Dependency Modeling for Information Fusion with Applications in Visual Recognition

MA Jinhua

A thesis submitted in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Principal Supervisor: Prof. YUEN Pong Chi

Hong Kong Baptist University

August 2013

Declaration

I hereby declare that this thesis represents my own work which has been done under the guidance of my principal supervisor Prof. YUEN Pong Chi after registration for the degree of PhD at Hong Kong Baptist University, and has not been previously included in a thesis, dissertation submitted to this or other institution for a degree, diploma or other qualification.

Signature:_____

Date: August 2013

Abstract

While many pattern recognition algorithms have been developed in the last forty years, classifying images/videos in practical applications still faces the challenges of self/mutual occlusions, clustered backgrounds, illumination variations, etc. In order to improve the recognition performance, many systems are designed by fusing multiple complementary features for various classification tasks. This thesis addresses the independent assumption issue in the fusion process and proposes two novel frameworks for dependency modeling.

Under some mild assumptions, the first approach uses a linear combination of posterior probabilities to model the feature dependency. Based on the linear combination property, this thesis proposes a Linear Classifier Dependency Modeling (LCDM) method for classifier level fusion. Under the linear dependency modeling framework, this thesis shows that more information about the class label is available in feature level, so LCDM is generalized to feature level and the Linear Feature Dependency Model (LFDM) is proposed.

Since it is almost impossible to verify whether the assumptions in existing methods are valid in practice applications, fusion method with less demanding assumption should give better performance. In the second approach, this thesis develops an Analytic Dependency Model (ADM) for score level fusion without the assumptions in existing fusion algorithms. With the proposed ADM, this thesis gives an equivalent condition to the independent assumption from probabilistic properties of marginal distributions. Since the ADM may contain infinite number of undeter-

mined coefficients, this thesis further proposes the Reduced Analytic Dependency Model (RADM) based on the convergent properties of analytic functions.

While the proposed fusion methods overcome some limitations in existing approaches, the fusion performance can be further improved by combining more discriminative features. Among feature extraction algorithms, supervised manifold learning has been successfully applied to many image classification problems. However, for video applications, existing manifold learning methods do not take full advantage of the global constraint of temporal labels. To overcome this limitation, this thesis proposes a new Supervised Spatio-Temporal Neighborhood Topology Learning (SSTNTL) method for video classification.

The proposed methods have been extensively evaluated on publicly available databases such as PASCAL VOC 2007, Columbia Consumer Video, Hollywood Human Action, etc., and convincing experimental results have been achieved. In short, the major contributions of this thesis are summarized as follows.

- A linear dependency modeling framework is developed for classifier level and feature level fusion.
- A Reduced Analytic Dependency Model (RADM) is derived for score level fusion with less demanding assumption.
- A Supervised Spatio-Temporal Neighborhood Topology Learning (SSTNTL) method is proposed for video classification.

Acknowledgements

I would like to express my sincere gratitude to my supervisor Prof. Pong Chi Yuen for providing me the opportunity to study and work in the exciting and challenging areas of pattern recognition and computer vision. His precious advices, inspiration, and research enthusiasm have guided me to overcome the problems encountered in my postgraduate study. I believe that I will benefit from what I have learnt from him throughout my career in the future.

I would like to thank all the faculty members and staffs in the Department of Computer Science at Hong Kong Baptist University. They gave me kind assistance and provide me updating facilities and supports for research.

I would like to give my special gratitude to my master's supervisor, Prof. Jian-Huang Lai for his valuable suggestions on my research. I also would like to thank all the postgraduate students in the department of computer science for discussing research problems and sharing the joys together. Especially, I would like to thank Dr. C Liu, Dr. W Zou, Dr. Y Feng, Mr. J Li and Mr. X Lan for their help and discussion in my research. I would like to thanks all the reviews who provided helpful comments to improve the quality of my research works.

Most importantly, I would like to express my heartfelt thanks to my parents for their sincere love, encouragement and deep understanding from my birth.

The research in this thesis was partially supported by the Science Faculty Research Grant of Hong Kong Baptist University, NSFC Research Grants 61128009 and 61172136, and NSFC-GuangDong Research Grant U0835005.

Table of Contents

List of Tables

List of Figures

List of Symbols

A	Action sequence represented by feature vectors $\vec{x}_1, \cdots, \vec{x}_{N_A}$
$\mathcal{B}(\cdot, \cdot)$	Euclidean ball
$\mathscr{B}$	Topological base
$I(\cdot, \cdot)$	Mutual information between two random variables
j	Training sample index
J	Number of training sample
l	Class label index
L	Number of classes
$\mathcal{L}$	Laplacian matrix
m	Feature index
M	Number of features
$\mathcal{N}(\cdot)$	Neighborhood of a data point
$\Pr(\cdot \mid \cdot)$	Conditional probability
s	Classification score
$\vec{x}$	Feature vector
ω	Class label

List of Abbreviations

DN	Combination rule under Dependent Normal assumption
GRLF	Graph-regularized Robust Late Fusion
IN	Combination rule under Independent Normal assumption
LCDM	Linear Classifier Dependency Modeling
LFDM	Linear Feature Dependency Modeling
LP-B	Multi-class Linear Programming Boosting
LPBoost	Two-class Linear Programming Boosting
LPP	Locality Preserving Projection
LSDA	Locality Sensitive Discriminant Analysis
LSTDE	Local Spatio-Temporal Discriminant Embedding
MKL	Multiple Kernel Learning
RADM	Reduced Analytic Dependency Modeling
RM	Reduced Multivariate polynomial combination model
SLPP	Supervised Locality Preserving Projection
SSC	Signal Strength-based Combination approach
SSTNTL	Supervised Spatio-Temporal Neighborhood Topology Learning

Chapter 1

Introduction

In this chapter, the research background is first introduced in Section 1.1. Then, the motivations and contributions of this thesis are reported in Sections 1.2–1.3. Finally, Section 1.4 gives an brief overview of this thesis.

1.1 Background

Information fusion is an active research topic in computer vision and pattern recognition, since detection/recognition performance can be improved by making use of complementary information from multiple features/sensors. The importance of information fusion is also shown by a wide range of successful applications such as biometrics authentication, object detection, image classification, event recognition from videos, etc. During the past twenty years, many fusion algorithms [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] have been developed in the computer vision and pattern recognition community. These fusion methods can be categorized into two approaches, i.e. score level fusion (late fusion) and feature level fusion (early fusion).

The framework of score level fusion approach is shown in Fig. 1.1. Given images or videos as input, feature descriptors, e.g. SIFT [23], HOG [24], Laplacianfaces [25],

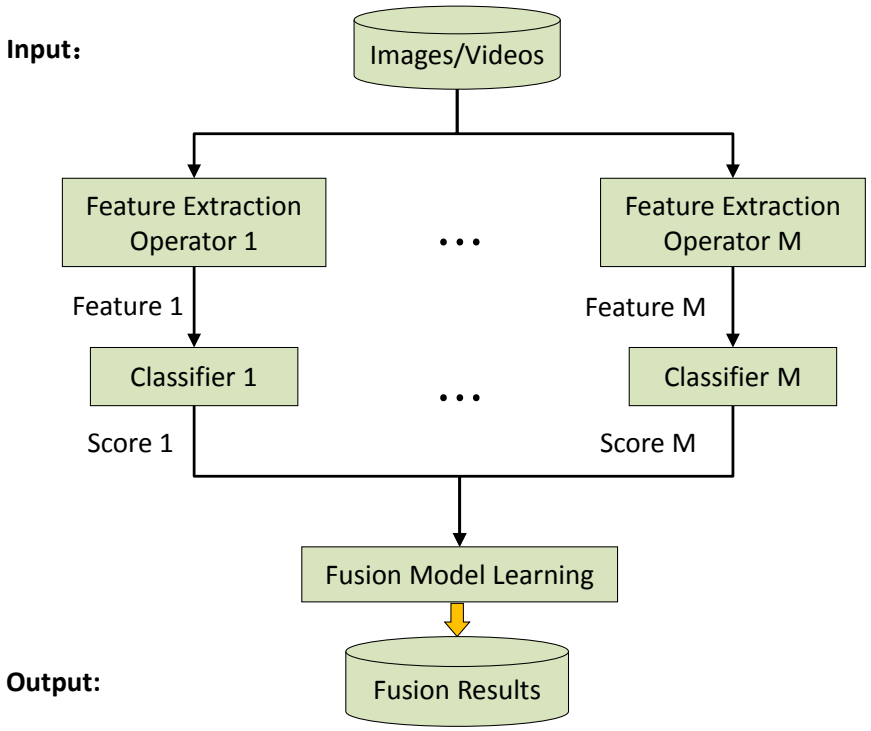


Figure 1.1: Score Level Fusion Framework for visual recognition

can be employed to extract complementary feature vectors for information fusion. After feature extraction, classifiers are trained on each feature representation to approximate the posterior probabilities. Then, the classification scores are combined to obtain final decisions as the fusion result. For example, Sum rule [2] computes the summation of the scores from the same object to obtain the final decision value. Although score level fusion methods can combine multiple classification scores efficiently, the potential loss of correlation information in the mixed feature space is a disadvantage of this fusion approach.

In contrast to score level fusion, as shown in Fig. 1.2, the feature level fusion scheme combines the feature representations directly without going through the classifier learning process for each feature. And the combination output can be a fusion representation or decision value. For example, the concatenation of selected normalized features [9] gives a fused vector representation, while Multiple Kernel Learning (MKL) methods [14] [18] [19] return a fused confidence score. Comparing with the

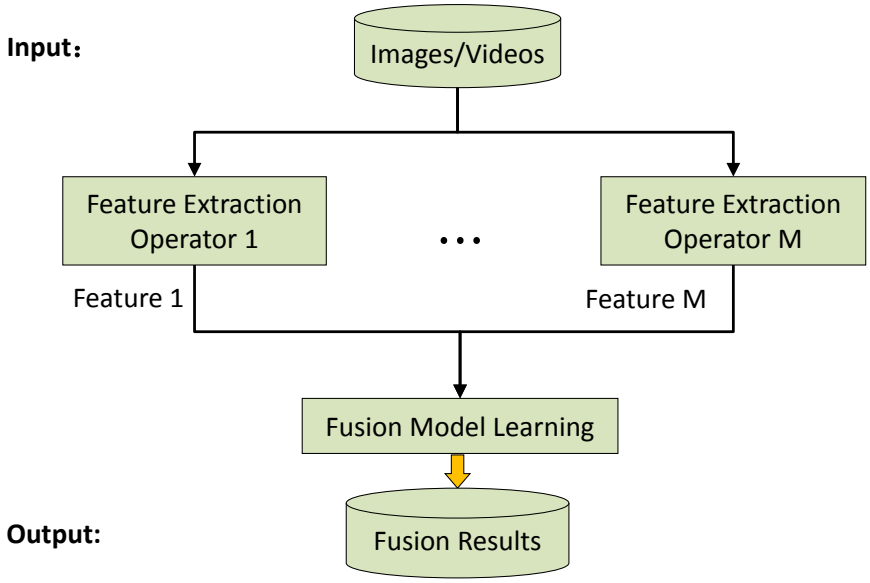


Figure 1.2: Feature Level Fusion Framework for visual recognition

score level fusion approach, fusion in feature level contains more information than that in classifier level according to the Data Processing Inequality (DPI) [26]. However, feature level fusion may not perform as well as expected due to the following two reasons. First, it is almost impossible to accurately estimate the joint distribution of multiple features, since feature dimensions are normally very high. Second, it is difficult to combine features from different modalities into a common representation, as the feature modalities can be different, e.g. feature representation can be a vector or a set of points.

1.2 Motivations of This Project

Many computer vision and pattern recognition applications face the challenges of self/mutual occlusions, clustered backgrounds, illumination variations, etc. In order to improve the recognition performance, many systems are designed by fusing information from multiple features which provide complementary cues for better prediction. While many combination rules [2] [8] [12] have been studied and devel-

oped in the last two decades, it is a general assumption that features are distributed conditionally independent with each other given class label. With this assumption, the joint distribution of the multiple features can be expressed as the product of probabilities of each feature. The conditionally independent assumption could simplify the problem, but it may not be valid in many practical applications. Therefore, this thesis proposes to model the dependency between multiple scores/features for better visual recognition.

Instead of utilizing the conditionally independent assumption, information fusion can be performed by estimating the joint distribution of multiple classifiers/features. However, it needs numerous data to estimate the joint density [27] accurately, when the dimension of classifiers/features is large. To deal with this problem, normal assumption was employed in [7] [13] to develop the fusion models. While most fusion methods were derived under certain assumptions, it is almost impossible to verify whether these assumptions are valid in practice. Thus, this thesis develops a fusion method with less demanding assumption which gives better recognition performance.

As shown in Fig. 1.1 and Fig. 1.2, feature extraction is a very important component in the fusion system. With more discriminative features, the fusion performance can be further improved. Among feature extraction algorithms, supervised manifold learning methods [28] [29] [30] [31] have been successfully applied to many image classification problems. Although these methods show that label information is important, they hardly take the temporal cue into account. For video applications, the local[1] embedding method [32] exploits the temporal information to develop the manifold learning algorithm, but it does not take full advantage of the global constraint of temporal labels, which is also useful for classifying videos. Consequently, this thesis proposes a method which incorporates the class label information as well

[1]It should be noticed that the local structure in the manifold learning methods refers to the small neighborhood of a data point in a manifold, while the local features as mentioned in the following sections refers to local descriptors of interest points detected in images or videos.

as the global information in temporal labels to learn the manifold structure.

1.3 Contributions of This Thesis

In this thesis, a novel framework for dependency modeling is developed, and two methods, namely Linear Classifier Dependency Modeling (LCDM) and Linear Feature Dependency Modeling (LFDM) are proposed for classifier level and feature level fusion, respectively. Inspired by the Bayesian model with independent assumption, it is proved that linear combination of the posterior probabilities of each classifier can model dependency under mild assumptions. Based on the linear combination property, LCDM is developed, and LFDM is derived by generalizing LCDM to feature level. The optimization problems in LCDM and LFDM are formulated as standard linear programming problems, which therefore, have analytic solutions. Comparing the proposed fusion methods in classifier level and feature level, it is shown that LFDM outperforms LCDM in two ways. First, it is proved that feature level contains more information about the label than that in classifier level under the proposed linear dependency modeling framework. Second, by analyzing the sensitivity to the density estimation errors in these two methods, it is shown that the upper bound of the error factor in LFDM is much smaller than that in LCDM. These works have been published in [33] [34].

Using analytic functions on posterior probabilities of each feature, this thesis develops another novel framework for score level dependency modeling without the assumptions in previous methods. It is shown that Product rule [2] (with independent assumption) and LCDM (without independent assumption) can be unified by the proposed framework. With the analytic dependency model (ADM), an equation system is derived from the properties of marginal distributions. And an equivalent condition to the independent assumption is presented in this thesis based on the structure of the solution to the derived equation system. Since the ADM may contain infinite number of undetermined coefficients, a reduced form of the ADM (RADM),

which can model dependency, is proposed from the convergent properties of analytic functions. After that, the RADM coefficients are learned by a new unconstrained quadratic programming problem, which minimizes the empirical classification errors and approximates the dependency modeling constraint. These works have been published in [35] and submitted to the International Journal of Computer Vision (IJCV) [36].

The advantages of the proposed fusion methods, LCDM, LFDM and RADM, are summarized as follows. All these methods can model dependency explicitly by making use of probabilistic properties. In addition, they are derived without normal assumption and will not suffer from the difficulty in joint distribution estimation with high-dimensionality problem. Moreover, RADM gives superior advantage over LCDM by removing the assumption that posterior probabilities will not deviate very much from the priors.

Since the fusion performance can be further improved by combining highly discriminative features, this thesis explores the manifold learning approach for video applications and proposes a novel method to learn the manifold structure from the aspect of neighborhood topology. By analyzing the topology for video recognition, this thesis combines the spatial distribution and label information to construct the Supervised Spatial (SS) neighborhood topology. And it is shown that the temporal adjacent neighborhood is contained in the SS neighbors. In order to take full advantage of the temporal information, this thesis constructs the novel Temporal Pose Correspondence (TPC) neighborhood by the global constraint of temporal labels with the help of dynamic time warping (DTW). Then, a new method, namely Supervised Spatio-Temporal Neighborhood Topology Learning (SSTNTL) is developed by fusing the SS and TPC neighborhoods for video recognition. The proposed SSTNTL not only discovers the local structure with label information but also preserves the global constraint of temporal labels in action sequences. These works have been published in [37].

1.4 Thesis Overview

The rest of this thesis is organized as follows:

Chapter 2 reviews the probabilistic and non-probabilistic fusion methods. The databases used for evaluation in this thesis are introduced.

Chapter 3 presents the proposed Supervised Spatio-Temporal Neighborhood Topology Learning (SSTNTL) for feature extraction in video applications. The proposed method is developed by analyzing the topological bases, which is an important concept in manifold. According to the topological analysis for action recognition in videos, the spatial distribution with label information and global constraint of temporal labels are combined to construct the final neighborhood topology. Then, the proposed SSTNTL is evaluated on five publicly available databases.

Chapter 4 reports the proposed linear dependency model for score level and feature level fusion. Inspired by the Bayesian model with independent assumption, it is proved that linear combination of the posterior probabilities of each classifier can model dependency under mild assumptions. Based on the linear combination property, two methods, namely Linear Classifier Dependency Modeling (LCDM) and Linear Feature Dependency Modeling (LFDM) are developed for classifier level and feature level fusion, respectively. Experimental results and their analysis are presented at the end of this chapter.

Chapter 5 presents the proposed Reduced Analytic Dependency Model for score level fusion. Unifying Product rule [2] (with independent assumption) and LCDM (without independent assumption), the Analytic Dependency Model (ADM) is proposed. Since the ADM may contain infinite number of undetermined coefficients, a reduced form is derived from the convergent properties of analytic functions. At last, results on databases for various recognition tasks are given.

Chapter 6 concludes this thesis and discusses the future directions.

Chapter 2

Related Works and Databases

Existing fusion methods can be categorized into probabilistic and non-probabilistic approaches. A brief review on both techniques is given in Section 2.1 and Section 2.2, respectively. And the databases used for evaluating the proposed methods are introduced in Section 2.3.

2.1 Probabilistic Fusion Methods

2.1.1 Fusion Models with Independent Assumption

In [2], a theoretical framework was developed for combining classifiers. According to Bayesian theory, under the assumption that classifier scores are distributed independently, the posterior probability is given by the following equation,

$$\Pr(\omega_l|\vec{x}_1,\cdots,\vec{x}_M) = \frac{\Pr(\omega_l)\prod_{m=1}^{M}\Pr(\vec{x}_m|\omega_l)}{\Pr(\vec{x}_1,\cdots,\vec{x}_M)} = \frac{P_0}{\Pr(\omega_l)^{M-1}}\prod_{m=1}^{M}\Pr(\omega_l|\vec{x}_m) \quad (2.1.1)$$

where ω_l denotes the label, M is the number of features, $\vec{x}_m$ is the m-th feature, and $P_0 = \frac{\prod_{m=1}^{M}\Pr(\vec{x}_m)}{\Pr(\vec{x}_1,\cdots,\vec{x}_M)}$. Product rule [2] was then derived by (2.1.1). Moreover, if the discriminatory information is highly ambiguous due to high levels of noise, it may be suitable to assume that the posterior probability of each feature vector will not

deviate dramatically from the prior probability [2]. In this situation, the posterior probability of a feature vector $\vec{x}_m$ can be expressed as

$$\Pr(\omega_l|\vec{x}_m) = \Pr(\omega_l)(1 + \delta_{lm}) \tag{2.1.2}$$

where δ_{lm} is a small number. Substituting $\Pr(\omega_l|\vec{x}_m)$ by (2.1.2) and expanding the product term in (2.1.1), Sum rule [2] was induced as the following equation,

$$\Pr(\omega_l|\vec{x}_1, \cdots, \vec{x}_M) = P_0[(1 - M)\Pr(\omega_l) + \sum_{m=1}^{M} \Pr(\omega_l|\vec{x}_m)] \tag{2.1.3}$$

Based on the Product rule and Sum rule, Kittler *et al.* [2] justified that the commonly used classifier combination rules, i.e. Max, Min, Median and Majority Vote, can be derived. It was shown that Sum rule outperforms other classifier combination schemes in their experiments.

Besides the combination rules developed under the Bayesian framework [2], Nandakumar *et al.* [12] proposed to find the optimal combination of the match scores based on the likelihood ratio test. In their approach, the distribution of each genuine or impostor match score is estimated by the finite Gaussian mixture model [38], and the joint distribution is computed by taking advantage of the Product rule (2.1.1) under the conditionally independent assumption. Apart from the likelihood ratio based fusion method [12], Terrades *et al.* [13] tackle the classifier combination problem using a non-Bayesian probabilistic framework. Under the assumptions that classifiers can be combined linearly and the scores follow independent normal (IN) distribution, the IN combination rule was derived [13].

2.1.2 Fusion Models without Independent Assumption

Without conditionally independent assumption, the posterior probability of classifiers can be computed by joint distribution estimation. For example, in [5], Parzen window density estimation [39] is used to estimate the joint density of a selected set of scores. When a large number of samples are available and the number of classifiers is small, the density estimated using Parzen window approach is very close to

the true one [5]. While this may not be the case for many practical applications, the error in the estimated density could be very large. As a result, the dependency between matching scores was considered by employing copula models in [7]. With the copula function under multivariate normal distribution assumption, the joint density of matching scores can be modeled and used to compute the likelihood ratio statistics for score fusion. Terrades *et al.* [13] also made use of normal distribution assumption, and proposed to fuse classifiers by a linear combination model. When features are not conditionally independent, the covariance matrix in the normal distribution is not diagonal. In this case, the dependent normal (DN) combination problem [13] was formulated into a constrained quadratic programming problem, which can be solved by nonlinear programming techniques [40].

2.2 Non-Probabilistic Fusion Methods

Besides the probabilistic fusion methods [2] [5] [7] [12] [13], the Optimal Weighting Method (OWM) [3], LPBoost approaches [4] [14] aimed at determining the correct weighting for linear combination by minimizing the least square error and 1-norm soft margin error, respectively. A multi-class variant of the two-class LPBoost [4] is presented in [14] and denoted as LP-B. This linear combination method determines the correct weights B_{lm} by learning the following linear programming (LP) problem

$$\begin{aligned} &\min_{B,\rho,\xi} -\rho + \frac{1}{\nu J}\sum_{j=1}^{J}\xi_j \\ s.t.\ \ &i)\ \sum_{m=1}^{M} B_{y_j m} h_{m,y_j}(\vec{x}_j) - \sum_{m=1}^{M} B_{lm} h_{m,l}(\vec{x}_j) \geq \rho - \xi_j, \forall j, \omega_l \neq y_j \\ &ii)\ \xi_j \geq 0, \forall j \\ &iii)\ \sum_{m=1}^{M} B_{lm} = 1, \forall l \end{aligned} \tag{2.2.4}$$

where $\vec{x}_j$ and y_j denote object j and the corresponding label, ρ represents the margin, ξ_j is the slack variable for object j, and $h_{m,l}$ gives the confidence by feature m on class l.

In order to describe the nonlinear input-output relationships, the reduced multivariate polynomial (RM) was introduced in [6]. Since the number of terms increases exponentially with the model order in the multivariate polynomial, Toh *et al.* [6] proposed to approximate the full polynomial by modified lumped multinomial. Then, the optimal RM model was learned by a weight-decay regularization problem in [6]. Different from score level fusion methods [3] [4] [6] [14], Multiple Kernel Learning (MKL) methods [14] [18] [19] can learn the classifiers and combination weights simultaneously. In the setting of MKL for feature combination, kernel K_m is constructed by feature m, so the multi-feature fusion problem is formulated as a multiple kernel learning problem.

While the linear or non-linear weighting approaches [3] [4] [6] [14] [18] [19] learn the score/feature fusion models with the help of labeled training data, the Signal Strength-based Combination (SSC) [20] approach and robust late fusion (RLF) method [21] fuse classification scores in an unsupervised manner. SSC was derived based on the signal strength concept and uncertainty degree for ensemble learning. With the marginal distribution graph analysis in [20], it was shown that SSC could increase the margin to support the final decision. On the other hand, Ye *et al.* [21] proposed to convert the score vectors from each feature into pairwise score relation matrix, whose entries represent the comparative relationships of scores between any two test samples. Under the assumption that the relation matrices can be decomposed into a shared rank-two matrix plus sparse errors, the fused score vector was obtained by fitting to the recovered low-rank score relation matrix. Based on the low-rank and sparse properties, the noise components could be eliminated. In order to cooperate with the feature level information, a graph based regularization term is added in the low-rank and sparse model. And the Graph-regularized Robust Late Fusion (GRLF) method was proposed in [21].

Figure 2.1: Visualization of the pixel average feature for digits from 0 to 9

2.3 Databases for Evaluation

2.3.1 Digit Database

The Digit database [41] contains ten digits from 0 to 9, and 200 examples for each digit. Six types of features (76 Fourier coefficients, 216 profile correlations, 64 Karhunen-Love coefficients, 240 pixel averages in 2×3 windows, 47 Zernike moments and 6 morphological features) are extracted in [41] and available on the website[1]. The pixel average feature for some samples is visualized in Fig. 2.1. We randomly select 20 samples of each digit for training and retain the rest for testing. Two kinds of classifiers with different parameters, i.e. k-NN classifiers for $k = 1, 3$ and SVM[2] classifiers for soft margin parameter $C = 0.1, 100$, are used to evaluate the fusion methods. Five-fold cross validation (CV) is performed with the training data to select the best parameters in the fusion models. And the CV outputs are used to train the combination models. These experiments are repeated ten times on this database

2.3.2 Oxford 17 Flower Database

Oxford 17 Flower database [43] contains 17 different types of flowers with 80 images per category. Some example images are shown in Fig. 2.2. Seven different types of features including shape, color, texture, HSV, HoG, SIFT internal, and SIFT boundary, are extracted using the method reported in [43] [44]. Distance matrices of these features and three predefined splits of the database (17×40 for training,

[1] `http://archive.ics.uci.edu/ml/datasets/Multiple+Features`

[2] The publicly available library [42] is adopted in this thesis to generate the SVM scores.

