV and HOG3D+IFV) are calculated by the NBNN classifier. The linear SVM is applied to LFF+IFV and HOG3D+IFV.

detector and the corresponding points from the depth channel. As we can see, LFF outperforms HOG, HOF, MBH and HOG3D in the RGB and depth channels and the RGB-D concatenation scheme. Although HON4D, as a descriptor specifically designed for depth sequences, achieves better performance in the depth channel, it can only be extracted from depth data and the recognition accuracies are relatively low. In contrast, our LFF is considered to be a general feature descriptor for both RGB and depth data and LFF in the RGB-D concatenation scheme reaches the highest accuracy in the experiment of feature comparison.

Since SPP is a projection for learning binary codes, we can also compare our SPP algorithm with other hashing methods. In our experiments, we compare the proposed method against seven general hashing algorithms including KSH [100], BRE [76], MLH [112], LSH [43], SpH [169], AGH [101], PCAH [165], BSSC [137] and RBM [57]. All the above methods are computed on the same extracted LFFs for a unified standard. All the compared methods are then evaluated on five different lengths of codes (32, 48, 64, 80, 96) and their results at 96-bit, which appear to be the best, are reported. Under the same experimental setting, all the parameters used in the compared methods have been strictly chosen according to their original papers. We list the compared results in Table 3.3 where RGB channel and depth channel represent only employing the methods in RGB and depth respectively, RGB-D fusion is the procedure of our algorithm and RGB-D cat is the concatenation of the features gained in RGB channel and depth channel. The results of the above mentioned other hashing methods in RGB-D fusion are not consistently higher than that in RGB-D concatenation, since not all of them preserve data structure. The training sizes are 70, 25 and 4 for datasets SKIG, MSRDailyActivity3D and CAD-60, respectively. Table 3.3 also reports the recognition accuracies of LFF and HOG3D using the improved Fisher vector (IFV) [120], for which 200 Gaussians are used in the GMM. The results show two phenomena: 1. LFF as a continuous feature outperforms other discrete histogram based features; 2. SPP outperforms other hashing methods.

3.5.4 Statistical Significance Test

To show the statistical significance of improvements, we conduct a t-test on the MAP improvements. In testing the null hypothesis that the population mean is equal to a specified value μ_0, the statistic

$$t = \frac{\bar{x} - \mu_0}{s/\sqrt{m}}$$

Table 3.4 t-Test on performance improvements.

Methods / Datasets	LFF+SPP vs. LFF+KSH	LFF+SPP vs. LFF+BRE	LFF+SPP vs. LFF+MLH	LFF+SPP vs. LFF+SpH	LFF+SPP vs. LFF+LSH	LFF+SPP vs. HOG3D+SPP
SKIG	9.97×10^{-13}	4.31×10^{-12}	1.52×10^{-14}	3.09×10^{-12}	1.45×10^{-14}	2.49×10^{-7}
MSRDailyActivity3D	3.98×10^{-12}	9.72×10^{-12}	3.26×10^{-13}	2.27×10^{-12}	5.78×10^{-16}	1.52×10^{-6}
CAD-60	3.57×10^{-13}	8.58×10^{-15}	3.55×10^{-15}	8.88×10^{-14}	1.46×10^{-17}	2.34×10^{-6}

Table 3.5 Recognition accuracy (%) of LFF and dense trajectory features on the UCF YouTube and HMDB51 datasets.

Feature	UCF YouTube	HMDB51
Trajectory	67.5	28.0
HOG	72.6	27.9
HOF	70.0	31.5
MBH	80.6	43.2
Trajectory/HOG/HOF/MBH combined	84.1	46.6
LFF (r = 1)	79.6	41.5
LFF (r = 3)	84.3	45.8
LFF (r = 5)	**85.2**	**46.9**
LFF (r = 7)	84.7	46.0
LFF (r = 9)	83.2	45.5

The LFF features are extracted along the same trajectories in the video sequences as the dense trajectory features.

is used, where $\bar{x}$ is the sample mean, s is the sample standard deviation of the sample and m is the sample size. Then the degree of freedom used in the test is $m-1$. We set $m = 10$ and code length $d = 96$ for this experiment. Table 3.4 lists the one-tail results of the t-test, which shows that the improvements are statistically significant.

3.5.5 Results on RGB Video dataset

To further illustrate the effectiveness of LFF, in this experiment, we compare the RGB version of LFF with the state-of-the-art feature: dense trajectory features on the **UCF YouTube** [91] and **HMDB51** [75] datasets for action recognition. The UCF YouTube dataset contains 1168 video sequences collected from 11 action categories. Most of them are sports activities, which are drawn from existing YouTube videos; therefore, the dataset contains large variations and approximates a real-world database. For this dataset, we deliberately use the full-sized sequences without any bounding boxes as the input to evaluate our method's robustness against complex and noisy backgrounds. We use the Leave-One-Out setup, i.e., testing on each original sequence while training on all the other sequences. The HMDB51 dataset contains 6849 realistic action sequences collected from a variety of movies and online videos. Specifically, it has 51 action classes and each has at least 101 positive samples. We adopt the official setting of [75] with three train/test splits. Each split has 70 training and 30 testing clips for each class. Table 3.5 illustrates that our proposed LFF ($r = 5$) can achieve competitive results with dense trajectory feature (DTF) which produces the state-of-the-art performance on recent publications [161, 162]. Note that for fair comparison of feature descriptors, all the compared features are extracted around the same points, i.e., the points on the trajectories.

3.6 Summary

The basic goal of this chapter is to obtain a fused local binary representation for RGB-D action recognition. To achieve this goal, we first introduced a continuous local descriptor called Local Flux Feature (LFF) based on the gradient field of video data, which is more suitable for the discretization of binary codes than histogram based local descriptors. After acquiring LFFs from RGB and depth channels, we applied the Structure Preserving Projection (SPP) to learn discriminative local binary representations. SPP preserves the characteristics in two levels including pairwise structure of local features and the relationship between video samples and classes at the same time without the collapse of data structure. The systematical experiments have shown not only the high efficiency of the proposed local binary representations, but also its superior performance than other local features and other hashing methods in terms of recognition accuracy on three RGB-D datasets.

Chapter 4

Binary Set Embedding for Cross-modal Retrieval

4.1 Introduction

The binarization scheme proposed in the last chapter is a supervised method. However, the label information of samples is unavailable for some situations. In this chapter, we extend it to an unsupervised algorithm and apply it to cross-modal retrieval for multimedia data.

In the current multimedia era, an image always appears with a description of text content on public knowledge websites such as the Wikipedia or photo sharing/social media websites such as Filckr and Facebook. Due to the diversity of the query and the multiple modalities of input, the multimedia similarity search, a.k.a. cross-modal retrieval, is becoming a critical problem and a ubiquitous searching method on the Internet [32, 118]. Nonetheless, the traditional nearest neighbor search in information retrieval is neither scalable nor efficient when facing the explosion of multimedia data. To conquer this problem, binary code representations, or hashing methods, provide a fast search mechanism through the bit XOR operation and the time complexity of similarity search is simply $O(1)$ if all the binary codes are stored. In addition, a more discriminative representation could be acquired if the algorithm sufficiently learns the intrinsic structure and the semantic information of multimedia data.

Notwithstanding the successful results achieved by the recent hashing methods, the lack of incorporating the visual features with the corresponding linguistic understanding makes them uncompetitive for the challenging cross-modal tasks. A major drawback is the use of global histogram-based representations, which would bring the quantization error during the codebook construction and lose the structure of local features and words. The document-oriented representations such as latent Dirichlet allocation (LDA) [14] need to be re-trained when the text is modified, a new paragraph is added to the dataset or a new dataset is built. This operation largely increases the computational complexity and the aforementioned cross-modality algorithms are also required to be implemented again. In addition, single-vector representations cannot comprehensively and precisely characterize the samples which have multiple tags or topics and the scenario with large intra-class variations and small inter-class discrepancies.

In this chapter, we aim to exploit the semantic connection between images and their corresponding documents in low-level features, either visual or textual, i.e., local features. The local feature descriptors for images such as SIFT [103] and even deep features [97] have been well studied. The construction of local features for texts can be done by the *word vector* techniques [107, 157] in natural language processing, which have been shown the superiority in machine translation. Once the learning phase for local features is completed, the coding function is fixed for any new sample (image-text pair) since each word has been assigned a unique hash code. Apparently, one of the requirements for our algorithm is that the cross-modal links based on local features need to be established. However, it is impossible and unrealistic to build a one-to-one correspondence between local feature points from different modalities. Therefore, we consider the relationship between the sets composed of local features from image and text domains. Taking the image-text pair of a car as an example, two SIFT features are close to each other if they are visually similar and two word vectors are close to each other if they are semantically similar. Meanwhile, the cross-modal algorithm must also connect the local feature set of the image “car” and the word vector set of the corresponding description of the car for semantic understanding of images.

To achieve the above objective, we propose a novel cross-modal hashing scheme called Binary Set Embedding (BSE) which is illustrated in Fig. 4.1. Due to the different distributions of image and text data, BSE learns two orthogonal projections and projects local features (image or text) into a common low-dimensional Hamming space. In this way, for each sample, the image features and the corresponding linguistic features are encoded to similar hash codes by BSE. In the meantime, we also take the geometric structures of each modality into account for preserving the intra-modal similarity. Given a local feature, its source information, i.e., the image (text) from which it is extracted, is also provided. Consequently, relationships in two layers: element-to-element and set-to-set which are e-

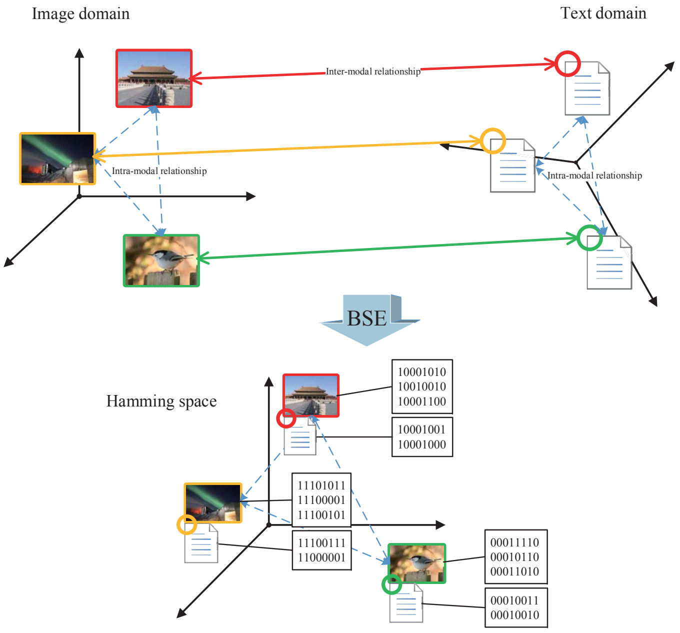


Fig. 4.1 Illustration of BSE. BSE encodes all local features in image and text domains into a common Hamming space.

quivalent to the structures of data points and images (texts) represented by local feature sets respectively, are simultaneously preserved in the lower-dimensional Hamming space. It is worthwhile to highlight several properties of the proposed approach:

- Our work associates images with semantic information in a fundamental level. The binary codes learned from local image features are semantically more robust than the word-frequency histogram.
- BSE assigns a binary code for each local feature. The encoding of local features reduces the sparsity of the final hash table and improves the usage of hash codes, which enables hash codes to achieve competitive performance with a short length.
- Last but not least, the local features for the text domain, i.e., word vectors, are independent of any specific datasets and can be trained offline, which makes BSE more universal in realistic applications.

4.2 Related Work

One of the most popular hashing algorithms with the idea of preserving the similarity is Locality Sensitive Hashing (LSH) [43], which pursuits the maximum probability of the collision for similar data. Specifically, LSH maps similar points in the original data space into the same hash buckets with high probability and dissimilar data pairs into the same hash buckets with low probability. Afterwards, a number of cross-modal hashing schemes have been proposed to discover the relationship among different modalities of multimedia data. Cross-Modality Similarity Search Hashing (CMSSH) [18] embeds incommensurable data into a common metric space by a boosting algorithm. With extended spectral hashing [169], Kumar et al. [78] proposed Cross-View Hashing (CVH) to generate binary codes for each modality via Canonical Correlation Analysis (CCA). Multimodal Latent Binary Embedding (MLBE) [186] is another cross-modal hashing method considering both the inter-modal and intra-modal similarity via a probabilistic model. To learn the hash function with good generalization, Co-Regularized Hashing (CRH) [185] was proposed to project data far from 0. Zhu et al. [188] proposed a linear method for multimedia search to reduce the computational complexity. Recently, Inter-Media Hashing (IMH) [144] was proposed to explore the correlations among different modalities and learn hashing functions by a linear regression model. Instead of learning codes for each specific view, both Composite Hashing with Multiple Information Sources (CHMIS) [179] and Collective Matrix Factorization Hashing (CMFH) [33] learn unified hash codes for each sample.

4.3 Binary Set Embedding

In this section, we introduce our Binary Set Embedding (BSE) algorithm. We first describe the intra-modal and inter-modal structures and then associate them into one objective function. With the orthogonality constraint, BSE outputs the orthogonal projections for each modality.

4.3.1 Notations and Problem Statement

Since our task is the similarity search between the image domain and the text domain, we consider s image-text sample pairs $S_1, \cdots, S_s$ containing the local feature sets from the image and text domains. For the i-th sample pair S_i, we denote its local feature sets in image and text domains by $X_i = \{\mathbf{x}_{i1}, \cdots, \mathbf{x}_{in_i}\}$ with $\mathbf{x}_{ij} \in \mathbb{R}^{D_1}$, and $Y_i = \{\mathbf{y}_{i1}, \cdots, \mathbf{y}_{im_i}\}$ with $\mathbf{y}_{ij} \in \mathbb{R}^{D_2}$, respectively. In this way, we have the union of the local feature sets $X = \bigcup_{i=1}^{s} X_i$ and $Y = \bigcup_{i=1}^{s} Y_i$. Without loss of generality, we denote $X = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$ and $Y = \{\mathbf{y}_1, \cdots, \mathbf{y}_M\}$, where $N = \sum_{i=1}^{s} n_i$ and $M = \sum_{i=1}^{s} m_i$.

Considering the different properties of image and text domains, we aim to seek two projections $\Theta_1 \in \mathbb{R}^{D_1 \times d}$ and $\Theta_2 \in \mathbb{R}^{D_2 \times d}$ for X and Y respectively to build the hash functions with the same code length:

$$H_1(\mathbf{x}) = \mathrm{sgn}(\Theta_1^T \mathbf{x}) \text{ and } H_2(\mathbf{y}) = \mathrm{sgn}(\Theta_2^T \mathbf{y}). \tag{4.1}$$

It is noticeable that during the code learning stage, we use $\{-1,+1\}$ to encode local features and employ centralized data $\mathbf{x}_i - \frac{1}{N}\sum_{j=1}^{N}\mathbf{x}_j$ and $\mathbf{y}_i - \frac{1}{M}\sum_{j=1}^{M}\mathbf{y}_j$ instead of $\mathbf{x}_i$ and $\mathbf{y}_i$ respectively, $i = 1, \cdots, s$. In the indexing phase, we use $\{0,1\}$ to represent codes for hash lookup.

4.3.2 Intra-modal Relationship

For the unsupervised analysis based on local feature descriptors, we are given not only the local features themselves, but also their source information, i.e., the sample from which they are extracted. We first discuss the connection between local features and the connection between images for the image domain. Then we have the similar objective functions for the text domain.

Element-to-Element Structure

We hope that the pairwise structure of local features in the original space could be preserved in the lower-dimensional Hamming space. Without class information, we employ the K-means clustering on X to divide the set $\mathscr{P}_1 = \{(i,j)|\mathbf{x}_i, \mathbf{x}_j \in X\}$ into two categories, i.e., positive pairs and negative pairs. Specifically, we divide X into K clusters by the K-means clustering and define the pairwise label for $(\mathbf{x}_i, \mathbf{x}_j)$ as follows:

$$\ell_{ij}^X = \begin{cases} +1, & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are in the same cluster} \\ -1, & \text{otherwise} \end{cases},$$

Moreover, we also expect that, for a positive pair, the effect on the objective function will increase when their distance decreases, and for a negative pair, conversely, its importance will be reduced when the paired features are closer to each other. Then by the Gaussian function, we assign the following weight for each pair with parameter σ:

$$W_{ij}^{X} = \begin{cases} \exp\left(-\frac{\|\mathbf{x}_i-\mathbf{x}_j\|^2}{\sigma^2}\right), & \ell_{ij}^{X}=1 \\ \exp\left(-\frac{1}{\sigma^2\|\mathbf{x}_i-\mathbf{x}_j\|^2}\right), & \ell_{ij}^{X}=-1 \end{cases},$$

where $\|\cdot\|$ is the L^2-norm. It is easy to find that $W_{ij}^{X} \in (0,1)$ and satisfies our requirement. Hence, preserving the feature-to-feature structure in the image domain is to maximize

$$\sum_{(i,j)\in\mathscr{P}_1} W_{ij}^{X}\ell_{ij}^{X}\langle H_1(\mathbf{x}_i),H_1(\mathbf{x}_j)\rangle. \tag{4.2}$$

Similarly, for the text domain, we also have the following objective function to be maximized:

$$\sum_{(i,j)\in\mathscr{P}_2} W_{ij}^{Y}\ell_{ij}^{Y}\langle H_2(\mathbf{y}_i),H_2(\mathbf{y}_j)\rangle, \tag{4.3}$$

where $\mathscr{P}_2$ is the pair set for the text domain, and W_{ij}^{Y} and ℓ_{ij}^{Y} are the pairwise weights and the pairwise labels in the text domain, respectively.

Set-to-Set Structure

The set-to-set structure can be regarded as a higher-level connection among local features to balance the sensitivity of the clustering information in the above element-to-element structure. This structure is constructed on the samples when each of them is represented by a set of local features. For image i, X_i represents its local feature set $\{\mathbf{x}_{i1},\cdots,\mathbf{x}_{in_i}\}$. We use the image-to-image (I2I) distance derived from [16] to measure the set-to-set distance from image i to image j, which can be regarded as an approximation of the Kullback-Leibler divergence and is defined as:

$$d_{ij} = \sum_{\mathbf{x}\in X_i} \|\mathbf{x}-NN_j(\mathbf{x})\|^2, \tag{4.4}$$

where $NN_j(\mathbf{x})$ is the nearest neighbor of local feature $\mathbf{x}$ in image j. Since generally $d_{ij} \neq d_{ji}$, we update the symmetric distance $D_{ij} = (d_{ij}+d_{ji})/2$ as the I2I distance between image i and image j. By Gaussian function, we can define the following I2I similarity with the smooth parameter σ_X:

$$S_{ij}^X = \exp\left(-\frac{D_{ij}^2}{2\sigma_I^2}\right),\ i,j = 1,\cdots,s. \tag{4.5}$$

Although the number of local features in one image is much smaller than N, the nearest neighbor search (NN-search) for all images is still time-consuming. We hope to use the cluster information in the above element-to-element section for the reduction of complexity. We denote the clusters of the K-means on X by $C_1,\cdots,C_K$. Without loss of generality, supposing the local features of image j are in $C_1,\cdots,C_{K_1}$ and the order of distances from corresponding centroids to $\mathbf{x} \in X_i$ is from nearest to farthest, the range of NN-search in $\mathscr{X}_j$ is reduced to $(C_1 \cup \cdots \cup C_{\lceil (K_1)^\delta \rceil}) \cap \mathscr{X}_j$, where $0 < \delta < 1$ and $\lceil \cdot \rceil$ is the ceiling function. This reduction of range is based on the assumption that the centroid of the cluster where the true nearest neighbor locates is also close to $\mathbf{x}$. In fact, it holds when $K \to N$. After the reduction of the searching range, the average complexity is reduced from $O(N^2)$ to $O(NK^{1+\delta})$ and we only need to compute the distances from $\mathbf{x}$ to the cluster centroids, which has been done in the K-means.

After applying the encoding algorithm, the I2I distance in Hamming space becomes

$$\widehat{D}_{ij}^X = \frac{1}{2}\left(\sum_{\mathbf{x}\in X_i} \|H_1(\mathbf{x}) - NN_j(H_1(\mathbf{x}))\|^2 + \sum_{\mathbf{x}\in X_j} \|H_1(\mathbf{x}) - NN_i(H_1(\mathbf{x}))\|^2\right). \tag{4.6}$$

Therefore, to preserve the I2I structure of the original image domain by giving the penalty S_{ij}^X to the mapped distance $\widehat{D}_{ij}^X$, a reasonable objective function is to minimize

$$\sum_{i,j} \widehat{D}_{ij}^X \cdot S_{ij}^X. \tag{4.7}$$

Likewise, preserving the set-to-set structure in the text domain is to minimize the following similar objective function:

$$\sum_{i,j} \widehat{D}_{ij}^Y \cdot S_{ij}^Y, \tag{4.8}$$

where $\widehat{D}_{ij}^Y$ and S_{ij}^Y are the encoded set-to-set distance and the set-to-set similarity in the text domain, respectively.

4.3.3 Inter-modal Relationship

Local features in the image domain and the text domain have different distributions. For precise retrieval, we need to encode the local features from similar samples to close hash codes no matter they are in the image domain or the text domain. Without class information, we are only concerned about the relationship between the image local features and the text local features from the same sample.

For each sample pair S_i, the local feature set from the image domain and the local feature set from the text domain are denoted by X_i and Y_i, respectively, $i = 1, \cdots, s$. Generally speaking, it is impossible to construct a one-to-one correspondence between X_i and Y_i; even a nearest neighbor relationship in the Hamming space cannot be built since the correspondence between visual features and semantic information is unknown by the algorithm. Then using the I2I distance to measure the connection between X_i and Y_i is not applicable. Therefore, we minimize the distance of all the local feature pairs in the set $\{(\mathbf{x},\mathbf{y})|\mathbf{x} \in X_i, \mathbf{y} \in Y_i\}$ for the i-th image-text pair in the Hamming space. In other words, our goal for the inter-modal relationship is to maximize the following sum of the inner products:

$$\sum_{i=1}^{s} \sum_{\mathbf{x}\in X_i, \mathbf{y}\in Y_i} \langle H_1(\mathbf{x}), H_2(\mathbf{y})\rangle. \tag{4.9}$$

4.3.4 Objective Function and Optimization

Spectral relaxation First, let us transform Eqs. (4.2), (4.3), (4.7), (4.8) and (4.9) to the functions on Θ_1 and Θ_2. Motivated by [100, 169], we relax the discrete sign function to a real-valued continuous function by using its signed magnitude, i.e., $\text{sgn}(x) \approx x$. In this way, the objective function in the element-to-element part of the image domain, i.e., Eq. (4.2) becomes

$$\begin{aligned}
&\sum_{(i,j)\in\mathscr{P}_1} W_{ij}^X \ell_{ij}^X \langle \Theta_1^T \mathbf{x}_i, \Theta_1^T \mathbf{x}_j \rangle \\
=&\sum_{(i,j)\in\mathscr{P}_1} W_{ij}^X \ell_{ij}^X (\Theta_1^T \mathbf{x}_i)^T \Theta_1^T \mathbf{x}_j \\
=&\sum_{(i,j)\in\mathscr{P}_1} W_{ij}^X \ell_{ij}^X \text{tr}(\Theta_1^T \mathbf{x}_i (\Theta_1^T \mathbf{x}_j)^T) \\
=&\sum_{(i,j)\in\mathscr{P}_1} W_{ij}^X \ell_{ij}^X \text{tr}(\Theta_1^T \mathbf{x}_i \mathbf{x}_j^T \Theta_1) \\
=&\,\text{tr}(\Theta_1^T L_X \Theta_1),
\end{aligned} \tag{4.10}$$

where $L_X = \sum_{(i,j)\in\mathscr{P}_1} W_{ij}^X \ell_{ij}^X \mathbf{x}_i \mathbf{x}_j^T$. With a similar transformation, Eq. (4.3) for the text domain becomes

$$\mathrm{tr}(\Theta_2^T L_Y \Theta_2), \tag{4.11}$$

where $L_Y = \sum_{(i,j)\in\mathscr{P}_2} W_{ij}^Y \ell_{ij}^Y \mathbf{y}_i \mathbf{y}_j^T$.

Additionally, for the I2I distance, we make a statistical approximation on the computation of projected I2I distances due to the large amount of local features. That is, we exchange the operation of NN-search and $H_1(\cdot)$ for all $\mathbf{x} \in X_i$ during the optimization, i.e., $\sum_{\mathbf{x}\in X_i} \|H_1(\mathbf{x}) - NN_j(H_1(\mathbf{x}))\|^2 \approx \sum_{\mathbf{x}\in X_i} \|H_1(\mathbf{x}) - H_1(NN_j(\mathbf{x}))\|^2$. In fact, the pairwise structure has been preserved in the objective function (4.2), which ensures the correctness of the exchange operation. Then we have the following projected distance $\widehat{d_{ij}}$ in the optimization:

$$\begin{aligned}
\widehat{d_{ij}} &\approx \sum_{\mathbf{x}\in X_i} \|\Theta_1^T \mathbf{x} - \Theta_1^T NN_j(\mathbf{x})\|^2 \\
&= \sum_{\mathbf{x}\in X_i} \|\Theta_1^T (\mathbf{x} - NN_j(\mathbf{x}))\|^2 \\
&= \sum_{\mathbf{x}\in X_i} \left(\Theta_1^T (\mathbf{x} - NN_j(\mathbf{x}))\right)^T \Theta_1^T (\mathbf{x} - NN_j(\mathbf{x})) \\
&= \sum_{\mathbf{x}\in X_i} \mathrm{tr}(\Theta_1^T (\mathbf{x} - NN_j(\mathbf{x}))(\mathbf{x} - NN_j(\mathbf{x}))^T \Theta_1).
\end{aligned}$$

If we denote

$$D_X = \frac{1}{2}\sum_{i,j} S_{ij}^X \Big(\sum_{\mathbf{x}\in X_i} (\mathbf{x} - NN_j(\mathbf{x}))(\mathbf{x} - NN_j(\mathbf{x}))^T + \sum_{\mathbf{x}\in X_j} (\mathbf{x} - NN_i(\mathbf{x}))(\mathbf{x} - NN_i(\mathbf{x}))^T \Big),$$

then the objective function in the set-to-set part of the image domain, i.e., Eq. (4.7) can be written as:

$$\mathrm{tr}(\Theta_1^T D_X \Theta_1). \tag{4.12}$$

Certainly, for the text domain, we also have the similar trace form:

$$\mathrm{tr}(\Theta_2^T D_Y \Theta_2), \tag{4.13}$$

where

$$D_Y = \frac{1}{2}\sum_{i,j} S_{ij}^Y \Big(\sum_{\mathbf{y}\in Y_i} (\mathbf{y} - NN_j(\mathbf{y}))(\mathbf{y} - NN_j(\mathbf{y}))^T + \sum_{\mathbf{y}\in Y_j} (\mathbf{y} - NN_i(\mathbf{y}))(\mathbf{y} - NN_i(\mathbf{y}))^T \Big).$$

And for the inter-modal relationship, Eq. (4.9) is simply relaxed to

$$\mathrm{tr}(\Theta_1^T A \Theta_2), \tag{4.14}$$

where $A = \sum_{i=1}^{n} \sum_{\mathbf{x} \in X_i, \mathbf{y} \in Y_i} \mathbf{x}\mathbf{y}^T$.

Objective function Without loss of generality, we let the objective dimension (code length) $d = 1$, i.e., Θ_1 and Θ_2 are column vectors. Furthermore, we place the intra-modal relationship and the inter-modal relationship at equally important positions. Therefore, combining the above functions on Θ_1 and Θ_2 and the norm constraint $||\Theta_1|| = ||\Theta_2|| = 1$, we have our final optimization problem:

$$\underset{||\Theta_1||=||\Theta_2||=1}{\arg\max} \frac{\Theta_1^T A \Theta_2}{(\Theta_1^T (\lambda D_X - L_X)\Theta_1)(\Theta_2^T (\lambda D_Y - L_Y)\Theta_2)}, \tag{4.15}$$

where λ is the parameter for balancing the effect of the element-to-element and set-to-set structures.

Optimization Let us denote $B_X = \lambda D_X - L_X$ and $B_Y = \lambda D_Y - L_Y$ which are two symmetric matrices. We change the norm constraints to $\Theta_1^T B_X \Theta_1 = 1$ and $\Theta_2^T B_Y \Theta_2 = 1$, since it is always possible to restore the final norm to $||\Theta_1|| = ||\Theta_2|| = 1$. Then we can define the Lagrangian function:

$$L(\Theta_1, \Theta_2) = \Theta_1^T A \Theta_2 - \alpha(\Theta_1^T B_X \Theta_1 - 1) - \beta(\Theta_2^T B_Y \Theta_2 - 1),$$

where α and β are the Lagrangian coefficients. To find the optimal solution, we let the derivatives of L with respect to Θ_1 and Θ_2 be zeros to obtain:

$$\frac{\partial L}{\partial \Theta_1} = A\Theta_2 - 2\alpha B_X \Theta_1 = 0, \tag{4.16}$$

$$\frac{\partial L}{\partial \Theta_2} = A^T \Theta_1 - 2\beta B_Y \Theta_2 = 0. \tag{4.17}$$

Multiplying Θ_1^T and Θ_2^T on the left-hand-side of the above equations respectively, we have

$$\begin{aligned}\Theta_1^T A\Theta_2 - 2\alpha = 0,\\ \Theta_2^T A^T \Theta_1 - 2\beta = 0.\end{aligned}$$

Then we only need to find the maximum α. From Eqs. (4.16) and (4.17), we also have

$$A\Theta_2 = 2\alpha B_X \Theta_1 \text{ and } A^T\Theta_1 = 2\alpha B_Y \Theta_2. \tag{4.18}$$

By transforming the above equations to the form of block matrix, we have

$$\begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix}\begin{pmatrix} \Theta_1 \\ \Theta_2 \end{pmatrix} = 2\alpha \begin{pmatrix} B_X & 0 \\ 0 & B_Y \end{pmatrix}\begin{pmatrix} \Theta_1 \\ \Theta_2 \end{pmatrix}. \tag{4.19}$$

As a result, to find the optimal solution of (4.15) is equivalent to solve the generalized eigen-decomposition problem (4.19).

4.3.5 Orthogonality Constraint

Until now we have only computed the projection vector for the first dimension. It is noticeable that our objective function (4.15) is similar to the canonical correlation analysis (CCA). However, the relative orthogonality constraints in CCA cannot reflect realistic intention for our scheme. In this section, we use the induction method to compute the remaining vectors successively and make them mutually orthogonal by using the matrix composed by previous output vectors as shown in Section 2.3.5. With this orthogonalization procedure, we can realize our whole algorithm.

Suppose we have gained first p vectors $\Theta_1 = [\mathbf{a}_1, \cdots, \mathbf{a}_p]$ and $\Theta_2 = [\mathbf{b}_1, \cdots, \mathbf{b}_p]$. We need to find the solutions $\mathbf{a}_{p+1}$ and $\mathbf{b}_{p+1}$ to the optimization problem (4.15) with the orthogonal constraints

$$\mathbf{a}_1^T\mathbf{a}_{p+1} = \cdots = \mathbf{a}_p^T\mathbf{a}_{p+1} = \mathbf{b}_1^T\mathbf{b}_{p+1} = \cdots = \mathbf{b}_p^T\mathbf{b}_{p+1} = 0.$$

If we project all the local features in the image and text domains onto the subspaces $span(\mathbf{a}_1, \cdots, \mathbf{a}_p)^\perp$ and $span(\mathbf{b}_1, \cdots, \mathbf{b}_p)^\perp$, respectively, then the optimization process will be in these two subspaces and the output vectors will satisfy the orthogonal constraints. In fact, we only need to solve the linear equations $\Theta_1^T Z = 0$ and $\Theta_2^T Z = 0$ with the unknown variable Z to obtain the orthonormal basis $P_1 \in \mathbb{R}^{D_1 \times (D_1 - p)}$ and $P_2 \in \mathbb{R}^{D_2 \times (D_2 - p)}$ of the spaces $span(\mathbf{a}_1, \cdots, \mathbf{a}_p)^{\perp}$ and $span(\mathbf{b}_1, \cdots, \mathbf{b}_p)^{\perp}$, respectively, which is commonly used in linear algebra. With the basis P_1 and P_2, the projections are simply as follows:

$$\begin{aligned} \mathbb{R}^{D_1} &\to \mathbb{R}^{D_1 - p} \cong span(\mathbf{a}_1, \cdots, \mathbf{a}_p)^{\perp} \\ \mathbf{x} &\mapsto P_1^T \mathbf{x} \end{aligned}$$

and

$$\begin{aligned} \mathbb{R}^{D_2} &\to \mathbb{R}^{D_2 - p} \cong span(\mathbf{b}_1, \cdots, \mathbf{b}_p)^{\perp} \\ \mathbf{y} &\mapsto P_2^T \mathbf{y}. \end{aligned}$$

In this case, we need to update all the matrices related to the local feature data:

$$A \leftarrow P_1^T A P_2, \; B_X \leftarrow P_1^T B_X P_1, \; B_Y \leftarrow P_2^T B_Y P_2.$$

Now we can repeat the eigen-decomposition procedure described in the above optimization section and output the optimal solutions $\mathbf{a}_{p+1} \in \mathbb{R}^{D_1 - p}$ and $\mathbf{b}_{p+1} \in \mathbb{R}^{D_2 - p}$. Finally, we recover $\mathbf{a}_{p+1}$ and $\mathbf{b}_{p+1}$ to the vectors in $\mathbb{R}^{D_1}$ and $\mathbb{R}^{D_2}$ by updating $\mathbf{a}_{p+1} \leftarrow P_1 \mathbf{a}_{p+1}$ and $\mathbf{b}_{p+1} \leftarrow P_2 \mathbf{b}_{p+1}$, respectively. We summarize BSE in Algorithm 4.

4.4 Voting Scheme for Local Feature Indexing

Having obtained the projection matrices Θ_1 and Θ_2, we can easily embed the training local features into binary hash codes by Eq. (4.1). For the query local features $\widehat{\mathbf{x}}$ and $\widehat{\mathbf{y}}$, their hash codes are obtained by $H(\widehat{\mathbf{x}}) = \mathrm{sgn}(\Theta_1^T(\widehat{\mathbf{x}} - \frac{1}{N}\sum_{j=1}^{N} \mathbf{x}_j))$ and $H(\widehat{\mathbf{y}}) = \mathrm{sgn}(\Theta_2^T(\widehat{\mathbf{y}} - \frac{1}{M}\sum_{j=1}^{M} \mathbf{y}_j))$, respectively. Nevertheless, traditional linear search (e.g., Hamming distance ranking) with complexity $O(N)$ is not fast any more for our local feature hashing scenario, since N denotes the total number (at least 3M for a large-scale database) of local features. To accomplish the local feature based visual retrieval, in this chapter, we introduce a fast voting scheme for local feature indexing [63]. We first build the Hamming lookup table (a.k.a. the hashing table) for all the hash codes from image and text domains. Given a query, we can find the bucket of corresponding hash codes in near constant time $O(1)$, and return all the data in the bucket as the retrieved results whether they are in image and text domains.

Algorithm 4 Binary Set Embedding

```
Input: The local feature sets $X$ and $Y$ from image and text domains respectively, the number of centroids $K$ in the K-means, the parameter $\delta$ for the NN-search, the balancing parameter $\lambda$ and the objective dimension (code length) $d$.
Output: The projection matrices $\Theta_1$ and $\Theta_2$ for the local features in image and text domains respectively.
Preprocessing: centralize $\mathbf{x}_i \leftarrow \mathbf{x}_i - \frac{1}{N}\sum_{k=1}^{N}\mathbf{x}_k$, $\mathbf{y}_j \leftarrow \mathbf{y}_j - \frac{1}{M}\sum_{k=1}^{M}\mathbf{y}_k$ for $i = 1,\cdots,N$, $j = 1,\cdots,M$;
Construct local feature pairing sets $\mathscr{P}_1$ and $\mathscr{P}_2$, and their corresponding pairwise labels $\ell_{ij}^{X}$ and $\ell_{ij}^{Y}$ by K-means clustering for image and text domains, respectively;
Compute the weights $W_{ij}^{X}$ and $W_{ij}^{Y}$ for the element-to-element structure and the similarities $S_{ij}^{X}$ and $S_{ij}^{Y}$ for the set-to-set structure;
Initialization: $\Theta_1 \leftarrow \emptyset$, $\Theta_2 \leftarrow \emptyset$, $P_1 \leftarrow I_{D_1}$ and $P_2 \leftarrow I_{D_2}$;
for $i = 1$ to $d$ do
    Project training local features in image and text domains onto the subspaces $span(\Theta_1)^{\perp}$ and $span(\Theta_2)^{\perp}$ by using the basis $P_1$ and $P_2$, respectively;
    Solve the generalized eigen-decomposition problem (4.19) to obtain the vector $\begin{pmatrix} \mathbf{a}_i \\ \mathbf{b}_i \end{pmatrix}$ corresponding to the largest generalized eigenvalue;
    Recover $\mathbf{a}_i \leftarrow P_1\mathbf{a}_i$ and $\mathbf{b}_i \leftarrow P_2\mathbf{b}_i$;
    Update $\Theta_1 \leftarrow [\Theta_1, \mathbf{a}_i]$ and $\Theta_2 \leftarrow [\Theta_2, \mathbf{b}_i]$, and let $P_1$ and $P_2$ be the orthonormal basis of $span(\Theta_1)^{\perp}$ and $span(\Theta_2)^{\perp}$ by solving the corresponding linear equations respectively.
end for
```

After construction of the Hamming lookup table over the training set, we store the corresponding indices for all the hash codes of local features. In this way, for a text query Q, we search the hash code $H(\mathbf{q}_i)$ for each local feature $\mathbf{q}_k \in Q$ in the query $Q = \{\mathbf{q}_1, \cdots, \mathbf{q}_m\}$ over the Hamming lookup table within Hamming radius r and return the possible images' indices. It is noteworthy that the same bucket in the Hamming lookup table may store the indices from different images. Therefore, we vote and accumulate the times of each image's index appearing in relevant buckets and then rank them in a decreasing order. Specifically, we assign a vector $\mathbf{v} = (v_1, \cdots, v_n) \in \mathbb{R}^n$ for the query with the subscripts corresponding to the indices of the images in the gallery. Then we update $v_i \leftarrow v_i + 1$ if there exists a local feature from sample i, which is within Hamming radius r in one Hamming lookup. The final retrieved samples are returned according to the descending order of $(v_1, \cdots, v_n)$. And for the image query, retrieval for the text results is performed by the same voting procedure. We summarize the above voting scheme in Algorithm 5.

Algorithm 5 Voting Scheme for Local Feature Indexing

```
Input: The local feature sets $X$ and $Y$ from image and text domains respectively, the local feature
    set of query text (image) $Q = \{\mathbf{q}_1, \cdots, \mathbf{q}_m\}$, Hamming radius $r$ and the learned projections $\Theta_1$
    and $\Theta_2$.
Output: The retrieved images (texts) ranked by similarity.
Encoding all the local features into Hamming space via the Eq. (4.1) with $\Theta_1$ and $\Theta_2$;
Construct Hamming lookup table over the training set;
for $i = 1$ to $m$ do
    For the query hash code $H(\mathbf{q}_i)$, store all the possible image (text) indices fall into the Hamming lookup table within Hamming radius $r$;
    Assign vector $\mathbf{v} = (v_1, \cdots, v_n) \in \mathbb{R}^n$ for the query $Q$ and update $v_i \leftarrow v_i + 1$ if image (text) $i$ appears in one Hamming lookup;
end for
Sort $(v_1, \cdots, v_n)$ in decreasing order;
return All the relevant images (texts) as the retrieved results.
```

4.5 Experiments

In this section, we evaluate the proposed BSE method on two public datasets: the Wiki dataset and the NUS-WIDE dataset for cross-modal retrieval tasks. The relevant results show that our BSE significantly outperforms several state-of-the-art methods.

Table 4.1 MAP comparison on the Wiki and NUS-WIDE datasets.

Task	**Method**	**Wiki**						**NUS-WIDE**					
		16 bits	**32 bits**	**48 bits**	**64 bits**	**80 bits**	**96 bits**	**16 bits**	**32 bits**	**48 bits**	**64 bits**	**80 bits**	**96 bits**
Image to Text	CVH	0.210	0.163	0.142	0.129	0.137	0.132	0.371	0.382	0.426	0.413	0.405	0.393
	IMH	0.221	0.224	0.232	0.220	0.213	0.208	0.498	0.492	0.473	0.477	0.468	0.466
	MLBE	0.242	0.237	0.231	0.235	0.223	0.210	0.483	0.472	0.465	0.463	0.474	0.472
	CMSSH	0.231	0.233	0.238	0.242	0.245	0.247	0.501	0.504	0.510	0.513	0.515	0.518
	CHMIS	0.237	0.240	0.245	0.248	0.248	0.251	0.492	0.497	0.513	0.515	0.521	0.524
	CMFH	0.256	0.259	0.261	0.263	0.265	0.270	0.551	0.562	0.568	0.570	0.574	0.583
	QCH	0.238	0.251	0.253	0.257	0.261	0.264	0.517	0.538	0.546	0.555	0.561	0.566
	BSE	**0.260**	**0.268**	**0.272**	**0.277**	**0.281**	**0.284**	**0.572**	**0.574**	**0.574**	**0.580**	**0.583**	**0.597**
Text to Image	CVH	0.310	0.202	0.187	0.153	0.140	0.137	0.422	0.403	0.395	0.390	0.427	0.438
	IMH	0.503	0.496	0.483	0.493	0.462	0.467	0.493	0.508	0.512	0.504	0.492	0.497
	MLBE	0.483	0.432	0.319	0.262	0.231	0.220	0.510	0.501	0.472	0.486	0.488	0.493
	CMSSH	0.305	0.312	0.320	0.323	0.328	0.331	0.508	0.514	0.523	0.527	0.529	0.533
	CHMIS	0.237	0.240	0.245	0.248	0.248	0.251	0.492	0.497	0.513	0.515	0.521	0.524
	CMFH	0.601	0.605	0.612	0.618	0.625	0.633	0.650	0.674	0.688	0.707	0.707	0.711
	QCH	0.316	0.357	0.369	0.427	0.456	0.471	0.554	0.583	0.588	0.601	0.624	0.633
	BSE	**0.614**	**0.618**	**0.625**	**0.633**	**0.638**	**0.640**	**0.671**	**0.684**	**0.710**	**0.721**	**0.728**	**0.732**

All the compared methods (except “BSE”) utilize vector of locally aggregated descriptors (VLAD) in this table.

4.5.1 Datasets

The **Wiki** dataset [124] collects samples from Wikipedia “featured articles”, containing 2866 image-text pairs in 10 semantic classes. For each image, a set of 128-*d* SIFT [103] local features are extracted around salient points. For each text, we utilize the novel word-to-vector technique [107] to extract the 200-*d* semantic *word vectors* trained from the *first billion characters from Wikipedia*[1] for each word. Following the setting in the original paper [124], we take 2173 image-text pairs as the training set and the remaining 693 image-text pairs as the query set.

The **NUS-WIDE** dataset [27] contains around 270,000 web images associated with 81 ground truth concept classes. As in [101], we only use the most frequent 21 concept classes, each of which has abundant relevant images ranging from 5,000 to 30,000. Unlike other datasets, each image in the NUS-WIDE dataset is assigned with multiple semantic labels (tags). In our work, two images belong to the same class, only if they share at least one common tag. Similarly, each image or text sample is represented by a set of SIFT features or a set of word vectors, respectively, as in the Wiki dataset. We further sample randomly 100 images from each of the selected 21 tags to form a query set of 2,100 images with the rest serving as the training set, since some of the remaining 60 tags contain too few images for the retrieval task.

[1] https://code.google.com/p/word2vec/

4.5.2 Compared Methods and Experimental Settings

In our experiments, since few works focus on the local feature representation based hashing scheme for cross-modal retrieval, we can only systematically compare the proposed BSE method with six prevailing global hashing methods for cross-modal retrieval tasks: CVH [78], MLBE [186], IMH [144], CMSSH [18], CHMIS [179], CMFH [33] and QCH [170]. For fair comparison, all the methods are implemented on the same SIFT features and word vectors in image and text domains, respectively. Specifically, for the global methods, we use the vector of locally aggregated descriptors[2] (VLAD) [64] to embed sets of SIFT/word vectors from each image/text into an integrated representation. For CVH, IMH, CMSSH and CMFH, the view-specific hashing codes can be learned while CHMIS is a cross-view fusion method which learns integrated hash codes. We implement CVH, IMH ourselves and utilize the public codes of MLBE, CMSSH, CHMIS, CMFH and QCH to calculate the results. All the parameters in compared methods are strictly selected according to their original publications.

For BSE, the parameter K for K-means is chosen from one of $\{100, 200, \cdots, 1000\}$ via 10-fold cross-validation on the training data and the best performed value of K is selected. Furthermore, the balancing parameter λ is also selected from one of $\{0.05, 0.1, \cdots, 0.5\}$, which yields the best performance by 10-fold cross-validation on the training set. δ for the NN-search is always fixed at 0.5 and the Hamming radius r is equal to 3.