Figure 2.2: Example images from Flower database (Images from the same columns are from the same classes)

17×20 for validation, and 17×20 for testing) are available on their website[3]. Following [14], kernel SVM classifiers are trained for each of the seven features. The kernel matrices are defined as $\exp\left(-d(\vec{x}, \vec{x}')/\eta\right)$, where d is the distance and η is the mean of pairwise distances. The parameter introduced in the soft margin SVM is selected from $C \in \{10^{-3}, \cdots, 10^{3}\}$ by five-fold CV in the training set. The CV outputs of the SVMs are used to train the weights for score level fusion.

2.3.3 CMU PIE Face Database

CMU PIE face database [45] contains 68 subjects with 41,368 images captured under different poses, illuminations and expressions. This thesis uses 105 near frontal-view face images for each individual, randomly select six for training, four for validation and the rest for testing. Some selected images are shown in Fig. 2.3. Four types of features, Eigenfaces [46], Fisherfaces [46], Laplacianfaces [25] and local binary patterns (LBP) [47] are extracted for each image. Parameters introduced from these features are determined as suggested in their papers [25] [46] [47]. Linear SVM was employed to trained the classifiers by the training data. The best SVM parameter is selected by the validation set from $C \in \{10^{-3}, \cdots, 10^{3}\}$. The SVM outputs of the training data are used to train the weights for classifier fusion methods. This

[3] `http://www.robots.ox.ac.uk/~vgg/data/flowers/17/index.html`

Figure 2.3: Example images from CMU PIE face database

Figure 2.4: Example images from FERET face database

experiment is repeated ten times on this database.

2.3.4 FERET Face Database

In FERET database [48], there are 14,126 images from 1,199 individuals. This thesis selects 72 individuals with six near frontal-view face images per person under different face expressions. Some selected images are shown in Fig. 2.4. In the experiments using this database, six images for each individual are randomly separated into training, validation and testing sets with equal size, i.e. two images for each set. The image features and classification scores are extracted as in the CMU PIE database.

2.3.5 Weizmann Human Action Database

Weizmann database [49] contains 93 videos from nine persons, each performing ten actions, i.e. "bend", "jumping jack", "jump in place on two legs", "jump forward on two legs", "galloping sideways", "skip", "run", "walk", "wave one hand", and "wave two hands". Example images from the videos are shown in Fig. 2.5. Nine-fold

Figure 2.5: Example images from videos representing all the ten actions in Weizmann

Figure 2.6: Example segmented images from videos in Weizmann database

cross validation protocol is employed to evaluate the proposed method. The average recognition rate is recorded. Eight-fold CV is performed on the training data, and the CV outputs are used to train the learning models and select the best parameters.

For manifold learning methods, the information saliency method [50] is employed to extract regions of interest and detect periodic motion cycles following the procedures in [51] [52]. While several action units can be detected for one video clip, one cycle per video is used in the experiments (refer to [51] [52] for details). Fig. 2.6 shows some example segmented images from the videos representing the ten actions in Weizmann database. The extracted images are normalized into 100×100 pixels and represented by feature vectors with dimension 10000.

For the fusion methods, the bag-of-features approach is adopted for this database to extract eight types of features for the videos. First, space-time interest points are detected by [53] for each video. For each interest point, eight local descriptors, namely 1) intensity (Int), 2) intensity difference (IntDif), 3) histograms of optical

flow (HoF) without grid, 4) histograms of gradient (HoG) without grid, 5) HoF with 2D grid (HoF2D), 6) HoG with 2D grid (HoG2D), 7) HoF with 3D grid (HoF3D), and 8) HoG with 3D grid (HoG3D), are generated based on the $9 \times 9 \times 5$ cuboid centered by the interest point. For the first descriptor, pixel intensities of the local cuboids are considered as feature vectors, whose dimension is $9 \times 9 \times 5 = 405$. For the second descriptor, pixel intensities of the local cuboids in two successive frames are subtracted with each other to generate the feature. So the dimension is $9 \times 9 \times 4 = 324$. For the third and forth features, histogram features based on gradient and optical flow orientation are computed on each spatial blocks in local cuboids like [24], then HoGs and HoFs are concatenated together over time dimension respectively in the local cuboids. 8 orientations are used to compute the histogram, so the dimension of these two features is $8 \times 5 = 40$. For the fifth and sixth features, in the spirit to the SIFT descriptor [23], spatial blocks in local cuboids over time are divided into 2×2 arrays, and histograms of 8 orientations are computed on each sub-block individually. Thus, the dimension for them is $2 \times 2 \times 8 \times 5 = 160$. The last two features proposed in [54] which is similar to the previous two, divide the local cuboids into $2 \times 2 \times 2$ 3D arrays, so the dimension is $2 \times 2 \times 2 \times 8 = 64$. After different kinds of features have been generated, we follow the general bag-of-features representation [54] to form the feature vectors for classifier combination and feature fusion. Training data are used to construct the vocabulary. And the number of clusters is set as 100 in this database. Linear SVM classifiers are employed as mentioned in Section 2.3.3.

2.3.6 KTH Human Action Database

There are 25 subjects performing six actions under four scenarios in KTH database [55]. The six actions include "boxing", "hand clapping", "hand waving", "jogging", "running" and "walking". The four scenarios are outdoors (S1), outdoors with scale variations (S2), outdoors with different clothes (S3) and indoors (S4). Example

Figure 2.7: Example images from videos in KTH (All six actions and four scenarios are presented)

Figure 2.8: Example segmented images from videos in KTH database

images from the videos are shown in Fig. 2.7.

For manifold learning methods, regions of interest and motion cycles are extracted similar to the procedures in Weizmann database (refer to [51] [52] for details). Fig. 2.8 shows some example segmented images from the videos representing the six actions in KTH database. The extracted images are normalized into 100×100 pixels and represented by two kinds of image features. The first one converts gray-scale images into vectors with dimension 10000, while the second one computes the GIST feature [56] on each extracted image. For the GIST feature, the parameters about the number of orientation per scale and the number of blocks are selected from {4,8}. Three training/testing protocols (refer to [57] for details about the relationship between the performance and evaluation protocol) are used for evaluation and will be further elaborated in the experiment section later.

For evaluation of the fusion methods, the common setting in [55] is employed to separate the video set into training (8 persons), validation (8 persons), and testing

Figure 2.9: Example segmented images from videos in UCF sport database

(9 persons) sets. Eight local features including intensity, intensity difference, HoF, HoG, HoF2D, HoG2D, HoF3D and HoG3D, are extracted from videos as reported in Section 2.3.5. Training and validation data are used to construct the vocabulary. And the number of clusters is set as 600 in this database. Linear SVM classifiers are employed as mentioned in Section 2.3.3. Eight-fold CV is performed on the training data, and the CV outputs are used to train the fusion models. The best parameters are selected by the validation set on this database.

2.3.7 UCF Sports Database

UCF sports database [58] contains ten types of sports actions, including diving (Div), golf swinging (Gol), kicking (Kic), lifting (Lif), horseriding (Rid), running (Run), skateboarding (Ska), swinging at the bench (SwB), swinging at the high bar (SwH), and walking (Wal). There are totally 150 real videos in this database and each action class has different number of training samples. Bounding boxes are provided to segment regions of interest. Fig. 2.9 shows the images representing different actions after the segmentation with the bounding boxes. GIST feature is computed on each extracted image as for the KTH database. Following the protocol in [58], the leave-one-sample-out cross validation setting is employed for evaluation.

2.3.8 Hollywood Human Action Database

Hollywood Human Action (HOHA) database [54] consists of eight types of actions, including Answer Phone (AnP), Get out of Car (GoC), Hand Shake (HS), Hug Per-

Figure 2.10: Example bounding boxes and images from videos in HOHA database

son (HP), Kiss (Ki), Sit Down (SiD), Sit Up (SiU), and Stand Up (StU). Following the evaluation protocol in [54], 219 and 211 videos with "clean" annotations are used for training and testing, respectively. The annotations provided by this database are used to segment the action unit from videos temporally. For the manifold learning methods, regions of interest are extracted manually as indicated by the bounding boxes in Fig. 2.10. And GIST feature is computed on each extracted region as for the KTH database. For the fusion methods, both the proposed spatio-temporal manifold representation and the eight local features as mentioned in Section 2.3.5 are used for experiments. Linear SVM classifiers are employed as reported in Section 2.3.3. Five-fold CV is performed on the training data, and the CV outputs are used to train the fusion models and select the best parameters.

2.3.9 Hand Gesture Action Database

Cambridge-Gesture database [59] consists of 900 image sequences of nine hand gesture classes under five kinds of illuminations. The nine hand gesture actions include Flat-Leftward (FL), Flat-Rightward (FR), Flat-Contract (FC), Spread-Leftward (SL), Spread-Rightward (SR), Spread-Contract (SC), V-Shape-Leftward (VL), V-Shape-Rightward (VR), and V-Shape-Contract (VC). Each class and illumination set contains 20 sequences. Example images from this database are shown in Fig. 2.11. Following the experimental protocol in [59] [60], 20 sequences per class with plain illumination is randomly partitioned into 10 sequences for training and

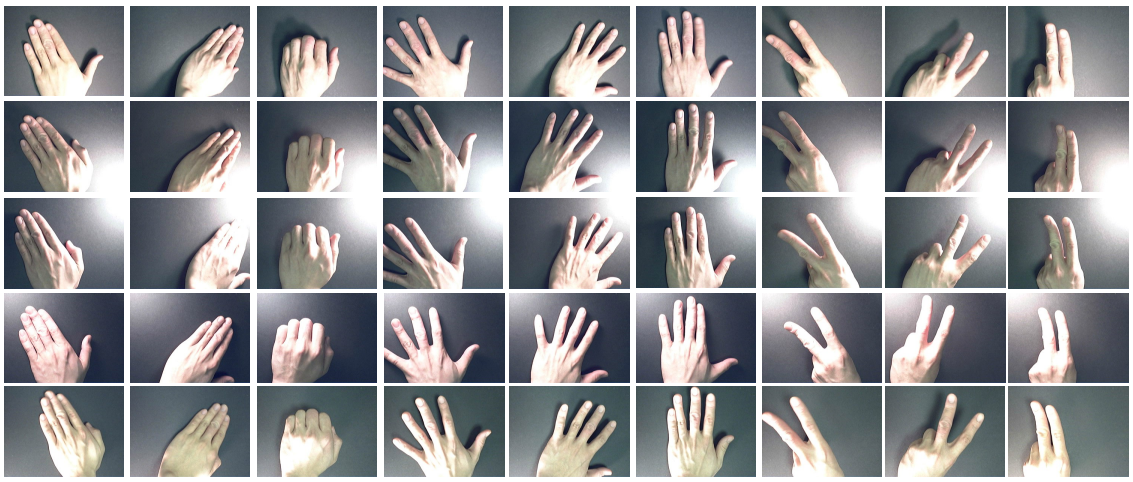


Figure 2.11: Example images from sequences in Cambridge-Gesture database

the other 10 sequences for validation, while testing is performed for the data sets with the other four illuminations. GIST feature is computed on each extracted image as for the KTH database.

2.3.10 VOC 2007 Object Categorization Database

PASCAL VOC 2007 [61] is one of the benchmark databases for visual object recognition in realistic scenes. There are around 10,000 consumer images of 20 different object categories, which are collected from the photo-sharing website. With the default split, 5,011 images are used for training, while 4,952 images for testing. Eight features reported in [62], including RGB, HSV, LAB, dense SIFT, Harris SIFT, dense HUE and Harris HUE with 3×1 horizontal decomposition of images as well as GIST descriptor, are available on the website[4] and employed in this experiment. Following the settings in [62], kernel SVMs are used for this experiment, and the kernel matrices are defined as $\exp\left(-d(\vec{x}, \vec{x}')/\eta\right)$, where d is the distance and η is the mean of pairwise distances. The SVM parameter is selected from $C \in \{10^{-3}, \cdots, 10^{3}\}$. Five-fold CV is performed on the training data, and the learning models and the best parameters are determined by the CV results. In order to reduce the impact

[4] `http://lear.inrialpes.fr/pubs/2010/GVS10/`

of wiggles in the recall and precision curve, the fusion methods are evaluated by the interpolated average precision as reported in [63].

2.3.11 Columbia Consumer Video Database

Columbia Consumer Video (CCV) database [64] is a recently developed benchmark for consumer video analysis. This database contains 9,317 YouTube videos over 20 semantic categories, in which around half of the videos are used for training and the other half for testing. In this experiment, three online available features[5] are employed, including SIFT visual feature, spatial-temporal interest point (STIP) visual feature, and Mel-frequency cepstral coefficients (MFCC) audio features reported in [64], to evaluate the fusion methods. The training procedures are the same as in Section 2.3.10. Following [64], the average precision is calculated by the the uninterpolated recall and precision curve to evaluate the fusion methods.

[5] `http://www.ee.columbia.edu/ln/dvmm/CCV/`

Chapter 3

Manifold learning for Spatio-Temporal Feature Representation

3.1 Introduction

As discussed in Chapter 1, feature extraction is a very important step in the fusion process. Among feature extraction algorithms, manifold learning has been successfully applied to many visual applications including 3D body configuration recovery [65], face recognition [25], image based age estimation [66], etc. Many existing manifold learning methods, e.g. Isometric feature Map (Isomap) [67], Locally Linear Embedding (LLE) [68] and Laplacian Eigenmap (LE) [69], aim at discovering the intrinsic geometrical structure of a data manifold. These manifold learning frameworks are designed for general applications, but they do not fully consider the temporal information for videos applications.

In [70], Spatio-Temporal Isomap (ST-Isomap) is proposed to construct the local neighbors emphasizing the similarity between the temporal related blocks. Besides ST-Isomap, the temporal extension to Laplacian Eigenmap (TLE) is proposed in [71]

and achieves better performance. Both ST-Isomap and TLE are unsupervised methods. That means class label information does not take into account. In turn, it may not be efficiently applied to supervised dimensionality reduction problem. Recently, supervised manifold embedding has been proposed. Existing methods include Locality Sensitive Discriminant Analysis (LSDA) [29], Local Fisher Discriminant Analysis (LFDA) [28], Marginal Fisher Analysis (MFA) [30], and Supervised Locality Preserving Projection (SLPP) [31]. Although these methods show that label information is important, they hardly take temporal cue into account.

In this context, Jia and Yeung [32] proposed a Local Spatio-Temporal Discriminant Embedding (LSTDE) method to discover the local spatial and local temporal discriminant structures. LSTDE is derived by maximizing the inter-class variance and minimizing the intra-class variance based on the local spatio-temporal information. However, LSTDE does not fully consider the global constraints of the temporal orders in action sequences.

To overcome the limitations mentioned above and follow the supervised spatial neighborhood topology learning (SNTL) method [72], this thesis proposes a new method to learn the manifold structure by using supervised spatial and temporal pose correspondence information from the topological aspect of manifold. As we know, topology is an important concept in the definition of a manifold [73]. By analyzing the topological base in existing methods for action recognition, a new topology is defined by spatial distribution and class label information, which are used to construct the supervised spatial (SS) neighborhood. However, the construction of SS neighbors does not take full advantage of the temporal information. Thus, by further analyzing the global constraint of temporal labels in action sequences, the temporal pose correspondence (TPC) neighborhood is developed. The supervised spatial information and global constraint of temporal labels are fused by taking the union of SS and TPC neighbors. At last, the optimal linear projection functions preserving neighborhood relationship are obtained by solving a generalized eigenvalue problem. The proposed method namely, Supervised Spatio-Temporal Neighborhood

Topology Learning (SSTNTL), not only can discover local structures with the help of label information, but also can comparing the global similarity of action sequences to separate different actions.

The rest of this chapter is organized as follows. Section 3.2 and Section 3.3 review some related works on video analysis and manifold learning, respectively. In section 3.4, the proposed method for spatio-temporal feature extraction is presented. Experimental results and a brief summary of this chapter are reported in Sections 3.5 and 3.6, respectively.

3.2 Review on Manifold Learning for Video Analysis

Video analysis is an active research topic in computer vision and pattern recognition, due to a wide range of potential applications, such as intelligent video surveillance, perceptual interface and content-based video retrieval. While many algorithms and systems have been developed in the last decade [74] [75] [76], recognizing actions in videos still remains challenging. In action recognition, a key issue is to extract useful action information from raw video data. So far various approaches have been proposed to extract features from video sequences, such as key frame extraction [77] [78] [79], space-time interest point detection and description [80] [54] [81], key point trajectory based approaches [82] [83] [84], etc.

In the last few years, there has been increasing interest in analyzing actions using manifold embedding methods, since action data may lie on a low-dimensional manifold embedded in the high-dimensional image space [85]. In [65], the gait manifolds under different viewpoints were learned by LLE and used for viewpoint estimation, 3D configuration recovery, new instance synthesis, etc. In [86], manifold representations learned by LE were used for tracking and 3D motion reconstruction.

Along this direction, manifold learning was used for action recognition in [32] [71] [85] [87]. In [85] [87], Locality Preserving Projection (LPP) [88] was used to learn the dynamic shape manifolds for action matching. Since LPP is closely related to the proposed method, a detailed introduction to LPP was presented in Section 3.3. In order to take advantage of the temporal information, TLE was proposed in [71] for unsupervised dimensionality reduction of time series. TLE follows the LE framework and constructs the neighborhood by the adjacent temporal and repetition temporal information. As demonstrated in [71], TLE ensured the temporal coherence and improved the generalization properties of the embedded low dimensional spaces. Besides LPP and TLE, LSTDE [32] was proposed to discover the local spatio-temporal discriminant structures for human action recognition. LSTDE constructs the projections from three aspects: minimizing the Euclidean distances between close data points of the same action class, maximizing the Euclidean distances between close data points of different classes, and maximizing the principal angles between video segments of close data points from different classes. Experimental results [32] demonstrated that LSTDE can improve the recognition performance over some representative manifold embedding methods.

3.3 Revisiting Locality Preserving Projection and Its Supervised Version

Locality preserving projection (LPP) [88] is a linear approximation of the nonlinear Laplacian eigenmap (LE) [69]. Suppose $\vec{x}_1, \vec{x}_2, \cdots, \vec{x}_N$ are N samples which lie on a manifold in R dimension Euclidean space. The problem of LPP is to find a transformation matrix P which best preserves the neighborhood relationship of the manifold containing $\vec{x}_1, \vec{x}_2, \cdots, \vec{x}_N$. Suppose $\vec{x}_1, \vec{x}_2, \cdots, \vec{x}_N$ are represented by $\vec{z}_1, \vec{z}_2, \cdots, \vec{z}_N$ in r ($\ll R$) dimension Euclidean space, where $\vec{z}_n = P^T \vec{x}_n$. The algorithmic procedure of LPP is summarized as follows [88],

1. Constructing the adjacency graph: Let G be a graph with N nodes. An edge is connected between nodes i and j, if $\vec{x}_i \in \mathcal{N}(\vec{x}_j)$ or $\vec{x}_j \in \mathcal{N}(\vec{x}_i)$, where $\mathcal{N}(\vec{x}_i)$ denotes a small neighborhood of $\vec{x}_i$. There are two ways to define $\mathcal{N}(\vec{x}_i)$: i) k-nearest neighbors (k-NN): $\mathcal{N}(\vec{x}_i)$ is made up of data points among the k nearest neighbors of $\vec{x}_i$ and ii) ε-neighborhood: $\mathcal{N}(\vec{x}_i) = \{\vec{x}_n | \|\vec{x}_n - \vec{x}_i\| < \varepsilon\}$.

2. Choosing the weights: There are two variations to choose weights, i.e. simple minded, $w_{ij} = 1$, or heat kernel, $w_{ij} = e^{-\|\vec{x}_i - \vec{x}_j\|^2/t}$, $t \in \mathbb{R}$, if and only if nodes i and j are connected by an edge; $w_{ij} = 0$, otherwise.

3. Eigenmap: Compute the eigenvectors and eigenvalues for the generalized eigenvalue problem:

$$X\mathcal{L}X^T\vec{e} = \lambda X\mathcal{D}X^T\vec{e} \tag{3.3.1}$$

where $X = (\vec{x}_1, \vec{x}_2, \cdots, \vec{x}_N)$, $\mathcal{D}$ is a diagonal matrix with $\mathcal{D}_{ii} = \sum_j w_{ij}$ and $\mathcal{L} = \mathcal{D} - \mathcal{W}$ is the Laplacian matrix.

Let column vectors $\vec{e}_1, \vec{e}_2, \cdots, \vec{e}_r$ be the eigenvectors of (3.3.1), such that $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_r$. Thus, the projection matrix and the embedding are given by $P = (\vec{e}_1, \vec{e}_2, \cdots, \vec{e}_r)$ and $\vec{z}_n = P^T\vec{x}_n$.

On the other hand, a supervised version of LPP (SLPP) is proposed in [31]. SLPP is based on the framework of LPP. In SLPP, the adjacency graph is constructed by class label information, i.e. an edge is connected between nodes i and j, if $\vec{x}_i$ and $\vec{x}_j$ are from the same class. Except for this, the other procedure is the same as that in LPP.

3.4 Supervised Spatio-Temporal Neighborhood Topology Learning

The block diagram of the action recognition framework is shown in Fig. 3.1. After preprocessing raw videos, a sequence of images containing the regions of interest are

obtained. For non-periodic actions, the whole sequences with segmented region of interest are denoted as action units and used for further processing. For periodic actions, one cycle in the action sequence is extracted by period detection method and denoted as an action unit similar to those of non-periodic actions. Each action unit is then represented by a set of feature vectors. Denote the normalized feature vectors by chronological order for action unit A as $\vec{x}_1, \cdots, \vec{x}_{N_A}$, where N_A is the number of segmented images in A. After that, each feature representation $\vec{x}_n$ is mapped to the embedded manifold by the proposed spatio-temporal manifold learning method. At last, a set-based classifier is employed on the sequence $\vec{z}_1, \cdots, \vec{z}_{N_A}$, and the action label is obtained.

As discussed in Section 3.3, the major difference between LPP and SLPP is the adjacency graph. In this section, it is shown that the adjacency graph is intimately related to the topological base from mathematical perspective. As we know, topology is an important concept in the definition of a manifold, so we will first give a topological analysis in the context of action recognition. Then, this thesis proposes to construct the neighborhood topology by the supervised spatial as well as temporal pose correspondence information. Finally, employing the locality preserving property in LPP, the proposed SSTNTL is learned by solving a generalized eigenvalue problem.

3.4.1 Topological Analysis for Action Recognition

In action recognition, actions are regarded as data points on a manifold. Supposes there are L classes of actions and data points from all the actions lie on a smooth and compact manifold. Different actions may be close to each other on the manifold, because different actions share similar poses. This is illustrated in Fig. 3.2, which shows two actions, namely "run" and "walk" It can be seen that poses (frames) indicated with red rectangle in "run" and "walk" actions are similar. As such, the distance between certain data points from these two actions will be small. This

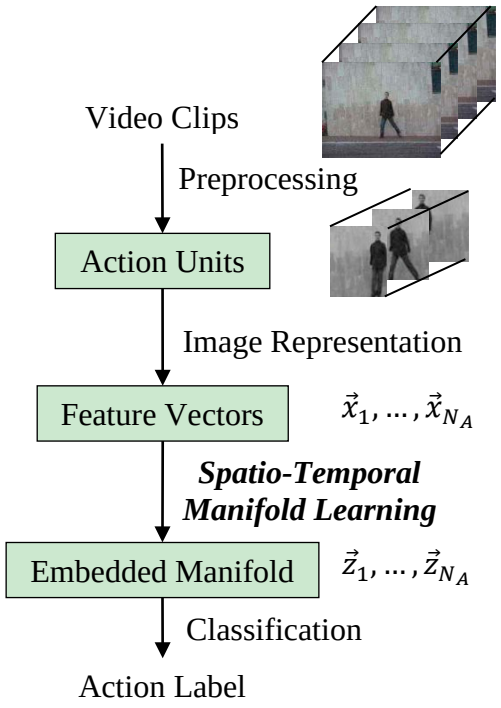


Figure 3.1: Manifold learning based action recognition framework

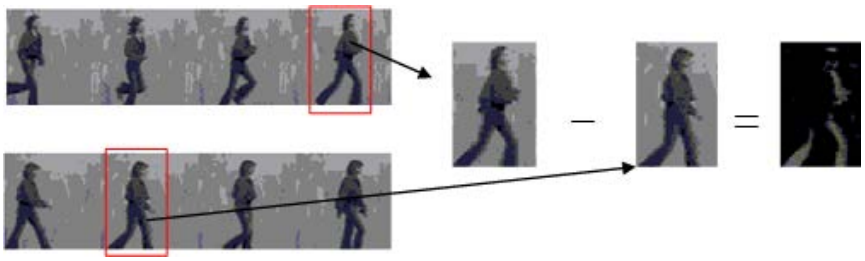
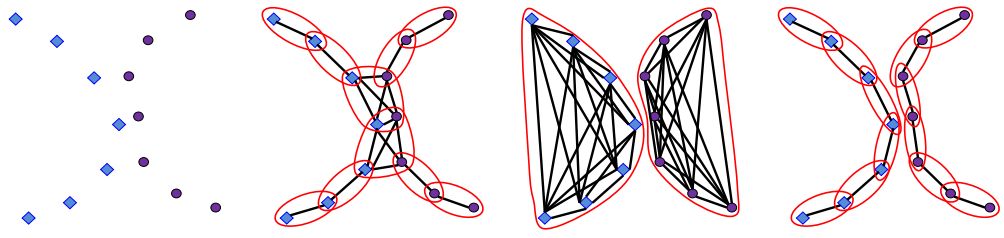


Figure 3.2: Similar pose in two different actions



(a) (b) (c) (d)

Figure 3.3: (a) Visualization of two action sequences. Visualization of the topological base and adjacency graphs in (b) LPP, (c) SLPP and (d) the proposed supervised spatial method.

means sequences representing these two actions may be close with each other. In this case, the two actions in Fig. 3.2 can be represented by data points (diamond and circle represent different actions) as shown in Fig. 3.3(a). Denote the two actions as A_i and A_j. And let $A = A_i \cup A_j$. Usually, the topology defined on A is induced by the Euclidean topology. Denote this topology as τ_e. Since topology is too abstract to understand, the topological base [89] is investigated instead. In mathematics, if $\tau = \overline{\mathscr{B}}$, where $\overline{\mathscr{B}}$ represents a family of sets generated by the family of sets $\mathscr{B}$, i.e. $\overline{\mathscr{B}} = \{U \subset X | \forall \vec{x} \in U, \exists \mathcal{B} \in \mathscr{B}, s.t. \vec{x} \in \mathcal{B} \subset U\}$, $\mathscr{B}$ is called the topological base of the topological space (A, τ). With this representation, τ_e can be written as

$$\tau_e = \overline{\mathscr{B}_e}, \quad \mathscr{B}_e = \{A \cap \mathcal{B}(\vec{x}_n, \varepsilon) | \vec{x}_n \in A, \varepsilon > 0\} \tag{3.4.2}$$

where $\mathcal{B}(\vec{x}_n, \varepsilon)$ is a ball centered at $\vec{x}_n$ and with radius ε.