For the query phase, we use the voting scheme introduced in Section 4.4 to retrieve neighbors of the query. We further report the mean average precision (MAP) of the top 50 retrieved images/documents for both of the datasets. It is defined as

$$\text{MAP} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{50}\sum_{j=1}^{50}P(ij),$$

where $|Q|$ is the size of the query set and $P(ij)$ indicates the precision of the top j retrieved texts (images) of the i-th image (text). In addition, all of the methods are evaluated on six different lengths of codes $\{16, 32, 48, 64, 80, 96\}$. The selection of training and test samples is repeated five times for all the datasets and the compared methods, and we report the averages as the final results.

[2] The best number of clusters K used in VLAD is selected via 10-fold cross-validation on the training data from $K = 100$ to $K = 1000$ with step 100.

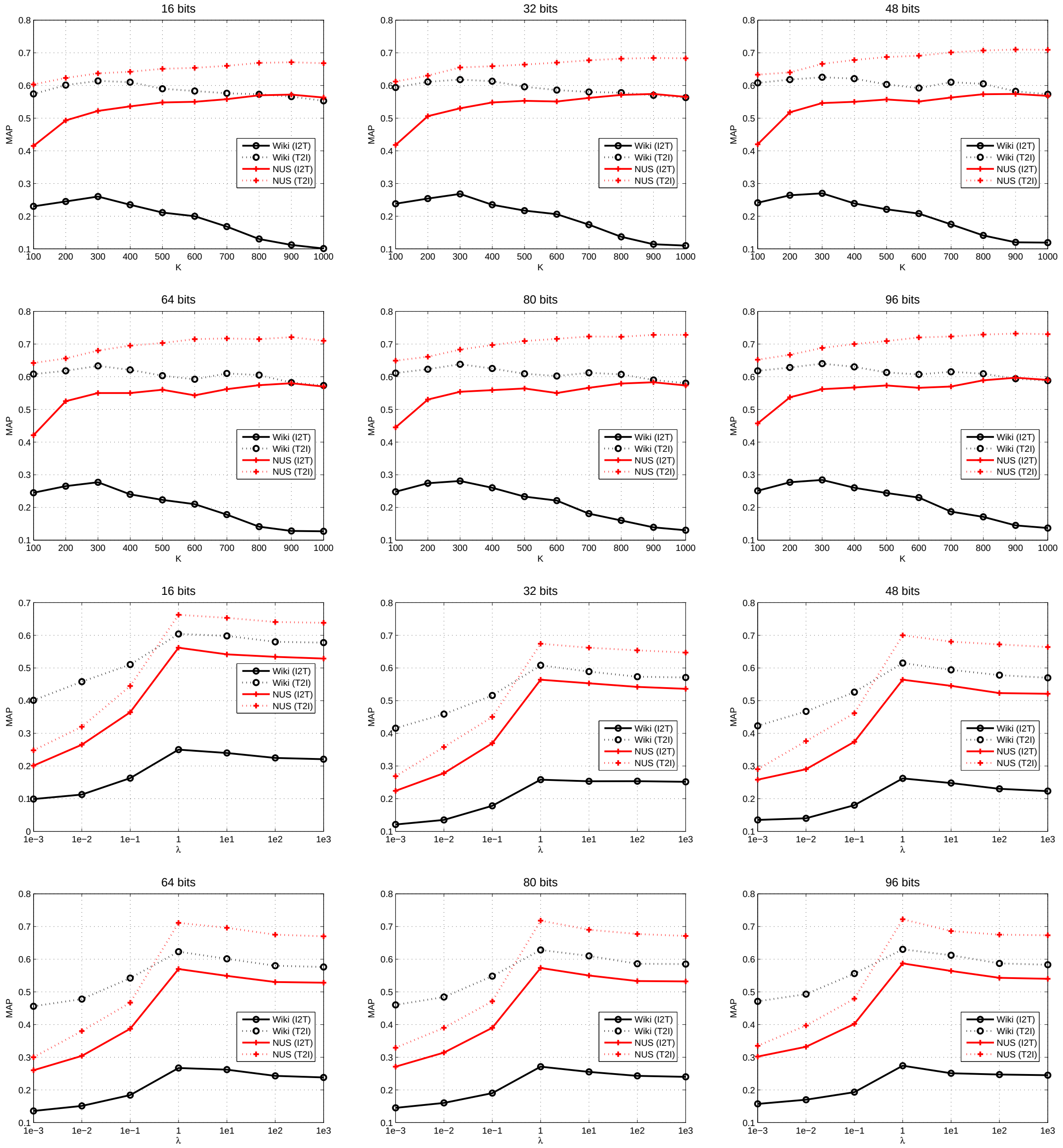
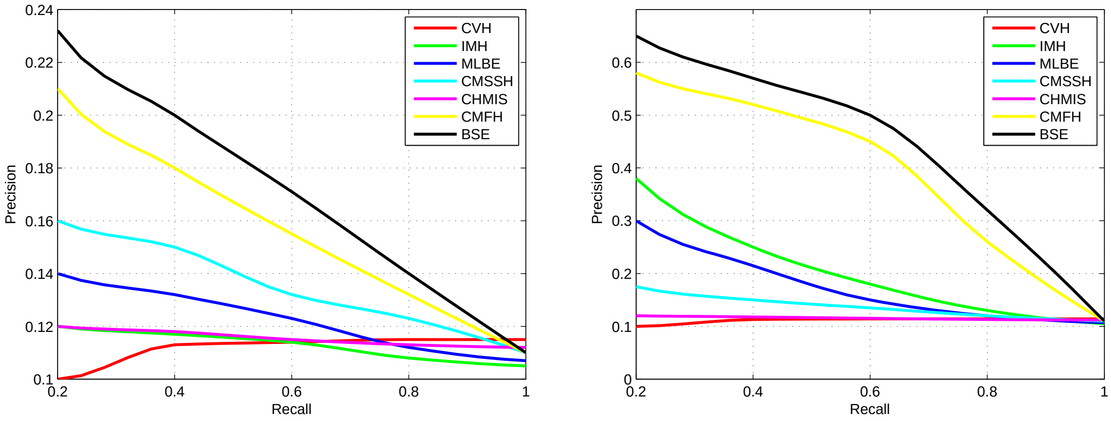


Fig. 4.2 Comparison of the MAP of BSE with respect to parameters K and λ on the Wiki and NUS-WIDE datasets with different bit lengths.

Table 4.2 Computational complexity for the test phase with the 48-bit codes on the Wiki and NUS-WIDE datasets.

Complexity Datasets	Task	Training time (s)	Average coding time (ms) for each local feature	Average querying time (ms) for each image/text	HashTable size (MB)
Wiki	Image to text	987.16	1.62	203.46	6.75
(4×10^4 pairs)	Text to image	-	1.70	97.24	21.63
NUS-WIDE	Image to text	2172.50	1.53	157.20	3.61
(8×10^5 pairs)	Text to image	-	1.79	21.4	254.34



(a) Wiki (I2T)

(b) Wiki (T2I)

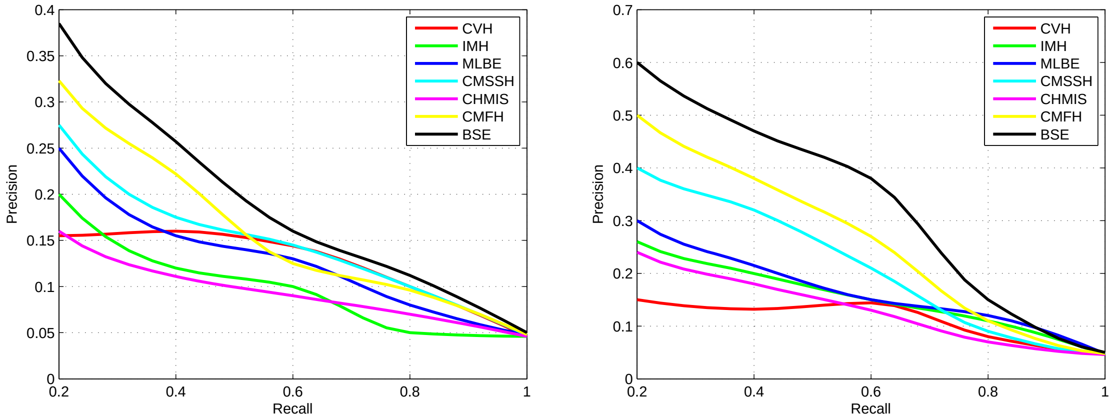
Fig. 4.3 The precision-recall curves of all compared algorithms on the Wiki dataset with the code length of 32 bits.

4.5.3 Results and Discussions

In this section, we will show the compared results of BSE and other methods, the parameter sensitivity analysis and the training size sensitivity analysis, respectively.

Table 4.1 illustrates the MAP on both Wiki and NUS-WIDE datasets. Since we focus on the cross-modal retrieval task, we show the corresponding results on two aspects respectively: image query vs. text database (I2T) and text query vs. image database (T2I). From the table, we can observe that the MAP of a text query is generally higher than that of an image query. The reason is that the text can better describe the semantic meaning of the image-text pairs than the image. Given an image query, since it only describes the low-level visual information, it is difficult to find semantically similar texts for it accurately.

From Table 4.1, it is easy to discover that the searching accuracies from CVH, IMH and MLBE are always fluctuant with the increase of the code length. Specifically, in terms of IMH and MLBE, the best performances are usually achieved with small bits (i.e., 16 and 32 bits, respectively) for both I2T and T2I on two datasets. For CVH, the highest results constantly appear with 16 bits on the Wiki dataset, while the best results are obtained



(a) NUS-WIDE (I2T)

(b) NUS-WIDE (T2I)

Fig. 4.4 The precision-recall curves of all compared algorithms on the NUS-WIDE dataset with the code length of 32 bits.

with 96 bits on the NUS-WIDE dataset. Besides, we can also find that with the code length increasing, the results calculated by CMSSH, CHMIS and CMFH are getting higher on both datasets. In particular, CHMIS always achieves better performance than CMSSH for I2T search, but obtains lower accuracies than CMSSH for T2I search. Since CHMIS is regarded as a cross-view fusion method, it cannot directly compute the separated codes for image and text domains respectively. Thus, the same integrated codes are used for I2T and T2I and give the same performance on these two domains. Different from all above conventional methods, our proposed BSE method successfully considers the relationship between local features on inter/intra data structures and completes retrieval via the local hash based feature indexing scheme. The related results demonstrate our BSE can achieve significantly better performance than CVH, IMH, MLBE, CMSSH, CHMIS and QCH for both I2T and T2I on Wiki and NUS-WIDE datasets and even outperforms the recent CMFH method. It is noticeable that CMFH's results are slightly lower than the results in the original paper [33]. The reason is the use of word vectors which can be trained offline and independent of any specific dataset unlike the provided dataset-oriented LDA representation. In addition, we used 21 most frequent classes of the NUS-WIDE dataset, which is larger than the ten largest concepts used in their paper. Beyond those, the precision-recall curves with the code length of 32 are also shown in Figs. 4.3 and 4.4. By measuring the area under curve (AUC), it can be obviously observed that BSE consistently performs better than other state-of-the-art methods. Moreover, the computational complexity for the test phase with the 48-bit codes on the Wiki and NUS-WIDE datasets is in Table 4.2.

To make our method more convincing, Table 4.3 gives a comparison between the proposed BSE and other cross-modal metric learning methods which also map multiple modalities into a shared space. In particular, we use VLAD to construct global representations for images and texts as mentioned before and then CCA and supervised CCA (SCCA) [170] are utilized to learn the real-valued low-dimensional data for cross-modal retrieval. It is noticeable that the improvements for the text-to-image task are more significant than those obtained for the image-to-text task. The main reason is that the algorithms could gain more precise semantic information from the given text query than the given image query. Text samples and class labels directly reflect the semantic information while understanding complex images with multiple objects is still a difficult task.

4.5.4 Parameter Sensitivity

In this section, we illustrate the sensitivity of two parameters: the number of clusters K and the balance parameter λ, on the Wiki and NUS-WIDE datasets with different bit lengths. We report the best results for a fixed parameter with varying other parameters in Fig. 4.2. As we can see from the figure, the results on two datasets at all different code lengths have the similar tendency. For the parameter K, we can observe that a small value of K ($K = 300$) in the K-means works better for the Wiki dataset with all bit lengths, since it is a relatively small dataset containing only 2173 image-text data with 10 semantic classes for training. While for the NUS-WIDE dataset, the best value of K always tends to be large ($K = 900$) for both I2T and T2I search. Furthermore, from the whole perspective, the tendency of the accuracies on NUS-WIDE with the change of K goes stably, which indicates that our final results are not sensitive to the choice of K. In Fig. 4.2, we also demonstrate the sensitivity of the balance parameter λ. It is discovered that with the increase of λ, the search results always rapidly grow and then slightly drop down with all bit lengths. The best results are usually achieved when $\lambda = 1$ on both Wiki and NUS-WIDE datasets. However, the final accuracies are more sensitive on the NUS-WIDE dataset when λ takes various values compared with those on the Wiki dataset. The comparison has shown the fact that we can reduce the range near the best point in the future tuning of parameters, i.e., the range for tuning K is proportional to the training size and the range for tuning λ is around 1. Additionally, we evaluate the effectiveness of element-to-element structure preserving and set-to-set structure preserving in Eq. (4.15) on both datasets, respectively. From Table 4.4 we can observe that only preserving element-to-element structure (i.e., $\lambda = 0$) or set-to-set structure (i.e., $\lambda = +\infty$) individually cannot achieve the best performance. To further

Table 4.3 MAP comparison with state-of-the-art cross-modal metric learning methods on both datasets.

Dataset	Task	Code length	CCA	SCCA	BSE
Wiki	Image to Text	8	0.171	0.224	0.237
		16	0.178	0.218	0.260
		24	0.180	0.213	0.265
		32	0.179	0.210	0.268
		48	0.175	0.212	0.272
Wiki	Text to Image	8	0.201	0.460	0.608
		16	0.214	0.427	0.614
		24	0.233	0.401	0.615
		32	0.246	0.388	0.618
		48	0.244	0.372	0.625
NUS-WIDE	Image to Text	8	0.428	0.465	0.567
		16	0.420	0.460	0.572
		24	0.413	0.454	0.573
		32	0.404	0.451	0.574
		48	0.397	0.446	0.574
NUS-WIDE	Text to Image	8	0.433	0.472	0.665
		16	0.427	0.470	0.671
		24	0.419	0.465	0.677
		32	0.405	0.453	0.684
		48	0.401	0.448	0.710

All the compared methods (except “BSE") utilize vector of locally aggregated descriptors (VLAD) in this table.

explore the advantages of the orthogonality constraint in Eq. (4.15), the results of BSE without orthogonal projection learning in Section 4.3.5 are also included in Table 4.4, where the learning procedure is similar to CCA.

4.5.5 Training Size Sensitivity

For the training phase, although we always fix the number of the training samples as mentioned in Section 4.5.1, the number of constructed local feature pairs for element-to-element preserving can be varied. Theoretically, more local pairs used in the training phase will lead to better results. If there exists N local features, the maximum number of pairs can be N^2. However, N for large datasets can be over a million. It is infeasible to utilize all the local pairs in training due to the computational costs. Thus, in our experiments, we randomly select a subset of the pairs, which contains 30% positive pairs and 70% negative pairs, sim-

Table 4.4 MAP comparison with different settings of the proposed BSE

Task	Method	Wiki						NUS-WIDE					
		16 bits	**32 bits**	**48 bits**	**64 bits**	**80 bits**	**96 bits**	**16 bits**	**32 bits**	**48 bits**	**64 bits**	**80 bits**	**96 bits**
Image to Text	Only element-to-element structure preserving	0.098	0.121	0.135	0.136	0.145	0.157	0.201	0.224	0.258	0.260	0.271	0.302
	Only set-to-set structure preserving	0.202	0.251	0.223	0.238	0.240	0.245	0.528	0.536	0.521	0.528	0.532	0.540
	BSE without orthogonality constraint	0.244	0.250	0.253	0.259	0.266	0.270	0.546	0.558	0.567	0.573	0.575	0.582
	BSE	**0.260**	**0.268**	**0.272**	**0.277**	**0.281**	**0.284**	**0.572**	**0.574**	**0.574**	**0.580**	**0.583**	**0.597**
Text to Image	Only element-to-element structure preserving	0.401	0.415	0.423	0.456	0.460	0.471	0.248	0.268	0.290	0.300	0.329	0.335
	Only set-to-set structure preserving	0.577	0.570	0.570	0.576	0.583	0.585	0.638	0.647	0.664	0.670	0.671	0.674
	BSE without orthogonality constraint	0.608	0.612	0.618	0.623	0.627	0.634	0.666	0.677	0.692	0.703	0.715	0.720
	BSE	**0.614**	**0.618**	**0.625**	**0.633**	**0.638**	**0.640**	**0.671**	**0.684**	**0.710**	**0.721**	**0.728**	**0.732**

"Only element-to-element structure preserving" refers to $\lambda = 0$ in Eq. (4.15). On the contrary, "Only set-to-set structure preserving" refers to $\lambda = +\infty$ in Eq. (4.15). "BSE without orthogonality constraint" indicates solving Eq. (4.15) under CCA-like solution without orthogonal projection optimization.

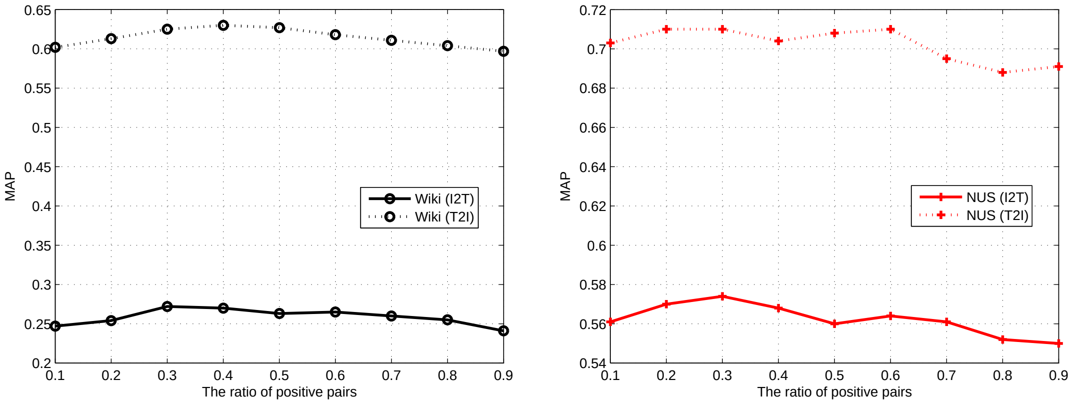
Table 4.5 Effect of Training Pair Size on MAP at 48-bit.

Datasets	Pair Size	Image to Text	Text to Image
Wiki	1×10^4	0.202	0.558
	2×10^4	0.224	0.586
	4×10^4	0.251	0.608
	8×10^4	0.272	0.625
NUS-WIDE	2×10^5	0.493	0.640
	4×10^5	0.518	0.672
	8×10^5	0.541	0.695
	1.6×10^6	0.574	0.710

ilar to [137], during the training phase. Table 4.5 illustrates the corresponding results by varying the number of pairs. Obviously, the proposed BSE can achieve significantly better results when the pair number equals 8×10^4 and 1.6×10^6 on two datasets with the total numbers of local feature pairs 4×10^{11} and 1×10^{13} respectively. In addition, different ratios for the number of positive and negative feature pairs with 48 bits codes on both Wiki and NUS-WIDE datasets are illustrated in Fig. 4.5, as well. From Fig. 4.5, it is observed that the performance on Wiki is quite stable when varying the ratio of positive and negative pairs, while for NUS-WIDE there always exists fluctuation in terms of MAP.

4.5.6 Generalization

In this experiment, we train the hash functions of different methods on the combination of the Wiki and NUS-WIDE datasets. Since the local features from the image and text domains that we used, i.e., SIFT and word vectors, are irrelevant to any specific dataset, we can unite the features of the Wiki and NUS-WIDE datasets together to form a larger dataset. For the



(a) Wiki (b) NUS-WIDE

Fig. 4.5 MAP (48 bits) via different ratios of positive/negative pair construction on the Wiki and NUS-WIDE datasets

Table 4.6 MAP comparison on the combination of the Wiki and NUS-WIDE datasets.

Task	Code length	CVH	IMH	MLBE	CMSSH	CHMIS	CMFH	BSE
Image to Text	16	0.264	0.301	0.306	0.341	0.317	0.351	**0.441**
	32	0.251	0.296	0.313	0.350	0.329	0.358	**0.450**
	48	0.244	0.304	0.315	0.357	0.336	0.367	**0.457**
	64	0.237	0.310	0.311	0.368	0.348	0.407	**0.465**
	80	0.230	0.323	0.320	0.372	0.364	0.397	**0.479**
	96	0.227	0.335	0327	0.380	0.382	0.385	**0.487**
Text to Image	16	0.337	0.501	0.487	0.401	0.317	0.620	**0.681**
	32	0.279	0.493	0.445	0.408	0.329	0.627	**0.693**
	48	0.256	0.481	0.348	0.419	0.336	0.631	**0.703**
	64	0.233	0.458	0.364	0.400	0.348	0.630	**0.736**
	80	0.247	0.453	0.356	0.403	0.364	0.649	**0.738**
	96	0.252	0.444	0.348	0.398	0.382	0.664	**0.742**

global methods, we still use the above VLAD representations. As shown in Table 4.6, the results of almost every method are between the corresponding ones of the Wiki and NUS-WIDE datasets in Table 4.1. Generally, the text-to-image results on the combined dataset are better than the ones on the Wiki dataset since more sufficient semantic information for images can be learned in the larger dataset. In contrast, the image-to-text results on the combined dataset are lower than the ones on the NUS-WIDE dataset for the reason that the images in NUS-WIDE are only with several tags rather than documents and the retrieval results are possibly the words in Wiki. Additionally, our method has significantly outperformed the other state-of-the-art cross-modality hashing methods and improved the text-to-image MAP compared with the results on both datasets.

4.6 Summary

In this chapter, a novel cross-modal hashing scheme called Binary Set Embedding (BSE) has been presented. Aiming for a general representation that is independent of any dataset, we have employed local feature descriptors for both image and text modalities. BSE associates the local feature set of images with the semantic information of the corresponding documents and embeds them into a common Hamming space. Due to the nature of local features, BSE simultaneously preserves the element-to-element and set-to-set structures which are correspondent to the data points and the source information of local features respectively in the intra-model relationship. Extensive results have shown that BSE outperforms state-of-the-art methods in terms of cross-modal retrieval tasks. Our future work aims to generalize our approach to carry out the cross-modal task for data from multiple modalities. In addition, we will collect a new large-scale image-text dataset to achieve more complex and challenging cross-modal retrieval tasks.