The topology in LPP is constructed with fixed ε in τ_e, which can be written as

$$\tau_{\text{LPP}} = \overline{\mathscr{B}_{\text{LPP}}}, \quad \mathscr{B}_{\text{LPP}} = \{A \cap \mathcal{B}(\vec{x}_n, \varepsilon_{\text{LPP}}) | \vec{x}_n \in A\} \tag{3.4.3}$$

where ε_{LPP} is the parameter for the ε-neighborhood. For suitable parameter ε_{LPP}, the topological base in LPP can be represented by the ovals in Fig. 3.3(b). Based on the topological base, the adjacency graph in LPP is constructed by putting a link between any two points in the same oval, which is also shown in Fig. 3.3(b). From Fig. 3.3(b), it can be seen that close points from two different actions are connected together, so projections learned by LPP will map these points into low-dimensional space close to each other. However, this may not be good for classification purpose.

Besides the Euclidean topology, the topology in SLPP is given by

$$\tau_{\text{SLPP}} = \overline{\mathscr{B}_{\text{SLPP}}}, \quad \mathscr{B}_{\text{SLPP}} = \{A_i, A_j\} \tag{3.4.4}$$

The topological base and adjacency graph in SLPP are shown as ovals and edges in Fig. 3.3(c). Since SLPP only consider the class information, every two points from the same class are connected by an edge on the graph in Fig. 3.3(c). Therefore, the topology in SLPP hardly contains any local information of the data.

3.4.2 Supervised Spatial Neighborhood Topology Construction

According to the analysis in the above section, we can see that the topological base plays an important role in both LPP and SLPP. With the spatial and class label information of the data, the topological bases $\mathscr{B}_{\text{LPP}}$ in LPP and $\mathscr{B}_{\text{SLPP}}$ in SLPP can be constructed, respectively. However, $\mathscr{B}_{\text{LPP}}$ cannot separate data points from the same class, and $\mathscr{B}_{\text{SLPP}}$ hardly contains any local information. To overcome this limitation, we take the intersection of $\mathscr{B}_{\text{LPP}}$ and $\mathscr{B}_{\text{SLPP}}$ and define the supervised spatial topological base as

$$\mathscr{B}_{\text{SS}} = \{\mathcal{B}_{\text{LPP}} \cap \mathcal{B}_{\text{SLPP}} | \mathcal{B}_{\text{LPP}} \in \mathscr{B}_{\text{LPP}}, \mathcal{B}_{\text{SLPP}} \in \mathscr{B}_{\text{SLPP}}\} \tag{3.4.5}$$

Since $\mathcal{B}_{\text{LPP}} = A \cap \mathcal{B}(\vec{x}_n, \varepsilon_{\text{LPP}})$ and $\mathcal{B}_{\text{SLPP}} = A_i$ or A_j, $\mathscr{B}_{\text{SS}}$ in (3.4.5) can be rewritten and the novel topology can be defined as

$$\tau_{\text{SS}} = \overline{\mathscr{B}_{\text{SS}}}, \mathscr{B}_{\text{SS}} = \{A_k \cap \mathcal{B}(\vec{x}_n, \varepsilon_{\text{SS}}) | \vec{x}_n \in A_k, k = i \text{ or } j\} \tag{3.4.6}$$

where ε_{SS} is the radius of a ball. The topological base $\mathscr{B}_{\text{SS}}$ with (3.4.6) and the adjacency graph constructed by $\mathscr{B}_{\text{SS}}$ are shown in Fig. 3.3(d). With this new topology and adjacency graph, A can be separated as two connected components, A_i and A_j. In addition, close points from the same action are connected by an edge on the graph. Therefore, the proposed supervised spatial topology τ_{SS} not only preserves local structure of data points from the same class but also separates data points from different classes.

Besides spatial and label information, the supervised spatial topology τ_{SS} contains temporal adjacent information as well. As shown in Fig. 3.4, action sequences are characterized by poses deforming continuously over time. In mathematics, the temporal continuity means that for suitable ε_{SS}, there exists a positive integer δ, s.t. $\|\vec{x}_{n+t} - \vec{x}_n\| \leq \varepsilon_{\text{SS}}$, when $|t| \leq \delta$. By the definition of the supervised spatial topological base, it has

$$\vec{x}_{n+t} \in \mathcal{B}_{\text{SS}} \in \mathscr{B}_{\text{SS}}, \quad \text{when } |t| \leq \delta \tag{3.4.7}$$

This equation means that the temporal adjacent neighbors are contained in the supervised spatial neighbors, which is indicated by the red ovals in Fig. 3.4. Moreover, the construction of $\mathscr{B}_{\mathrm{SS}}$ avoids the non-trivial selection problem of the adjacent parameter, which is another advantage of the proposed method.

In practice, it is difficult to choose an optimal $\varepsilon_{\mathrm{SS}}$ when using ε neighborhood to construct the adjacency graph. Therefore, we construct the adjacency graph using k-NN. Since the number of data points from different classes may not be the same, we choose a percentage parameter a^{SS} instead of k to construct the neighborhood of each data point. The algorithmic procedure to construct the supervised spatial neighborhood $\mathcal{N}_{\mathrm{SS}}(\vec{x}_n)$ for each $\vec{x}_n$ is stated in Fig. 3.5.

3.4.3 Temporal Pose Correspondence Neighborhood Topology Construction

Although the supervised spatial neighborhood $\mathcal{N}_{\mathrm{SS}}$ contains temporal adjacent neighbors as mentioned in the above section, the construction of $\mathcal{N}_{\mathrm{SS}}$ still does not take full advantage of the temporal information. If we consider different sequences of the same action "bend" as shown in Fig. 3.6, it can be observed that different sequences share similar poses deforming similarly over time. However, if the neighborhood is constructed only by spatial information, the corresponding poses in different sequences of the same action may not be close to each other, due to the background and appearance changes. Thus, the temporal pose correspondence (TPC) between sequences of the same action may not be discovered by the supervised spatial neighbors. In this context, we propose to construct the neighborhood by the temporal pose correspondence directly.

Denote two action units of the same class as $A_i = \{\vec{x}_{i1}, \cdots, \vec{x}_{iN_i}\}$ and $A_j = \{\vec{x}_{j1}, \cdots, \vec{x}_{jN_j}\}$. Suppose feature vectors $\vec{x}_{in_i}$ in action sequence A_i and $\vec{x}_{jn_j}$ in A_j be corresponding to underlying action poses $\vec{u}_{in_i}$ and $\vec{u}_{jn_j}$, respectively. In this context, the pose relationship which may vary in speed can be found out by employing

Figure 3.4: Visualization of the temporal continuity in an action sequence

Algorithm 3.1. Constructing $\mathcal{N}_{\mathrm{SS}}$

Input: Training samples $\vec{x}_1, \cdots, \vec{x}_N$;
Corresponding labels $y_1, \cdots, y_N$;
Percentage parameter a^{SS};

Output: Supervised spatial neighborhood $\mathcal{N}_{\mathrm{SS}}$;

for $l = 1, \cdots, L$
Compute number of samples for class l, J_l;
Calculate k-NN parameter $k_l^{\mathrm{SS}} = a^{\mathrm{SS}} J_l$
endfor;
for $n = 1, \cdots, N$
Select the $k_{y_n}^{\mathrm{SS}}$ nearest samples respect to $\vec{x}_n$
from class y_n as $\mathcal{N}_{\mathrm{SS}}(\vec{x}_n)$;
endfor;
return $\mathcal{N}_{\mathrm{SS}}(\vec{x}_1), \cdots, \mathcal{N}_{\mathrm{SS}}(\vec{x}_N)$.

Figure 3.5: Algorithm 3.1: Construction of the SS neighborhood

Figure 3.6: Topological visualization of the TPC neighborhood

Dynamic Time Warping (DTW) [90] on the underlying pose sequences $\vec{u}_{i1}, \cdots, \vec{u}_{iN_i}$ and $\vec{u}_{j1}, \cdots, \vec{u}_{jN_j}$. Denote the warping function as

$$f = [(f_i(1), f_j(1)), \cdots, (f_i(T), f_j(T))] \tag{3.4.8}$$

where $1 \leq f_i(t) \leq N_i$, $1 \leq f_j(t) \leq N_j$, $f_i(t-1) \leq f_i(t)$, $f_j(t-1) \leq f_j(t)$, and T is an integer, s.t. $T \geq \max\{N_i, N_j\}$. DTW finds the optimal warping function f^* by solving the following optimization problem

$$\min_f \sum_{t=1}^{T} \omega(t) \|\vec{u}_{if_i(t)} - \vec{u}_{jf_j(t)}\| \tag{3.4.9}$$

The distance between corresponding poses $\vec{u}_{if_i^*(t)}$ and $\vec{u}_{jf_j^*(t)}$ must be small, though the distance between original feature vectors $\vec{x}_{if_i^*(t)}$ and $\vec{x}_{jf_j^*(t)}$ may be large due to background and appearance changes. However, underlying poses $\vec{u}_{i1}, \cdots, \vec{u}_{iN_i}$ and $\vec{u}_{j1}, \cdots, \vec{u}_{jN_j}$ are unknown, so the pose relationship cannot be obtained by directly solving optimization problem (3.4.9).

Alternatively, we model the relationship between feature vectors and underlying poses as follows. Suppose there is a projection for each sequence, which maps the feature vector subtracted by the background vector to the underlying pose, i.e.

$$\vec{u}_{in_i} = Q_i(\vec{x}_{in_i} - \vec{v}_i), \vec{u}_{jn_j} = Q_j(\vec{x}_{jn_j} - \vec{v}_j) \tag{3.4.10}$$

In equation (3.4.10), vectors $\vec{v}_i, \vec{v}_j$ and projection matrices Q_i, Q_j model the backgrounds and appearance changes in each action unit, respectively. Substituting $\vec{u}_{in_i}$ and $\vec{u}_{jn_j}$ with (3.4.10) into (3.4.9), the objective function becomes

$$\sum_{t=1}^{T} \omega(t) \|Q_i(\vec{x}_{in_i} - \vec{v}_i) - Q_j(\vec{x}_{jn_j} - \vec{v}_j)\| \tag{3.4.11}$$

Since vectors $\vec{v}_i, \vec{v}_j$ and projection matrices Q_i, Q_j are unknown variables in the optimization function (3.4.11), we minimize the upper bound of (3.4.11) given by (3.4.12)

instead.

$$\begin{aligned} &\|Q_i\| \sum_{t=1}^{T} \omega(t) \|\vec{x}_{if_i(t)} - \vec{x}_{jf_j(t)}\| \\ &+ \|Q_i - Q_j\| \sum_{t=1}^{T} \omega(t) \|\vec{x}_{jf_j(t)}\| \\ &+ \|Q_j\vec{v}_j - Q_i\vec{v}_i\| \sum_{t=1}^{T} \omega(t) \end{aligned} \tag{3.4.12}$$

Since $\sum_{t=1}^{T} \omega(t)$ is a constant respect to t and feature vectors $\vec{x}_{j1}, \cdots, \vec{x}_{jN_j}$ are normalized, minimizing objective function (3.4.12) is equivalent to solving the following optimization problem

$$\min_{f} \sum_{t=1}^{T} \omega(t) \|\vec{x}_{if_i(t)} - \vec{x}_{jf_j(t)}\| \tag{3.4.13}$$

This means that DTW performing on feature sequences $\vec{x}_{i1}, \cdots, \vec{x}_{iN_i}$ and $\vec{x}_{j1}, \cdots, \vec{x}_{jN_j}$ is equivalent to finding the minimum upper bound of the optimal pose matching (3.4.9).

The optimal warping function f^* obtained by solving (3.4.13) with DTW gives the correspondence between similar poses in different sequences of the same action. However, the difference between the corresponding frames may be very large due to the background and appearance changes. Consequently, if corresponding frames of the same pose in different sequences are in the same neighborhood, the manifold structure may be destroyed. Thus, neighboring corresponding frames are selected as the base of the temporal pose correspondence neighborhood topology, i.e.

$$\tau_{\text{TPC}} = \overline{\mathscr{B}_{\text{TPC}}}, \mathscr{B}_{\text{TPC}} = \{\mathcal{C}(\vec{x}_n) \cap \mathcal{B}(\vec{x}_n, \varepsilon_{\text{TPC}}) | \vec{x}_n \in X\} \tag{3.4.14}$$

where X is the universal set and $\mathcal{C}(\vec{x}_n)$ denotes the set containing poses in different sequences corresponding to $\vec{x}_n$. The topological base constructed by the temporal pose correspondence is indicated by the red ovals in Fig. 3.6.

Similar to the construction of the supervised spatial neighborhood, the temporal pose correspondence neighborhood is constructed by k-NN with a percentage parameter a^{TPC} instead of k. Denote J_l be the number of sequences for action l and

Algorithm 3.2. Constructing $\mathcal{N}_{\mathrm{TPC}}$

Input: Training sequences $A_1, \cdots, A_J$;
Corresponding labels $y_1, \cdots, y_J$;
Percentage parameter a^{TPC};

Output: Temporal pose correspondence neighborhood $\mathcal{N}_{\mathrm{TPC}}$;

for $l = 1, \cdots, L$
  Find action sequences with label ω_l;
  for $i_l = 1, \cdots, J_l$
    for $j_l = i + 1, \cdots, J_l$
      Obtain warping function f^* by solving (3.4.13);
      Add new elements to pose corresponding sets by f^*;
    endfor;
  endfor;
endfor;
for $n = 1, \cdots, N$
  Count the number of elements in $\mathcal{C}(\vec{x}_n)$, $\#\big(\mathcal{C}(\vec{x}_n)\big)$;
  Set $k_n^{\mathrm{TPC}} = a^{\mathrm{TPC}} * \#\big(\mathcal{C}(\vec{x}_n)\big)$;
  Select the k_n^{TPC} nearest samples in $\mathcal{C}(\vec{x}_n)$ as $\mathcal{N}_{\mathrm{TPC}}(\vec{x}_n)$;
endfor;
return $\mathcal{N}_{\mathrm{TPC}}(\vec{x}_1), \cdots, \mathcal{N}_{\mathrm{TPC}}(\vec{x}_N)$.

Figure 3.7: Algorithm 3.2: Construction of the TPC neighborhood

$J = J_1 + \cdots + J_L$, where L is the number of classes. The algorithmic procedure to construct the temporal pose correspondence neighborhood $\mathcal{N}_{\mathrm{TPC}}(\vec{x}_n)$ for each $\vec{x}_n$ is stated in Fig. 3.7.

3.4.4 Supervised Spatial and Temporal Pose Correspondence Neighborhood Topology Learning

As mentioned in Section 3.4.2 and Section 3.4.3, the topological base $\mathscr{B}_{\mathrm{SS}}$ contains supervised spatial information, while $\mathscr{B}_{\mathrm{TPC}}$ consists of temporal pose correspondence. In order to ensure that the topological base embodies both information, we take the union of $\mathscr{B}_{\mathrm{SS}}$ and $\mathscr{B}_{\mathrm{TPC}}$, and the fused topology namely, supervised

spatio-temporal neighborhood topology is defined as

$$\tau_{\mathrm{ST}} = \overline{\mathscr{B}_{\mathrm{ST}}}, \mathscr{B}_{\mathrm{ST}} = \{\mathcal{B}_{\mathrm{ST}} | \mathcal{B}_{\mathrm{ST}} \in \mathscr{B}_{\mathrm{SS}} \text{ or } \mathcal{B}_{\mathrm{ST}} \in \mathscr{B}_{\mathrm{TPC}}\} \tag{3.4.15}$$

Based on (3.4.15), the fused neighborhood of each $\vec{x}_n$ is given by

$$\mathcal{N}_{\mathrm{ST}}(\vec{x}_n) = \mathcal{N}_{\mathrm{SS}}(\vec{x}_n) \cup \mathcal{N}_{\mathrm{TPC}}(\vec{x}_n) \tag{3.4.16}$$

Suppose manifold $\mathscr{M}_{\mathrm{ST}}$ is defined with the neighborhood topology τ_{ST}. In [69], it is shown that the optimal mapping f preserving locality on manifold $\mathscr{M}_{\mathrm{ST}}$ is given by solving the following optimization problem

$$\arg \min_{f,\ s.t.\ \|f\|_{\mathscr{M}_{\mathrm{ST}}}=1} \int_{\mathscr{M}_{\mathrm{ST}}} \|\nabla f\|^2 \tag{3.4.17}$$

In order to avoid the out-of-sample problem, the mapping f is restricted to be linear. In [88], it is shown that the optimal linear projections preserving locality can be obtained by solving the following optimization problem,

$$\arg \min_{\vec{e},\ s.t.\ \vec{e}^T X \mathcal{D} X^T \vec{e}=1} \vec{e}^T X \mathcal{L} X^T \vec{e} \tag{3.4.18}$$

where $\mathcal{L}$ and $\mathcal{D}$ are the Laplacian and diagonal matrices as defined in Section 3.3.

Since the Laplacian $\mathcal{L}$ and diagonal matrix $\mathcal{D}$ are sparse with special structure, $X\mathcal{L}X^T$ and $X\mathcal{D}X^T$ can be computed by the following equations

$$X\mathcal{L}X^T = \begin{pmatrix} X_1 \cdots X_L \end{pmatrix} \begin{pmatrix} \mathcal{L}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathcal{L}_L \end{pmatrix} \begin{pmatrix} X_1^T \\ \vdots \\ X_L^T \end{pmatrix} = \sum_{l=1}^{L} X_l \mathcal{L}_l X_l^T \tag{3.4.19}$$

$$X\mathcal{D}X^T = \begin{pmatrix} X_1 \cdots X_L \end{pmatrix} \begin{pmatrix} \mathcal{D}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathcal{D}_L \end{pmatrix} \begin{pmatrix} X_1^T \\ \vdots \\ X_L^T \end{pmatrix} = \sum_{l=1}^{L} X_l \mathcal{D}_l X_l^T \tag{3.4.20}$$

At last, the algorithmic procedure of the proposed SSTNTL method is presented in Fig. 3.8.

Algorithm 3.3. SSTNTL

Input: Training sequences $A_1, \cdots, A_J$;
Corresponding labels $y_1, \cdots, y_J$;
Parameters $a^{\mathrm{SS}}, a^{\mathrm{TPC}}, r$;

Output: Projection matrix $P = (\vec{e}_1, \cdots, \vec{e}_r)$;

Construct $\mathcal{N}_{\mathrm{SS}}$ by Algorithm 3.1;
Construct $\mathcal{N}_{\mathrm{TPC}}$ by Algorithm 3.2;
Construct $\mathcal{N}_{\mathrm{ST}}$ by (3.4.16);
Set matrices $\Sigma_{\mathcal{L}} = 0$ and $\Sigma_{\mathcal{D}} = 0$;
for $l = 1, \cdots, L$
    Compute $\Sigma_{\mathcal{L}} = \Sigma_{\mathcal{L}} + X_l \mathcal{L}_l X_l^T$ and $\Sigma_{\mathcal{D}} = \Sigma_{\mathcal{D}} + X_l \mathcal{D}_l X_l^T$;
endfor;
Solve the eigenvalue problem $\Sigma_{\mathcal{L}} \vec{e} = \lambda \Sigma_{\mathcal{D}} \vec{e}$;
Sort the eigenvectors $\vec{e}_1, \cdots, \vec{e}_{d'}$ by eigenvalues $\lambda_1 \leq \cdots \leq \lambda_r$;
return $P = (\vec{e}_1, \cdots, \vec{e}_r)$.

Figure 3.8: Algorithm 3.3: The algorithmic procedure of SSTNTL

3.5 Experiments

In this section, we evaluate the proposed SSTNTL on five publicly available action databases: Weizmann [49], KTH [55], UCF [58], Hollywood human action [54] and Cambridge-Gesture [59] database. In the following sections, we first give a brief introduction to the experimental settings and the classifier used for the manifold learning methods in Section 3.5.1. Then, the results on these five databases are reported in Sections 3.5.2-3.5.6, respectively. At last, we compare SSTNTL with and without TPC neighbors in Section 3.5.7.

3.5.1 Settings and Classifier

For the video databases used in the experiments, we follow the procedures as described in Fig. 3.1 to preprocess the videos and represent the extracted images for each database as discussed in Section 2.3. After obtaining the image features, Principal Component Analysis (PCA) [39] is used for dimensionality reduction to

avoid the singular matrix problem and improve the computational efficiency. The dimension of PCA is determined by keeping around 90% energy.

As shown in Fig. 3.8, parameters a^{SS}, a^{TPC}, r need to be determined in SSTNTL. Since it is difficult and time-consuming to search the best combination of the three parameters simultaneously, we determine the parameters in a two-stage scheme. First, setting $r = 60$, a^{SS} and a^{TPC} are selected from $\{0.02, 0.04, \cdots, 0.2\}$ and $\{0.1, 0.2, \cdots, 0.9\}$ respectively. Second, the best dimension r is selected from two to the determined PCA dimension with step size two for fixed a^{SS} and a^{TPC}. On the other hand, the parameter selection procedure in other methods is similar to that in SSTNTL. The neighborhood parameter in LPP and LSDA is selected from the same set as a^{SS} in SSTNTL. Since LSTDE is time-consuming with large neighborhood parameter, the neighborhood parameter in LSTDE is selected from $\{0.01, 0.02, 0.03\}$ instead. In order to avoid extra parameter selection by the heat kernel weighting, the weight matrix is calculated by the simple minded method as mentioned in Section 3.3.

After feature extraction, we use a nearest neighbor framework in the classification stage. Let A_q be a query action sequence. The class label of A_q is given by

$$y_q = y_{\arg\min_j d(P^T Z_q, P^T Z_j)} \tag{3.5.21}$$

where d measures the distance between embedded feature sequences Z_q and Z_j. Since median Hausdorff distance gives more robust results as mentioned in [85], we define d as equation (3.5.22) in the following experiments.

$$d(Z_q, Z_j) = \underset{n_j}{\mathrm{median}} \min_{n_q} \|\vec{z}_{qn_q} - \vec{z}_{jn_j}\| + \underset{n_q}{\mathrm{median}} \min_{n_j} \|\vec{z}_{qn_q} - \vec{z}_{jn_j}\| \tag{3.5.22}$$

3.5.2 Results on Weizmann Human Action Database

The recognition accuracies of SSTNTL with different values of a^{SS} and a^{TPC}, and fixed $r = 60$ on this database are shown in Fig. 3.9. The darker element of the matrix in Fig. 3.9 represents higher recognition rate. The columns are the recognition rates

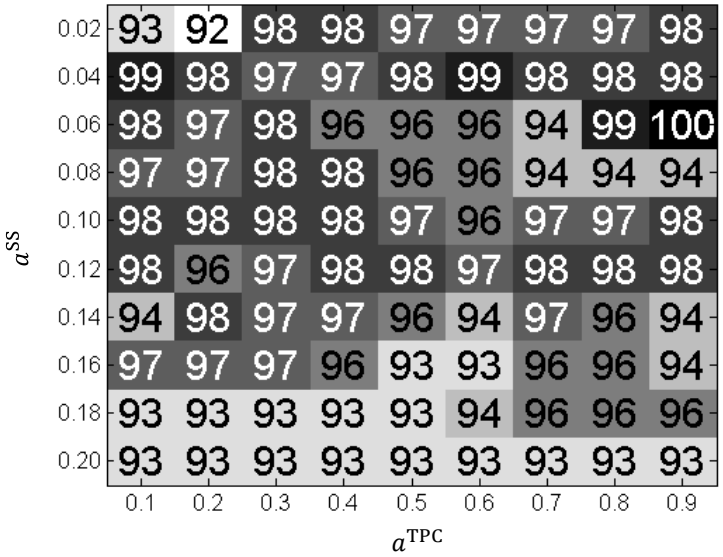


Figure 3.9: Recognition accuracy (%) of SSTNTL with different values of a^{SS} and a^{TPC}, and fixed $r = 60$ on Weizmann database

with changing a^{SS} for specific a^{TPC}, while rows are recognition rates with changing a^{TPC} for specific a^{SS}. From Fig. 3.9, we can see that the highest accuracy of 100% is achieved in this database, when $a^{\mathrm{SS}} = 0.06$ and $a^{\mathrm{TPC}} = 0.9$. From the last row in Fig. 3.9, we observe that the recognition rates do not change when $a^{\mathrm{SS}} = 0.2$ for different a^{TPC}. This may be due to the reason that the supervised spatial neighborhood $\mathcal{N}_{\mathrm{SS}}$ already contains the temporal pose correspondence neighborhood $\mathcal{N}_{\mathrm{TPC}}$, when $a^{\mathrm{SS}} = 0.2$. Moreover, the recognition accuracy changes rapidly from 92% to 98% when $a^{\mathrm{SS}} = 0.02$ and a^{TPC} changes slightly from 0.2 to 0.3. On the other hand, the accuracy has little change when a^{TPC} changes significantly from 0.3 to 0.9. These results show that the accuracy may not be a continuous function of the parameters. And, a stable and better performance can be achieved with larger a^{TPC}.

Fig. 3.10 shows the recognition accuracies of different methods with different dimensions. From Fig. 3.10, we can see that the recognition rates of SSTNTL are higher than those of all the other methods in every dimension. On the other hand, when the dimension is less than 20, SSTNTL, LPP and SLPP outperform LSDA and LSTDE a lot. This result suggests that the neighborhood topology learning

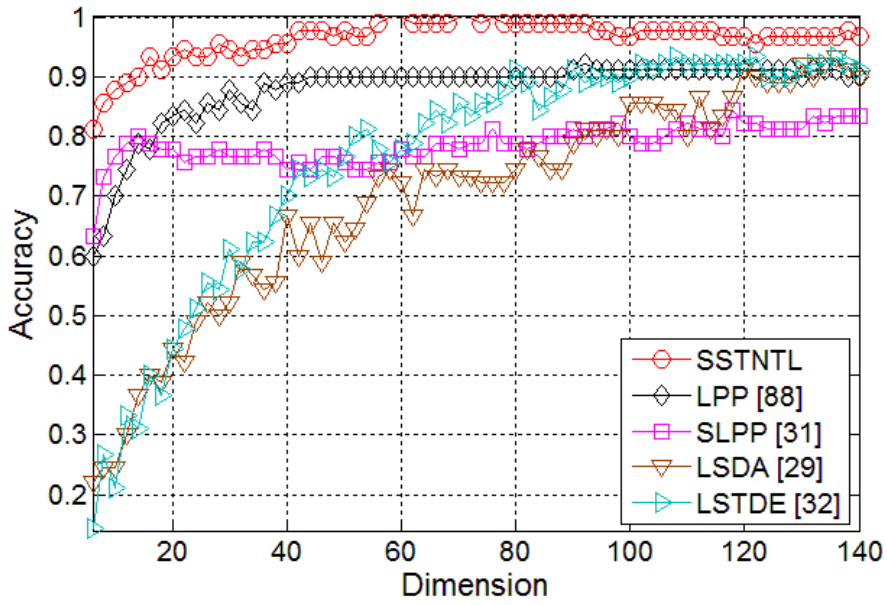


Figure 3.10: Recognition accuracy with different embedded dimensions for the best neighborhood parameters on Weizmann database

methods with clear manifold interpretation are much better than others, when the number of selected dimensions is small.

We compare the best recognition performances of different manifold embedding methods on this database in the second column of Table 3.1. SSTNTL achieves the highest recognition accuracy of 100%, and outperform other embedding methods by 6.7%. On the other hand, we compare SSTNTL with state-of-the-art algorithms[1] in Table 3.2. Our method outperforms space-time interest point based methods [80] [81] [91] as well as sparse representation approach [92]. Since Weizmann database is relatively simple, boosted exemplar learning (BEL) [79], local trinary patterns (LTP) [93] and the proposed method give the perfect performance of 100% recognition accuracy. While BEL classifies actions based on exemplars and LTP extends the local binary patterns, these two methods do not consider the global constraint of temporal labels. Thus, the proposed method outperforms them on the more challenging databases as shown in Table 3.3, Table 3.4 and Table 3.7.

[1]Please be noticed that the results of state-of-the-art methods are extracted from their papers under the same setting.

Database (Feature)	Weizmann (Gray)	KTH (Gray)	KTH (GIST)	UCF (GIST)	HOHA (GIST)
SSTNTL	**100.0**	**79.6**	**94.4**	**91.3**	**44.5**
LPP [88]	92.2	72.2	89.4	88.6	29.2
SLPP [31]	84.4	70.8	88.4	86.6	27.0
LSDA [29]	93.3	75.0	83.3	84.6	29.4
LSTDE [32]	93.3	75.9	80.6	82.6	26.5

Table 3.1: Recognition accuracies (%) of manifold embedding methods on different databases

Method	Accuracy
SSTNTL	**100.0**
LTP [93]	**100.0**
BEL [79]	**100.0**
Sparse Representation [92]	98.9
Effective Codebook [91]	95.4
BoW [80]	90.0
3D Gradients [81]	84.3

Table 3.2: Recognition accuracy (%) comparison with state-of-the-art action recognition systems on Weizmann database

3.5.3 Results on KTH Human Action Database

Three training/testing protocols (refer to [57] for details about the relationship between the performance and evaluation protocol) are used for evaluation and the results are reported as follows.

Following the split setting in [55], the KTH database is divided into training (eight persons), validation (eight persons) and testing (nine persons) sets. Eight-fold cross-validation is performed on the training and validation sets to find optimal parameters and model. We first compare SSTNTL with other manifold embedding

Method (train/test setting: split)	Accuracy
SSTNTL	94.4
Product Manifold [60]	**96.0**
Neighbor Hierarchy [94]	94.5
Dense Trajectory [84]	94.2
BoD+MKGPC [95]	94.1
Effective Codebook [91]	92.6
Harris 3D + HoF [96]	92.1
Random Forest [97]	91.8
HoG + HoF [54]	91.8
3D Gradients [81]	91.4
LTP [93]	90.1

Table 3.3: Recognition accuracy (%) comparison with state-of-the-art action recognition systems on KTH database with split setting

methods using gray-scale and GIST features in the third and fourth columns of Table 3.1, respectively. From Table 3.1, we can see that SSTNTL outperforms other methods using both image features, while the performances with GIST are better than those with gray-scale feature. Then, We compare the proposed method with state-of-the-art algorithms under the split setting in Table 3.3. From Table 3.3, we can see that the proposed method outperforms most of the existing methods and is comparable with the hierarchical approach [94]. While Product Manifold (PM) [60] achieves the highest accuracy, the result is obtained by performing PM on action sequences with manually spatio-temporal alignment.

For the second protocol, we evaluate the proposed method under leave-one-person-out (LOO) setting using all the four scenarios. The results are reported in Table 3.4. As shown in Table 3.4, the proposed method achieves the second highest accuracy of 96.3%. PM is better than the proposed method by 0.7%. However, the result with PM is obtained by performing spatio-temporal alignment manually.

Method (train/test setting: LOO)	Accuracy
SSTNTL	96.3
Product Manifold [60]	**97.0**
MoSIFT [57]	96.3
TCCA [59]	95.3
BEL [79]	95.3
Tracklet [83]	94.5
BoW [80]	83.3

Table 3.4: Recognition accuracy (%) comparison with state-of-the-art action recognition systems on KTH database with LOO setting

Method	S1	S2	S3	S4	Mean
SSTNTL	**98.0**	88.7	**96.0**	**100.0**	**95.7**
Tracklet [83]	**98.0**	**92.7**	92.0	96.7	94.8
AFMKL [98]	96.7	91.3	93.3	96.7	94.5
HSTM [99]	95.6	87.4	90.7	94.7	92.1

Table 3.5: Recognition accuracy (%) comparison with state-of-the-art action recognition systems on KTH database under each scenario

While the proposed method detects the regions of interest automatically, the result is also comparable.

At last, we compare SSTNTL with existing methods under each scenario with leave-one-person-out setting. The results are shown in Table 3.5. SSTNTL outperforms others under scenarios of outdoor, outdoor with clothes variation and indoor. Especially, the proposed method achieves 100% accuracy for the indoor scenario. This convinces that the proposed method is very effective under controlled setting. On the other hand, Tracklet [83] and AFMKL [98] are better than the proposed method under scenario two of outdoor with scale variation. The reason can be explained as follows. The features used in SSTNTL are obtained by motion detection.

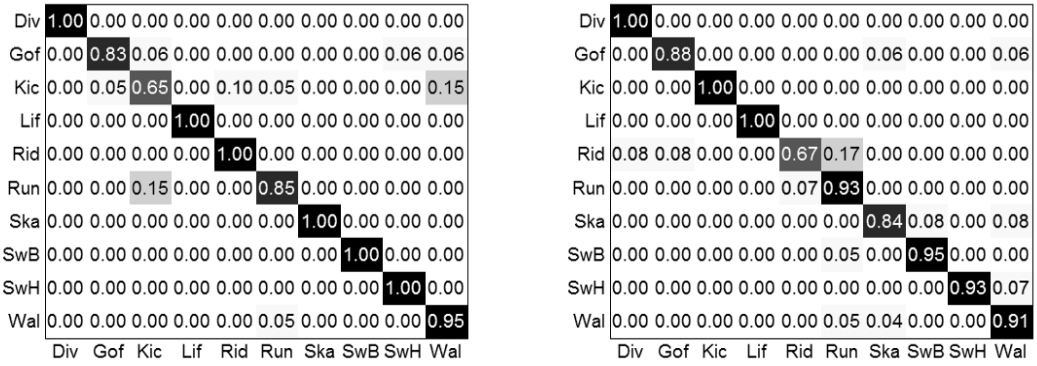
Method	Accuracy
SSTNTL	**91.3**
AFMKL [98]	**91.3**
Dense Trajectory [84]	88.2
Neighbor Hierarchy [94]	87.3
Dense HoF [96]	85.6
Sparse Representation [92]	83.8
LTP [93]	79.2

Table 3.6: Recognition accuracy (%) comparison with state-of-the-art action recognition systems on UCF database

Since motion detection under scenario two with scale variations is more challenged than that under the other three scenarios, the detected moving regions in scenario two must be less robust. Thus, local feature based methods, Tracklet and AFMKL, without motion detection outperform the proposed method under scenario two.

3.5.4 Results on UCF Sports Database

The last but one column in Table 3.1 shows the recognition rates of different manifold embedding methods on UCF Sports Database. The same conclusion can be drawn that SSTNTL outperforms other manifold embedding methods on this database. In Table 3.6, we further compare the proposed method with state-of-the-art action recognition systems. From Table 3.6, we can see that SSTNTL gives the highest accuracy 91.3% together with the augmented features (context and appearance distribution features) multiple kernel learning (AFMKL) method [98]. And the proposed method outperforms dense trajectory [84], space-time interest point (STIP) based methods [94] [96], sparse representation [92], and local trinary patterns (LTP) [93]. This is because the scene representations may be discriminative for some sport actions. And the proposed method not only captures the temporal variation,



(a) SSTNTL (b) AFMKL [98]

Figure 3.11: Confusion matrices on UCF sport database

but also makes use of the discriminative representations of image frames. Fig. 3.11 shows the confusion matrices of SSTNTL and AFMKL, which both achieves the best recognition accuracy. From Fig. 3.11(a), we can see that the proposed method achieves 100% accuracy for six in ten actions, including diving, lifting, riding horse, skateboarding, swinging at the bench and at the high bar. Comparing the confusion matrices in Fig. 3.11(a) and Fig. 3.11(b), the performance of the proposed method is better than or equal to that of AFMKL in classifying seven out of ten actions.

3.5.5 Results on HOHA Database

The average (Avg) values of the per-class precisions of the manifold embedding methods on HOHA database are recorded in the last column of Table 3.1. From Table 3.1, we can see that the average precision of SSTNTL is 15% higher than those of other manifold embedding methods on this database. This convinces the effectiveness of SSTNTL in the more challenging database, comparing to other manifold embedding methods.

In Table 3.7, we compare the proposed method with state-of-the-art action recognition systems in per-class precisions and their average. From Table 3.7, we can see that the average precision of SSTNTL is higher than the baseline method [54] by combining Histogram of Oriented Gradients (HoG) and Optical Flow (HoF), as well

Method	AnP	GoC	HS	HP	Ki	SiD	SiU	StU	Avg
SSTNTL	40.0	**62.5**	**44.4**	38.1	44.2	30.2	**44.4**	52.4	44.5
BoD + MKGPC [95]	**43.4**	46.8	44.1	**46.9**	**57.3**	**46.2**	38.4	57.1	**47.5**
HoG + HoF [54]	32.1	41.5	32.3	40.6	53.3	38.6	18.2	50.5	38.4
LTP [93]	35.1	32.0	33.8	28.3	57.6	36.2	13.1	58.3	36.8
Tracklet [83]	33.0	27.0	20.1	34.5	53.7	27.4	19.0	**60.0**	34.3
3D Gradients [81]	18.6	22.6	11.8	19.8	47.0	32.5	7.0	38.0	24.7

Table 3.7: Recognition precision (%) comparison with state-of-the-art action recognition systems on Hollywood database

as recently proposed descriptors [81] [93] [83]. And the proposed method outperforms others for classifying actions including Get out of Car, Hand Shake and Sit Up. The performance of the proposed method is comparable with, but lower than the multiple kernel Gaussian process classifier (MKGPC) [95]. This is attributed to the combination of bag-of-detector (BoD) scene descriptors, HoG, HoF and 3D Gradients for MKGPC, while the proposed method is based on one kind of feature.

3.5.6 Results on Hand Gesture Action Database

Recognition accuracies in different illumination sets on this database are presented in Table 3.8. Compared with the manifold embedding methods, SSTNTL outperforms others under the four different illumination sets. This convinces that SSTNTL is better than LPP and SLPP with other topologies, as well as the local discriminative algorithms even with temporal information.

In order to further compare the generalization ability of the manifold embedding methods, the 2D visualizations of the training and testing data with illumination set three are shown in Fig. 3.12. For each method, the 2D embedding of the training data is presented in the upper row, while the one of the testing data is in the lower row. From Fig. 3.12, we can see that the 2D embedding of the testing data deviates

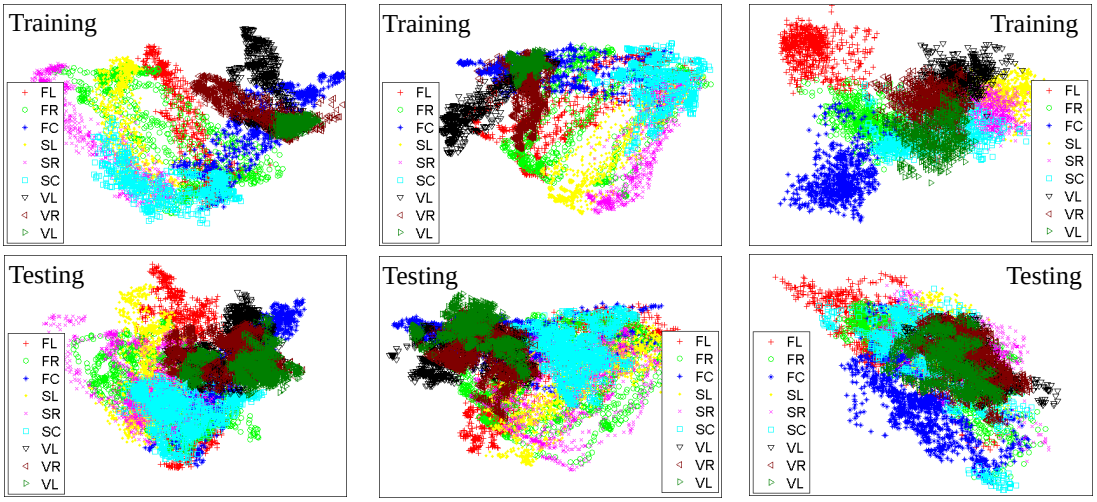
Method	Set 1	Set 2	Set 3	Set 4	Mean
SSTNTL	**89**	**89**	85	**91**	**89**
LPP [88]	85	84	75	89	83
SLPP [31]	66	65	53	68	63
LSDA [29]	66	67	58	68	65
LSTDE [32]	68	66	69	80	71
TCCA [59]	81	81	78	86	82
PM [60]	**89**	86	**89**	87	88

Table 3.8: Recognition accuracy (%) of different methods on Cambridge-Gesture database

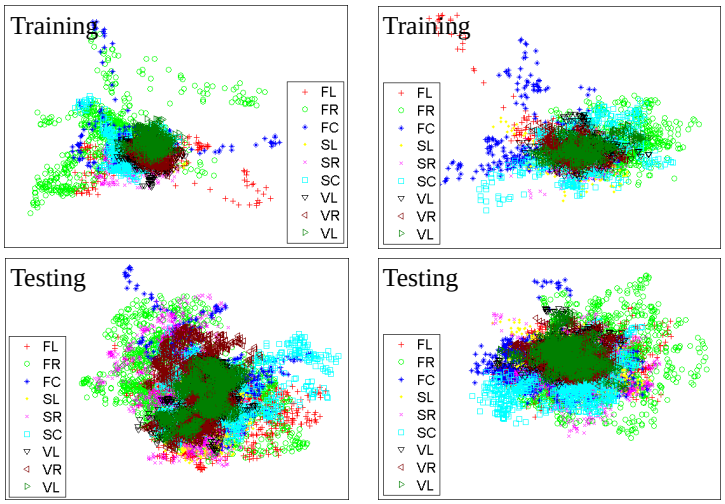
to a different degree from that of the training data by different method.

It is not easy to compare different methods by the 2D embeddings using visualization. As mentioned in [100], the median Hausdorff distance defined by (3.5.22) can be used to quantify the similarity between the 2D embeddings. Denote the training and testing embeddings as $Z^{\text{train}} = \{Z_1^{\text{train}}, \cdots, Z_L^{\text{train}}\}$ and $Z^{\text{test}} = \{Z_1^{\text{test}}, \cdots, Z_L^{\text{test}}\}$, where Z_l^{train} and Z_l^{test} are the embeddings for action class l. With these notations, we first measure the shape similarity between the training and testing data as two data sets. And the first similarity index is given by $1/d(Z^{\text{train}}, Z^{\text{test}})$, where d is a distance function defined by (3.5.22). With label information, we further calculate the average difference between the training and testing data of each class. And the second index is defined by $L/\sum_{l=1}^{L} d(Z_l^{\text{train}}, Z_l^{\text{test}})$.

The similarity quantification scores for these two indexes are shown in the second and third columns in Table 3.9. The similarity measures in Table 3.9 are consistent with each other for different methods. However, different from the results in Table 3.8, SLPP outperforms LSDA and LSTDE in terms of the two similarity measures. In this case, we further show the recognition rate with embedded dimension two and different dimension in the last column of Table 3.9 and Fig. 3.13, respectively. From Table 3.9, we can see that when the embedded dimension is two, the



(a) SSTNTL (b) LPP [88] (c) SLPP [31]



(d) LSDA [29] (e) LSTDE [32]

Figure 3.12: 2D visualization of different methods with the training and testing data on Cambridge-Gesture database

Method	Similarity 1	Similarity 2	Accuracy (dim=2)
SSTNTL	**0.941**	**0.698**	**22.2**%
LPP [88]	0.904	0.682	21.1%
SLPP [31]	0.578	0.485	18.3%
LSDA [29]	0.530	0.462	16.7%
LSTDE [32]	0.499	0.462	13.9%

Table 3.9: Two similarity measures of the 2D embeddings for training and testing data and recognition accuracy with embedded dimension two on Cambridge-Gesture database

recognition rates of the neighborhood topology learning methods are higher than those of LSDA and LSTDE, so the similarity results in Table 3.9 are reasonable. On the other hand, Fig. 3.13 indicates that SSTNTL, LPP and SLPP outperform LSDA and LSTDE, when the dimension is less than 50, which is similar to the results in Weizmann database. And these results show that the generalization ability of the topology based manifold learning methods is better than that of the local discriminability based methods, when the reduced dimension is low. And SSTNTL gives the best performance in this experiment.

We also compare the proposed method with state-of-the-art gesture action recognition algorithms. The Tensor Canonical Correlation Analysis (TCCA) [59] and Product Manifold (PM) [60] are used for comparison with the manifold embedding methods. From Table 3.8, we can see that both PM and SSTNLT get the highest accuracy of 89% for illumination set one. The product manifold method outperforms the others for set three, while SSTNTL outperforms others for illumination sets two and four. Compared with the mean recognition rates, SSTNTL achieves the highest mean accuracy of 89% in this database.

At last, the confusion matrices of the best four algorithms are shown in Fig. 3.14. Comparing the diagonal elements in the confusion matrices, SSTNTL outperforms LPP on eight actions, TCCA and product manifold method on five actions, respectively. This means SSTNTL outperforms others for classifying more than half of the

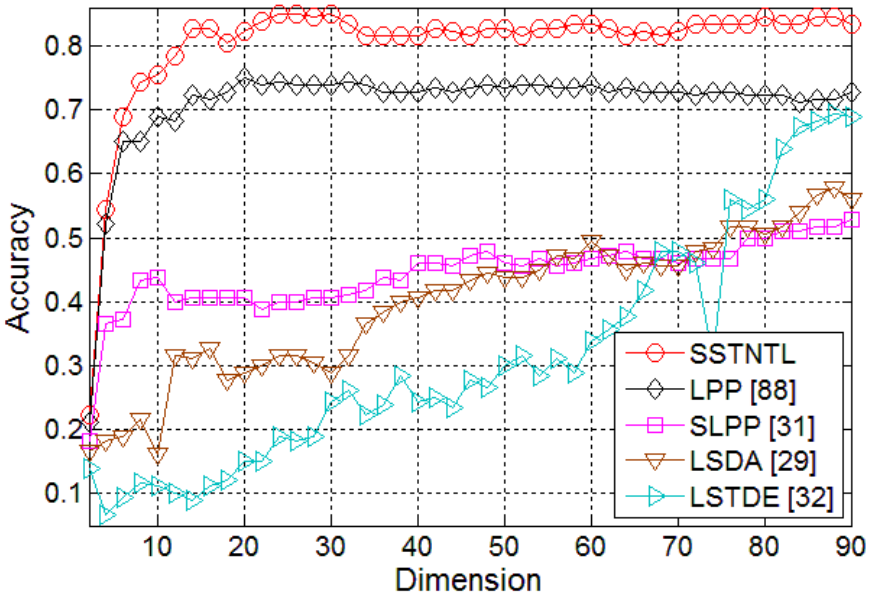
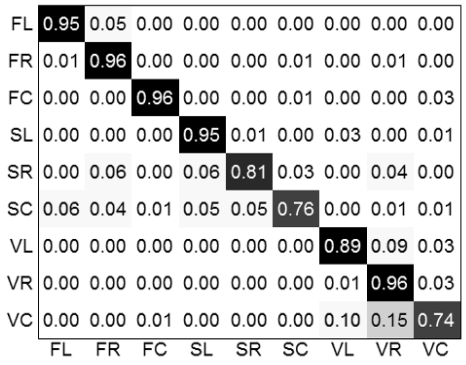
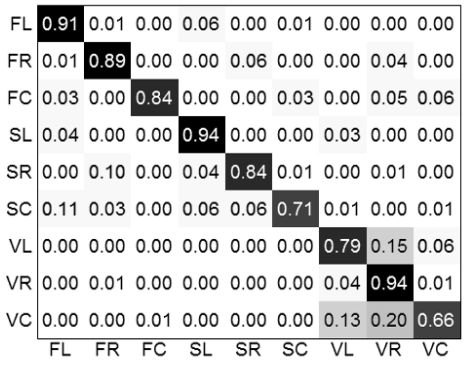


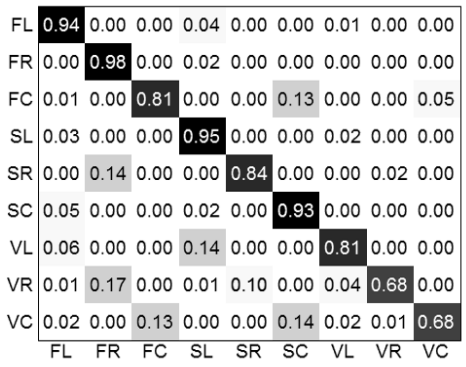
Figure 3.13: Recognition accuracy with the best neighborhood parameter and different embedded dimensions on Cambridge-Gesture database



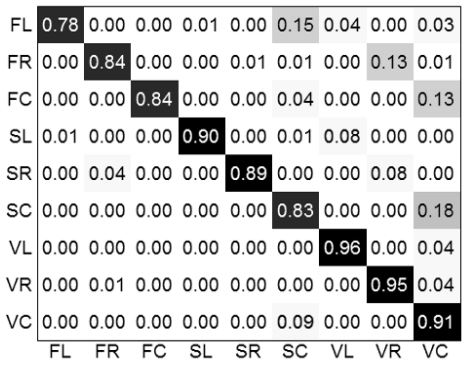
(a) SSTNTL



(b) LPP [88]



(c) TCCA [59]



(d) PM [60]

Figure 3.14: Confusion matrix on Hand Gesture confusion matrices

Method	Weizmann	KTH	UCF	HOHA	Gesture
With TPC	**100.0**	**94.4**	**91.3**	**44.5**	**89**
Without TPC	95.6	90.3	89.3	32.3	80

Table 3.10: Recognition accuracies (%) of SSTNTL with and without TPC neighborhood on different databases

gesture actions in this database. Overall speaking, SSTNTL is also performing well for hand gesture action recognition.