Chapter 5

Kernelized Multiview Projection for Multimedia Data Fusion

5.1 Introduction

Traditional feature reduction techniques as the proposed algorithms in previous chapters are mainly based on single feature representations, a.k.a. single-view representation, either global [62, 129] or local [16, 107]. For local methods, descriptors such as SIFT [103] are computed for each detected or densely sampled point, then the Bag-of-Words scheme or its improved version is employed to embed these local features into a holistic representation. On the one hand, local feature based methods tend to be more robust and effective in challenging scenarios, while this kind of representation is often not precise and informative because of the quantization error during the codebook construction and the loss of structural relationships among local features. On the other hand, global representations [30, 114] describe the image as a whole. Unfortunately, global methods are sensitive to shift, scaling, occlusion and cluttering, which commonly exist in realistic images.

Notwithstanding the remarkable results achieved by both local and global methods in some cases, most of them are still based on a single view (feature representation). In realistic applications, variations in lighting conditions, intra-class differences, complex backgrounds and viewpoint and scale changes all lead to obstacles for robust feature extraction. Naturally, single representations cannot handle realistic tasks to a satisfactory extent.

In practice, a typical sample can be represented by different views/features, e.g., gradient, shape, color, texture and motion. Generally speaking, these views from different feature spaces always maintain their particular statistical characteristics. Accordingly, it is desirable to incorporate these heterogeneous feature descriptors into one compact representation, leading to the multiview learning approaches. These techniques have been designed for multiview data classification [189], clustering [13] and feature selection [182]. For such multiview learning tasks, the feature representations are usually very high-dimensional for each view. However, little effort has been paid to learning low-dimensional and compact representations for multiview computer vision tasks. Thus, how to obtain an effective low-dimensional embedding to discover the discriminative information from all views is a worthy research topic, since the effectiveness and efficiency of the methods drop exponentially as the dimensionality increases, which is commonly referred to as the curse of dimensionality.

Existing multiview embedding techniques include the multiview spectral embedding (MSE) [172] and the multiview stochastic neighbor embedding (m-SNE) [174], which have explored the locality information and probability distributions for the fusion of multiview data respectively. Recently, Han et al. [50] proposed a sparse unsupervised dimensionality reduction to obtain a sparse representation for multiview data. However, these methods are only defined on the training data and it remains unclear how to embed the new test data due to their nonlinearity. In other words, they suffer from the *out-of-sample* problem [8], which heavily restricts their applicability in realistic and large-scale vision tasks.

In this chapter, to tackle the *out-of-sample* problem, we propose a novel unsupervised multiview subspace learning method called kernelized multiview projection (KMP), which can successfully learn the projection to encode different features with different weights achieving a semantically meaningful embedding. KMP considers different probabilistic distributions of data points and the locality information among data simultaneously. Instead of using the multiview features directly, the kernel matrices from multiple views enable KMP to normalize the scales and the dimensions of different features. In fact, we show that the fusion of multiple kernels is actually the concatenation of features in the high-dimensional reproducing kernel Hilbert space (RKHS), while the learning phase of KMP remains in the low-dimensional space. Having obtained kernels for each view in RKHS, KMP can not only fuse the views by exploring the complementary property of different views as multiple kernel learning (MKL) [45, 80, 159], but also find a common low-dimensional subspace where the distribution of each view is sufficiently smooth and discriminative.

5.2 Related Work

A simple multiview embedding framework is to concatenate the feature vectors from different views together as a new representation and utilize an existing dimensionality reduction method directly on the concatenated vector to obtain the final multiview representation. Nonetheless, this kind of concatenation is not physically meaningful because each view has a specific characteristic. Besides, the relationship between different views is ignored and the complementary nature of intrinsic data structure of different views is not sufficiently explored.

One feasible solution is proposed in [102] called distributed spectral embedding (DSE). In this multiview scheme, a spectral embedding scheme is first performed on each view, respectively, producing the individual low-dimensional representations. After that, a common compact embedding is finally learned to guarantee that it would be similar with all single-view's representations as much as possible. Although the spectral structure of each view can be effectively considered for learning a multiview embedding via DSE, the complementarity between different views is still neglected.

To effectively and efficiently learn the complementary nature of different views, multi-view spectral embedding (MSE) is introduced in [172]. The main advantage of MSE is that it can simultaneously learn a low-dimensional embedding over all views rather than separate learning as in DSE. Additionally, MSE shows better effectiveness in fusing different views in the learning phase.

However, both DSE and MSE are based on nonlinear embedding, which leads to a serious computational complexity problem and the *out-of-sample* problem [8]. In particular, when we apply them to classification or retrieval tasks, the methods have to be re-trained for learning the low-dimensional embedding when new test data are used. Due to their nonlinearity nature, this will cause heavily computational costs and even become impractical for realistic and large-scale scenarios.

Towards solving the out-of-sample problem for multiview embedding, we propose a unsupervised projection method, namely, KMP. It is noteworthy that, as a linear method, a projection is learned via the proposed KMP using all of the training data. Nevertheless, different from non-linear approaches, once the learning phase finishes, the projection will be fixed and can be directly applied to embed any new test sample without re-training.

5.3 Kernelized Multiview Projection

5.3.1 Notations

Given N training samples $\{S_1, \cdots, S_N\}$ and M different descriptors for multiview feature extraction, $X_p^i \in \mathbb{R}^{D_i}$ represents the feature vector for the i-th view and p-th sample. Since the dimensions of various descriptors are different, kernel matrices $K_1, \cdots, K_M \in \mathbb{R}^{N \times N}$ are constructed by the kernel functions such as the RBF kernel and the polynomial kernel, for the fusion of different views in the same scale. Our task is to output an optimal projection matrix $P \in \mathbb{R}^{N \times d}$ and weights $(\alpha_1, \cdots, \alpha_M)$ satisfying $\sum_{i=1}^{M} \alpha_i = 1$ for kernel matrices such that the fused feature matrix $Y = [\mathbf{y}_1, \cdots, \mathbf{y}_N]^T = KP = (\sum_{i=1}^{M} \alpha_i K_i)P$ can represent original multiview data comprehensively.

The projection learning of KMP is based on the similarity matrix W_i in which $(W_i)_{pq}$ represents the similarity value between the p-th sample and the q-th sample. Generally, the similarity can be calculated by some kernel function. For instance, the heat kernel function is popularly used to measure the similarity between two samples. With the above similarity matrix, we can define the Laplacian matrix $L_i = D_i - W_i$ for the i-th view, $i = 1, 2, \cdots, M$, where D_i is a diagonal matrix with $(D_i)_{pp} = \sum_q (W_i)_{pq}$. However, the mechanisms of similarity computation for images and videos are different since their feature structures are different. We first introduce the learning phase of KMP. Section 5.4 will describe different ways to calculate the similarity matrix for images and videos.

5.3.2 Multiview Kernel Fusion

Due to the complementary nature of different descriptors, we assign different weights for different views. The goal of KMP is to find the basis of a subspace in which the lower-dimensional representation can preserve the intrinsic structure of original data. Therefore, we impose a set of nonnegative weights $\alpha = (\alpha_1, \cdots, \alpha_M)$ on the similarity matrices $W_1, \cdots, W_M$ and we have the fused similarity matrix $W = \sum_{i=1}^{M} \alpha_i W_i$, fused diagonal matrix $D = \sum_{i=1}^{M} \alpha_i D_i$ and the fused Laplacian matrix $L = \sum_{i=1}^{M} \alpha_i L_i$.

For the kernel matrix, we also define the fused kernel matrix $K = \sum_{i=1}^{M} \alpha_i K_i$. In fact, suppose ϕ_i is the substantial feature map for kernel K_i, i.e., $K_i = \phi_i(X^i)^T \phi_i(X^i)$, then the fused kernel value is computed by the feature vector concatenated by the mapped vectors via $\phi_1, \cdots, \phi_M$, since we have

$$\begin{aligned} K &= \sum_{i=1}^{M} \alpha_i K_i = \sum_{i=1}^{M} \alpha_i \phi_i(X^i)^T \phi_i(X^i) \\ &= \begin{bmatrix} \sqrt{\alpha_1}\phi_1(X^1) \\ \vdots \\ \sqrt{\alpha_M}\phi_M(X^M) \end{bmatrix}^T \begin{bmatrix} \sqrt{\alpha_1}\phi_1(X^1) \\ \vdots \\ \sqrt{\alpha_M}\phi_M(X^M) \end{bmatrix} \\ &= \phi(X)^T \phi(X), \end{aligned}$$

where $\phi(\cdot) = [\sqrt{\alpha_1}\phi_1(\cdot)^T, \cdots, \sqrt{\alpha_M}\phi_M(\cdot)^T]^T$ is the fused feature map and $X = (X^1, \cdots, X^M)$ is the M-tuple consisting of features from all the views.

To preserve the fused locality information, we need to find the optimal projection for the following optimization problem:

$$\arg\min_{\mathbf{v}} \sum_{p,q} \|\mathbf{v}^T \psi_p - \mathbf{v}^T \psi_q\|_2^2 (W)_{pq}, \tag{5.1}$$

where ψ_p is the fused mapped feature, i.e., $[\psi_1, \cdots, \psi_N] = \phi(X)$. Through simple algebra derivation, the above optimization problem can be transformed to the following form:

$$\arg\min_{\mathbf{v}} \mathrm{tr}(\mathbf{v}^T \phi(X) L \phi(X)^T \mathbf{v}). \tag{5.2}$$

With the constraint $\mathrm{tr}(\mathbf{v}^T \phi(X) D \phi(X)^T \mathbf{v}) = 1$, minimizing the objective function in Eq. (5.2) is to solve the following generalized eigenvalue problem:

$$\phi(X) L \phi(X)^T \mathbf{v} = \lambda \phi(X) D \phi(X)^T \mathbf{v}. \tag{5.3}$$

Note that each solution of problem (5.3) is a linear combination of $\psi_1, \cdots, \psi_N$, and there exists an N-tuple $\mathbf{p} = (p_1, \cdots, p_N)^T \in \mathbb{R}^N$ such that $\mathbf{v} = \sum_{i=1}^{N} p_i \psi_i = \phi(X)\mathbf{p}$. For matrix V consisting of all the linearly independent solutions of problem (5.3), there exists a matrix P such that $V = \phi(X)P$. Therefore, with the additional constraint $\mathrm{tr}(P^T \phi(X) D \phi(X)^T P) = 1$, we can formulate the new objective function as follows:

$$\begin{aligned}&\arg\min_{P,\alpha} \mathrm{tr}(P^T KLKP)\\&\text{s.t. } \mathrm{tr}(P^T KDKP) = 1,\ \sum_{i=1}^{M} \alpha_i = 1,\ \alpha_i \geq 0,\end{aligned} \tag{5.4}$$

or in the form associated with the norm constraint:

$$\arg\min_{P,\alpha} \frac{\mathrm{tr}(P^T KLKP)}{\mathrm{tr}(P^T KDKP)},\ \text{s.t.} \sum_{i=1}^{M} \alpha_i = 1,\ \alpha_i \geq 0. \tag{5.5}$$

5.3.3 Alternate Optimization via Relaxation

In this section, we employ a procedure of alternate optimization [10] to derive the solution of the optimization problem. To the best of our knowledge, it is difficult to find its optimal solution directly, especially for the weights in (5.5).

First, for a fixed α, finding the optimal projection P is simply reduced to solve the generalized eigenvalue problem

$$KLK\mathbf{p} = \lambda KDK\mathbf{p}, \tag{5.6}$$

and set $P = [\mathbf{p}_1, \cdots, \mathbf{p}_d]$ corresponds to the smallest d eigenvalues based on the Ky-Fan theorem [11].

Next, to optimize α, we derive a relaxed objective function from the original problem. The output of the relaxed function can ensure that the value of the objective function in (5.5) is in a small neighborhood of the true minimum.

We fix the projection P to update α individually. Without loss of generality, we first consider the condition that $M = 2$, i.e., there are only two views. Then the optimization problem (5.5) is reduced to

$$\arg\min_{P,\alpha} \frac{\mathrm{tr}(P^T KLKP)}{\mathrm{tr}(P^T KDKP)},\ \alpha_1 + \alpha_2 = 1,\ \alpha_1, \alpha_2 \geq 0. \tag{5.7}$$

For simplicity, we denote $L_{ijk} = \mathrm{tr}(P^T K_i L_k K_j P)$ and $D_{ijk} = \mathrm{tr}(P^T K_i D_k K_j P)$, $i, j, k \in \{1, 2\}$. Then we can simply find that $L_{ijk} = L_{jik}$ and $D_{ijk} = D_{jik}$.

Relaxation With the Cauchy-Schwarz inequality [51], the relaxation for the objective function in (5.7) is shown in the following equation:

$$
\begin{aligned}
\frac{\mathrm{tr}(P^T KLKP)}{\mathrm{tr}(P^T KDKP)} &= \frac{\mathrm{tr}\left(P^T(\alpha_1 K_1+\alpha_2 K_2)(\alpha_1 L_1+\alpha_2 L_2)(\alpha_1 K_1+\alpha_2 K_2)P\right)}{\mathrm{tr}\left(P^T(\alpha_1 K_1+\alpha_2 K_2)(\alpha_1 L_1+\alpha_2 L_2)(\alpha_1 K_1+\alpha_2 K_2)P\right)} \\
&= \frac{\alpha_1^3 L_{111}+2\alpha_1^2\alpha_2 L_{121}+\alpha_1\alpha_2^2 L_{221}+\alpha_1^2\alpha_2 L_{112}+2\alpha_1\alpha_2^2 L_{122}+\alpha_2^3 L_{222}}{\alpha_1^3 D_{111}+2\alpha_1^2\alpha_2 D_{121}+\alpha_1\alpha_2^2 D_{221}+\alpha_1^2\alpha_2 D_{112}+2\alpha_1\alpha_2^2 D_{122}+\alpha_2^3 D_{222}} \\
&\le \frac{1}{\alpha_1^3 L_{111}+2\alpha_1^2\alpha_2 L_{121}+\alpha_1\alpha_2^2 L_{221}+\alpha_1^2\alpha_2 L_{112}+2\alpha_1\alpha_2^2 L_{122}+\alpha_2^3 L_{222}} \\
&\quad\times\left(\frac{(\alpha_1^3 L_{111})^2}{\alpha_1^3 D_{111}}+\frac{(2\alpha_1^2\alpha_2 L_{121})^2}{2\alpha_1^2\alpha_2 D_{121}}+\frac{(\alpha_1\alpha_2^2 L_{221})^2}{\alpha_1\alpha_2^2 D_{221}}+\frac{(\alpha_1^2\alpha_2 L_{112})^2}{\alpha_1^2\alpha_2 D_{112}}+\frac{(2\alpha_1\alpha_2^2 L_{122})^2}{2\alpha_1\alpha_2^2 D_{122}}+\frac{(\alpha_2^3 L_{222})^2}{\alpha_2^3 D_{222}}\right) \\
&= \frac{1}{\alpha_1^3 L_{111}+2\alpha_1^2\alpha_2 L_{121}+\alpha_1\alpha_2^2 L_{221}+\alpha_1^2\alpha_2 L_{112}+2\alpha_1\alpha_2^2 L_{122}+\alpha_2^3 L_{222}}\times\left(\alpha_1^3 L_{111}\frac{L_{111}}{D_{111}}\right. \\
&\quad\left.+2\alpha_1^2\alpha_2 L_{121}\frac{L_{121}}{D_{121}}+\alpha_1\alpha_2^2 L_{221}\frac{L_{221}}{D_{221}}+\alpha_1^2\alpha_2 L_{112}\frac{L_{112}}{D_{112}}+2\alpha_1\alpha_2^2 L_{122}\frac{L_{122}}{D_{122}}+\alpha_2^3 L_{222}\frac{L_{222}}{D_{222}}\right) \\
&= \sum_{i,j,k\in\{1,2\}} w_{ijk}(\alpha_1,\alpha_2)\frac{L_{ijk}}{D_{ijk}},
\end{aligned}
\tag{5.8}
$$

where w_{ijk} is the coefficient of $\frac{L_{ijk}}{D_{ijk}}$ and $\sum_{i,j,k\in\{1,2\}} w_{ijk} = 1$. In this way, the objective function in (5.7) is relaxed to a weighted sum of $\frac{L_{ijk}}{D_{ijk}}$. Thus, minimizing the weighted sum of the right-hand-side in (5.8) can lower the objective function value in (5.7). Note that

$$
\alpha_1^2\alpha_1 = \frac{1}{2}\alpha_1\cdot\alpha_1\cdot 2\alpha_2 \le \frac{1}{2}\left(\frac{\alpha_1+\alpha_1+2\alpha_2}{3}\right)^3 = \frac{4}{27},
$$

and then the weights without containing α_1^3 and α_2^3 are always smaller than a constant. Therefore, we only ensure that a part of the terms in the weighted sum is minimized, i.e., to solve the following optimization problem:

$$
\underset{\alpha_1,\alpha_2}{\arg\min}\, w_{111}\frac{L_{111}}{D_{111}}+w_{222}\frac{L_{222}}{D_{222}},\ \text{s.t. } w_{111}+w_{222}=1. \tag{5.9}
$$

Since w_{111} and w_{222} are the functions of (α_1,α_2), we first find the optimal weights without parameters (α_1,α_2). To avoid trivial solution, we assign an exponent $r>1$ for each weight.

By denoting $\gamma_1 = w_{111}$ and $\gamma_2 = w_{222}$, the relaxed optimization will be

$$\arg\min_{\gamma_1,\gamma_2} \gamma_1^r \frac{L_{111}}{D_{111}} + \gamma_2^r \frac{L_{222}}{D_{222}}, \text{ s.t. } \gamma_1 + \gamma_2 = 1, \gamma_1, \gamma_2 \geq 0. \tag{5.10}$$

For (5.10), we have the Lagrangian function with the Lagrangian multiplier η:

$$L(\gamma_1, \gamma_2, \eta) = \gamma_1^r \frac{L_{111}}{D_{111}} + \gamma_2^r \frac{L_{222}}{D_{222}} - \eta(\gamma_1 + \gamma_2 - 1). \tag{5.11}$$

We only need to set the derivatives of L with respect to γ_1, γ_2 and η to zeros as follows:

$$\frac{\partial L}{\partial \gamma_1} = r\gamma_1^{r-1} \frac{L_{111}}{D_{111}} - \eta = 0, \tag{5.12}$$

$$\frac{\partial L}{\partial \gamma_2} = r\gamma_2^{r-1} \frac{L_{222}}{D_{222}} - \eta = 0, \tag{5.13}$$

$$\frac{\partial L}{\partial \eta} = \gamma_1 + \gamma_2 - 1 = 0. \tag{5.14}$$

Then γ_1 and γ_2 can be calculated by

$$\begin{aligned} \gamma_1 &= \frac{(L_{222}D_{111})^{\frac{1}{r-1}}}{(L_{222}D_{111})^{\frac{1}{r-1}} + (L_{111}D_{222})^{\frac{1}{r-1}}}, \\ \gamma_2 &= \frac{(L_{111}D_{222})^{\frac{1}{r-1}}}{(L_{222}D_{111})^{\frac{1}{r-1}} + (L_{111}D_{222})^{\frac{1}{r-1}}}. \end{aligned} \tag{5.15}$$

Having acquired γ_1 and γ_2, we can obtain α_1 and α_2 by the corresponding relationship between the coefficients of the functions in (5.9) and (5.10):

$$\frac{\alpha_1^3 L_{111}}{\alpha_2^3 L_{222}} = \frac{w_{111}}{w_{222}} = \frac{\gamma_1^r}{\gamma_2^r}. \tag{5.16}$$

With the constraint $\alpha_1 + \alpha_2 = 1$, we can easily find that

$$\begin{aligned} \alpha_1 &= \frac{(\gamma_1^r L_{222})^{\frac{1}{3}}}{(\gamma_1^r L_{222})^{\frac{1}{3}} + (\gamma_2^r L_{111})^{\frac{1}{3}}}, \\ \alpha_2 &= \frac{(\gamma_2^r L_{111})^{\frac{1}{3}}}{(\gamma_1^r L_{222})^{\frac{1}{3}} + (\gamma_2^r L_{111})^{\frac{1}{3}}}. \end{aligned} \tag{5.17}$$

Hence, for the general M-view situation, we also have the corresponding relaxed problems:

$$\underset{\sum_{i=1}^{M}\alpha_i=1}{\arg\min} \sum_{i,j,k\in\{1,\cdots,M\}} w_{ijk}(\alpha_1,\cdots,\alpha_M)\frac{L_{ijk}}{D_{ijk}} \tag{5.18}$$

and

$$\underset{\gamma_1,\cdots,\gamma_M}{\arg\min} \sum_{i=1}^{M}\gamma_i^r\frac{L_{iii}}{D_{iii}}, \text{ s.t. } \sum_{i=1}^{M}\gamma_i=1,\ \gamma_i\geq 0. \tag{5.19}$$

The coefficients $(\gamma_1,\cdots,\gamma_M)$ and $(\alpha_1,\cdots,\alpha_M)$ can be obtained in similar forms:

$$\gamma_i=\frac{(D_{iii}/L_{iii})^{\frac{1}{r-1}}}{\sum_{j=1}^{M}(D_{jjj}/L_{jjj})^{\frac{1}{r-1}}},\ i=1,\cdots,M \tag{5.20}$$

and

$$\alpha_i=\frac{(\gamma_i^r/L_{iii})^{\frac{1}{3}}}{\sum_{j=1}^{M}(\gamma_j^r/L_{jjj})^{\frac{1}{3}}},\ i=1,\cdots,M. \tag{5.21}$$

Convergence Although the weight α obtained in the above procedure is not the global minimum, the objective function is ensured in a range of small values. We let F_1 and F_2 be the objective functions in (5.5) and (5.18), respectively, and let

$$F_3=\sum_{i=j=k} w_{ijk}\frac{L_{ijk}}{D_{ijk}}=\sum_{i=1}^{M} w_{iii}\frac{L_{iii}}{D_{iii}}. \tag{5.22}$$

We can find that $F_1\leq F_2$ and if there exists $\alpha_i=1$ for some i, then $F_1=F_2=F_3$. During the alternate procedure, for optimizing P, F_1 is minimized, and for optimizing α, F_3 is minimized. Denote $m_1=\max(F_1-F_3)$ and $(P_1,\alpha_1)=\arg\max(F_1-F_3)$, then we have

$$\min F_3+m_1\leq F_3(P_1,\alpha_1)+(F_1-F_3)(P_1,\alpha_1)=F_1(P_1,\alpha_1)\leq\max F_1,$$

and we can define the following nonnegative continuous function:

$$F_4(P,\alpha)=\max\Big(F_1(P,\alpha),\min_{\alpha}\big(F_3(P,\alpha)+m_1\big)\Big). \tag{5.23}$$

Note that $\min_\alpha\big(F_3(P,\alpha)+m_1\big)$ is independent of α, thus for any P, there exists α_0, such that $F_1(P,\alpha_0)=\min_\alpha\big(F_3(P,\alpha)+m_1\big)$. If we impose the above alternate optimization on F_4, F_4 is nonincreasing and therefore converges. Though α does not converge to a fixed point, the value of F_1 is reduced into a small district, which is smaller than $\min_\alpha F_3$ plus a constant. It is also worthwhile to note that F_3 is actually the weighted sum of the objective functions for preserving each view's locality information. However, the optimization for F_3 still learns information from each view separately, i.e., the locality similarity is not fused. We summarize the KMP in Algorithm 6.