3.5.7 Comparing SSTNTL with and without TPC Neighbors

In the last experiment, we compare SSTNTL with and without temporal pose correspondence (TPC) neighbors constructed by the DTW method. From Table 3.10, we can see that SSTNTL with TPC neighbors outperforms that without TPC neighbors on different databases. This convince that the neighborhood constructed by the global constraint of temporal lables help to improve the performance.

In order to further show that the proposed method can discover the corresponding poses, we compare the supervised spatial neighborhood and TPC neighborhood on KTH database. Fig. 3.15 shows the TPC neighbors which cannot be detected by the supervised spatial information. From Fig. 3.15, we can see that most of the detected TPC neighbors are corresponding poses for the referred images on the left hand side. This gives the reason why SSTNTL with TPC neighbors is better.

3.6 Summary

In this chapter, a novel manifold learning method, namely Supervised Spatio-Temporal Neighborhood Topology Learning (SSTNTL) is proposed for action classification. S-

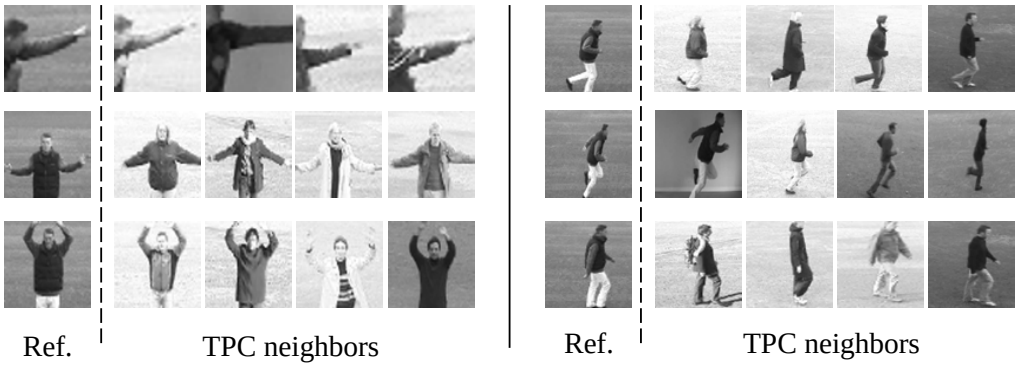


Figure 3.15: Example TPC neighbors do not belong to the SS neighborhood

tarting from analyzing the topological characteristics in the context of action recognition, we proposed to construct the neighborhood topology from two aspects. First, the spatial distribution containing local structures, as well as the label information separating data points from different class are used to construct the supervised spatial topology. Second, the temporal pose correspondence neighborhood is constructed by discovering the global constraints of temporal labels in action sequences of the same class. These two neighborhood topologies are fused by taking the union of them. Based on the fused topology, SSTNTL employs the locality preserving property in LPP, and solves the generalized eigenvalue problem to obtain the best projections that not only separate data points from different classes, but also preserve local structures and global constraints of temporal labels.

The proposed algorithm is evaluated on five publicly available action databases. Experimental results show that SSTNTL outperforms the neighborhood topology learning methods with other topologies, as well as the local discriminative algorithms even with temporal information. On the other hand, by analyzing the 2D visualizations of the training and testing data, it is shown that the generalization ability of the topology learning methods is better than that of the local discriminability based methods, when the reduced dimension is low. And SSTNTL shows the best generalization ability. In addition, compared with state-of-the-art action recognition algorithms, SSTNTL gives convincing performance for both human and gesture action recognition.

Chapter 4

Linear Dependency Modeling

4.1 Introduction

Instead of extracting a high discriminative feature, fusion has been proposed to improve the recognition performance [11] [14]. One popular approach to make use of all features is to train a classifier for each of them, and then combine the classification scores to draw a conclusion. While many classifier combination techniques [2] [8] [12] have been studied and developed in the last decade, it is a general assumption that classification scores are conditionally independent distributed. With this assumption, the joint probability of all the scores can be expressed as the product of the marginal probabilities. The conditionally independent assumption could simplify the problem, but may not be valid in many practical applications.

In [5], instead of taking the advantage of conditionally independent assumption, the classifier fusion method is proposed by estimating the joint distribution of multiple classifiers and performance is improved. However, when the number of classifiers is large, it needs numerous data to accurately estimate the joint distribution [27]. On the other hand, Terrades *et al.* [13] proposed to combine classifiers in a non-Bayesian framework by linear combination. Under dependent normal assumption (DN), they formulated the classifier combination problem into a constraint quadratic optimiza-

tion problem. Nevertheless, if normal assumption is not valid, the combination will not be optimal. Apart from these methods, LPBoost [4] and its multi-class variants [14] aim at determining the correct weighting for linear classifier combination to improve the classification performance. In [101], multi-class LPBoost is extended to online setting, so that it does not need to solve the linear programming (LP) problem from scratch for every new sample added to the system. These boosting based combination methods can model dependency implicitly as well.

All the above-mentioned methods perform the fusion based on classification scores and implicitly assume that the feature dependency can be reflected by the score dependency. However, the Data Processing Inequality (DPI) [26] indicates that feature level contains more information than that in classifier level, so feature level dependence modeling should be performed directly. The key problem is that the dimension of feature is normally high and it is very hard to estimate the joint distribution accurately in feature level. Multiple Kernel Learning (MKL) [102] can be considered as a feature level fusion method and finds the optimal combination of kernels of different features. To solve the complexity problem, fast methods [103] [104] are proposed. Recently, variants of MKL have been developed and used for object recognition [62] [105] [106]. MKL can be considered as implicit dependency modeling technique based on the feature kernels, but dependency information may lose when original features are converted into kernel matrices.

In this chapter, a novel framework for dependency modeling is developed, and two methods, namely Linear Classifier Dependency Modeling (LCDM) and Linear Feature Dependency Modeling (LFDM) are proposed for classifier level and feature level fusion, respectively. Inspired by the Bayesian model with independent assumption, we prove that linear combination of the posterior probabilities of each classifier can model dependency under mild assumptions. Under the framework of linear dependency modeling, it can be shown that more information about class label is available in feature level, so we generalize LCDM to feature level and propose LFDM. Finally, the optimal models for LCDM and LFDM are obtained by solving

a standard linear programming problem, which maximizes the margins between the genuine and imposter posterior probabilities.

The rest of this chapter is organized as follows. Section 4.2 will report the proposed method. Experimental results and a brief summary of this chapter are given in Sections 4.3 and Sections 4.4, respectively.

4.2 Linear Dependency Modeling

With conditionally independent assumption, the posterior probability can be computed as in (2.1.1) or (2.1.3). However, as discussed in Section 4.1, this assumption may not be true in many practical applications. Thus, in this chapter, we remove the independent assumption and study the dependent case. In this section, we propose a new linear dependency modeling method to model dependency. We first derive the model in classifier level and then proceed to feature level. Finally, we present a learning method to learn the optimal linear dependency model under marginal criterion.

4.2.1 Linear Dependency Modeling in Classifier Level

Let us consider the case that there exists m', s.t. $\Pr(\omega_l|\vec{x}_{m'}) = 0$ and $\Pr(\omega_l|\vec{x}_m) = 1$ for $1 \leq m \leq M$, $m \neq m'$. In this case, there are $M-1$ classifiers giving strong confidences on assigning the class label as ω_l, and one classifier concludes other label. When $M \gg 1$, we can ignore the m' classifier giving $\Pr(\omega_l|\vec{x}_{m'}) = 0$, and the posterior probability $\Pr(\omega_l|\vec{x}_1, \cdots, \vec{x}_M)$ is expected to be large. However, with conditionally independent assumption, from (2.1.1), the posterior probability by all classifiers becomes zeros. This means, in this case, the posterior probability only depends on one single classifier which satisfies $\Pr(\omega_l|\vec{x}_{m'}) = 0$ and the other classifiers will not have any contribution on it.

In order to avoid the posterior probability to be zero and dominated only by one classifier, we add a term to $\Pr(\omega_l|\vec{x}_{m'})$. This action also affect the posterior probability of some classifiers. So, for all classifiers, we add terms to model the dependency between them. Moreover, the classifier scores with dependency terms cannot deviate much from the original ones. Otherwise the original decision by each single classifier will take little effect on the final decision. Therefore, the values of the dependency terms must be small. Under the assumption that posterior probability of each classifier will not deviate dramatically from the prior as in Sum rule [2], δ_{lm}, $m = 1, \cdots, M$ given by (2.1.2) are small values. In this context, we define the small dependency term for feature m and class ω_l as the multiplication of dependency weight α_{lm}, prior probability $\Pr(\omega_l)$ and small value δ_{lm}. In order to ensure that the dependency term $\alpha_{lm}\Pr(\omega_l)\delta_{lm}$ is still not too large and bounded by the fixed values of prior probability $\Pr(\omega_l)$ and small number δ_{lm}, we restrict $|\alpha_{lm}| \leq 1$, so that $|\alpha_{lm}\Pr(\omega_l)\delta_{lm}| \leq |\Pr(\omega_l)\delta_{lm}|$.

Adding dependency term $\alpha_{lm}\Pr(\omega_l)\delta_{lm}$ to each $\Pr(\omega_l|\vec{x}_m)$, the posterior probability becomes

$$\Pr(\omega_l|\vec{x}_1, \cdots, \vec{x}_M) = \frac{P_0}{\Pr(\omega_l)^{M-1}} \prod_{m=1}^{M} (\Pr(\omega_l|\vec{x}_m) + \alpha_{lm}\Pr(\omega_l)\delta_{lm}) \tag{4.2.1}$$

In the above equation (4.2.1), α_{lm}, $m = 1, \cdots, M$ are the weights to model the dependency between classifiers. If all the classifiers are independent with each other, $\alpha_{lm} = 0$ for $m = 1, \cdots, M$ and the posterior probability with dependency terms given by (4.2.1) becomes the Product model as in (2.1.1) with the independent assumption. On the other hand, when classifiers are dependent, $\alpha_{lm} \neq 0$ and the posterior formulation (4.2.1) models dependency.

The product formulation (4.2.1) contains high order terms of the dependency weights, which make it difficult to determine the optimal model. Thus, we further expand the product formulation as follows. With the definition of the small number

δ_{lm} in (2.1.2), the dependency model (4.2.1) becomes

$$\Pr(\omega_l|\vec{x}_1,\cdots,\vec{x}_M) = P_0 * \Pr(\omega_l)\prod_{m=1}^{M}(1+\beta_{lm}\delta_{lm}) \tag{4.2.2}$$

where $\beta_{lm} = \alpha_{lm} + 1$, $0 \leq \beta_{lm} \leq 2$. Since the terms $\delta_{l1},\cdots,\delta_{lM}$ defined by (2.1.2) are assumed to be small, the second and higher order terms of them can be neglected. If we expand the product term in the right hand side of (4.2.2), the posterior probability becomes

$$\Pr(\omega_l|\vec{x}_1,\cdots,\vec{x}_M) = P_0 * \Pr(\omega_l)(1+\sum_{m=1}^{M}\beta_{lm}\delta_{lm}) \tag{4.2.3}$$

Substituting δ_{lm} by (2.1.2) into (4.2.3), we obtain

$$\Pr(\omega_l|\vec{x}_1,\cdots,\vec{x}_M) = P_0 * (\sum_{m=1}^{M}\beta_{lm}[\Pr(\omega_l|\vec{x}_m) - \Pr(\omega_l)] + \Pr(\omega_l)) \tag{4.2.4}$$

This gives the linear dependency model easier to be determined.

After that, the summation of weights $\sum_{m=1}^{M}\beta_{lm}$ should be normalized with respect to class label l. So $\sum_{m=1}^{M}\beta_{lm}$ is a constant Q, i.e. $\sum_{m=1}^{M}\beta_{lm} = Q$. Constant Q is set to be M due to the following two reasons.

First, the general dependency model given by (4.2.4) includes the independent case. When the independent assumption holds, $\beta_{lm} = 1$ for all m, which results in $Q = \sum_{m=1}^{M}\beta_{lm} = M$.

Second, constraint $\sum_{m=1}^{M}\beta_{lm} = M$ gives a balance between under-learning and over-learning, which is elaborated as follows. With constraint $0 \leq \beta_{lm} \leq 2$ for the dependency model, Q can be chosen from 0 to $2M$. When $Q = 0$, all dependency weights β_{lm} become zero. In this case, the classifier scores $\Pr(\omega_l|\vec{x}_m)$ do not have any contribution to the posterior probability $\Pr(\omega_l|\vec{x}_1,\cdots,\vec{x}_M)$. And the classification only depends on the prior probability $\Pr(\omega_l)$, which is not reasonable. Thus, the range for constant Q should be $0 < Q \leq 2M$. When Q is too large, e.g. $Q = 2M$, the coefficients β_{lm} are all equal to two and need not to be determined by the training data. Thus, the model may suffer from under-learning problem. On the other hand, if Q is too small, e.g. $0 < Q \leq 2$, there may have only one non-zero coefficient with

value Q. Under this situation, if there is a strong classifier which is "perfect" for the training data, all the coefficients are zero except the one for the "perfect" classifier. While multiple classifiers/features provide complementary information for the class label and the "perfect" classifier may not be suitable for the testing data, the model suffers from over-learning problem. Thus, Q cannot be too large nor too small. And the constraint $\sum_{m=1}^{M} \beta_{lm} = M$ with a moderate constant M helps to reduce the problem of under-learning and over-learning.

Finally, the Linear Classifier Dependency Model (LCDM) is summarized as equation (4.2.4) with constraints $0 \leq \beta_{lm} \leq 2$ and $\sum_{m=1}^{M} \beta_{lm} = M$. When classifiers are independent with each other, $\beta_{lm} = 1$ for $m = 1, \cdots, M$ and LCDM becomes the Sum rule as in (2.1.3). On the other hand, when classifiers are dependent, $\beta_{lm} \neq 1$ and the LCDM method models dependency.

4.2.2 Linear Dependency Modeling in Feature Level

Given the feature vectors $\vec{x}_1, \cdots, \vec{x}_M$, where $\vec{x}_m = (x_{m1}, \cdots, x_{mN_m})$ and N_m is the dimension of feature m. $\Pr(\omega_l|\vec{x}_m)$ can be computed by estimating the joint distribution $\Pr(x_{m1}, \cdots, x_{mN_m}|\omega_l)$. Since $\vec{x}_1, \cdots, \vec{x}_M$ are normally high dimensional vectors, it needs numerous samples to estimate the joint density accurately [27]. In general, the probability is hard to be determined accurately.

In classifier combination methods [2] [13], the posterior probabilities $\Pr(\omega_l|\vec{x}_m)$ are viewed as scores calculated from certain classifiers, such as support vector machines (SVM). In this scenario, classifiers are trained individually for a single feature under certain criteria. However, there is no guarantee that they are optimal after fusion.

In the scenario that feature vectors are given, $\Pr(\omega_l|x_{mn})$ can be computed by one-dimensional probability estimation. Let us denote $\mathcal{S} = (s_{11}, \cdots, s_{LM})$ and $\mathcal{F} = (f_{111}, \cdots, f_{LMN_M})$, where $s_{lm} = \Pr(\omega_l|\vec{x}_m)$ and $f_{lmn} = \Pr(\omega_l|x_{mn})$. In the

classification problems, information from $\mathcal{S}$ in classifier level or $\mathcal{F}$ in feature level is used to infer the class label ω_l. From the aspect of information quantity of $\mathcal{S}$ and $\mathcal{F}$ about the class labels, we have the following proposition. (Please refer to the appendices for the proof of this proposition.)

Proposition 1.

Under the framework of linear dependency modeling by (4.2.4)*, $I(\mathcal{S}, \mathcal{Y}) \leq I(\mathcal{F}, \mathcal{Y})$, where $\mathcal{S}$ and $\mathcal{F}$ are the random vectors of the classification scores and features respectively, $\mathcal{Y}$ is the label viewed as a random variable, and $I(\cdot, \cdot)$ represents the mutual information.*

The above proposition indicates the relationship between classifier level and feature level fusion from the information quantity aspect. And this proposition implies that feature level contains more information about the class label than that in classifier level.

Based on the above analysis, we propose to model the dependency in feature level. Let us consider the posterior probability by all features $\Pr(\omega_l|\vec{x}_1, \cdots, \vec{x}_M)$ again. Given the feature vectors $\vec{x}_1, \cdots, \vec{x}_M$, it can be rewritten as

$$\Pr(\omega_l|\vec{x}_1, \cdots, \vec{x}_M) = \Pr(\omega_l|x_{11}, \cdots, x_{1N_1}, \cdots, x_{M1}, \cdots, x_{MN_M}) \tag{4.2.5}$$

With the representation (4.2.5), each element in the feature vector can be viewed as a single classifier. Similar to the analysis in Section 4.2.1, the dependency term for each x_{mn} can be written as $\alpha_{lmn}\Pr(\omega_l)\delta_{lmn}$, where α_{lmn} satisfies $|\alpha_{lmn}| \leq 1$ and each δ_{lmn} is a small number, s.t.

$$\Pr(\omega_l|x_{mn}) = \Pr(\omega_l)(1 + \delta_{lmn}) \tag{4.2.6}$$

Then, the posterior probability analogous to (4.2.1) is given by

$$\begin{aligned} &\Pr(\omega_l|x_{11}, \cdots, x_{1N_1}, \cdots, x_{M1}, \cdots, x_{MN_M}) \\ &= \frac{P_0'}{\Pr(\omega_l)^{M-1}} \prod_{m=1}^{M} \prod_{n=1}^{N_m} (\Pr(\omega_l|x_{mn}) + \alpha_{lmn}\Pr(\omega_l)\delta_{lmn}) \end{aligned} \tag{4.2.7}$$

where $P_0' = \frac{\prod_{m=1}^{M}\prod_{n=1}^{N_m}\Pr(x_{mn})}{\Pr(\vec{x}_1,\cdots,\vec{x}_M)}$. With (4.2.6), the above equation can be rewritten as

$$\begin{aligned}&\Pr(\omega_l|x_{11},\cdots,x_{1N_1},\cdots,x_{M1},\cdots,x_{MN_M})\\&= P_0' * \Pr(\omega_l)\prod_{m=1}^{M}\prod_{n=1}^{N_m}(1+\gamma_{lmn}\delta_{lmn})\end{aligned} \tag{4.2.8}$$

where $\gamma_{lmn} = \alpha_{lmn} + 1$. As δ_{lmn} defined in (4.2.6) measures the difference between posterior $\Pr(\omega_l|x_{mn})$ and prior $\Pr(\omega_l)$, δ_{lmn} is assumed to be small. By expanding the product in (4.2.8) and neglecting the terms of second and higher orders, it has

$$\begin{aligned}&\Pr(\omega_l|x_{11},\cdots,x_{1N_1},\cdots,x_{M1},\cdots,x_{MN_M})\\&= P_0' * \Pr(\omega_l)(1+\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{lmn}\delta_{lmn})\end{aligned} \tag{4.2.9}$$

Substituting δ_{lmn} by (4.2.6) into (4.2.9), the posterior probability with dependency modeling in feature level is given as

$$\begin{aligned}&\Pr(\omega_l|x_{11},\cdots,x_{1N_1},\cdots,x_{M1},\cdots,x_{MN_M})\\&= P_0' * (\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{lmn}[\Pr(\omega_l|x_{mn}) - \Pr(\omega_l)] + \Pr(\omega_l))\end{aligned} \tag{4.2.10}$$

At last, applying the analysis for the LCDM constraints in Section 4.2.1 to feature level, we have two constraints, $\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{lmn} = \sum_{m=1}^{M} N_m$ and $0 \le \gamma_{lmn} \le 2$, for the proposed Linear Feature Dependency Model (LFDM).

4.2.3 Learning Optimal Linear Dependency Model

The optimal LCDM and LFDM can be learned by different criteria to optimize the classification performance. In this section, we consider the marginal criterion, and give detailed derivations on how to learn the optimal LFDM. Similar derivations can be applied to LCDM.

Let the training samples and corresponding labels be $\vec{x}_j = (x_{j11},\cdots,x_{jMN_M})$ and y_j, $j = 1,\cdots,J$, respectively. If all the training samples are correctly classified, $\Pr(y_j|\vec{x}_j) > \max_{\omega_l \ne y_j}\Pr(\omega_l|\vec{x}_j)$. Under this condition, the difference between the

genuine and imposter probabilities $\Pr(y_j|\vec{x}_j) - \max_{\omega_l \neq y_j} \Pr(\omega_l|\vec{x}_j)$ is large. In turn, the performance will be good. Considering the marginal criterion, and inspired by LPBoost [4] and its multiclass generalization [14], the objective function for learning the best model is defined as

$$\begin{aligned} &\min_{\gamma,\rho,\xi} -\rho + \frac{1}{\nu J}\sum_{j=1}^{J}\sum_{\omega_l \neq y_j} \xi_{jl} \\ s.t.\ \ &i)\ \Pr(y_j|\vec{x}_j) - \Pr(\omega_l|\vec{x}_j) \geq \rho - \xi_{jl}, \forall j, \omega_l \neq y_j \\ &ii)\ \Pr(\omega_l|\vec{x}_j) \geq 0, \forall j, \omega_l \\ &iii)\ \xi_{jl} \geq 0, \forall j, \omega_l \neq y_j \end{aligned} \tag{4.2.11}$$

where ν is a positive parameter, s.t. $\nu \in (0, 1)$. Let us denote $\Pr(\omega_l|x_{jmn}) - \Pr(\omega_l)$ in (4.2.10) as p_{jlmn} and $K = \sum_{m=1}^{M} N_m$. Suppose the prior probabilities are the same, i.e. $\Pr(\omega_l) = \frac{1}{L}, \forall \omega_l$, where L is the number of classes. Adding the normalization and range constraints to γ_{lmn} as mentioned in Section 4.2.2, substituting (4.2.10) into (4.2.11), and ignoring P_0' as a constant respective to class label, the optimization problem (4.2.11) becomes

$$\begin{aligned} &\min_{\gamma,\rho,\xi} -\rho + \frac{1}{\nu J}\sum_{j=1}^{J}\sum_{\omega_l \neq y_j} \xi_{jl} \\ s.t.\ &i)\ \sum_{m=1}^{M}\sum_{n=1}^{N_m} \gamma_{y_j mn} p_{jy_j mn} - \sum_{m=1}^{M}\sum_{n=1}^{N_m} \gamma_{lmn} p_{jlmn} \geq \rho - \xi_{jl}, \forall j, \omega_l \neq y_j \\ &ii)\ \sum_{m=1}^{M}\sum_{n=1}^{N_m} \gamma_{lmn} p_{jlmn} + \frac{1}{L} \geq 0, \forall j, \omega_l \\ &iii)\ \xi_{jl} \geq 0, \forall j, \omega_l \neq y_j \\ &iv)\ \sum_{m=1}^{M}\sum_{n=1}^{N_m} \gamma_{lmn} = K, \forall l \\ &v)\ 0 \leq \gamma_{lmn} \leq 2, \forall l, m, n \end{aligned} \tag{4.2.12}$$

Optimization problem (4.2.12) is theoretically sound and purely derived from (4.2.11) mathematically. However, according to the following proposition, optimization problem (4.2.12) may not have feasible solution in practice. (Please refer to the appendices for the proof of this proposition.)

Proposition 2.

If there exist j_0 *and* l_0*, s.t.* $Pr(\omega_{l_0}|x_{j_0mn}) < Pr(\omega_{l_0})(1-\frac{1}{K}), \forall m,n$*, then constraints* $ii)$ *and* $iv)$ *in* (4.2.12) *cannot be hold at the same time.*

When the fusion is performed in feature level, K will be a very large number, i.e. $\frac{1}{K}$ is very small. In this situation, The condition in the above proposition is likely to be satisfied. Thus, optimization problem (4.2.12) will not have feasible solution, and one of the two constraints should be removed.

Constraint $ii)$ in (4.2.11) is to ensure that the posterior probability is non-negative. From (4.2.11) to (4.2.12), the posterior probability is substituted with (4.2.10). Let us check the derivation from (4.2.8) to (4.2.10). We can see that equation (4.2.10) is the first order approximation to the posterior probability. Since the summation of the neglected terms of second and higher orders may be positive or negative, constraint $ii)$ in (4.2.12) cannot guarantee that the posterior probability is non-negative. On the other hand, since γ_{lmn} is less than or equal to 2 and δ_{lmn} is a small number, e.g. $|\delta_{lmn}| \leq \frac{1}{2}$, the posterior probability in product formulation is always non-negative according to (4.2.8). Thus, we remove the constraint $ii)$ in (4.2.12). Consequently, the final optimization problem becomes

$$\begin{aligned}
&\min_{\gamma,\rho,\xi} -\rho + \frac{1}{\nu J}\sum_{j=1}^{J}\sum_{\omega_l \neq y_j}\xi_{jl} \\
s.t.\ i)&\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{y_jmn}p_{jy_jmn} - \sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{lmn}p_{jlmn} \geq \rho - \xi_{jl}, \forall j, \omega_l \neq y_j \\
ii)&\ \xi_{jl} \geq 0, \forall j, \omega_l \neq y_j \\
iii)&\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{lmn} = K, \forall l \\
iv)&\ 0 \leq \gamma_{lmn} \leq 2, \forall l,m,n
\end{aligned} \tag{4.2.13}$$

The optimal LFDM is learned by (4.2.13). Similarly, the optimal LCDM can be obtained by applying (4.2.4) into (4.2.11) and following the same derivation procedures from (4.2.11) to (4.2.13). The optimization problem for LCDM is given

as

$$\begin{aligned}
&\min_{\beta,\rho,\xi} -\rho + \frac{1}{\nu J}\sum_{j=1}^{J}\sum_{\omega_l \neq y_j}\xi_{jl} \\
s.t.\ i)\ &\sum_{m=1}^{M}\beta_{y_j m}h_{jy_j m} - \sum_{m=1}^{M}\beta_{lm}h_{jlm} \geq \rho - \xi_{jl}, \forall j, \omega_l \neq y_j \\
ii)\ &\xi_{jl} \geq 0, \forall j, \omega_l \neq y_j \\
iii)\ &\sum_{m=1}^{M}\beta_{lm} = M, \forall l \\
iv)\ &0 \leq \beta_{lm} \leq 2, \forall l, m, n
\end{aligned} \tag{4.2.14}$$

where $h_{jlm} = \Pr(\omega_l|\vec{x}_{jm}) - \Pr(\omega_l)$. The optimization problems (4.2.14) and (4.2.13) for LCDM and LFDM are standard linear programming problems. Thus, the solutions can be determined by any off the shelf techniques, e.g. [40]. Since our experiments are performed in the Matlab environment, a Matlab build-in function with the interior point method is employed.