During the testing phase, having acquired the data from each view $X^1_{test}, \cdots, X^M_{test}$ of a test video sequence v_{test}, we first compute the kernel values to form the representation of v_{test} in RKHS of each view:

$$\mathbf{k}^i_{test} = (k_i(v_1, v_{test}), \cdots, k_i(v_N, v_{test})),\ i = 1, \cdots, M,$$

where $k_i(\cdot,\cdot)$ is the kernel function. Using the weights $(\alpha_1, \cdots, \alpha_M)$ optimized by Algorithm 1, we have the fused representation of v_{test}: $\mathbf{k}_{test} = \sum_{i=1}^M \alpha_i \mathbf{k}^i_{test}$. Then the final fused representation of v_{test} in the reduced space is $\mathbf{y}_{test} = \mathbf{k}_{test} P$.

Algorithm 6 Kernelized Multiview Projection

```
Input: The training samples $\{S_1, \cdots, S_N\}$ and parameter $r > 1$.
Output: The projection matrix $P \in \mathbb{R}^{N \times d}$ and the weights $\alpha = (\alpha_1, \cdots, \alpha_M) \in \mathbb{R}^M$ for
    kernel matrices.
Extract multiple features from each training image and obtain data matrices $X^i_p$, $p = 1, \cdots, N$, $i = 1, \cdots, M$;
Compute the similarity matrices $W_1, \cdots, W_M$ and the Laplacian matrices $L_1 \cdots, L_M$ for each view;
Compute the kernel matrices $K_1, \cdots, K_M \in \mathbb{R}^{N \times N}$ and the Laplacian matrices $L_1, \cdots, L_M \in \mathbb{R}^{N \times N}$ for $M$ views;
Initialize $\alpha \leftarrow (\frac{1}{M}, \cdots, \frac{1}{M})$;
repeat
    Compute the fused kernel matrix $K = \sum_{i=1}^M \alpha_i K_i$ and the fused Laplacian matrix $L = \sum_{i=1}^M \alpha_i L_i$;
    Compute $P$ by solving the generalized eigenvalue problem (5.6);
    Compute coefficients $\gamma = (\gamma_1, \cdots, \gamma_M)$ by Eq. (5.20);
    Transform $\gamma$ to $\alpha$ by Eq. (5.21);
until $F_4$ defined in Eq. (5.23) converges.
```

5.4 Similarity Construction

5.4.1 Computation for Images

For each view of images, we value the similarity of each sample pair by using the neighbors of each point. The construction of W_i is illustrated below via the ℓ^1-graph [26], which is demonstrated to be robust to data noise, automatically sparse and adaptive to the neighborhood.

For each X_p^i, we find the coefficients $\boldsymbol{\beta} \in \mathbb{R}^{N-1}$ such that $X_p^i = B\boldsymbol{\beta}$, where

$$B = [X_1^i, \cdots, X_{p-1}^i, X_{p+1}^i, \cdots, X_N^i] \in \mathbb{R}^{D_i \times (N-1)}.$$

Considering the noise effect, we can rewrite it as $X_p^i = B'\boldsymbol{\beta}'$, where $B' = [B, I] \in \mathbb{R}^{D_i \times (D_i+N-1)}$ and $\boldsymbol{\beta}' \in \mathbb{R}^{D_i+N-1}$. Thus, seeking the sparse representation for X_p^i leads to the following optimization problem:

$$\arg\min_{\boldsymbol{\beta}'} \|X_p^i - B'\boldsymbol{\beta}'\|_2, \text{ s.t. } \|\boldsymbol{\beta}'\|_1 < \varepsilon, \tag{5.24}$$

where ε is the parameter with a small value. This problem can be solved by the orthogonal matching pursuit [116].

Considering different probabilistic distributions that exist over the data points and the natural locality information of the data, we first employ the Gaussian mixture model (GMM) on the training data for each view. On the one hand, it has been proved that data in the high-dimensional space do not always follow the same distribution, but are naturally clustered into several groups. On the other hand, realistic data distributions basically follow the same form, i.e., Gaussian distribution. In this case, G clusters are obtained by the unsupervised GMM clustering for each view. Thus, we can solve the above problem (5.24) using the data from the same cluster to represent each point rather than the whole data points B, which is also regarded as a solution to alleviate the computational complexity of problem (5.24).

In particular, for $\boldsymbol{\beta}' = (\beta_1, \cdots, \beta_{D_i+N-1})$, we can first set $\beta_q = 0$ if X_q^i and X_p^i are in different clusters, $\forall q \neq p$, then solve the above problem. Now the similarity matrix $W_i \in \mathbb{R}^{N\times N}$ can be defined as: $(W_i)_{pp} = 0$, $\forall p$, $(W_i)_{pq} = |\beta_q|$ if $q < p$, and $(W_i)_{pq} = |\beta_{q-1}|$ if $q > p$. To ensure the symmetry, we update $W_i \leftarrow (W_i^T + W_i)/2$. Then we set the diagonal matrix $D_i \in \mathbb{R}^{N\times N}$ with $(D_i)_{pp} = \sum_q (W_i)_{pq}$ and the Laplacian matrix $L_i = D_i - W_i$ for each view i.

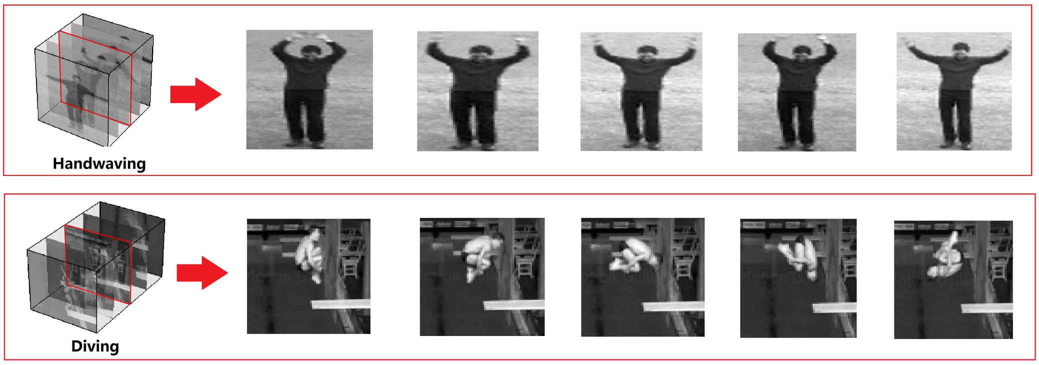


Fig. 5.1 Illustration of selected middle frames from actions “Handwaving" and “Diving".

5.4.2 Computation for Videos

Incremental Naive Bayes Keyframe Selection

In a video sequence, however, not all of the poses are informative and discriminative for action recognition. Some poses may carry neither complete nor accurate information and would even contain common patterns shared by various action types. Since these poses in a video sequence cannot represent the action well and would cause confusion during the classification phase, a weakly supervised method, termed Incremental Naive Bayes Filter (INBF), has been carried out to filter the noisy representation and keep the relatively representative and discriminative poses, i.e., the key poses.

For each action category, ten action sequences are randomly selected. We choose a small set of discriminative poses for a certain action type from each action sequence as the INBF initial positive samples (labeled as $y = 1$), and the remaining frames are adopted as the negative ones ($y = 0$). As illustrated in Fig. 5.1, the five frames in the middle of an action sequence are selected as discriminative poses. We repetitively apply the above procedure to each action type. INBF is then regarded as an unsupervised online learning strategy.

For the i-th feature view, the representation of each pose (frame) s can be written as $\mathbf{x}^i(s) = (x^i_1(s), \cdots, x^i_D(s)) \in \mathbb{R}^D$. Since all the features we extracted are based on statistical histograms, we assume all elements in x^i are independently distributed and model them with a naive Bayes classifier:

$$P(\mathbf{x}^i) = \log \frac{\Pi_{m=1}^{D} \Pr(x^i_m|y=1)\Pr(y=1)}{\Pi_{m=1}^{D} \Pr(x^i_m|y=0)\Pr(y=0)} = \sum_{m=1}^{D} \log \frac{\Pr(x^i_m|y=1)}{\Pr(x^i_m|y=0)}. \tag{5.25}$$

Note that we make the assumption of a uniform prior, i.e., $\Pr(y=1) = \Pr(y=0)$, and

$y \in \{0,1\}$ is a binary variable which represents the negative and positive sample labels, respectively.

Furthermore, in either statistics or physics, real-world data distribution empirically follows the same form, i.e., Gaussian distribution. Thus, the conditional distributions $x_m^i|y=1$ and $x_m^i|y=0$ in the classifier $P(\mathbf{x}^i)$ are assumed to be Gaussian distributed with the four-tuple $(\mu_{y=1}^m, \mu_{y=0}^m, \sigma_{y=1}^m, \sigma_{y=0}^m)$, which satisfy

$$x_m^i|y=1 \sim N(\mu_{y=1}^m, \sigma_{y=1}^m) \text{ and } x_m^i|y=0 \sim N(\mu_{y=0}^m, \sigma_{y=0}^m).$$

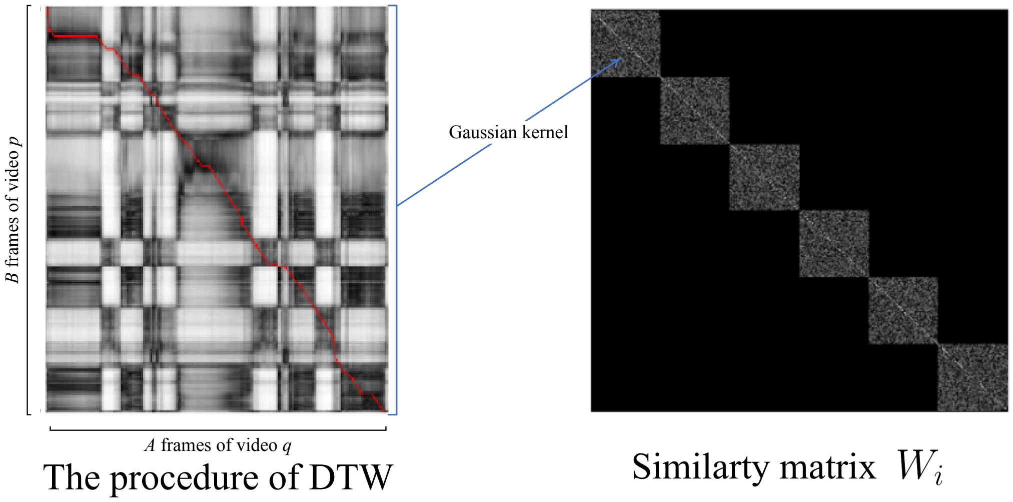
Up to now, for a certain feature view, we can initialize a group of naive Bayes models for each action type, and the training sequence is successively employed through all the models. The Gaussian parameters in INBF can be then incrementally updated as follows:

$$\begin{aligned} &\mu_{y=1}^m \leftarrow \lambda\mu_{y=1}^m + (1-\lambda)\mu_{y=1}, \\ &\sigma_{y=1}^m \leftarrow \sqrt{\lambda(\sigma_{y=1}^m)^2 + (1-\lambda)(\sigma_{y=1})^2 + \lambda(1-\lambda)(\mu_{y=1}^m - \mu_{y=1})^2}, \end{aligned} \tag{5.26}$$

where $\mu_{y=1} = \frac{1}{S}\sum_{s|y(s)=1} x_m^i(s)$, $\sigma_{y=1} = \sqrt{\frac{1}{S}\sum_{s|y(s)=1}(x_m^i(s) - \mu_{y=1})^2}$, $\lambda > 0$ denotes the learning rate of INBF, and $S = |\{s|y(s)=1\}|$. And $\mu_{y=0}^m$ and $\sigma_{y=0}^m$ have similar update rules. The above solutions are easily obtained by maximum likelihood estimation. In this way, we can use INBF to keep the representative frames for the later learning phase and discard irrelevant frames to decrease the influence of noise. The process of INBF is summarized in Algorithm 7.

Algorithm 7 Incremental Naive Bayes Keyframe Selection

```
Input: 10 randomly selected action sequences from each category; the total number of
    actions in each category $N_c$.
Output: The selected keyframes for action sequences.
Manually select 5 representative frames from each sequence of the target category as
    the positive samples and label them as $y = 1$, otherwise $y = 0$;
for $m = 1, \cdots, N_c$ do
    Calculate $\mu_{y=1}^m$, $\sigma_{y=1}^m$, $\mu_{y=0}^m$ and $\sigma_{y=0}^m$;
    Update $\mu_{y=1}^{m+1} = \lambda\mu_{y=1}^m + (1-\lambda)\mu_{y=1}$;
    Update $\sigma_{y=1}^{m+1} = \sqrt{\lambda(\sigma_{y=1}^m)^2 + (1-\lambda)(\sigma_{y=1})^2 + \lambda(1-\lambda)(\mu_{y=1}^m - \mu_{y=1})^2}$;
    Update $\mu_{y=0}^m$ and $\sigma_{y=0}^m$ by using similar rules;
end for
return The cleaned action sequence for each target action category.
```



The procedure of DTW

Similarty matrix W_i

Fig. 5.2 Illustration of the similarity matrix construction.

RBF Sequential Kernel Construction

For the i-th view, since we extract features from the frames of video sequences, each video sequence can be described by a set of features with a sequential order (along the temporal axis). The similarity between video v_p and video v_q under view i: $k_i(v_p, v_q)$ can be measured via Dynamic Time Warping (DTW) [9]. Therefore, the kernel function can be defined as: $k_i(v_p, v_q) = \exp(-\frac{DTW(X_p^i, X_q^i)^2}{2\sigma^2})$, where $DTW(X_p^i, X_q^i)$ indicates the sequential distance computed via DTW and σ is a standard deviation in the RBF kernel. In this way, we can easily obtain the kernel matrices for different views using the above equation.

Similarity Calculation

Based on the above kernel construction, we can obtain kernel matrices $K_1, \cdots, K_M \in \mathbb{R}^{N \times N}$ with the same size for M views with different dimensions. Furthermore, we use the label of training video sequences to supervise the calculation of the similarity matrix W_i for the i-th view. Then each component of W_i is computed as follows:

$$(W_i)_{pq} = \begin{cases} \exp(-\frac{DTW(X_p^i, X_q^i)^2}{2\sigma^2}), & C(p) = C(q) \\ 0, & otherwise \end{cases}, \tag{5.27}$$

where $C(p)$ is the label function which indicates the label of video v_p and $p, q = 1, \cdots, N$. In fact, the similarity matrix W_i is a block matrix consisting of some submatrices of kernel matrix K_i as illustrated in Fig. 5.2. Then we have the diagonal matrix D_i in which $(D_i)_{pp} = \sum_q (W_i)_{pq}$ and the Laplacian matrix $L_i = D_i - W_i$ for each view i.

5.5 Experiments on Image Classification

In this section, we evaluate our Kernelized Multiview Projection (KMP) on three image datasets: CMU PIE, CIFAR10 and SUN397 respectively. The **CMU PIE** face dataset [21] contains $41,368$ images from 68 subjects (people). Following the settings in [21], we select $11,554$ front face images, which are manually aligned and cropped into 32×32 pixels. Further, $7,500$ images are used as the training set and the remaining $4,054$ images are used for testing. The **CIFAR10** dataset [156] is a labeled subset of the 80-million tiny images collection. It consists of a total of $60,000$ 32×32 color images in 10 classes. The entire dataset is partitioned into two parts: a training set with $50,000$ samples and a test set with $10,000$ samples. The **SUN397** dataset [173] contains $108,754$ scene images in total from 397 well-sampled categories with at least 100 images per category. We randomly select 50 samples from each category to construct the training set and the rest of samples are the test set. Thus, there are $19,850$ and $88,904$ images in the training set and test set, respectively.

5.5.1 Compared Methods and Settings

For image classification, each image can be usually described by different feature representations, i.e., multiview representation, in high-dimensional feature spaces. In this paper, we adopt four different feature representations: HOG [30], LBP [3], ColorHist and GIST [114] to describe each image. Table 5.1 illustrates the original dimensions of these features.

We compare our proposed KMP with two related multi-kernel fusion methods. In particular, the RBF kernels[1] for each view are adopted in the proposed KMP method:

$$K = \sum_{i=1}^{M} \alpha_i K_i,$$

where the weight α_i is obtained via alternate optimization. AM indicates that the kernels

[1] Our approach can work with any legitimate kernel function, though we focus on the popular RBF kernel in this paper

Table 5.1 Dimensions of four features for image classification.

Feature representation	**Dimension**
Histogram of oriented gradients (HOG)	225
Local binary pattern (LBP)	256
Color histogram (ColorHist)	192
GIST	384
Total dimension	1057

are combined by arithmetic mean:

$$K_{AM} = \frac{1}{M}\sum_{i=1}^{M} K_i,$$

and GM denotes the combination of kernels through geometric mean:

$$K_{GM} = (\prod_{i=1}^{M} K_i)^{\frac{1}{M}}.$$

Besides, we also include the best performance of the single-view-based spectral projection (BSP), the average performance of the single-view-based spectral projection (ASP) and the concatenation of single-view-based embeddings (CSP) in our compared experiments. In particular, AM and GM are incorporated with the proposed KMP framework. BSP, ASP and CSP are based on the kernelized extension of Discriminative Partition Sparsity Analysis (DPSA) [93] technique. In addition, two non-linear embedding methods, distributed spectral embedding (DSE) and multiview spectral embedding (MSE), are adopted in our comparison, as well. In DSE and MSE, the Laplacian eigenmap (LE) [7] is adopted. For all these compared embedding methods, the RBF-SVM is adopted to evaluate the final performance.

All of the above methods are evaluated on seven different code lengths: $\{20, 30, \cdots, 80\}$. Under the same experimental setting, all the parameters used in the compared methods have been strictly chosen according to their original papers. For KMP and MSE, the optimal balance parameter r for each dataset is selected from one of $\{2, 3, \cdots, 10\}$, which yields the best performance by 10-fold cross-validation on the training set. The number of the GMM clusters G in KMP is selected from one of $\{10, 20, \cdots, 100\}$ with a step of 10 via cross-validation on the training data. The same procedure occurs on the selection of sparsity

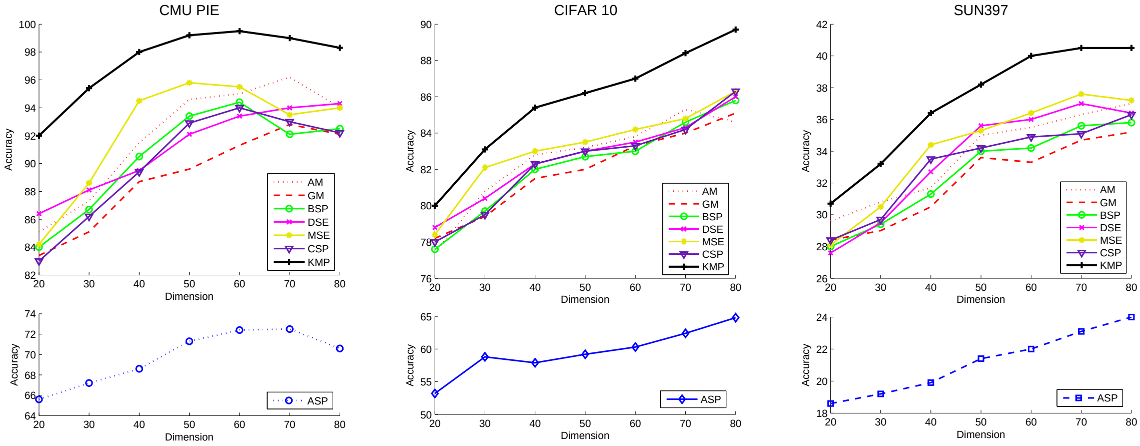


Fig. 5.3 Performance comparison (%) of KMP with different multiview embedding methods on the three datasets.

hyperparameter ε from one of $\{5,8,10,12,15,18,20\}$. The best smooth parameter σ in the construction of the RBF kernel and RBF-SVM is also chosen by the cross-validation on the training data. Since the clustering procedure has uncertainty, all experiments are performed five times repeatedly and each of the results in the following section is the averages of five runs.

5.5.2 Results

In Table 5.2, we first illustrate the performance of the original single-view representations on all the three datasets. In detail, we extract original feature representations under one certain view and then directly feed them to the SVM for classification. From the comparison, we can easily observe that the GIST features consistently outperform the other descriptors on the CMU PIE and CIFAR10 datasets but HOG takes the superior place on the SUN397 dataset. The lowest accuracy is always obtained by ColorHist. Furthermore, we also include the long representation, which is concatenated by all the four original feature representations, into this comparison. It is shown that in most of the time the concatenated representation can reach better performance than single view representations, but is always significantly worse than the proposed KMP. Additionally, the results of the multiple kernel learning based on SVM (MKL-SVM) [45] are listed in Table 5.2 using the same four feature descriptors. Specifically, the best accuracies achieved by KMP are 99.5%, 89.7% and 40.5% on the CMU PIE, CIFAR10, and SUN397, respectively.

In Fig. 5.3, seven different embedding schemes are compared with the proposed KMP

Table 5.2 Performance comparison (%) between the SVM using multiple features through KMP and the SVM using single original features. The numbers in parentheses indicate the dimensions of the representations. For MKL-SVM, ℓ^1-graph is also used to construct the kernel matrix for each view and then MKL-SVM is applied to final classification.