4.2.4 Sensitivity to Density Estimation Error

In order to compare the proposed LCDM and LFDM for dependency modeling in classifier level and feature level respectively, we study the error sensitivity properties of these two methods.

In the LCDM model, we implicitly assume that the posterior probability $\Pr(\omega_l|\vec{x}_m)$ is known or computed correctly. However, this may not be the case, because the number of samples is limited in training stage. Let us denote the estimated probability as $\widehat{\Pr}(\omega_l|\vec{x}_m)$. The density estimation error for the true density is given by

$$e_{lm} = \widehat{\Pr}(\omega_l|\vec{x}_m) - \Pr(\omega_l|\vec{x}_m) \tag{4.2.15}$$

In practice, the estimated probabilities are used instead of the true ones. Consequently, according to (4.2.4), the LCDM model is changed to

$$\widehat{\Pr}(\omega_l|\vec{x}_1, \cdots, \vec{x}_M) = P_0 * [\Pr(\omega_l)(1-M) + \sum_{m=1}^{M}\beta_{lm}\widehat{\Pr}(\omega_l|\vec{x}_m)] \tag{4.2.16}$$

Substituting $\widehat{\Pr}(\omega_l|\vec{x}_m)$ with (4.2.15) into (4.2.16), it has

$$\widehat{\Pr}(\omega_l|\vec{x}_1,\cdots,\vec{x}_M) = P_0 * [\Pr(\omega_l)(1-M) + (1+E_c)\sum_{m=1}^{M}\beta_{lm}\Pr(\omega_l|\vec{x}_m)] \quad (4.2.17)$$

where E_c is the error factor in LCDM and given by the following equation

$$E_c = \frac{\sum_{m=1}^{M}\beta_{lm}e_{lm}}{\sum_{m=1}^{M}\beta_{lm}\Pr(\omega_l|\vec{x}_m)} \quad (4.2.18)$$

Similarly, in the LFDM model, the error factor is given by

$$E_f = \frac{\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{lmn}\epsilon_{lmn}}{\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{lmn}\Pr(\omega_l|x_{mn})} \quad (4.2.19)$$

where $\epsilon_{lmn} = \widehat{\Pr}(\omega_l|x_{mn}) - \Pr(\omega_l|x_{mn})$.

Since it is hard to compare the error factors E_c in LCDM and E_f in LFDM directly, we investigate the upper bounds instead. Denote U_c and U_f as the upper bounds of E_c and E_f respectively. We have the following equations

$$E_c \leq \frac{\sum_{m=1}^{M}\beta_{lm}|e_{lm}|}{\sum_{m=1}^{M}\beta_m^l\Pr(\omega_l|\vec{x}_m)} = U_c \quad (4.2.20)$$

$$E_f \leq \frac{\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{lmn}|\epsilon_{lmn}|}{\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{lmn}\Pr(\omega_l|x_{mn})} = U_f \quad (4.2.21)$$

By analyzing the denominators and numerators of U_c in (4.2.20) and U_f in (4.2.21) respectively, it can be shown that U_f is smaller than U_c, which is elaborated as follows.

We first consider the denominators of U_c in (4.2.20) and U_f in (4.2.21). According to (2.1.2) and the normalization constraint, the denominator in (4.2.20) can be approximated by

$$\sum_{m=1}^{M}\beta_{lm}\Pr(\omega_l|\vec{x}_m) = \Pr(\omega_l)\sum_{m=1}^{M}\beta_{lm}(1+\delta_{lm}) \approx \Pr(\omega_l) * M \quad (4.2.22)$$

Similarly, the denominator in (4.2.21) can be approximated by

$$\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{lmn}\Pr(\omega_l|x_{mn}) \approx \Pr(\omega_l) * \sum_{m=1}^{M}N_m \quad (4.2.23)$$

Since $1 \ll N_m$, we get

$$\sum_{m=1}^{M} \beta_{lm} \Pr(\omega_l|\vec{x}_m) \ll \sum_{m=1}^{M} \sum_{n=1}^{N_m} \gamma_{lmn} \Pr(\omega_l|x_{mn}) \tag{4.2.24}$$

This equation shows that the denominator in U_c is much smaller than that in U_f. On the other hand, we compare the numerators of U_c in (4.2.20) and U_f in (4.2.21). It is reported in [27] that density estimation in high dimensional space is much difficult than that in one-dimensional space. And, the required sample size increases dramatically with dimension to achieve the given estimation accuracy. That means the absolute value of the estimation error in classifier level is much larger than that in feature level, i.e. $|e_{l'm'}| \gg |\epsilon_{lmn}|$. If $|e_{l'n'}| > 2N_m|\epsilon_{lmn}|$, according to the range and normalization constraint, the numerator in U_c is larger than that in U_f, i.e.

$$\sum_{m=1}^{M} \beta_{lm}|e_{lm}| > \sum_{m=1}^{M} \sum_{n=1}^{N_m} \gamma_{lmn}|\epsilon_{lmn}| \tag{4.2.25}$$

With (4.2.24) and (4.2.25), we have $U_f \ll U_c$. This means the upper bound of the error factor in feature level is much smaller than that in classifier level. Thus, LFDM is better than LCDM in the worst case.

4.2.5 Remarks

In this section, we summarize our proposed methods and indicate their advantages over exiting methods for dependency modeling.

- LCDM is an "optimal" classifier-level fusion method in the sense of classifier dependency modeling given in Section 4.2.1. The training procedure of LCDM is described in Fig. 4.1.
- LFDM is an "optimal" feature-level combination method in the sense of feature dependency modeling given in Section 4.2.2. The algorithmic procedure of LFDM is shown in Fig. 4.2.
- Considering h_{jlm} in (4.2.14) and p_{jlmn} in (4.2.13) as classification scores, the derived formulations (4.2.14) and (4.2.13) look like the objective function of

Algorithm 1. LCDM

Input: Training samples $\mathcal{O}_1, \cdots, \mathcal{O}_J$;
Corresponding labels $y_1, \cdots, y_J$;
Selected descriptors $\mathcal{D}_1, \cdots, \mathcal{D}_M$;

Output: Trained classifiers $\phi_{11}, \cdots, \phi_{LM}$;
Classifier dependency weight $\vec{\beta}$;

for $m = 1, \cdots, M$
  for $j = 1, \cdots, J$
    Get feature vector $\vec{x}_{jm} = \mathcal{D}_m(\mathcal{O}_j)$;
  endfor;
endfor;
Train classifiers $\phi_{11}, \cdots, \phi_{LM}$ for the *M* descriptors, giving confidence on the *L* classes;
for $l = 1, \cdots, L$
  for $m = 1, \cdots, M$
    for $j = 1, \cdots, J$
      Compute hypothesis $h_{jlm} = \phi_{lm}(\vec{x}_{jm}) - \frac{1}{L}$;
    endfor;
  endfor;
endfor;
Solve the optimization problem (4.2.14) to obtain $\vec{\beta}$;
return $\vec{\beta}$ and $\phi_{11}, \cdots, \phi_{LM}$.

Figure 4.1: Algorithm 1: The training procedure of LCDM

Algorithm 2. LFDM

Input: Training samples $\mathcal{O}_1, \cdots, \mathcal{O}_J$;
Corresponding labels $y_1, \cdots, y_J$;
Selected descriptors $\mathcal{D}_1, \cdots, \mathcal{D}_M$;

Output: Estimated distributions $\mathcal{P}_{111}, \cdots, \mathcal{P}_{LMN}$;
Feature dependency weight vector $\vec{\gamma}$;

for $m = 1, \cdots, M$
  for $j = 1, \cdots, J$
    Get feature vector $\vec{x}_{jm} = \mathcal{D}_m(\mathcal{O}_j)$;
  endfor;
endfor;
Estimate the posterior distributions $\mathcal{P}_{111}, \cdots, \mathcal{P}_{LMN}$;
for $l = 1, \cdots, L$
  for $j = 1, \cdots, J$
    for $m = 1, \cdots, M$
      for $n = 1, \cdots, N_m$
        Compute $p_{jlmn} = \mathcal{P}_{lmn}(x_{jmn}) - \frac{1}{L}$;
      endfor;
    endfor;
  endfor;
endfor;
Solve the optimization problem (4.2.13) to obtain $\vec{\gamma}$;
return $\vec{\gamma}$ and $\mathcal{P}_{111}, \cdots, \mathcal{P}_{LMN}$.

Figure 4.2: Algorithm 2: The training procedure of LFDM

LP-B in [14]. Though the goal and the starting points of the LCDM and LFDM are different from boosting methods, we come up with a similar optimization problem with extra constraints $0 \leq \beta_{lm} \leq 2$ in LCDM and $0 \leq \gamma_{lmn} \leq 2$ in LFDM. Boosting methods aim at finding the correct weighting for classifier combination, while our objective is to model the feature dependency. Without the range constraint $0 \leq \beta_{lm} \leq 2$ or $0 \leq \gamma_{lmn} \leq 2$, the dependency weight β_{lm} or γ_{lmn} can be a large value, so that the derivation from (4.2.2) to (4.2.3), or from (4.2.8) to (4.2.10) is invalid. On the other hand, this causes the posterior probability in product form given as (4.2.2) or (4.2.8) be negative. Thus, the constraint $0 \leq \beta_{lm} \leq 2$ or $0 \leq \gamma_{lmn} \leq 2$ is very important for dependency model. This can be observed from the experimental results.

- Compared with the dependency modeling technique [5] by estimating the joint density, LCDM or LFDM does not need to estimate the joint distribution of the scores or features. Consequently, our methods do not suffer from the difficulty in joint distribution estimation with high dimensionality problem.

- Compared with the combination rule under dependent normal (DN) assumption [13], LCDM and LFDM are derived without the distribution assumption. In many practical applications, data may not follow normal distribution, so our methods could have better performance in the non-normal cases.

- Compared with the linear combination methods, LPBoost [14] [4] and MK-L [102] [103] [104], our methods are derived from a probabilistic framework and can model dependency explicitly.

- Compared with feature level fusion method, MKL [102] [103] [104], LFDM models the dependency with the feature vectors directly and does not rely on the kernel matrices. In the situation that some elements in the feature vectors are noisy, the constructed kernels will be noisy as well. Since the combination by MKL is based on the noisy kernels, the performance may degrade. However, the proposed LFDM will not suffer from this problem.

4.3 Experiments

In this section, we evaluate the proposed Linear Classifier Dependency Modeling (LCDM) and Linear Feature Dependency Modeling (LFDM) methods, with synthetic data as well as four real image/video databases. The real image/video databases are Oxford 17 Flower [43], Digit [41], Weizmann [49] and KTH [55]. First, we compare the proposed LCDM with state-of-art classifier combination methods with synthetic data and the Flower database in Sections 4.3.1 and 4.3.2 respectively. Then, we evaluate the LCDM with different types of classifiers and LFDM with different probability estimation methods on the Digit database in Section 4.3.3. After that, we compare the best performances achieved by the LCDM and LFDM on Weizmann and KTH action databases in Section 4.3.4. Finally, analysis on the dependencies between classifiers or features and their relationship with the learning results is presented in Section 4.3.5.

4.3.1 Results on Synthetic Data

Since it is impossible to know the intrinsic distributions in practical applications, we use synthetic data to simulate the classifier scores, similar to [13] [107]. In this experiment, four types of classifier distributions, namely Independent Normal (IndNormal), Dependent Normal (DepNormal), Independent Non-Normal (IndNonNor), Dependent Normal (DepNonNor) are used to evaluate the LCDM and other classifier combination methods. For each distribution, 2000 samples (500 positive and 500 negative for training and testing respectively) of 40 dimensions, which simulate 40 classifiers, are randomly generated by Matlab built-in function. In order to avoid the possibility that the results could be influenced by the random number generation, we run the experiments 200 times. For multi-class methods, LP-B and the proposed LCDM, we estimate the genuine and imposter posterior probabilities by each classifier as in [108], i.e. $\Pr(+|s_m)$ and $\Pr(-|s_m)$, where s_m represents

Test \ Method	IndNormal	DepNormal	IndNonNor	DepNonNor
Sum [2]	95.34±0.71	86.44±1.16	74.67±1.50	63.33±0.89
LPBoost [4]	96.81±0.48	95.29±0.85	90.10±1.61	68.95±1.30
LP-B [14]	97.51±0.48	93.17±1.30	89.00±0.08	69.37±1.90
IN [13]	**97.67±0.40**	92.52±0.98	84.80±1.65	65.46±0.92
DN [13]	97.56±0.42	**95.64±0.89**	91.41±1.38	69.84±1.35
LCDM	97.66±0.46	93.88±0.93	**93.00±0.07**	**72.14±1.52**

Table 4.1: Mean accuracy (%) and standard deviation on synthetic data

the score. The parameter ν[1] in the proposed method and LP-B is selected from $\{0.05, 0.1, \ldots, 0.95\}$.

Table 4.1 shows the mean recognition rates and standard deviations of the six combination methods with different data distributions. From Table 4.1, we can observe that the proposed LCDM outperforms Sum rule and LP-B under all data distributions. This is because Sum rule is derived under independent assumption without training and LP-B does not model the dependency explicitly. On the other hand, IN and DN achieve the best performance with independent and dependent normal distribution respectively, while the performance of LCDM is comparable with IN and DN. This is reasonable because the IN and DN combination rules are derived under normal assumption. For the non-normal distributions, the proposed LCDM outperforms the other methods.

4.3.2 Results on Oxford 17 Flower Database

Results with synthetic data show that the proposed dependency modeling method, LCDM is effective. In this section, the classifier combination methods are evaluated

[1]This parameter is selected from the same set of values in the following databases except Weizmann and KTH.

Single feature		Classifier Combination	
feature	accuracy	method	accuracy
Color	61.14±1.81	Sum [2]	85.39±3.14
Shape	70.16±1.28	IN [13]	85.49±1.72
Texture	62.90±2.44	DN [13]	84.22±1.91
HSV	61.44±0.47	LPBoost [4]	82.65±0.82
HoG	58.94±4.14	LP-β [14]	85.50±3.00
SIFTint	70.40±1.43	LP-B [14]	85.52±2.37
SIFTbdy	59.37±3.40	LCDM	**86.27±2.43**

Table 4.2: Mean accuracy (%) and standard deviation on Oxford Flower database

on a real database for the task of flower classification. Table 4.2 shows the mean accuracy and standard deviation for each feature in the left column and for combination methods in the right column. From Table 4.2, we can see that the proposed LCDM outperforms all existing methods. Moreover, the recognition accuracies of the combination methods are much higher than that of a single feature. For example, the Sum rule can have the accuracy of 85.39%. This implies the classifiers are relatively independent to each other, which is due to the reason that the features are extracted by different properties of the instances, e.g. shape and color. While the features are not very dependent in this case, it is impossible to obtain totally independent features. Therefore, the proposed LCDM can further improve the performance by modeling the dependency between the classification scores.

4.3.3 Results on Digit Database

On the Flower database, LCDM gets the highest accuracy with the best kernel SVM classifier for each feature. In this section, we evaluate the classifier level combination methods with different classifiers and feature level fusion with different probability estimation methods on the Digit database.

	1-NN	3-NN	SVM C=0.1	SVM C=100
Sum [2]	93.76±0.72	**94.76±0.70**	94.82±0.60	96.23±0.66
IN [13]	95.06±0.71	93.40±0.84	94.75±0.61	95.63±1.08
DN [13]	94.77±0.55	93.50±0.89	94.82±0.59	94.93±1.09
LPBoost [4]	94.88±0.64	94.18±0.71	89.71±2.31	96.41±0.41
LP-B [14]	94.13±1.09	93.53±0.81	94.95±0.55	96.57±0.35
LCDM	**95.18±0.56**	94.46±0.71	**95.34±0.66**	**96.79±0.60**

Table 4.3: Mean accuracy (%) and standard deviation of classifier level fusion methods on Multiple Feature Digit database

Since the covariance matrices in the Gaussian based classifier are degenerate for most features, k nearest neighbor (k-NN) classifiers for $k = 1, 3$ and SVM classifiers for $C = 0.1, 100$ are used to evaluate the classifier combination methods. The results which are summarized in Table 4.3 show that LCDM gets highest recognition rates with all classifiers except 3-NN, and the performance of LCDM is very close to that of Sum combination in 3-NN. This ensures that the LCDM can be used with different classifiers and improve the recognition performance in most cases.

In the proposed LFDM for feature dependency modeling, the one-dimensional probability $\Pr(\omega_l|x_{jmn})$ needs to be estimated. So we evaluate LFDM with different probability estimation methods, including kernel density estimation (KDE) via diffusion [108] and kernel smoothing density estimation (KSD) [109]. We also choose different kernels to estimate the probability with KSD, but the results are the same. On the other hand, for comparison, the classifier combination algorithms are extended to feature level fusion and denoted as Sum-F, IN-F, DN-F, LPBoost-F, and LP-B-F. The results are recorded on the right hand side in Table 4.4. It can be seen that the proposed LFDM outperforms others using different probability estimation methods. Another observation is that DN-F in feature level gets an obvious improvement compared to the other methods without explicit dependency modeling, while

	KDE [108]	KSD [109]
Sum-F [2]	92.91±0.80	93.63±0.59
IN-F [13]	93.47±0.82	94.37±0.86
DN-F [13]	97.00±0.62	96.87±0.89
LPBoost-F [4]	95.79±1.03	95.34±1.28
LP-B-F [14]	94.36±1.13	95.44±1.15
LFDM	**97.05±0.60**	**97.09±0.62**

Table 4.4: Mean accuracy (%) and standard deviation of feature level fusion methods on Multiple Feature Digit database

it is not the case with the classifier combination methods as shown in Table 4.3. This implies that the dependency information is more useful and important for the fusion process in feature level.

4.3.4 Results on Human Action Databases

In this section, we give a detailed evaluation of the proposed methods on the two standard human action databases, Weizmann and KTH. It is important to point out that the main objective of this experiment is to evaluate the performance of the classifier level and feature level combination methods given a set of action features, but not state-of-art human action recognition algorithms. In Weizmann database, data from eight persons of each validation are used to train classifiers and the best combinations, as well as estimate the posterior probability. In KTH database, each classifier and the classifier combination result are trained using the training set, and selected by the validation. And the posterior probability in LFDM is estimated on the training and validation sets. The best parameter in MKL is selected from $\{10^{-3}, \cdots, 10^{3}\}$. On the other hand, for the linear programming based methods, we choose the best parameter ν from 0.1 to 0.9 with 0.1 incremental step.

Feature \ Dataset	Weizmann	KTH
Int	74.44	77.31
IntDif	**82.22**	**78.70**
HoF	73.33	75.46
HoG	66.67	53.24
HoF2D	67.78	68.96
HoG2D	61.11	58.33
HoF3D	75.56	75.00
HoG3D	64.44	65.74

Table 4.5: Recognition accuracy (%) of each feature on Weizmann and KTH database

The highest recognition accuracy of each feature is recorded in Table 4.5. Similar patterns on the two databases can be found in Table 4.5. The second feature with intensity difference over time gives the highest accuracies on both databases. And the HoF based features are better than the HoG based ones. This means temporal information is very important for action recognition. After the best classifiers are obtained, we evaluate the score combination methods based on them. The highest recognition accuracies are presented on the top rows in Table 4.6. Similar to the results on the Flower database, all the combination methods outperform the best single feature. On the other hand, LCDM gives the best performances together with IN on Weizmann and LP-B on KTH databases.

Similar to the experiments in Section 4.3.3, we use different density estimation techniques to evaluate the feature level fusion methods. The recognition accuracies on the two databases are shown in Table 4.7. Since the combination method developed under the dependent normal assumption [13] requires to solve a constrained quadratic programming problem numerically which is very time-consuming with high dimensionality problem on KTH database, the result with DN-F for KTH is not available. From Table 4.7, we can see that LFDM outperforms the other fusion

Method \ Dataset	Weizmann	KTH
Sum [2]	84.44	84.72
IN [13]	85.56	84.26
DN [13]	84.44	83.80
LPBoost [4]	83.33	83.33
LP-B [14]	84.44	85.19
LCDM	85.56	85.19
MKL [103] [104]	81.11	82.41
Sum-F [2]	57.78	78.70
IN-F [13]	68.89	77.31
DN-F [13]	77.78	–
LPBoost-F [4]	68.89	75.93
LP-B-F [14]	70.00	76.56
LFDM	**86.67**	**88.43**

Table 4.6: Recognition accuracy (%) of the best performance on Weizmann and KTH database

methods much more, compared to the results on the Digit database. This convinces that LFDM works well when the distribution varies.

The highest accuracies of all the fusion methods mentioned above as well as MKL[2] are summarized in Table 4.6. From Table 4.6, we can see that the proposed LFDM obtains the highest accuracy of 86.67% and 88.43% on Weizmann and KTH databases respectively. These results convince that LFDM can model the dependency in feature level very well. Since feature level contains more information about the label than that in classifier level, if dependency of features can be well modeled, the fusion process performed in feature level outperforms that in classifier level. On the

[2]The results with SILP MKL [103] and simple MKL [104] are the same.

	Weizmann		KTH	
	KDE [108]	KSD [109]	KDE [108]	KSD [109]
Sum-F [2]	57.78	48.89	78.70	50.00
IN-F [13]	68.89	67.78	69.91	77.31
DN-F [13]	77.78	76.67	–	–
LPBoost-F [4]	67.78	68.89	75.93	75.00
LP-B-F [14]	70.00	65.56	76.56	75.46
LFDM	**86.67**	**86.67**	**87.96**	**88.43**

Table 4.7: Recognition accuracies (%) of feature level fusion methods with different density estimation techniques on Weizmann and KTH database

other hand, comparing DN-F and LFDM on Weizmann database and digit database in the previous section, respectively, the performance of LFDM is much better than that of DN-F on the action database, while their performances are very close on the digit database. This also validates that LFDM works well under different distributions.

We also perform additional experiments for verification on Weizmann and KTH databases with the best two classifier level methods and feature level fusion methods, respectively. The ROC curves are recorded and plotted in Fig. 4.3(a) and Fig. 4.3(b). Same conclusion is drawn that the proposed LFDM gives the largest areas under the ROC curves on these two databases. On the other hand, although the accuracies of LCDM and IN on Weizmann, as well as LCDM and LP-B on KTH are the same, LCDM outperforms the two methods in ROC measurement.