Method \ **Dataset**	**CMU PIE**	**CIFAR10**	**SUN397**
HOG	83.3	70.2	29.3
LBP	74.6	54.2	20.4
ColorHist	31.2	23.0	9.3
GIST	94.2	82.3	17.5
Concatenation	93.4	82.8	31.9
MKL-SVM	95.6	86.3	30.7
KMP	**99.5**(60)	**89.7**(80)	**40.5**(70)

on all the three datasets. From the comparison, the proposed KMP always leads to the best performance for image classification. Meanwhile, arithmetic mean (AM) and the single-view-based spectral projection (BSP) generally achieve higher accuracies than the best performance of geometric mean (GM) and the average performance of the single-view-based spectral projection (ASP). The concatenation of single-view-based embeddings (CSP) achieves competitive performance compared with BSP on all the three datasets. DSE always produces worse performance than MSE and sometimes even obtains lower results than CSP. However, DSE generates better performance than GM and ASP, since a more meaningful multiview combination scheme is adopted in DSE. Beyond that, it is obviously observed that, with different target dimensions, there are large differences among the final results. Fig. 5.4 plots the low-dimensional embedding results obtained by AM, GM, KMP, DSE and MSE on the CIFAR10 dataset. Our proposed KMP can well separate different categories, since it takes the semantically meaningful data structure of different views into consideration for embedding.

In addition, we can observe that with the increase of the dimension, all the curves of compared methods on the CIFAR10 and SUN397 datasets are climbing up except for DSE and MSE, both of which have a slight decrease on SUN397 when the dimension exceeds 70. However, on the CMU PIE dataset, the results in comparison always climb up then go down for almost every compared method except for DSE when the length of dimension increases (see Fig. 5.3). For instance, the highest accuracy on the CMU PIE dataset is on the dimension of 60 and the best performance on CIFAR10 and SUN397 happens when $d = 80$ and $d = 70$, respectively.

Table 5.3 Performance (%) of KMP with different r values on the CMU PIE dataset.

Dimension	r=2	r=3	r=4	r=5	r=6	r=7	r=8	r=9	r=10
d=20	87.0	87.0	87.5	87.8	**88.9**	88.7	88.0	88.0	87.4
d=30	89.4	90.1	90.5	91.0	91.3	**91.4**	**91.4**	90.7	89.3
d=40	87.2	89.0	89.4	91.2	92.0	93.5	93.5	**93.7**	93.2
d=50	84.8	95.1	95.5	96.0	96.4	97.3	**98.2**	97.9	97.5
d=60	97.3	97.5	98.4	98.2	98.7	99.2	99.6	**99.8**	99.7
d=70	96.2	96.4	96.9	97.2	97.9	98.2	98.5	**99.0**	98.7
d=80	96.5	96.8	97.2	97.5	97.1	97.4	98.0	98.3	**98.6**

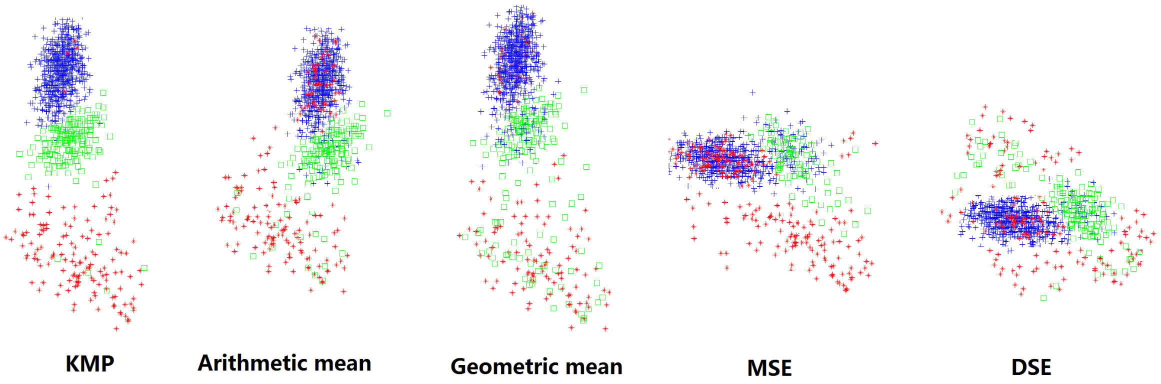


Fig. 5.4 Illustration of low-dimensional distributions of five different fusion schemes (illustrated with data of three categories from the CIFAR10 dataset).

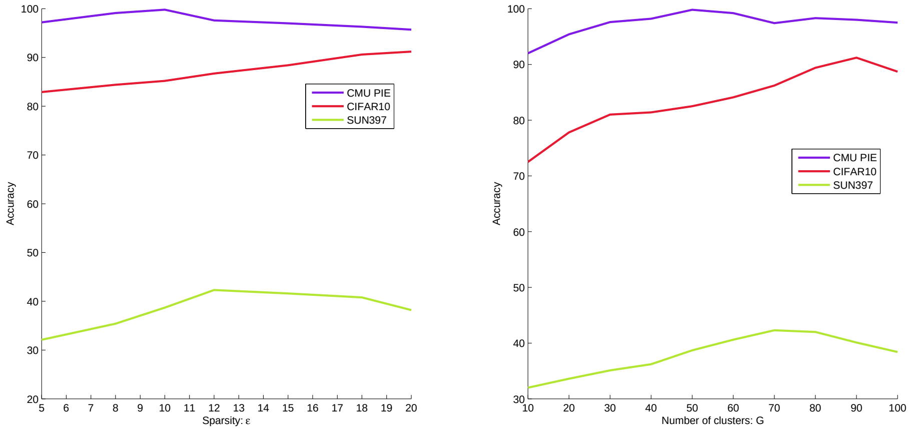


Fig. 5.5 The curves on the left side show the best performance on the training data when ε is equal to one value from $\{5, 8, 10, 12, 15, 18, 20\}$ while G varies its value in $\{10, 20, \cdots, 100\}$, and vice versa.

Table 5.4 Comparison of training and coding time (seconds) for learning 80 dimensional embedded features on the three datasets.

Dataset	**Phase**	DSE	MSE	MKL	**KMP**
CMU PIE	Training time	1148.24	716.79	873.72	755.28
	Coding time/query	1156.01	790.09	-	0.032
CIFAR 10	Training time	1683.70	1026.32	1098.97	991.54
	Coding time/query	1696.52	1072.18	-	0.041
SUN397	Training time	2804.91	1778.74	1678.14	1694.10
	Coding time/query	2812.36	1784.50	-	0.036

Furthermore, some parameter sensitivity analysis is carried out. Table 5.3 illustrates the performance variation of KMP with respect to the parameter r on the CMU PIE dataset; the target dimensionality of the low-dimensional embedding d is fixed at $\{20, 30, \cdots, 80\}$ with a step of 10, respectively. By adopting the 10-fold cross-validation scheme on the training data, it is demonstrated that higher dimensions prefer a larger r in our KMP. Finally, Fig. 5.5 shows the variation of parameters G and ε on all three datasets. The general tendency of these curves is consistently shown as “rise-then-fall”. It can be also seen from this figure that a larger training set needs larger values of G and ε, and vice versa.

5.5.3 Time Consumption Analysis

In this section, we compare the training and coding time of the proposed KMP algorithm with other methods. As we can see from Table 5.4, our method can achieve competitive training time compared with the state-of-the-art multiview and multiple kernel learning methods. Since there is no embedding procedure in MKL, the coding time is not applicable for MKL. Due to the nature of DSE and MSE, they need to be re-trained when receiving a new test sample. In contrast, once the projection and weights are gained by KMP, they are fixed for all test samples and implemented in a fast way. All the experiments are completed using Matlab 2014a on a workstation configured with an i7 processor and 32GB RAM.

5.6 Experiments on Action Recognition

In this section, we evaluate KMP systematically on five action datasets: KTH [133], UCF YouTube [91], UCF Sports [126], Hollywood2 [104] and HMDB51 [75] respectively. Some representative frames of these datasets are illustrated in Fig. 5.6. In the rest of this section, we will first introduce the details of the used datasets and their corresponding experimental settings. After that, the compared results will be presented and discussed.

5.6.1 Datasets

The **KTH** dataset is the benchmark dataset commonly used for action recognition with 599 video clips. Particularly, it contains six different action classes (i.e., boxing, handclapping, handwaving, jogging, running and walking), which are performed by 25 subjects under 4 different scenarios. Following the pre-processing step mentioned in [178], the coarse 3D bounding boxes are extracted from all the raw action sequences and further normalized into an equal size of 100×100 of each frame. In our experiments, we adopt two usually used settings to compare the final results. The first one is the original experimental setting of the authors, i.e., divide the data into a test set with 9 subjects: 2, 3, 5, 6, 7, 8, 9, 10, 22 and the rest form the training set. We finally report the average accuracy over all classes as the performance measure. The other setting is the common leave-one-person-out cross-validation.

The **UCF YouTube** dataset contains 1168 video clips with 11 action categories: *basketball shooting, biking/cycling, diving, golf swinging, horse back riding, soccer juggling, swinging, tennis swinging, trampoline jumping, volleyball spiking, and walking with a dog*. We also extract the bounding boxes according to the original paper [91]. Each frame of the sequences is further normalized into the size of 100×100. This dataset is relatively challenging due to large variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background, and illumination conditions. Following the original setup in [91], a leave-one-out scheme is adopted. The average accuracy over all classes is reported as the final performance.

The **UCF Sports** dataset has 10 classes of human actions with 150 collected broadcast videos. This collection represents a natural pool of actions featured in a wide range of scenes and viewpoints with a large intra-class variability. For this dataset, we use the provided bounding boxes and resize each video frame to a normalized size of 100×100. In our experiments, we use a five-fold cross- validation setup mentioned in [126], adopting 4/5th of the total number of sequences in each category for training and the rest for testing. The final recognition rate is averaged over the five folds.

The **Hollywood2** dataset is a collection of 1707 action samples comprising 12 types of action from 69 different Hollywood movies. For this dataset, we deliberately use the

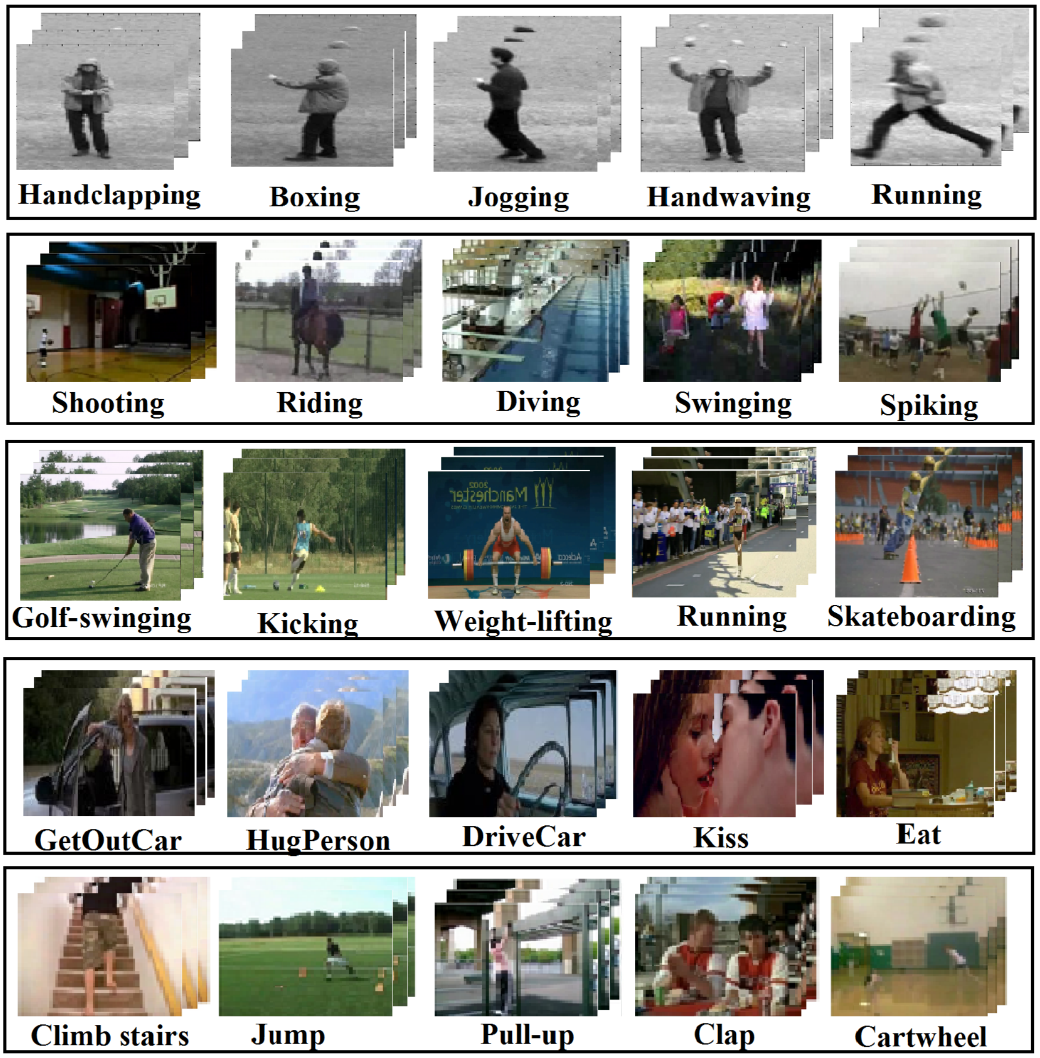


Fig. 5.6 Some example frames of five datasets: KTH, UCF YouTube, UCF Sports, Hollywood2 and HMDB51 (ordered from the top to the bottom).

full-sized sequences without any bounding boxes. In our experiments, we use the proposed KMP on a training set of 823 sequences and a test set with 884 sequences following the original setting.

The **HMDB51** dataset contains 6849 realistic action sequences collected from a variety of movies and online videos. Specifically, it has 51 action classes and each has at least 101 positive samples. In our experiments, coarse bounding boxes have been extracted from all the sequences through masks released with the dataset and initialized into the size of 100×120 for each frame. We adopt the official setting of [75] with three train/test splits. Each split has 70 training and 30 testing clips for each class.

5.6.2 Multiview Pose Feature Extraction

With the increasing complexity of recognition scenarios, using a single type of feature representation is difficult to satisfy the required accuracies in vision tasks, especially for some realistic applications.

Given a frame containing one pose, we would like to first describe it with multiview informative features. The descriptors are expected to capture the gradient, motion, texture and color information, which are the main cues of a pose. We, therefore, employ the histogram of optical flow (HOF) [82], the histogram of oriented gradients (HOG) [30], the local binary pattern (LBP) [3] and color histogram (ColorHist), respectively, for pose representation.

HOF: A fast and effective algorithm to capture the action movement based on the Lucas-Kanade optical flow. Specifically, we calculate HOF between any adjacent frames and each motion region is divided into sub-regions with a 5×5 grid. For each sub-region, a 12-bin histogram is computed to accumulate the motion orientation within 360 degrees. Thus, the length of the final vector of HOF is $5 \times 5 \times 12 = 300$.

HOG: A powerful gradient descriptor. In particular, a 9-bin histogram over [0,180] degrees is computed to accumulate the gradient orientation over a 5×5 cell. The length of the vector is $5 \times 5 \times 9 = 225$.

LBP: LBP features tolerate against illumination changes and are computationally efficient. The operator labels the pixels of an image by thresholding a 3×3 neighborhood of each pixel with the center value and considering the results as a binary number and a 256-bin histogram of the LBP labels computed over a region is used as a texture descriptor.

Note that, all the above three features are extracted on the gray-scale frames.

ColorHist: For each channel of RGB, a 64-bin histogram is used. Thus the final ColorHist has $3 \times 64 = 192$ dimensions.

Table 5.5 Dimensions of four features for action recognition.

Feature representation	Dimension
Histogram of oriented gradients (HOG)	225
Histograms of optical flow (HOF)	300
Local binary pattern (LBP)	256
Color histogram (ColorHist)	192
Total dimension	973

In this way, each pose from a video frame is represented by four different feature views which can describe the thorough information of this frame/pose.

5.6.3 Compared Methods and Settings

For action recognition, a video sequence can be usually described using different feature representations, i.e., multiview representation, in high dimensional feature spaces. In this paper, we adopt four different feature representations (i.e., HOG, HOF, LBP, ColorHist) to describe a video sequence. Table 5.5 illustrates the original dimensions of these features. We systematically compare our proposed KMP with two related multi-kernel fusion methods. In particular, KMP denotes that the RBF sequential kernels are combined by the proposed method: $K = \sum_{i=1}^{M} \alpha_i K_i$, where the weight α_i is obtained via alternate optimization. AM indicates that the kernels are combined by arithmetic mean: $K_{AM} = \frac{1}{M}\sum_{i=1}^{M} K_i$, and GM denotes the combination of kernels through geometric mean: $K_{GM} = (\prod_{i=1}^{M} K_i)^{\frac{1}{M}}$. Besides, we also include the best performance of the single-view-based spectral projection (BSP), the average performance of the single-view-based spectral projection (ASP) and concatenation of multiview embeddings in our compared experiments. All of AM, GM , BSP, ASP and multiview embedding concatenation are based on the locality preserving projections (LPP) [55] technique. In addition, two non-linear embedding methods distributed spectral embedding (DSE) and multiview spectral embedding (MSE) are adopted in our comparison, as well. In DSE and MSE, the Laplacian embedding (LE) [7] is adopted.

All of the above methods are evaluated on seven different lengths of codes {20, 30, 40, 50, 60, 70, 80}. Under the same experimental setting, all the parameters used in the compared methods have been strictly chosen according to their original papers. For KMP and MSE, the optimal balance parameter r for each dataset is selected from one of {2, 3, 4, 5, 6, 7, 8, 9, 10} with the step of 1, which yields the best performance by 9-fold cross-validation on the training data. The best σ in kernel construction is also selected by the cross-validation on the training data.

Table 5.6 Runtime(seconds) of the training and test phases with $d = 80$ on different datasets.

Datasets	Training time	Test time
KTH	460.15s	1.89s
UCF YouTube	1533.0s	4.12s
UCF Sports	170.9s	1.01s
Hollywood2	1220.5s	4.03s
HMDB51	3250.8s	12.95s

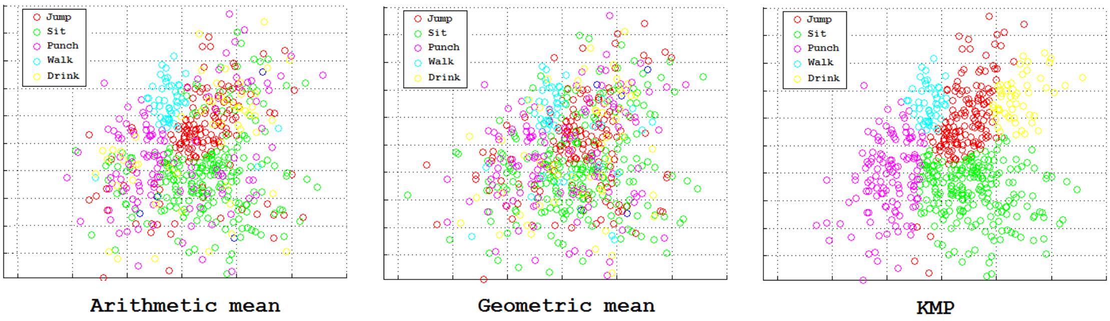


Fig. 5.7 Illustration of low-dimensional distributions of three different multi-kernel fusion schemes (illustrated with data of five actions inform the HMDB51 dataset).

5.6.4 Results

In Table 5.7, we first illustrate the performance of the single-view representation on all five datasets. In detail, we compute the RBF sequential kernel and weight matrix for a certain single view and input them to our KMP system. Since only a single view is used in KMP, it can be regarded as the procedure of kernelized LPP. From the comparison, we can easily observe that the HOG and HOF features consistently outperform the LBP descriptor in low dimensional feature space. The lowest accuracy is always obtained by ColorHist. Furthermore, we also include the long representation, which is concatenated by all the four low-dimensional feature representations, and the proposed KMP for multiview fusion based reduction into this comparison. It is shown that the concatenated representation can reach better performance than any of the single views, but is significantly lower than our KMP. Specifically, the best accuracies achieved by KMP are 97.5%, 87.6%, 95.8%, 64.3% and 49.8% on KTH, UCF YouTube, UCF Sport, Hollywood2 and HMDB51, respectively. Additionally, the results of the multiple kernel learning based on SVM (MKL-SVM) [45] are listed in Table 5.7 using the same four feature descriptors. The training time and the test time of KMP are listed in Table 5.6. The runtime of the training phase includes the multiview feature extraction, the INBF process, the construction of kernel matrices via DTW and

Table 5.7 Performance comparison (%) between the proposed KMP and single feature representations.

Dataset **Accuracy**	**KTH**	**UCF YouTube**	**UCF Sports**	**Hollywood2**	**HMDB51**
HOG	92.3 (50)	82.6 (70)	91.5 (50)	52.9 (70)	42.3 (50)
HOF	91.6 (70)	81.9 (70)	90.7 (50)	56.7 (70)	39.7 (50)
LBP	80.2 (50)	70.5 (40)	74.6 (30)	32.1 (30)	22.4 (30)
ColorHist	42.7 (20)	31.1 (30)	37.2 (30)	19.4 (20)	18.1 (30)
Concatenation	93.8 (190)	85.4 (210)	93.1 (160)	60.5 (190)	46.0 (160)
MKL-SVM	91.4	82.5	94.3	58.9	47.5
KMP	**97.5** (60)	**87.6**(80)	**95.8** (50)	**64.3** (80)	**49.8** (70)

The numbers in parentheses indicate the dimensions of the representations. For MKL-SVM, DTW is also used to construct the kernel matrix (as illustrated in Fig. 5.2) for each view and then MKL-SVM is applied to final classification.

Table 5.8 Performance comparison (%) on the KTH dataset with different feature fusion methods.

Method **Dimension**	Arithmetic mean (AM)	Geometric mean (GM)	BSP	ASP	DSE	MSE	KMP
d=20	86.8	84.5	85.6	72.4	85.9	86.0	**88.9**
d=30	88.7	83.6	88.4	74.4	88.0	87.7	**91.4**
d=40	91.6	86.2	91.0	71.3	89.6	91.7	**93.7**
d=50	93.0	90.4	92.3	73.6	92.5	93.9	**95.0**
d=60	93.3	90.7	91.5	75.3	93.8	94.2	**97.5**
d=70	93.6	92.0	91.8	74.8	93.8	93.5	**96.2**
d=80	92.5	91.1	92.1	75.0	93.3	93.7	**96.8**

Table 5.9 Performance comparison (%) on the UCF YouTube dataset with different feature fusion methods.

Method **Dimension**	Arithmetic mean (AM)	Geometric mean (GM)	BSP	ASP	DSE	MSE	KMP
d=20	72.9	71.8	71.5	58.2	72.1	73.6	**76.0**
d=30	75.0	74.2	72.8	59.4	74.0	75.2	**78.6**
d=40	79.5	77.4	77.7	62.5	78.2	80.8	**82.0**
d=50	82.3	80.8	80.3	61.8	81.3	82.5	**84.2**
d=60	82.1	81.3	80.9	64.2	81.7	82.5	**85.6**
d=70	82.9	82.2	82.6	66.0	83.0	83.3	**85.0**
d=80	84.2	83.0	82.3	66.3	83.5	84.5	**87.6**

Table 5.10 Performance comparison (%) on the UCF Sports dataset with different feature fusion methods.

Method **Dimension**	Arithmetic mean (AM)	Geometric mean (GM)	BSP	ASP	DSE	MSE	KMP
d=20	82.8	82.0	81.3	65.2	83.2	86.2	**88.5**
d=30	87.3	86.5	87.0	68.3	87.5	88.0	**91.6**
d=40	93.0	92.4	89.6	71.0	93.2	93.0	**94.7**
d=50	93.0	92.9	91.5	73.4	93.8	**95.8**	**95.8**
d=60	93.8	92.7	90.8	73.0	94.0	94.5	**95.5**
d=70	93.2	93.0	91.2	71.7	93.6	**95.1**	94.8
d=80	92.3	91.6	90.2	72.8	90.7	92.6	**94.3**

Table 5.11 Performance (%) of KMP with different *r* values on the KTH dataset.

Parameter value **Dimension**	r=2	r=3	r=4	r=5	r=6	r=7	r=8	r=9	r=10
d=20	87.0	87.0	87.5	87.8	**88.9**	88.7	88.0	88.0	87.4
d=30	89.4	90.1	90.5	91.0	91.3	**91.4**	**91.4**	90.7	89.3
d=40	87.2	89.0	89.4	91.2	92.0	93.5	93.5	**93.7**	93.2

the optimization of KMP.

In Tables 5.8–5.10, six different multiview embedding schemes are compared with the proposed KMP on the KTH, UCF YouTube and UCF Sports respectively. From the whole tendency, the proposed KMP always leads to the best performance for action recognition. Meanwhile, arithmetic mean (AM) and geometric mean (GM) achieve higher recognition accuracies than the best performance of the single-view-based spectral projection (BSP) and the average performance of the single-view-based spectral projection (ASP). DSE produces worse performance than MSE and sometimes even obtains lower results than AM, but generates better performance than others, since a more meaningful multiview combination scheme is adopted in DSE. Beyond these, it is obviously observed that, with different target dimensions, the final results change a lot. Although both KMP and MSE consider the similarity matrix of each view, KMP maps data into the RKHS which is more suitable for linearly inseparable data in realistic situations. Usually, the best results via KMP appear from $d = 50$ to $d = 80$. For instance, the highest accuracy on the KTH dataset is on the dimension of 60 and the best performance on the UCF Sports and UCF YouTube happens when $d = 50$ and $d = 80$, respectively.

Similar behaviors can also be seen on the Hollywood2 and HMDB51 datasets. From Fig. 5.8, we can observe that with the increase of the dimension, all the curves of compared methods on the Hollywood2 dataset are climbing up except for ASP and BSP, both of which have a decrease when the dimension exceeds 70. However, on the HMDB51 dataset, the

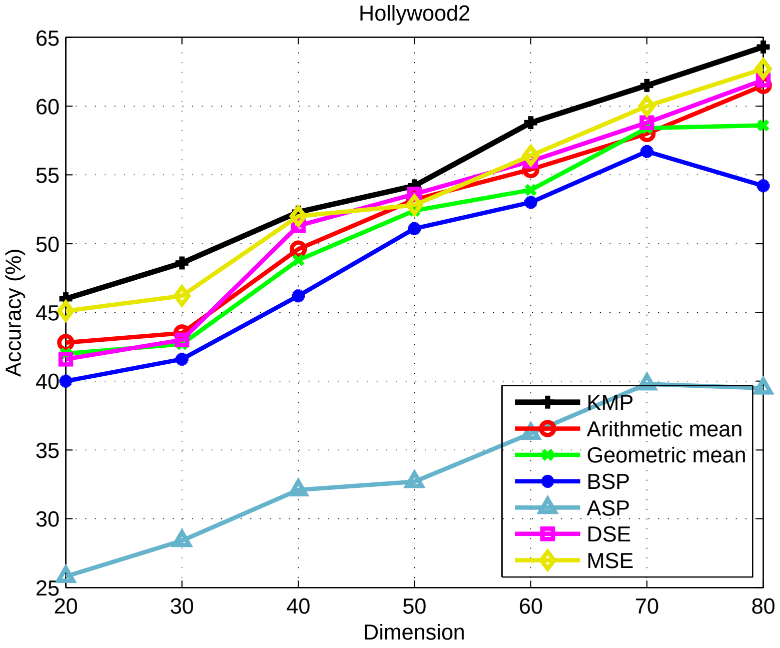


Fig. 5.8 Performance comparison (%) on the Hollywood2 dataset with different feature fusion methods.

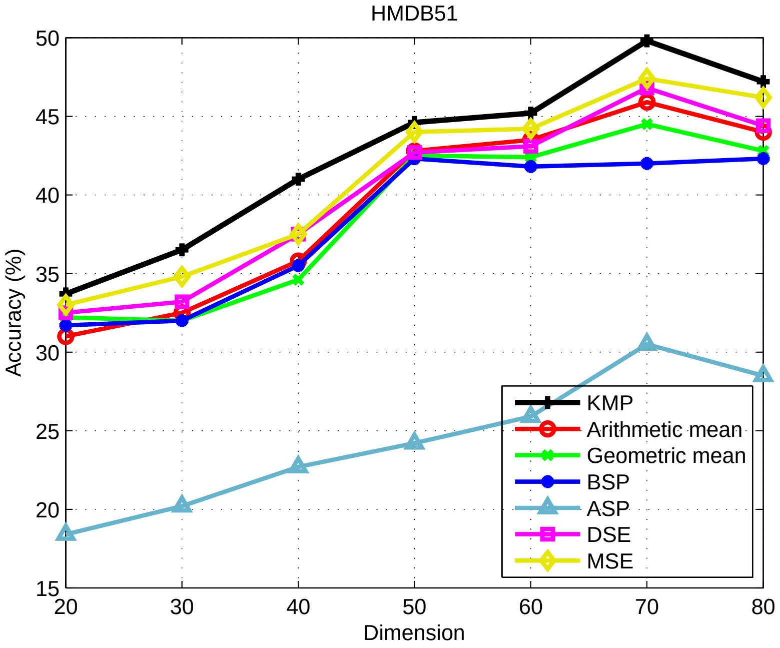


Fig. 5.9 Performance comparison (%) on the HMDB51 dataset with different feature fusion methods.

Table 5.12 The effectiveness (%) for INBF with $d = 80$ on different datasets.

Datasets	KMP with INBF	KMP without INBF
KTH	96.8	95.2
UCF YouTube	87.6	84.8
UCF Sport	94.3	91.5
Hollywood2	64.3	62.2
HMDB51	47.1	45.8

results in comparison always climb up then go down when the length of dimension increases (see Fig. 5.9). Besides, from these figures, we can also discover that all the curves have the same tendency of change. All of the above compared methods including MKL-SVM are trained on the same multiview features after INBF.

Furthermore, Table 5.11 illustrates the performance variation of KMP with respect to the balance parameter r; the dimensionality of the low-dimensional embedding d is fixed at 20,30 and 40 respectively on the KTH dataset. By adopting the 9-fold cross-validation scheme on the training data, it is demonstrated that the higher dimension prefers a larger r in our KMP. Moreover, Fig. 5.7 shows the low-dimensional (2-dimensional) embeddings obtained by AM, GM and KMP on the HMDB51 dataset. Our proposed KMP can well separate different categories, since it takes the semantically meaningful data structure of different views into consideration for embedding. The effectiveness of the INBF procedure in the training phase is demonstrated in Table 5.12.

At last, we also compare our results with the state-of-the-art approaches published in major vision conferences and journals in Table 5.13. In a sense, this kind of comparison is not fair enough, since different features with different methods are applied in different publications. Thus, we only treat this as a general evaluation of recent results. For the four datasets: KTH, UCF YouTube, UCF Sports and Hollywood2, our KMP approach either outperforms state-of-the-art methods or achieves the competitive results compared with published results. For the HMDB51 dataset, the proposed KMP has not shown better results than that reported in [163] and [142] due to the powerful features they introduced, but doubles the result shown in the original paper that introduced this dataset [75]. As a dimensionality reduction method, the proposed KMP can also adopt trajectory-based features or deep-learned features as different views for multiview learning. Considering that our action representation is semi-holistic and does not require an interest points detection phase, the results achieved by KMP are outstanding.

Table 5.13 Performance comparison (%) of KMP with state-of-the-art methods in the literature.

KTH		**UCF YouTube**		**UCF Sports**		**Hollywood2**		**HMDB51**	
Liu *et al.* [94]	93.5	Brendel *et al.* [17]	77.8	AFMKL [171]	91.3	Wang *et al.* [161]	58.3	Kuehne *et al.* [75]	22.8
Schindler and van Gool [131]	92.7	Le *et al.* [84]	75.8	GMKL [171]	85.2	Taylor *et al.* [154]	46.6	Sapienza *et al.* [130]	31.53
Wang *et al.* [164]	92.1	Bhattacharya *et al.* [12]	76.5	Wang *et al.* [161]	88.2	Ullah *et al.* [91]	53.2	Liu *et al.* [96]	36.5
Laptev *et al.* [82]	91.8	Sapienza *et al.* [130]	80.4	Le *et al.* [84]	86.5	Gilbert *et al.* [42]	50.9	Jiang *et al.* [66]	40.7
Jhuang *et al.* [65]	91.7	Wang *et al.* [161]	84.2	Kovashka and Grauman [74]	87.3	Le *et al.* [84]	53.3	Wang *et al.* [162]	46.6
Klaser *et al.* [72]	91.4	Kihl *et al.* [71]	87.6	O'Hara and Draper [113]	91.3	Jiang *et al.* [66]	59.5	Wang *et al.* [163]	**57.2**
Wang *et al.* [162]	94.2			Wang *et al.* [162]	88.0	Wang *et al.* [162]	58.2	Simonyan *et al.* [142]	55.4
Kihl *et al.* [71]	94.7			Vrigkas *et al.* [160]	95.1	Wang *et al.* [163]	64.3		
				Sun *et al.* [150]	86.6	Kihl *et al.* [71]	60.2		
Our method	**97.5**	Our method	**87.6**	Our method	**95.8**	Our method	**64.3**	Our method	49.8

5.7 Summary

In this chapter, we have presented an effective subspace learning framework called Kernelized Multiview Projection (KMP). KMP can encode a variety of features in different ways, to achieve a semantically meaningful embedding. Specifically, KMP is able to successfully explore the complementary property of different views and finally finds a unique low-dimensional subspace where the distribution of each view is sufficiently smooth and discriminative. KMP can be regarded as a fused dimensionality reduction method for multiview data. We have systematically evaluated our approach on three image datasets: CMU PIE, CIFAR10 and SUN397, and five human action datasets: KTH, UCF YouTube, UCF Sports, Hollywood2 and HMDB51. The corresponding results have shown the effectiveness and the superiority of our algorithm compared with other multiview embedding methods.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This thesis has addressed the feature reduction and representation learning problem for visual applications from the perspectives of dimensionality reduction and binary code learning. In this section, we conclude the main contributions of the thesis.

Firstly, a novel subspace learning algorithm called Local Feature Discriminant Projection (LFDP) for supervised dimensionality reduction of local features has been proposed. LFDP is able to efficiently seek a subspace to improve the discriminability of local features for classification. We have made three novel contributions in this work: (1) the proposed LFDP is a general supervised subspace learning algorithm which provides an efficient way for dimensionality reduction of large-scale local feature descriptors; (2) we introduce the Differential Scatter Discriminant Criterion (DSDC) to the subspace learning of local feature descriptors which avoids the matrix singularity problem; (3) we propose a generalized orthogonalization method to impose on projections, leading to a more compact subspace without redundancy. Extensive experimental validation on three benchmark datasets including UIUC-Sports, Scene-15 and MIT Indoor demonstrates that the proposed LFDP outperforms other dimensionality reduction methods and achieves state-of-the-art performance for image classification.

Secondly, to acquire a general feature for the video data, we convert the problem to describing the gradient fields of RGB and depth information of video sequences at first. With the local fluxes of the gradient fields, a new kind of continuous local descriptor called Local Flux Feature (LFF) is obtained. Then the LFFs from RGB and depth channels are fused into a Hamming space via the Structure Preserving Projection (SPP). Specifically, an orthogonal projection matrix is applied to preserve the pairwise structure with a shape constraint to avoid the collapse of data structure in the projected space. Furthermore, a bipartite graph structure of data is taken into consideration, which is regarded as a higher level connection between samples and classes than the pairwise structure of local features. The extensive experiments show not only the high efficiency of binary codes and the effectiveness of combining LFFs from RGB-D channels via SPP on various action recognition benchmarks of RGB-D data, but also the potential power of LFF for general action recognition. Moreover, for more comprehensive applications, the supervised SPP is extended to an unsupervised version to bridge the semantic gap between images and texts to a satisfactory level. A novel unsupervised binary coding algorithm called Binary Set Embedding (BSE) has been proposed to obtain semantically-preserving hash codes for local features from the image domain and words from text domain. BSE associates the image features with the word vectors learned from the human language instead of the provided documents from datasets. Extensive experiments demonstrate the superior performance of BSE compared with state-of-the-art cross-modal hashing methods using either image or text queries.

Finally, to make use of multimedia data from multiple sources, we propose a novel and general spectral coding algorithm called Kernelized Multiview Projection (KMP). Computing the kernel matrices from different features/views, KMP can encode different features with different weights to achieve a low-dimensional and semantically meaningful subspace where the distribution of each view is sufficiently smooth and discriminative. More crucially, KMP is linear for the reproducing kernel Hilbert space (RKHS), which allows it to be competent for various practical applications. We demonstrate KMP's performance for both image classification and action recognition on eight popular datasets and the results are consistently superior to state-of-the-art techniques.

6.2 Future Work

Based on the potential extensions of the current works on dimensionality reduction and binary code learning, in this section, we discuss some related research topics which will be explored in the future.

6.2.1 General and Efficient Discriminant Analysis

The proposed LFDP in Chapter 2 is designed for local features, which requires that each sample is represented by a group of data points. The previous methods such as LE, LLE, ISOMAP, LPP, NPE, LDP and LDE need at least $O(N^2)$ computational complexity given N data points. However, in the current "big data" era, N is usually at scale of millions or even billions, which cannot tolerate the algorithms with such high computational complexity. Besides, the reduced dimensionality of LDA is restricted by the number of classes and LDA also suffers from the matrix singularity problem. Recently, Hauberg et al. [53] proposed a scalable subspace learning method called Grassmann Average. Unlike traditional PCA, their algorithm is robust to the outliers for large-scale data. Since the label information is not involved in their method, the output data will sacrifice certain discriminative ability. To speed up large-scale classification, recognition and similarity search tasks, in the future, we aim to propose an efficient discriminant analysis method with at most linear computational complexity for holistic representations.

6.2.2 Hashing for 3D Object Data

With the development of depth and laser camera techniques, 3D retrieval/recognition is becoming a popular and attractive research direction. Using 3D information will also improve traditional image retrieval performance. However, retrieval/recognition algorithms are also of high computational complexity when the scale of 3D object datasets is growing larger. For example, in the industry of manufacture, there are usually millions of parts for assembling large industrial products such as airplanes, ships and constructions. Searching 3D parts in such a million-scale dataset would be extremely time-consuming. Therefore, it is desirable to propose a hashing method which can transfer 3D representations to binary codes to improve the efficiency of searching algorithms while preserving the discriminative ability of original features.

6.2.3 Hashing for Online Learning

Hashing is a popular and effective method for large-scale vision tasks. In many realistic application cases, data always are acquired in a stream form. The data on the Internet are changing and updating everyday. For most of current hashing algorithms, after the training phase, the hash functions are then fixed for all test data. If there is a new training sample added to the training set, the whole model needs to be retrained, which will cause a very high time complexity. The mechanism of online learning provides a feasible way to address this problem. For the incoming training data, the learned function only needs very few modification to obtain the update function. On the other hand, hash functions map data into binary codes, which are easy to update since the operation is just bit flipping. Recently, some efforts on this topic have been proposed such as Online Kernel-based Hashing (OKH) [60] and Online Sketching Hashing (OSH) [86]. However, their results are still unsatisfactory. Therefore, it is desirable to develop an efficient and sophisticated online hashing algorithm with sufficient discovery for data structure.

References

[1] Ahad, M. A. R., Tan, J. K., Kim, H., and Ishikawa, S. (2012). Motion history image: its variants and applications. *Machine Vision and Applications*, 23(2):255–281.

[2] Aho, A. V., Hopcroft, J. E., and Ullman, J. D. (1974). *The Design and Analysis of Computer Algorithms*. Addison-Wesley.

[3] Ahonen, T., Hadid, A., and Pietikäinen, M. (2004). Face recognition with local binary patterns. In *European Conference on Computer Vision*.

[4] Arfken, G. and Weber, H. (2005). *Mathematical Methods for Physicists*. Elsevier.

[5] Bay, H., Ess, A., Tuytelaars, T., and Gool, L. J. V. (2008). Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359.

[6] Bay, H., Tuytelaars, T., and Gool, L. J. V. (2006). SURF: Speeded up robust features. In *European Conference on Computer Vision*, pages 404–417.

[7] Belkin, M. and Niyogi, P. (2001). Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems*, pages 585–591.

[8] Bengio, Y., Paiement, J.-F., Vincent, P., Delalleau, O., Roux, N. L., and Ouimet, M. (2003). Out-of-sample extensions for LLE, Isomap, MDS, eigenmaps, and spectral clustering. In *Advances in Neural Information Processing Systems*.

[9] Berndt, D. J. and Clifford, J. (1994). Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370.

[10] Bezdek, J. C. and Hathaway, R. J. (2002). Some notes on alternating optimization. In *AFSS International Conference on Fuzzy Systems*.

[11] Bhatia, R. (1997). *Matrix analysis*. Springer-Verlag.

[12] Bhattacharya, S., Sukthankar, R., Jin, R., and Shah, M. (2011). A probabilistic representation for efficient large scale visual recognition tasks. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[13] Bickel, S. and Scheffer, T. (2004). Multi-view clustering. In *International Conference on Data Mining*.

[14] Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

[15] Bobick, A. F. and Davis, J. W. (2001). The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):257–267.

[16] Boiman, O., Shechtman, E., and Irani, M. (2008). In defense of nearest-neighbor based image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[17] Brendel, W. and Todorovic, S. (2010). Activities as time series of human postures. In *European Conference on Computer Vision*.

[18] Bronstein, M. M., Bronstein, A. M., Michel, F., and Paragios, N. (2010). Data fusion through cross-modality metric learning using similarity-sensitive hashing. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3594–3601.

[19] Cai, D., Bao, H., and He, X. (2011a). Sparse concept coding for visual analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2905–2910.

[20] Cai, D., He, X., and Han, J. (2007). Semi-supervised discriminant analysis. In *IEEE International Conference on Computer Vision*.

[21] Cai, D., He, X., and Han, J. (2011b). Speed up kernel discriminant analysis. *VLDB*, 20(1):21–33.

[22] Cai, D., He, X., Han, J., and Huang, T. S. (2011c). Graph regularized nonnegative matrix factorization for data representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1548–1560.

[23] Cai, D., He, X., Han, J., and Zhang, H. (2006). Orthogonal laplacianfaces for face recognition. *IEEE Transactions on Image Processing*, 15(11):3608–3614.

[24] Cai, H., Mikolajczyk, K., and Matas, J. (2011d). Learning linear discriminant projections for dimensionality reduction of image descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):338–352.

[25] Chen, H.-T., Chang, H.-W., and Liu, T.-L. (2005). Local discriminant embedding and its variants. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[26] Cheng, B., Yang, J., Yan, S., Fu, Y., and Huang, T. S. (2010). Learning with ℓ^1-graph for image analysis. *IEEE Transactions on Image Processing*, 19(4):858–866.

[27] Chua, T.-S., Tang, J., Hong, R., Li, H., Luo, Z., and Zheng, Y. (2009). Nus-wide: a real-world web image database from national university of singapore. In *Proceedings of the ACM international conference on image and video retrieval*. ACM.

[28] Cox, T. F. and Cox, M. A. (2000). *Multidimensional scaling*. CRC press.

[29] Cruz, L., Lucio, D., and Velho, L. (2012). Kinect and RGBD images: Challenges and applications. In *SIBGRAPI Conference on Graphics, Patterns and Images - Tutorials*, pages 36–49.

[30] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 886–893.

[31] Dalal, N., Triggs, B., and Schmid, C. (2006). Human detection using oriented histograms of flow and appearance. In *European Conference on Computer Vision*, pages 428–441.

[32] Datta, R., Joshi, D., Li, J., and Wang, J. Z. (2008). Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv.*, 40(2).

[33] Ding, G., Guo, Y., and Zhou, J. (2014). Collective matrix factorization hashing for multimodal data. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2083–2090.

[34] Dollár, P., Rabaud, V., Cottrell, G., and Belongie, S. (2005). Behavior recognition via sparse spatio-temporal features. In *2nd Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance.*

[35] Duchene, L. and Leclerq, S. (1988). An optimal transformation for discriminant and principal component analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(6):978–983.

[36] Engel, D. and Curio, C. (2008). Scale-invariant medial features based on gradient vector flow fields. In *International Conference on Pattern Recognition*, pages 1–4.

[37] Fathi, A. and Mori, G. (2008). Action recognition by learning mid-level motion features. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8.

[38] Freeman, W. T. and Roth, M. (1995). Orientation histograms for hand gesture recognition. In *International Workshop on Automatic Face and Gesture Recognition*, volume 12, pages 296–301.

[39] Fukunaga, K. (1990). *Introduction to statistical pattern recognition*. Academic press.

[40] Gaede, V. and Günther, O. (1998). Multidimensional access methods. *ACM Computing Surveys (CSUR)*, 30(2):170–231.

[41] Geng, B., Tao, D., Xu, C., Yang, L., and Hua, X. (2012). Ensemble manifold regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(6):1227–1233.

[42] Gilbert, A., Illingworth, J., and Bowden, R. (2011). Action recognition using mined hierarchical compound features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):883–897.