In the last experiment, we evaluate the sensitivity of parameter ν in the linear programming based methods. Fig. 4.3(c) and Fig. 4.3(d) show the recognition accuracies of these methods with different values of ν. It can be observed that the feature level fusion methods are less sensitive to parameter changed on these two databases. The reason is as follows. Parameter ν is related to the tradeoff between

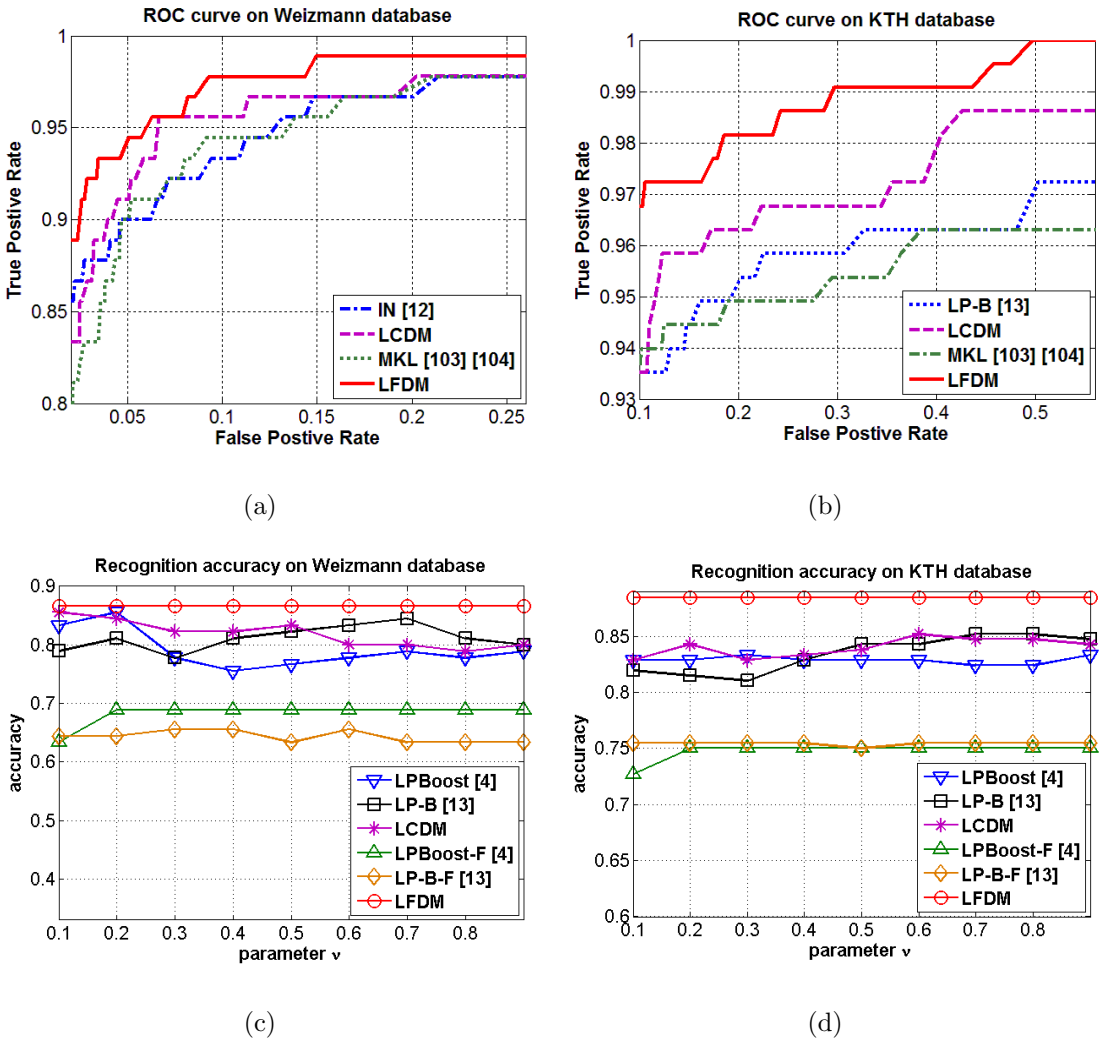


Figure 4.3: (a) and (b) are the ROC curves of the best two classifier level methods and feature level fusion methods on Weizmann and KTH database respectively. (c) and (d) are the recognition accuracy of the linear programming based methods in classifier level and feature level with different ν on Weizmann and KTH database respectively.

minimizing the classification error and maximizing the margin between genuine and imposter samples [4]. Since the feasible solution set in feature level is much larger than that in classifier level, the classification error is more likely to be zero in feature level, and the tradeoff between classification error and margin becomes less important. Besides, Fig. 4.3(c) and Fig. 4.3(d) also show that the proposed LFDM is insensitive to parameter and obtains much better results, compared with other feature level fusion methods. This is another advantage of the proposed model.

4.3.5 Analysis of Dependency and Learning Results

In this section, we would like to analyze the dependencies between classifiers or features and their relationship with the learning results for the real databases. From the theories of probability and statistics, some methods such as mutual information [26], product-moment correlation coefficient [110] and distance correlation [111], have been proposed to measure the dependency between two random variables. Nevertheless, computation of the mutual information suffers from the non-trivial problem of joint density estimation while the correlation coefficient only measures the linear dependencies. Analogous to correlation coefficient, distance correlation [111] quantifies dependency by measuring the difference between the joint characteristic function and the product of the marginal characteristic functions. Since distance correlation can be easily computed without the non-trivial estimation of the joint distribution of features and is able to measure both linear and non-linear dependencies [111], we employ it to measure the dependencies between classifiers or features for the four real databases in this experiment.

With the sampling sets $(Z_{1j}, Z_{2j}), j = 1, \cdots, J$ for random vectors $\mathcal{Z}_1$ and $\mathcal{Z}_2$, let us define the distance statistics as,

$$\begin{aligned} &d_{mij} = \|Z_{mi} - Z_{mj}\|, \;\; \bar{d}_{mi\cdot} = \frac{1}{J}\sum_{j=1}^{J} d_{mij}, \;\; \bar{d}_{m\cdot j} = \frac{1}{J}\sum_{i=1}^{J} d_{mij}, \\ &\bar{d}_{m\cdot\cdot} = \frac{1}{J^2}\sum_{i=1}^{J}\sum_{j=1}^{J} d_{mij}, \;\; D_{mij} = d_{mij} - \bar{d}_{mi\cdot} - \bar{d}_{m\cdot j} + \bar{d}_{m\cdot\cdot} \end{aligned} \tag{4.3.26}$$

for $m = 1$ or 2. With the distance statistics for each random vector $\mathcal{Z}_1$ and $\mathcal{Z}_2$ given by (4.3.26), the distance covariance is empirically given by

$$\mathcal{V}^2(\mathcal{Z}_1, \mathcal{Z}_2) = \frac{1}{J^2} \sum_{i=1}^{J} \sum_{j=1}^{J} D_{1ij} D_{2ij} \tag{4.3.27}$$

At last, the non-negative distance correlation $\mathcal{R}(\mathcal{Z}_1, \mathcal{Z}_2)$ [111] is computed by

$$\mathcal{R}^2(\mathcal{Z}_1, \mathcal{Z}_2) = \begin{cases} \frac{\mathcal{V}^2(\mathcal{Z}_1, \mathcal{Z}_2)}{\sqrt{\mathcal{V}^2(\mathcal{Z}_1)\mathcal{V}^2(\mathcal{Z}_2)}}, & \mathcal{V}^2(\mathcal{Z}_1)\mathcal{V}^2(\mathcal{Z}_2) > 0 \\ 0, & \mathcal{V}^2(\mathcal{Z}_1)\mathcal{V}^2(\mathcal{Z}_2) = 0 \end{cases} \tag{4.3.28}$$

where distance covariances $\mathcal{V}^2(\mathcal{Z}_1) = \mathcal{V}^2(\mathcal{Z}_1, \mathcal{Z}_1)$ and $\mathcal{V}^2(\mathcal{Z}_2) = \mathcal{V}^2(\mathcal{Z}_2, \mathcal{Z}_2)$. Non-negative $\mathcal{R}(\mathcal{Z}_1, \mathcal{Z}_2)$ in (4.3.28) satisfies $0 \leq \mathcal{R} \leq 1$, $\mathcal{R} = 0$ only if two random vectors are independent, and larger $\mathcal{R}$ indicates a larger degree of dependency between random vectors $\mathcal{Z}_1$ and $\mathcal{Z}_2$.

The distance correlation given by (4.3.28) is only for two random vectors, so we extend it to the case with multiple random vectors by averaging the correlations between any two different pairs of vectors. The measures of dependencies in classifier level and feature level are given by the following equations respectively.

$$\mathfrak{D}_c = \frac{2}{M(M-1)} \sum_{m_1=1}^{M} \sum_{m_2=m_1+1}^{M} \mathcal{R}(\mathcal{S}_{m_1}, \mathcal{S}_{m_2}) \tag{4.3.29}$$

$$\mathfrak{D}_f = \frac{2}{M(M-1)} \sum_{m_1=1}^{M} \sum_{m_2=m_1+1}^{M} \mathcal{R}(\mathcal{F}_{m_1}, \mathcal{F}_{m_2}) \tag{4.3.30}$$

where random vectors $\mathcal{S}_m$ and $\mathcal{F}_m$ are given by $\mathcal{S}_m = (s_{1m}, \cdots, s_{Lm})$ and $\mathcal{F}_m = (f_{1m1}, \cdots, f_{1mN_m})$ as in Section 4.2.2. Table 4.8 shows the dependency scores computed by (4.3.29) and (4.3.30) for the four real databases. For the Flower database, since original features have not been provided on their web site, we do not calculate the distance correlation in feature level. On the other hand, the dependency score in Digit database is the average of results from the four different classifiers recorded in Table 4.9.

Comparing the classifier dependencies ($\mathfrak{D}_c$) in Table 4.8, we can see that the Flower database has the smallest dependency. This is in line with the recognition

	Flower	Digit	Weizmann	KTH
$\mathfrak{D}_c$	0.2253	0.4485	0.4740	0.4926
$\mathfrak{D}_f$	–	0.5360	0.9373	0.9108

Table 4.8: Dependency indicators in classifier level and feature level for real databases

	1-NN	3-NN	SVM C=0.1	SVM C=100
$\mathfrak{D}_c$	0.2253	0.4485	0.4740	0.4926

Table 4.9: Dependency indicators with different classifiers in Digit database

performance in Section 4.3.2 that the independent methods, Sum and IN give convincing results in this database. However, the dependency score 0.2253 indicates a certain degree of dependency. Thus, LCDM gives the best recognition performance by learning the dependency automatically in this database.

Moreover, it can be seen from Table 4.8 that both Weizmann and KTH action databases have large feature dependency scores ($\mathfrak{D}_f$). This is reasonable as the action features are extracted from the same set of interest points with different descriptors. On the other hand, large dependency scores in action databases indicate that rich dependency information is available for dependency modeling. Therefore, LFDM significantly outperforms other feature level fusion methods in the action databases.

Finally, we compare the dependencies in classifier level and feature level, respectively. From Table 4.8 and Table 4.9, we can see that features in Weizmann and KTH action databases have a higher degree of dependency than that in Digit database, while it is not the case with the dependencies in classifier level. Moreover, Table 4.9 shows that dependencies with different classifiers differ from each other, although the same set of features is used in the Digit database. These observations show that classifier statistics cannot truly reflect the dependency characteristics in feature level. In other words, the dependency information is distorted in classifi-

er level. Therefore, LFDM which models dependency in feature level, outperforms LCDM, a classifier level dependency modeling method.

4.4 Summary

In this chapter, a new framework for dependency modeling between features by linear combination is designed and proposed for the tasks of recognition. Two methods, namely Linear Classifier Dependency Modeling (LCDM) and Linear Feature Dependency Modeling (LFDM) are developed based on the proposed framework. LCDM and LFDM are learned by solving the linear programming problems, which maximize the margins between the genuine and imposter posterior probabilities. LCDM and LFDM are designed for classifier and feature combination, respectively, without normal assumption.

Experimental results demonstrate that IN and DN [13] give the lowest error rates when the distributions are independent normal and dependent normal, respectively. However, normal assumption may not be valid in many practical applications, so the proposed LCDM and LFDM outperform existing classifier-level and feature-level fusion methods under non-normal distributions and on four real databases, respectively. Considering the classifier combination methods, the simple Sum rule is preferable, since it gets acceptable performance but take no additional time for training. In addition, analysis on dependencies between classifiers/features shows that statistics of classifier scores cannot truly reflect the dependency characteristics in feature level. Consequently, LFDM, which models dependency in feature explicitly, outperforms all existing classifier-level and feature-level fusion methods on the action databases.

Chapter 5

Reduced Analytic Dependency Modeling

5.1 Introduction

This chapter focuses on the assumption issue in the score level fusion methods. Under linear classifier combination assumption, optimal weighting method [3], LP-Boost [4] and its multi-class variants [14] determine the correct weighting for efficient classifier combination. Since linear methods are limited to linear separable systems, Toh *et al.* [6] developed a reduced multivariate polynomial model (RM) to describe the nonlinear input-output relationships for classifier fusion. Besides the linear or non-linear weighting approaches, He and Cao [20] proposed to efficiently integrate individual classifiers from the signal strength concept. In order to reduce the noise components in scores, the sparse technique with rank minimization was employed in [21]. And the robust late fusion (RLF) method [21] was developed based on the assumption that multiple score relation matrices can be decomposed into a shared low-rank matrix plus the sparse errors.

From probabilistic aspect, classifier fusion can be performed by estimating the joint distribution of multiple classifiers and performance is improved [5]. However,

when the number of classifiers is large, it needs numerous data to estimate the joint density accurately [27]. To deal with this problem, copula function with multivariate normal assumption was used to model the score dependency in [7]. On the other hand, Terrades *et al.* [13] proposed to combine classifiers in a non-Bayesian framework by linear combination. Under dependent normal (DN) assumption as in [7], they formulated the classifier combination problem into a constrained quadratic programming problem. Nevertheless, if normal assumption is not valid, these two methods will not be optimal.

Since the fused score represents the posterior probability, the fusion model taking probabilistic properties into account gives more convincing results as shown in chapter 4. On the other hand, most fusion methods are derived under certain assumptions, while it is hard to evaluate whether these assumptions are valid in practical applications. Thus, fusion method with less demanding assumption should give better performance. For these reasons, we develop a novel framework for dependency modeling with the analytic function assumption, which is easier to be satisfied. And a new method, namely Reduced Analytic Dependency Modeling (RADM) is proposed for score level fusion in this chapter.

Inspired by Product rule [2] (with independent assumption) and LCDM (without independent assumption), we propose to model dependency by analytic functions on posterior probabilities of each feature. With the analytic dependency model (ADM), an equation system is derived from the properties of marginal distributions. And an equivalent condition to the independent assumption is presented in this thesis based on the structure of the solution to the derived equation system. Since there may be infinite number of undetermined coefficients in the ADM, we propose a reduced form of it, based on the convergent properties of analytic functions. At last, the RADM learning algorithm is developed by taking advantages of label information from training data and probabilistic constraint derived from marginal distribution properties, under regularized least square criterion.

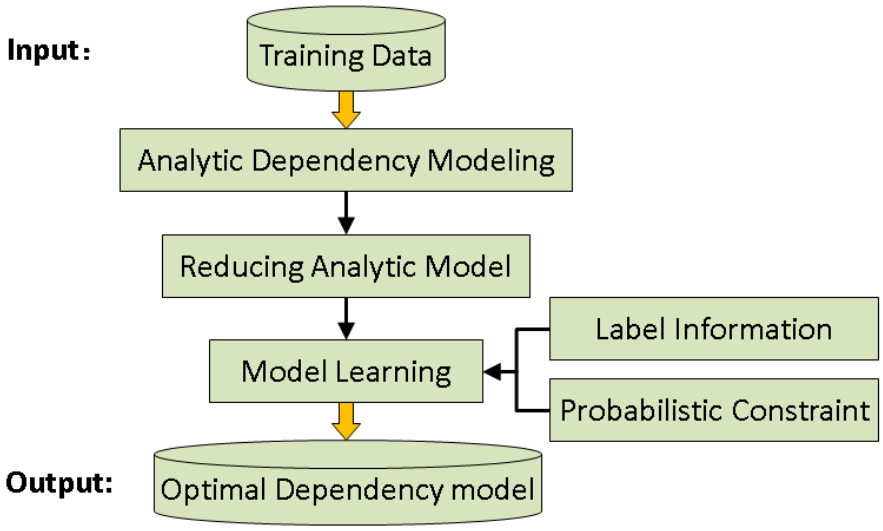


Figure 5.1: Proposed Reduced Analytic Dependency Modeling (RADM) framework

The rest of this chapter is organized as follows. Section 5.2 reports the proposed method. Experimental results and we have designed and proposed are given in Section 5.3 and Section 5.4, respectively.

5.2 Reduced Analytic Dependency Modeling

In this section, we propose a novel Reduced Analytic Dependency Modeling (RADM) method to model dependency for score level fusion. The block diagram of the proposed method is presented in Fig. 5.1 and briefly explained as follows. Given classification scores as training data, we propose to model dependency by analytic functions on them. Since the analytic dependency model (ADM) may contain infinite number of undetermined coefficients, the infinite model is reduced to finite one based on the convergent properties of analytic functions. With the reduced ADM, the optimal model is learned by incorporating label information from training data and probabilistic constraint derived from marginal distribution properties as shown in Fig. 5.1. Each component in this block diagram will be further elaborated in the following sections.

5.2.1 Analytic Dependency Modeling

Let us consider a combination problem that, there are M distinct feature descriptors $f_1, \cdots, f_M$ for any object $\mathcal{O}$. Denote feature representations $\vec{x}_1, \cdots, \vec{x}_M$ as $\vec{x}_m = f_m(\mathcal{O})$. The objective of dependency modeling is to estimate the posterior probability $\Pr(\omega_l|\vec{x}_1, \cdots, \vec{x}_M)$ for better classification performance. Usually, the dimension of the feature representations is very high. And, the modalities of feature representations can be different, e.g. $\vec{x}_m$ can be a vector or a set of points. Thus, direct dependency modeling in feature level is difficult. In turn, we consider modeling dependency by posterior probabilities of each feature, $\Pr(\omega_l|\vec{x}_m)$. Let us denote $s_{lm} = \Pr(\omega_l|\vec{x}_m)$ and $\vec{s}_l = (s_{l1}, \cdots, s_{lM})^T$. Since prior probabilities are not related to feature representations, $\Pr(\omega_l)$ is a positive constant c_l respect to $\vec{x}_1, \cdots, \vec{x}_M$. With these notations, denote two functions of the scores, h_{Product} as in equation (5.2.1) and h_{LCDM} as in equation (5.2.2).

$$h_{\text{Product}}(\vec{s}_l) = c_l^{1-M} \prod_{m=1}^{M} s_{lm} \tag{5.2.1}$$

$$h_{\text{LCDM}}(\vec{s}_l) = \sum_{m=1}^{M} a_{lm} s_{lm} + c_l(1 - M) \tag{5.2.2}$$

Then, the Product rule given by equation (2.1.1) under independent assumption and LCDM given by equation (4.2.4) for dependency modeling can be rewritten as equations (5.2.3) and (5.2.4), respectively.

$$\Pr(\omega_l|\vec{x}_1, \cdots, \vec{x}_M) = P_0 \cdot h_{\text{Product}}(\vec{s}_l) \tag{5.2.3}$$

$$\Pr(\omega_l|\vec{x}_1, \cdots, \vec{x}_M) \approx P_0 \cdot h_{\text{LCDM}}(\vec{s}_l) \tag{5.2.4}$$

As indicated in equations (5.2.3) and (5.2.4), the Product rule and LCDM can be formulated as two different functions h_{Product} and h_{LCDM} on posterior probabilities $s_{l1}, \cdots, s_{lM}$. This implies, if we choose a function different from h_{Product} with independent assumption, e.g. h_{LCDM}, the dependency can be modeled. On the other hand, LCDM was proposed under the assumption that posterior probabilities of each

classifier will not deviate dramatically from the priors as mentioned in [34]. However, without these assumptions, the function h_l for class ω_l on $s_{l1}, \cdots, s_{lM}$ should be different from h_{Product} and h_{LCDM}. Generally speaking, h_l can be any function which models dependency between feature representations by posterior probabilities $s_{l1}, \cdots, s_{lM}$, i.e.

$$\Pr(\omega_l|\vec{x}_1, \cdots, \vec{x}_M) = P_0 \cdot h_l(s_{l1}, \cdots, s_{lM}) \tag{5.2.5}$$

In order to write out the general function h_l explicitly, we propose to determine it by converged power series which is also known as analytic function. According to the definition of multivariate power series or analytic functions [112], h_l is given by the following equation,

$$h_l(\vec{s}_l; \vec{\alpha}_l) = \sum_{|\vec{\theta}|=0}^{\infty} \alpha_{l\vec{\theta}} \vec{s_l^{\theta}} \tag{5.2.6}$$

where $\vec{\theta} = (n_1, \cdots, n_M)^T$, $n_1, \cdots, n_M$ are non-negative integers, $|\vec{\theta}| = n_1 + \cdots + n_M$, $\vec{s_l^{\theta}} = \prod_{m=1}^{M} s_{lm}^{n_m}$ and $\vec{\alpha}_l = (\alpha_{l\vec{0}}, \cdots, \alpha_{l\vec{\theta}}, \cdots)^T$ is weighting coefficient vector in which $\vec{0} = (0, \cdots, 0)^T$.

With the analytic dependency model (ADM) given by equation (5.2.6), we further investigate how it can model dependency from probabilistic aspect. Let us consider the scores obtained by the first feature, i.e. $m = 1$, and rewrite the ADM function (5.2.6) according to the order of s_{l1} as,

$$h_l(\vec{s}_l; \vec{\alpha}_l) = \sum_{r=0}^{\infty} g_{l1r}(\tilde{s}_{l1}; \vec{\alpha}_{l1r}) s_{l1}^r \tag{5.2.7}$$

where $\tilde{s}_{l1} = (s_{l2}, \cdots, s_{lM})^T$ and g_{l1r} is an analytic function of $\tilde{s}_{l1}$ with coefficient vector $\vec{\alpha}_{l1r}$. On the other hand, posterior probabilities can be given by the Bayes' rule [110] as follow,

$$\begin{aligned} \Pr(\omega_l|\vec{x}_1) &= \frac{\Pr(\vec{x}_1|\omega_l)\Pr(\omega_l)}{\Pr(\vec{x}_1)} \\ \Pr(\omega_l|\vec{x}_1, \cdots, \vec{x}_M) &= \frac{\Pr(\vec{x}_1, \cdots, \vec{x}_M|\omega_l)\Pr(\omega_l)}{\Pr(\vec{x}_1, \cdots, \vec{x}_M)} \end{aligned} \tag{5.2.8}$$

Since conditional probability $\Pr(\vec{x}_1|\omega_l)$ in equation (5.2.8) can be viewed as the marginal probability of the joint density $\Pr(\vec{x}_1, \cdots, \vec{x}_M|\omega_l)$ over random measurements except $\vec{x}_1$ [110], we get

$$\Pr(\vec{x}_1|\omega_l) = \int \Pr(\vec{x}_1, \cdots, \vec{x}_M|\omega_l) d\vec{x}_2 \cdots d\vec{x}_M \tag{5.2.9}$$

With equations (5.2.5) (5.2.7) (5.2.8) and $P_0 = \frac{\prod_{m=1}^{M} \Pr(\vec{x}_m)}{\Pr(\vec{x}_1, \cdots, \vec{x}_M)}$ as mentioned in Section 2.1, the conditional joint density can be written as

$$\Pr(\vec{x}_1, \cdots, \vec{x}_M|\omega_l) = \frac{\prod_{m=1}^{M} \Pr(\vec{x}_m)}{\Pr(\omega_l)} \sum_{r=0}^{\infty} g_{l1r}(\tilde{s}_{l1}; \vec{\alpha}_{l1r}) s_{l1}^r \tag{5.2.10}$$

With notations $s_{l1} = \Pr(\omega_l|\vec{x}_1)$, substituting the probabilities $\Pr(\vec{x}_1|\omega_l)$ in (5.2.8) and $\Pr(\vec{x}_1, \cdots, \vec{x}_M|\omega_l)$ in (5.2.10) into equation (5.2.9), we get

$$s_{l1} = \int \prod_{m=2}^{M} \Pr(\vec{x}_m) [\sum_{r=0}^{\infty} g_{l1r}(\tilde{s}_{l1}; \vec{\alpha}_{l1r}) s_{l1}^r] d\vec{x}_2 \cdots d\vec{x}_M \tag{5.2.11}$$

According to Abel's Lemma [112], the series in (5.2.11) is uniformly converged. Thus, it can be integrated term by term, and equation (5.2.11) becomes

$$s_{l1} = \sum_{r=0}^{\infty} G_{l1r}(\vec{\alpha}_{l1r}) s_{l1}^r \tag{5.2.12}$$

where $G_{l1r}(\vec{\alpha}_{l1r}) = \int \prod_{m=2}^{M} \Pr(\vec{x}_m) g_{l1r} d\vec{x}_2 \cdots d\vec{x}_M$.

Without loss of generality, equation (5.2.12) is true for $m = 2, \cdots, M$. Comparing the left and the right hand sides in (5.2.12), the following equations can be obtained for $m = 1, \cdots, M$,

$$G_{lm1}(\vec{\alpha}_{lm1}) = 1 \tag{5.2.13}$$

$$G_{lm0}(\vec{\alpha}_{lm0}) = 0, G_{lm2}(\vec{\alpha}_{lm2}) = 0, G_{lm3}(\vec{\alpha}_{lm3}) = 0, \cdots \tag{5.2.14}$$

According to the definition (5.2.7), $g_{lmr}(\tilde{s}_{lm}; \vec{\alpha}_{lmr})$ is an analytic function similar to $h_l(\vec{s}_l; \vec{\alpha}_l)$ in equation (5.2.6) and the score vector $\tilde{s}_{lm}$ can be considered as mappings from feature representations $\vec{x}_1, \cdots, \vec{x}_{m-1}, \vec{x}_{m+1}, \cdots, \vec{x}_M$ to their posterior probabilities. Therefore, the integration of $\prod_{i \neq m} \Pr(\vec{x}_i) g_{lmr}(\tilde{s}_{lm}; \vec{\alpha}_{lmr})$ over feature representations except $\vec{x}_m$, which is denoted by $G_{lmr}(\vec{\alpha}_{lmr})$, is a linear function on

coefficient vector $\vec{\alpha}_{lmr}$. Without calculating the integration, we can observe that $\vec{\alpha}_{lm0} = \vec{0}, \vec{\alpha}_{lm2} = \vec{0}, \vec{\alpha}_{lm3} = \vec{0}, \cdots$ is a trivial solution to equation system (5.2.14) for $m = 1, \cdots, M$. Substituting this trivial solution into (5.2.7), we have the following proposition. (Please refer to the appendices for the proof of this proposition.)

Proposition 3.

Conditionally independent condition is equivalent to the situation that the solution to equation system (5.2.14) *is trivial, i.e.*

$$h_l(\vec{s}_l; \vec{\alpha}_l) = c_l^{1-M} \prod_{m=1}^{M} s_{lm} \Leftrightarrow \vec{\alpha}_{lm0} = \vec{0}, \vec{\alpha}_{lm2} = \vec{0}, \vec{\alpha}_{lm3} = \vec{0}, \cdots \tag{5.2.15}$$

This proposition gives an equivalent condition to the independent assumption from the structure of the solution to equation system (5.2.14). Considering the negative and inverse-negative propositions to the equivalent condition (5.2.15), if the solution to equation system (5.2.14) is non-trivial, the dependency between scores can be modeled. For general analytic functions, the weight vectors $\vec{\alpha}_{lm0}, \vec{\alpha}_{lm2}, \vec{\alpha}_{lm3}, \cdots$ are not necessary to be zeros. Consequently, the ADM can model dependency by setting non-trivial solution to the equation system (5.2.14).