[43] Gionis, A., Indyk, P., and Motwani, R. (1999). Similarity search in high dimensions via hashing. In *International Conference on Very Large Data Bases*, pages 518–529.

[44] Golub, G. H. and van Loan, C. F. (1996). *Matrix computations*. Johns Hopkins University Press.

[45] Gönen, M. and Alpaydin, E. (2011). Multiple kernel learning algorithms. *Journal of Machine Learning Research*, 12:2211–2268.

[46] Hadfield, S., Lebeda, K., and Bowden, R. (2014). Natural action recognition using invariant 3d motion encoding. In *European Conference on Computer Vision*, pages 758–771.

[47] Han, J., He, S., Qian, X., Wang, D., Guo, L., and Liu, T. (2013a). An object-oriented visual saliency detection framework based on sparse coding representations. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(12):2009–2021.

[48] Han, J., Ji, X., Hu, X., Zhu, D., Li, K., Jiang, X., Cui, G., Guo, L., and Liu, T. (2013b). Representing and retrieving video shots in human-centric brain imaging space. *IEEE Transactions on Image Processing*, 22(7):2723–2736.

[49] Han, J., Shao, L., Xu, D., and Shotton, J. (2013c). Enhanced computer vision with microsoft kinect sensor: A review. *IEEE Transactions on Cybernetics*, 43(5):1318–1334.

[50] Han, Y., Wu, F., Tao, D., Shao, J., Zhuang, Y., and Jiang, J. (2012). Sparse unsupervised dimensionality reduction for multiple view data. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(10):1485–1496.

[51] Hardy, G. H., Littlewood, J. E., and Pólya, G. (1952). *Inequalities*. Cambridge university press.

[52] Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning*. Springer.

[53] Hauberg, S., Feragen, A., and Black, M. J. (2014). Grassmann averages for scalable robust pca. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3810–3817.

[54] He, X., Cai, D., Yan, S., and Zhang, H. (2005). Neighborhood preserving embedding. In *IEEE International Conference on Computer Vision*, pages 1208–1213.

[55] He, X. and Niyogi, P. (2003). Locality preserving projections. In *Advances in Neural Information Processing Systems*, pages 153–160.

[56] Henry, P., Krainin, M., Herbst, E., Ren, X., and Fox, D. (2012). RGB-D mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments. *The International Journal of Robotics Research*, 31(5):647–663.

[57] Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507.

[58] Holte, M. B., Moeslund, T. B., Nikolaidis, N., and Pitas, I. (2011). 3d human action recognition for multi-view camera systems. In *International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, pages 342–349.

[59] Hua, G., Brown, M., and Winder, S. A. J. (2007). Discriminant embedding for local image descriptors. In *IEEE International Conference on Computer Vision*, pages 1–8.

[60] Huang, L., Yang, Q., and Zheng, W. (2013). Online hashing. In *International Joint Conference on Artificial Intelligence*.

[61] Hyvärinen, A., Karhunen, J., and Oja, E. (2004). *Independent component analysis*, volume 46. John Wiley & Sons.

[62] Jaakkola, T. and Haussler, D. (1999). Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems*, pages 487–493.

[63] Jegou, H., Douze, M., and Schmid, C. (2008). Hamming embedding and weak geometric consistency for large scale image search. In *European Conference on Computer Vision*, pages 304–317.

[64] Jegou, H., Douze, M., Schmid, C., and Pérez, P. (2010). Aggregating local descriptors into a compact image representation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3304–3311.

[65] Jhuang, H., Serre, T., Wolf, L., and Poggio, T. (2007). A biologically inspired system for action recognition. In *International Conference on Computer Vision*.

[66] Jiang, Y.-G., Dai, Q., Xue, X., Liu, W., and Ngo, C.-W. (2012). Trajectory-based modeling of human actions with motion reference points. In *European Conference on Computer Vision*.

[67] Joly, A. and Buisson, O. (2008). A posteriori multi-probe locality sensitive hashing. In *Proceedings of the 16th International Conference on Multimedia 2008, Vancouver, British Columbia, Canada, October 26-31, 2008*, pages 209–218.

[68] Joly, A., Buisson, O., and Frélicot, C. (2007). Content-based copy retrieval using distortion-based probabilistic similarity search. *IEEE Transactions on Multimedia*, 9(2):293–306.

[69] Ke, Y. and Sukthankar, R. (2004). PCA-SIFT: A more distinctive representation for local image descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 506–513.

[70] Ke, Y., Sukthankar, R., and Huston, L. (2004). Efficient near-duplicate detection and sub-image retrieval. In *ACM Multimedia*, volume 4, page 5.

[71] Kihl, O., Picard, D., and Gosselin, P.-H. (2015). A unified framework for local visual descriptors evaluation. *Pattern Recognition*, 48(4):1174–1184.

[72] Klaser, A. and Marszalek, M. (2008). A spatio-temporal descriptor based on 3d-gradients. In *British Machine Vision Conference*.

[73] Kläser, A., Marszalek, M., and Schmid, C. (2008). A spatio-temporal descriptor based on 3d-gradients. In *British Machine Vision Conference*, pages 1–10.

[74] Kovashka, A. and Grauman, K. (2010). Learning a hierarchy of discriminative space-time neighborhood features for human action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[75] Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., and Serre, T. (2011). Hmdb: A large video database for human motion recognition. In *IEEE International Conference on Computer Vision*.

[76] Kulis, B. and Darrell, T. (2009). Learning to hash with binary reconstructive embeddings. In *Advances in Neural Information Processing Systems*, pages 1042–1050.

[77] Kulis, B. and Grauman, K. (2009). Kernelized locality-sensitive hashing for scalable image search. In *IEEE International Conference on Computer Vision*.

[78] Kumar, S. and Udupa, R. (2011). Learning hash functions for cross-view similarity search. In *International Joint Conference on Artificial Intelligence*, pages 1360–1365.

[79] Lai, K., Bo, L., Ren, X., and Fox, D. (2011). Sparse distance learning for object recognition combining RGB and depth information. In *IEEE International Conference on Robotics and Automation*, pages 4007–4013.

[80] Lanckriet, G. R. G., Cristianini, N., Bartlett, P. L., Ghaoui, L. E., and Jordan, M. I. (2004). Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72.

[81] Laptev, I. (2005). On space-time interest points. *International Journal of Computer Vision*, 64(2-3):107–123.

[82] Laptev, I., Marszalek, M., Schmid, C., and Rozenfeld, B. (2008). Learning realistic human actions from movies. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[83] Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2169–2178.

[84] Le, Q. V., Zou, W. Y., Yeung, S. Y., and Ng, A. Y. (2011). Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[85] Lee, H., Battle, A., Raina, R., and Ng, A. Y. (2006). Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems*, pages 801–808.

[86] Leng, C., Wu, J., Cheng, J., Bai, X., and Lu, H. (2015). Online sketching hashing. In *IEEE conference on computer vision and pattern recognition*, pages 2503–2511.

[87] Li, L., Su, H., Lim, Y., and Li, F. (2014). Object bank: An object-level image representation for high-level visual recognition. *International Journal of Computer Vision*, 107(1):20–39.

[88] Li, L.-J. and Li, F.-F. (2007). What, where and who? classifying events by scene and object recognition. In *IEEE International Conference on Computer Vision*, pages 1–8.

[89] Lin, Y., Jin, R., Cai, D., Yan, S., and Li, X. (2013). Compressed hashing. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[90] Liu, B., Wang, Y., Shen, B., Zhang, Y., and Hebert, M. (2014a). Self-explanatory sparse representation for image classification. In *European Conference on Computer Vision*, pages 600–616.

[91] Liu, J., Luo, J., and Shah, M. (2009). Recognizing realistic actions from videos "in the wild". In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1996–2003.

[92] Liu, L. and Shao, L. (2013). Learning discriminative representations from RGB-D video data. In *International Joint Conference on Artificial Intelligence*.

[93] Liu, L. and Shao, L. (2014). Discriminative partition sparsity analysis. In *International Conference on Pattern Recognition*, pages 1597–1602.

[94] Liu, L., Shao, L., Li, X., and Lu, K. (2015a). Learning spatio-temporal representations for action recognition: A genetic programming approach. *IEEE Transaction Cybernetics*.

[95] Liu, L., Shao, L., and Rockett, P. (2013a). Boosted key-frame selection and correlated pyramidal motion-feature representation for human action recognition. *Pattern Recognition*, 46(7):1810–1818.

[96] Liu, L., Shao, L., and Rockett, P. (2013b). Boosted key-frame selection and correlated pyramidal motion-feature representation for human action recognition. *Pattern Recognition*, 46(7):1810–1818.

[97] Liu, L., Shen, C., Wang, L., van den Hengel, A., and Wang, C. (2014b). Encoding high dimensional local features by sparse coding based fisher vectors. In *Advances in Neural Information Processing Systems*, pages 1143–1151.

[98] Liu, L., Yu, M., and Shao, L. (2015b). Local feature binary coding for approximate nearest neighbor search. In *British Machine Vision Conference*.

[99] Liu, L., Yu, M., and Shao, L. (2015c). Multiview alignment hashing for efficient image search. *IEEE Transactions on Image Processing*, 24(3):956–966.

[100] Liu, W., Wang, J., Ji, R., Jiang, Y.-G., and Chang, S.-F. (2012). Supervised hashing with kernels. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2074–2081.

[101] Liu, W., Wang, J., Kumar, S., and Chang, S.-F. (2011). Hashing with graphs. In *International Conference on Machine Learning*, pages 1–8.

[102] Long, B., Philip, S. Y., and Zhang, Z. M. (2008). A general model for multiple view unsupervised learning. In *International Conference on Data Mining*.

[103] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

[104] Marszalek, M., Laptev, I., and Schmid, C. (2009). Actions in context. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[105] Mika, S., Rätsch, G., Weston, J., Schölkopf, B., Müller, K., Hu, Y.-H., Larsen, J., Wilson, E., and Douglas, S. (1999). Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing*, pages 41–48.

[106] Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630.

[107] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

[108] Muja, M. and Lowe, D. G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. In *VISAPP 2009 - Proceedings of the Fourth International Conference on Computer Vision Theory and Applications, Lisboa, Portugal, February 5-8, 2009 - Volume 1*, pages 331–340.

[109] Müller, M. and Röder, T. (2006). Motion templates for automatic classification and retrieval of motion capture data. In *ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 137–146.

[110] Muthu, R., Bouridane, A., and Khelifi, F. (2014). Minutiae based fingerprint image hashing. In *International Conference on Control, Decision and Information Technologies (CoDIT)*, pages 696–700.

[111] Ni, B., Wang, G., and Moulin, P. (2011). RGBD-hudaact: A color-depth video database for human daily activity recognition. In *IEEE International Conference on Computer Vision Workshops*, pages 1147–1153.

[112] Norouzi, M. and Fleet, D. J. (2011). Minimal loss hashing for compact binary codes. In *International Conference on Machine Learning*, pages 353–360.

[113] O'Hara, S. and Draper, B. A. (2012). Scalable action recognition with a subspace forest. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[114] Oliva, A. and Torralba, A. (2001). Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175.

[115] Oreifej, O. and Liu, Z. (2013). Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 716–723.

[116] Pati, Y. C., Rezaiifar, R., and Krishnaprasad, P. (1993). Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *The Twenty-Seventh Asilomar Conference on Signals, Systems and Computers*, pages 40–44.

[117] Paton, M. and Kosecka, J. (2012). Adaptive RGB-D localization. In *Conference on Computer and Robot Vision*, pages 24–31.

[118] Pereira, J. C., Coviello, E., Doyle, G., Rasiwasia, N., Lanckriet, G. R. G., Levy, R., and Vasconcelos, N. (2014). On the role of correlation and abstraction in cross-modal multimedia retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):521–535.

[119] Perronnin, F. and Dance, C. R. (2007). Fisher kernels on visual vocabularies for image categorization. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[120] Perronnin, F., Sánchez, J., and Mensink, T. (2010). Improving the fisher kernel for large-scale image classification. In *European Conference on Computer Vision*, pages 143–156.

[121] Poppe, R. (2010). A survey on vision-based human action recognition. *Image and vision computing*, 28(6):976–990.

[122] Prungsinchai, S., Khelifi, F., and Bouridane, A. (2012). Sub-images based image hashing with non-negative factorization. In *IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pages 781–784.

[123] Quattoni, A. and Torralba, A. (2009). Recognizing indoor scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 413–420.

[124] Rasiwasia, N., Pereira, J. C., Coviello, E., Doyle, G., Lanckriet, G. R. G., Levy, R., and Vasconcelos, N. (2010). A new approach to cross-modal multimedia retrieval. In *International Conference on Multimedia*, pages 251–260.

[125] Ren, X., Bo, L., and Fox, D. (2012). RGB-(D) scene labeling: Features and algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2759–2766.

[126] Rodriguez, M. D., Ahmed, J., and Shah, M. (2008). Action mach a spatio-temporal maximum average correlation height filter for action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[127] Roweis, S. T. and Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326.

[128] Sadeghi, F. and Tappen, M. F. (2012). Latent pyramidal regions for recognizing scenes. In *European Conference on Computer Vision*, pages 228–241.

[129] Sánchez, J., Perronnin, F., Mensink, T., and Verbeek, J. J. (2013). Image classification with the fisher vector: Theory and practice. *International Journal of Computer Vision*, 105(3):222–245.

[130] Sapienza, M., Cuzzolin, F., and Torr, P. H. (2012). Learning discriminative space-time actions from weakly labelled videos. In *British Machine Vision Conference*.

[131] Schindler, K. and Van Gool, L. (2008). Action snippets: How many frames does human action recognition require? In *IEEE Conference on Computer Vision and Pattern Recognition*.

[132] Schölkopf, B., Smola, A. J., and Müller, K. (1997). Kernel principal component analysis. In *Artificial Neural Networks - ICANN '97, 7th International Conference, Lausanne, Switzerland, October 8-10, 1997, Proceedings*, pages 583–588.

[133] Schuldt, C., Laptev, I., and Caputo, B. (2004). Recognizing human actions: A local SVM approach. In *International Conference on Pattern Recognition*, volume 3, pages 32–36.

[134] Scovanner, P., Ali, S., and Shah, M. (2007). A 3-dimensional sift descriptor and its application to action recognition. In *ACM International Conference on Multimedia*, pages 357–360.

[135] Sempena, S., Maulidevi, N. U., and Aryan, P. R. (2011). Human action recognition using dynamic time warping. In *International Conference on Electrical Engineering and Informatics*, pages 1–5.

[136] Shah, S. A. A., Bennamoun, M., Boussaid, F., and El-Sallam, A. A. (2013). A novel local surface description for automatic 3d object recognition in low resolution cluttered scenes. In *IEEE International Conference on Computer Vision Workshops*, pages 638–643.

[137] Shakhnarovich, G. (2005). *Learning task-specific similarity*. PhD thesis, Massachusetts Institute of Technology.

[138] Shao, L., Liu, L., and Li, X. (2014a). Feature learning for image classification via multiobjective genetic programming. *IEEE Transactions on Neural Networks and Learning Systems*, 25(7):1359–1371.

[139] Shao, L., Zhen, X., Tao, D., and Li, X. (2014b). Spatio-temporal laplacian pyramid coding for action recognition. *IEEE Transactions on Cybernetics*, 44(6):817–827.

[140] Shotton, J., Fitzgibbon, A. W., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., and Blake, A. (2011). Real-time human pose recognition in parts from single depth images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1297–1304.

[141] Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). Learning local feature descriptors using convex optimisation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1573–1585.

[142] Simonyan, K. and Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems*, pages 568–576.

[143] Sivic, J. and Zisserman, A. (2003). Video google: A text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision*, pages 1470–1477.

[144] Song, J., Yang, Y., Yang, Y., Huang, Z., and Shen, H. T. (2013). Inter-media hashing for large-scale retrieval from heterogeneous data sources. In *ACM SIGMOD International Conference on Management of Data*, pages 785–796.

[145] Spinello, L. and Arras, K. O. (2011). People detection in RGB-D data. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3838–3843.

[146] Spinello, L. and Arras, K. O. (2012). Leveraging RGB-D data: Adaptive fusion and domain adaptation for object detection. In *IEEE International Conference on Robotics and Automation*, pages 4469–4474.

[147] Springer, J., Xin, X., Li, Z., Watt, J., and Katsaggelos, A. K. (2013). Forest hashing: Expediting large scale image retrieval. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 1681–1684.

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

[149] Sugiyama, M. (2006). Local fisher discriminant analysis for supervised dimensionality reduction. In *International Conference on Machine Learning*, pages 905–912.

[150] Sun, L., Jia, K., Chan, T.-H., Fang, Y., Wang, G., and Yan, S. (2014). Dl-sfa: Deeply-learned slow feature analysis for action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2625–2632.

[151] Sung, J., Ponce, C., Selman, B., and Saxena, A. (2012). Unstructured human activity detection from RGBD images. In *IEEE International Conference on Robotics and Automation*, pages 842–849.

[152] Tao, D., Li, X., Wu, X., and Maybank, S. J. (2007). General tensor discriminant analysis and gabor features for gait recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(10):1700–1715.

[153] Tao, D., Li, X., Wu, X., and Maybank, S. J. (2009). Geometric mean for subspace selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):260–274.

[154] Taylor, G., Fergus, R., LeCun, Y., and Bregler, C. (2010). Convolutional learning of spatio-temporal features. In *European Conference on Computer Vision*.

[155] Tenenbaum, J. B., De Silva, V., and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323.

[156] Torralba, A., Fergus, R., and Freeman, W. T. (2008). 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970.

[157] Turney, P. D. and Pantel, P. (2010). From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141–188.

[158] van der Maaten, L. J., Postma, E. O., and van den Herik, H. J. (2009). Dimensionality reduction: A comparative review. Technical Report 2009–005, Tilburg centre for Creative Computing, Tilburg University, Tilburg, The Netherlands.

[159] Vedaldi, A., Gulshan, V., Varma, M., and Zisserman, A. (2009). Multiple kernels for object detection. In *IEEE International Conference on Computer Vision*, pages 606–613.

[160] Vrigkas, M., Karavasilis, V., Nikou, C., and Kakadiaris, I. A. (2014). Matching mixtures of curves for human action recognition. *Computer Vision and Image Understanding*, 119:27–40.

[161] Wang, H., Kläser, A., Schmid, C., and Liu, C.-L. (2011). Action recognition by dense trajectories. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3169–3176.

[162] Wang, H., Kläser, A., Schmid, C., and Liu, C.-L. (2013a). Dense trajectories and motion boundary descriptors for action recognition. *International journal of computer vision*, 103(1):60–79.

[163] Wang, H. and Schmid, C. (2013). Action recognition with improved trajectories. In *International Conference on Computer Vision*, pages 3551–3558.

[164] Wang, H., Ullah, M. M., Klaser, A., Laptev, I., Schmid, C., et al. (2009). Evaluation of local spatio-temporal features for action recognition. In *British Machine Vision Conference*.

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

[166] Wang, J., Liu, Z., Wu, Y., and Yuan, J. (2012b). Mining actionlet ensemble for action recognition with depth cameras. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1290–1297.

[167] Wang, X., Wang, B., Bai, X., Liu, W., and Tu, Z. (2013b). Max-margin multiple-instance dictionary learning. In *International Conference on Machine Learning*, pages 846–854.

[168] Wang, Z., Hu, Y., and Chia, L. (2010). Image-to-class distance metric learning for image classification. In *European Conference on Computer Vision*, pages 706–719.

[169] Weiss, Y., Torralba, A., and Fergus, R. (2008). Spectral hashing. In *Advances in Neural Information Processing Systems*, pages 1753–1760.

[170] Wu, B., Yang, Q., Zheng, W.-S., Wang, Y., and Wang, J. (2015). Quantized correlation hashing for fast cross-modal search. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 3946–3952.

[171] Wu, X., Xu, D., Duan, L., and Luo, J. (2011). Action recognition using context and appearance distribution features. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[172] Xia, T., Tao, D., Mei, T., and Zhang, Y. (2010). Multiview spectral embedding. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 40(6):1438–1446.

[173] Xiao, J., Hays, J., Ehinger, K. A., Oliva, A., and Torralba, A. (2010). SUN database: Large-scale scene recognition from abbey to zoo. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3485–3492.

[174] Xie, B., Mu, Y., Tao, D., and Huang, K. (2011). m-SNE: Multiview stochastic neighbor embedding. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 41(4):1088–1096.

[175] Xu, C. and Prince, J. L. (1998). Snakes, shapes, and gradient vector flow. *IEEE Transactions on Image Processing*, 7(3):359–369.

[176] Yang, H., Bai, X., Zhou, J., Ren, P., Zhang, Z., and Cheng, J. (2014). Adaptive object retrieval with kernel reconstructive hashing. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[177] Yang, J., Yu, K., Gong, Y., and Huang, T. S. (2009). Linear spatial pyramid matching using sparse coding for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1794–1801.

[178] Yao, A., Gall, J., and Van Gool, L. (2010). A hough transform-based voting framework for action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[179] Zhang, D., Wang, F., and Si, L. (2011). Composite hashing with multiple information sources. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 225–234.

[180] Zhang, L., Zhen, X., and Shao, L. (2014). Learning object-to-class kernels for scene classification. *IEEE Transactions on Image Processing*, 23(8):3241–3253.

[181] Zhang, T., Tao, D., and Yang, J. (2008). Discriminative locality alignment. In *European Conference on Computer Vision*, pages 725–738.

[182] Zhao, Z. and Liu, H. (2008). Multi-source feature selection via geometry-dependent covariance analysis. In *Third Workshop on New Challenges for Feature Selection in Data Mining and Knowledge Discovery*, pages 36–47.

[183] Zhen, X., Shao, L., and Zheng, F. (2014). Discriminative embedding via image-to-class distances. In *British Machine Vision Conference*.

[184] Zhen, X., Wang, Z., Yu, M., and Li, S. (2015). Supervised descriptor learning for multi-output regression. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1211–1218.

[185] Zhen, Y. and Yeung, D.-Y. (2012a). Co-regularized hashing for multimodal data. In *Advances in Neural Information Processing Systems*, pages 1376–1384.

[186] Zhen, Y. and Yeung, D.-Y. (2012b). A probabilistic model for multimodal hash function learning. In *SIGKDD*, pages 940–948.

[187] Zhu, F. and Shao, L. (2014). Weakly-supervised cross-domain dictionary learning for visual recognition. *International Journal of Computer Vision*, 109(1-2):42–59.

[188] Zhu, X., Huang, Z., Shen, H. T., and Zhao, X. (2013). Linear cross-modal hashing for efficient multimedia search. In *International Conference on Multimedia*, pages 143–152.

[189] Zien, A. and Ong, C. S. (2007). Multiclass multiple kernel learning. In *International Conference on Machine Learning*.