5.2.2 Reduced Model

The ADM may have infinite number of coefficients in which directly estimating the coefficient vector $\vec{\alpha}_l$ is infeasible. In turn, we propose to approximate the ADM based on the convergent properties of the series defined by equations (5.2.6) and (5.2.7).

Let us consider equation (5.2.6) again. According to the definition of convergence of series [113], for any positive number ε, there exists a positive integer K, such that $|\sum_{|\vec{\theta}|=K+1}^{\infty} \alpha_{l\vec{\theta}} \vec{s}_l^{\vec{\theta}}| \leq \varepsilon$. If ε tends to zero, the analytic function can be approximated by the following equation,

$$h_l(\vec{s}_l; \vec{\alpha}_l) \approx h_l(\vec{s}_l; \vec{\alpha}_l; K) = \sum_{|\vec{\theta}|=0}^{K} \alpha_{l\vec{\theta}} \vec{s}_l^{\vec{\theta}} \tag{5.2.16}$$

Without loss of generality, the series defined similar to equation (5.2.7) is converged for $m = 1, \cdots, M$, so h_l can be approximated by the first $R+1$ terms as follows,

$$h_l(\vec{s}_l; \vec{\alpha}_l) \approx h_l(\vec{s}_l; \vec{\alpha}_l; R) = \sum_{r=0}^{R} g_{lmr}(\tilde{s}_{lm}; \vec{\alpha}_{lmr}) s_{lm}^r \tag{5.2.17}$$

where R is a positive integer. Equations (5.2.16) and (5.2.17) indicate that the highest order of each term and each variable are K and R, respectively. Combining these two equations, the reduced analytic dependency model (RADM) is given by

$$\begin{gathered} h_l(\vec{s}_l; \vec{\alpha}_l; K, R) = \sum_{|\vec{\theta}|=0}^{K} \alpha_{l\vec{\theta}} \vec{s_l^{\theta}}, \\ \text{s.t. } \vec{\theta} = (n_1, \cdots, n_M)^T \text{ and } 0 \leq n_m \leq R \end{gathered} \tag{5.2.18}$$

Since the model order should be larger than or equal to the variable order, $K \geq R$. On the other hand, if $K > MR$, $h_l(\vec{s}_l; \vec{\alpha}_l; K, R) = h_l(\vec{s}_l; \vec{\alpha}_l; MR, R)$. This means the RADM degenerates to $h_l(\vec{s}_l; \vec{\alpha}_l; MR, R)$ when $K > MR$. Therefore, the relationship between the model order K and variable order R in the RADM is restricted as $R \leq K \leq MR$.

Denote the confidence vector $\vec{z}_l = (\vec{s_l^0}, \cdots, \vec{s_l^{\theta}}, \cdots)^T$, where $\vec{s_l^0}, \cdots, \vec{s_l^{\theta}}, \cdots$ are the terms in equation (5.2.18). With these notations, the RADM given by equation (5.2.18) can be written as $h_l(\vec{s}_l; \vec{\alpha}_l; K, R) = \vec{\alpha}_l^T \vec{z}_l$. The algorithmic procedure to obtain $\vec{z}_l$ in the RADM for class ω_l is given in Algorithm 5.1.

5.2.3 Model Learning

In this subsection, we present the RADM learning algorithm which minimizes the empirical classification error and approximates the dependency modeling constraint.

Supervised Model Learning

Given J training samples $\mathcal{O}_1, \cdots, \mathcal{O}_J$ and their corresponding labels $y_1, \cdots, y_J$, we formulate the learning problem in a supervised manner. In order to minimize the

Algorithm 5.1 Construct confidence vector $\vec{z}_l$ in RADM.

Require: Posterior probability scores $s_{l1}, \cdots, s_{lM}$ and model parameters K, R;

Set $\mathcal{D} = (0, 1, \cdots, R)^T$ and $\vec{z}_l = (1, s_{l1}, s_{l1}^2, \cdots, s_{l1}^R)^T$;
for $m = 2, 3, \cdots, M$ **do**
  Set $\tilde{\mathcal{D}} = (\mathcal{D}, \vec{0})$, where $\vec{0} = (0, \cdots, 0)^T$ with the same column dimension of $\mathcal{D}$;
  for $r = 1, 2, \cdots, R$ **do**
    Update $\tilde{\mathcal{D}} = (\tilde{\mathcal{D}}; (\mathcal{D}, r\vec{1}))$ which is the column concatenation of $\tilde{\mathcal{D}}$ and $(\mathcal{D}, r\vec{1})$, where $\vec{1} = (1, \cdots, 1)^T$ with the same dimension of $\mathcal{D}$;
    Update $\vec{z}_l = (\vec{z}_l; s_{lm}^r \vec{z}_l)$ which is the column concatenation of $\vec{z}_l$ and $s_{lm}^r \vec{z}_l$;
    Delete the rows in $\tilde{\mathcal{D}}$ and corresponding elements in $\vec{z}_l$ such that the summations of the rows in $\tilde{\mathcal{D}}$ are larger than K;
  end for
  Set $\mathcal{D} = \tilde{\mathcal{D}}$;
end for
return $\vec{z}_l$.

classification error, we consider the posterior probabilities as, $\Pr(\omega_l|\mathcal{O}_j)$ is equal to one, if $\omega_l = y_j$, and zeros, otherwise. On the other hand, with equation (5.2.5), the posterior probability is computed by $P_0 * h_l(\vec{s}_l; \vec{\alpha}_l)$, where $P_0 = \frac{\prod_{m=1}^{M} \Pr(\vec{x}_m)}{\Pr(\vec{x}_1, \cdots, \vec{x}_M)}$. Since estimations of the marginal probability $\Pr(\vec{x}_m)$ for each m and joint density $\Pr(\vec{x}_1, \cdots, \vec{x}_M)$ are difficult, direct computation of P_0 for $j = 1, \cdots, J$ may not be feasible. While larger joint density implies larger marginal probability, we assume that $\Pr(\vec{x}_{j1}, \cdots, \vec{x}_{jM})$ is proportional to $\Pr(\vec{x}_{jm})$ for $j = 1, \cdots, J$, which means P_0 is a positive constant respect to j. Denote $1/P_0 = b$. With the label information, we have the following equation,

$$h_l(\vec{s}_{jl}; \vec{\alpha}_l) = b\delta_{jl}, \quad \delta_{jl} = \begin{cases} 1, & \omega_l = y_j \\ 0, & \omega_l \neq y_j \end{cases} \tag{5.2.19}$$

where $\vec{s}_{jl} = (s_{jl1}, \cdots, s_{jlM})^T$, $s_{jlm} = \Pr(\omega_l|\vec{x}_{jm})$ and $\vec{x}_{jm} = f_m(\mathcal{O}_j)$.

According to equations (5.2.16) (5.2.17) (5.2.18), the reduced model $h_l(\vec{s}_l; \vec{\alpha}_l; K, R)$ approximates but is not exactly equal to $h_l(\vec{s}_{jl}; \vec{\alpha}_l)$. Denote $\vec{z}_{jl} = (\vec{s}_{jl}^{\vec{0}}, \cdots, \vec{s}_{jl}^{\vec{\theta}}, \cdots)^T$ in (5.2.18) for $j = 1, \cdots, J$ and $l = 1, \cdots, L$. The coefficient vector in the RADM can be learned by minimizing the normalized least square error between the analytic model $h_l(\vec{s}_{jl}; \vec{\alpha}_l)$ and the reduced model $h_l(\vec{s}_l; \vec{\alpha}_l; K, R)$. With equation (5.2.19), the error function is defined as follow,

$$E_{\mathrm{LS}}(\vec{\alpha}, b) = \frac{1}{2LJ} \sum_{l=1}^{L} \sum_{j=1}^{J} (\vec{\alpha}_l^T \vec{z}_{jl} - b\delta_{jl})^2 \tag{5.2.20}$$

where $\vec{\alpha}$ is the column concatenation of $\vec{\alpha}_1, \cdots, \vec{\alpha}_L$.

With the positive constraint $b > 0$, the objective function $E_{\mathrm{LS}}(\vec{\alpha}, b)$ in equation (5.2.20) is minimized when $\vec{\alpha}$ and b tend to zeros. However, this solution cannot separate the training samples from different classes with each other. On the other hand, equation (5.2.19) implies that the margins of the posteriors between different classes increase when b increases. Therefore, we propose to minimize the error function (5.2.20) and maximize the positive variable b at the same time. Then, the objective function is rewritten as,

$$E(\vec{\alpha}, b; \rho) = \frac{1}{2LJ} \sum_{l=1}^{L} \sum_{j=1}^{J} (\vec{\alpha}_l^T \vec{z}_{jl} - b\delta_{jl})^2 - \rho b \tag{5.2.21}$$

where ρ is a positive parameter balancing the least square error $E_{\mathrm{LS}}(\vec{\alpha}, b)$ and the positive variable b.

In order to avoid the parameter selection problem introduced by the balancing parameter ρ, we analyze the structure of the solution which minimizes the objective function (5.2.21), and have the following proposition. (Please refer to the appendices for the proof of this proposition.)

Proposition 4.

Denote $(\vec{\alpha}^*, b^*) = \arg\min_{\vec{\alpha},b}(E_{LS}(\vec{\alpha}, b) - b)$ *and* $(\vec{\alpha}_\rho^*, b_\rho^*) = \arg\min_{\vec{\alpha},b}(E_{LS}(\vec{\alpha}, b) - \rho b)$, *then* $\vec{\alpha}_\rho^* = \rho\vec{\alpha}$.

This proposition shows that the solution with the balancing parameter ρ can be

obtained by scaling the solution without ρ. Since the comparative relationships of the posterior probabilities remain the same with different scalings, the classification performance would not change by selecting different balancing parameters ρ. Therefore, we refine the error function (5.2.21) as

$$E(\vec{\alpha}, b) = \frac{1}{2LJ} \sum_{l=1}^{L} \sum_{j=1}^{J} (\vec{\alpha}_l^T \vec{z}_{jl} - b\delta_{jl})^2 - b \tag{5.2.22}$$

Probabilistic Constraint

According to the analysis in Section 5.2.1, the ADM models dependency by setting non-trivial solution to equations (5.2.13) and (5.2.14). While the reduced model learned by minimizing the error function (5.2.22) may not satisfy these two equations, the probabilistic constraint derived from marginal distribution properties needs to be considered in the fusion model. Although it is difficult to evaluate whether equations (5.2.13) and (5.2.14) are valid in practice, they are derived from and equivalent to the original equation (5.2.11). Thus, we approximate equation (5.2.11) empirically with the training data as follows.

As mentioned in Section 5.2.1, considering $m = 1$, equation (5.2.11) is derived by substituting equations (5.2.8) (5.2.10) into (5.2.9). Since the right hand side in equation (5.2.9) is the integration of the joint distribution given label ω_l, the empirical estimation of equation (5.2.11) only relies on the training samples from class ω_l. Thus, according to (5.2.7), equation (5.2.11) can be estimated by,

$$s_{j_1 l 1} = \frac{1}{J_l^{M-1}} \sum_{y_{j_2}=\omega_l} \cdots \sum_{y_{j_M}=\omega_l} h_l(s_{j_1 l 1}, \cdots, s_{j_M l M}; \vec{\alpha}_l) \tag{5.2.23}$$

where J_l is the number of training samples for class ω_l and $y_{j_1} = \omega_l$. Substituting equation (5.2.6) into the right hand side of (5.2.23), we get

$$s_{jl1} = \sum_{|\vec{\theta}|=0}^{\infty} \alpha_{l\vec{\theta}} s_{jl1}^{n_1} \prod_{m=2}^{M} t_{lmn_m}, \text{ s.t. } t_{lmn_m} = \frac{1}{J_l} \sum_{y_j=\omega_l} s_{jlm}^{n_m} \tag{5.2.24}$$

where $\vec{\theta} = (n_1, \cdots, n_M)^T$ is defined in equation (5.2.6).

Without lose of generality, equation (5.2.24) is valid for $m = 2, \cdots, M$. On the other hand, the reduced model (5.2.18) approximates but is not exactly equal to the analytic function in equation (5.2.23) or (5.2.24). Therefore, similar to the derivation in Section 5.2.3, we approximate equation (5.2.24) by minimizing the following normalized least square error

$$E_{\text{Marg}}(\vec{\alpha}) = \frac{1}{2L} \sum_{l=1}^{L} \frac{1}{MJ_l} \sum_{m=1}^{M} \sum_{y_j=\omega_l} (\vec{\alpha}_l^T \vec{q}_{jlm} - s_{jlm})^2 \tag{5.2.25}$$

where $\vec{q}_{jlm}$ is the cumulative confidence vector with elements defined in equation (5.2.24). The element with corresponding index vector $\vec{\theta} = (n_1, \cdots, n_M)$ in $\vec{q}_{jlm}$ is equal to $s_{jlm}^{n_m} \prod_{i \neq m} t_{lin_i}$. The algorithmic procedure to obtain $\vec{q}_{jlm}$ for $y_j = \omega_l$ is given by replacing s_{li}^r with t_{lir} for $i \neq m$ in Algorithm 5.1.

Supervised Learning with Probabilistic Constraint

In Section 5.2.3, we formulate the learning problem in a supervised manner, while the probabilistic constraint is derived from marginal distribution properties in Section 5.2.3. In this section, we learn the optimal coefficient vector $\vec{\alpha}$ in the RADM by minimizing the weighted combination of the error function (5.2.22) and marginal distribution constraint (5.2.25). On the other hand, a least square regularized term is added in the objective function, so that the fusion model suffers less from over-fitting problem. Therefore, the optimization problem becomes,

$$\min_{\vec{\alpha},b} (E(\vec{\alpha}, b) + \lambda E_{\text{Marg}}(\vec{\alpha}) + \frac{1}{2}\mu \vec{\alpha}^T \vec{\alpha}) \tag{5.2.26}$$

where λ and μ are positive parameters balancing the marginal distribution constraint and regularized term.

In other to solve this optimization problem, we convert the objective function in (5.2.26) to matrix formulation as follows. Denote the undetermined variable tuples $(\vec{\alpha}, b)$ as $\tilde{\alpha}$, and $\mathcal{Z}_l = (\vec{z}_{1l}, \cdots, \vec{z}_{Jl})$, $\vec{\delta}_l = (\delta_{1l}, \cdots, \delta_{Jl})^T$. The error function (5.2.22)

derived by making use of label information can be rewritten as

$$\begin{aligned} & E(\tilde{\alpha}) = \frac{1}{2}\tilde{\alpha}^T \mathcal{H} \tilde{\alpha} - \tilde{\alpha}^T \tilde{u}, \\ \text{s.t. } & \tilde{u} = (\vec{0}; 1), \\ & \mathcal{H} = \frac{1}{JL} \begin{pmatrix} \mathcal{Z}_1 \mathcal{Z}_1^T & & & -\mathcal{Z}_1 \vec{\delta}_1 \\ & \ddots & & \vdots \\ & & \mathcal{Z}_L \mathcal{Z}_L^T & -\mathcal{Z}_L \vec{\delta}_L \\ -\vec{\delta}_1^T \mathcal{Z}_1^T & \cdots & -\vec{\delta}_L^T \mathcal{Z}_L^T & \sum_{l=1}^{L} \vec{\delta}_l^T \vec{\delta}_l \end{pmatrix} \end{aligned} \tag{5.2.27}$$

On the other hand, let $\mathcal{Q}_{lm} = (\vec{q}_{j_1 lm}, \cdots, \vec{q}_{j_{J_l} lm})$ and $\vec{s}_{lm} = (s_{j_1 lm}, \cdots, s_{j_{J_l} lm})^T$ for $y_{j_m} = \omega_l$. The probabilistic constraint (5.2.25) derived from marginal distribution properties becomes

$$\begin{aligned} & E_{\text{Marg}}(\tilde{\alpha}) = \frac{1}{2}\tilde{\alpha}^T \mathcal{H}_{\text{Marg}} \tilde{\alpha} - \tilde{\alpha}^T \tilde{v} + c_0, \\ \text{s.t. } & H_l = \frac{1}{MJ_l} \sum_{m=1}^{M} \mathcal{Q}_{lm} \mathcal{Q}_{lm}^T, \\ & \vec{v}_l = \frac{1}{MJ_l} \sum_{m=1}^{M} \mathcal{Q}_{lm} \vec{s}_{lm}, \\ & \mathcal{H}_{\text{Marg}} = \frac{1}{L} \begin{pmatrix} H_1 & & & \vec{0} \\ & \ddots & & \vdots \\ & & H_L & \vec{0} \\ \vec{0}^T & \cdots & \vec{0}^T & 0 \end{pmatrix}, \tilde{v} = \frac{1}{L} \begin{pmatrix} \vec{v}_1 \\ \vdots \\ \vec{v}_L \\ 0 \end{pmatrix} \end{aligned} \tag{5.2.28}$$

where c_0 is a constant respect to $\tilde{\alpha}$. In addition, the regularization term is reformulated with $\tilde{\alpha}$ as

$$\vec{\alpha}^T \vec{\alpha} = \tilde{\alpha}^T \mathcal{H}_{\text{Reg}} \tilde{\alpha}, \text{ s.t. } \mathcal{H}_{\text{Reg}} = \begin{pmatrix} I & \vec{0} \\ \vec{0}^T & 0 \end{pmatrix} \tag{5.2.29}$$

where I represents the identity matrix. With the equations (5.2.27) (5.2.28) (5.2.29), we take the first derivative respect to $\tilde{\alpha}$ and obtain the optimal solution to optimization problem (5.2.26) as

$$\tilde{\alpha}^* = (\mathcal{H} + \lambda \mathcal{H}_{\text{Marg}} + \mu \mathcal{H}_{\text{Reg}})^{-1} (\tilde{u} + \lambda \tilde{v}) \tag{5.2.30}$$

Algorithm 5.2 Learning coefficient vector $\vec{\alpha}^*$ in RADM.

Require: Scores $s_{11}, \cdots, s_{JM}$, labels $y_1, \cdots, y_J$, and positive parameters λ, μ;

1: Construct confidence vector $\vec{z}_{jl}$ by Algorithm 5.1 for $j = 1, \cdots, J$ and $l = 1, \cdots, L$;

2: Construct cumulative vector $\vec{q}_{jlm}$ by modifying Algorithm 5.1 as mentioned in Section 5.2.3 for $y_j = \omega_l$, $l = 1, \cdots, L$, and $m = 1, \cdots, M$;

3: **for** $l = 1, \cdots, M$ **do**

4: Compute $\mathcal{Z}_l \mathcal{Z}_l^T$ and $\mathcal{Z}_l \vec{\delta}_l$ in equation (5.2.27);

5: Compute H_l and $\vec{v}_l$ in equation (5.2.28);

6: **end for**

7: Combine $\mathcal{Z}_1 \mathcal{Z}_1^T, \cdots, \mathcal{Z}_L \mathcal{Z}_L^T$ and $\mathcal{Z}_1 \vec{\delta}_1, \cdots, \mathcal{Z}_L \vec{\delta}_L$ to obtain $\mathcal{H}$ by equation (5.2.27);

8: Combine $H_1, \cdots, H_L$ and $\vec{v}_1, \cdots, \vec{v}_L$ to obtain $\mathcal{H}_{\text{Marg}}$ and $\vec{v}$ by equation (5.2.28);

9: Obtain the optimal solution $(\vec{\alpha}^*, b^*)$ by equation (5.2.30);

10: **return** $\vec{\alpha}^*$.

Recalling the analysis in Section 5.2.3, b is the ratio of the probability functions, and thus positive. However, it does not need to add this constraint $b > 0$ to the optimization problem (5.2.26) due to the following proposition. (Please refer to the appendices for the proof of this proposition.)

Proposition 5. *Let* $(\vec{\alpha}^*, b^*)$ *be the solution to the optimization problem* (5.2.26) *given by equation* (5.2.30)*, then it has* $b^* > 0$.

This proposition shows that the solution (5.2.30) to the unconstrained optimization problem (5.2.26) satisfies the positive constraint. And thus, it is optimal.

At last, the algorithmic procedure to train the optimal coefficients in the RADM is summarized in Algorithm 5.2.

5.3 Experiments

In this section, we compare the proposed RADM with state-of-the-art score level fusion algorithms, including Sum [2], IN [13], LP-B [14], RM [6], SSC [20], GRLF [21], DN [13] and LCDM, in six different domains of recognition problems, namely 1) Digit Recognition, 2) Flower Classification, 3) Face Recognition, 4) Human Action Recognition, 5) Object Categorization and 6) Consumer Video Understanding. The details are discussed in Sections 5.3.1– 5.3.6. In order to evaluate the probabilistic constraint (5.2.25) derived from marginal distribution properties, we compare the proposed method with $\lambda > 0$ and $\lambda = 0$ in equation (5.2.26) in Section 5.2.3. At last, the fusion performance with the proposed SSTNTL feature is reported in Section 5.3.8. It is important to point out that the main objective of these experiments is to evaluate the performance of different score level fusion methods, but not state-of-art digit, flower, face, human action, object and consumer video recognition algorithms.

5.3.1 Results on Digit Database

Since the probabilities are hard to be determined accurately due to the problem of limited training samples, we use existing classifier techniques [39], e.g. nearest neighbor (NN) classifiers or support vector machines, and normalize the classifier outputs by the double sigmoid method [8] to approximate the probabilities. We use five-fold CV to select the best parameters, and train the weights for classifier combination by the CV outputs. Positive parameters[1] λ and μ in equation (5.2.26) or (5.2.30) are selected from $\{10^{-4}, \cdots, 10^{4}\}$, while the variable order R is selected from one to four and the model order K is selected from one to eight with one step increment, respectively. In order to give a fair comparison, parameters in other score level fusion are selected as suggested in their papers [6] [14] [21].

Mean accuracies and standard deviations of the best single feature (BestFea) and

[1]The parameters are selected from the same sets of values in the following experiments.

Method	1-NN	3-NN	SVM C=0.1	SVM C=100
BestFea	93.69±0.48	84.28±0.69	93.34±0.50	94.73±0.65
Sum [2]	93.76±0.72	94.76±0.70	94.82±0.60	96.23±0.66
IN [13]	95.06±0.71	93.40±0.84	94.75±0.61	95.63±1.08
LP-B [14]	94.13±1.09	93.53±0.81	94.95±0.55	96.57±**0.35**
RM [6]	95.35±0.65	94.92±0.79	95.93±0.52	96.51±0.73
SSC [20]	94.23±1.11	93.36±1.32	94.64±0.94	95.39±0.77
GRLF [21]	95.71±**0.43**	**95.45**±0.67	95.80±0.50	96.28±0.57
DN [13]	94.77±0.55	93.50±0.89	94.82±0.59	94.93±1.09
LCDM	95.18±0.56	94.46±0.71	95.34±0.66	96.79±0.60
RADM	**95.72**±0.61	95.23±**0.59**	**96.30**±**0.35**	**96.98**±0.61

Table 5.1: Mean accuracy (%) and standard deviation on Multiple Feature Digit database

different classifier combination methods are reported in Table 5.1. From Table 5.1, we can see that the proposed RADM obtains the highest recognition rate of 96.98% with SVM classifier for $C = 100$ on this database. Moreover, the proposed method outperforms others with k-NN classifier for $k = 1$ and SVM classifiers for $C = 0.1, 100$. While GRLF achieves the highest accuracy with k-NN classifier for $k = 3$, the recognition rates of GRLF and RADM are close to each other and the standard derivation of RADM is smaller. These results indicate that RADM can improve the performance in most cases using different classifiers.

5.3.2 Results on Oxford 17 Flower Database

Since the results in Section 5.3.1 show that SVM gives better performance, we employ SVM classifiers on this and the following databases. Table 5.2 shows the accuracies under three splits, mean accuracies and standard deviations (Std) of different methods. From Table 5.2, we can see that recognition accuracies of the classifier fusion methods are much higher than that of the best single feature. This convinces

Method	Split 1	Split 2	Split 3	Mean±Std
BestFea	71.76	69.71	71.47	70.98±**1.11**
Sum [2]	87.94	81.76	86.76	85.49±3.28
IN [13]	86.47	83.24	86.18	85.29±1.79
LP-B [14]	86.76	83.53	87.06	85.78±1.96
RM [6]	86.18	83.53	86.76	85.49±1.72
SSC [20]	**88.82**	83.24	86.18	86.08±2.80
GRLF [21]	**88.82**	82.65	86.47	85.98±3.12
DN [13]	85.29	82.65	84.71	84.22±1.39
LCDM	87.94	83.82	87.06	86.27±2.17
RADM	**88.82**	**85.29**	**88.53**	**87.55**±1.96

Table 5.2: Accuracy (%) under three splits, mean accuracy and standard deviation (Std) on Oxford 17 Flowers database

that performance can be improved by combining different pieces of information. On the other hand, RADM, SSC and GRLF get the highest accuracy of 88.82% under the first split, while RADM outperforms the others under the second and third splits. Comparing the mean recognition rates, RADM obtains an improvement of 2.06% and 1.28% over the score fusion methods with independent assumption and those without independent assumption, respectively. This indicates that modeling dependency without specific assumptions like those in DN and LCDM helps to further improve the recognition performance. Overall speaking, RADM gives the best performance with highest mean accuracy and comparable standard deviation by better discovering the relationship between scores.

5.3.3 Results on Face Databases

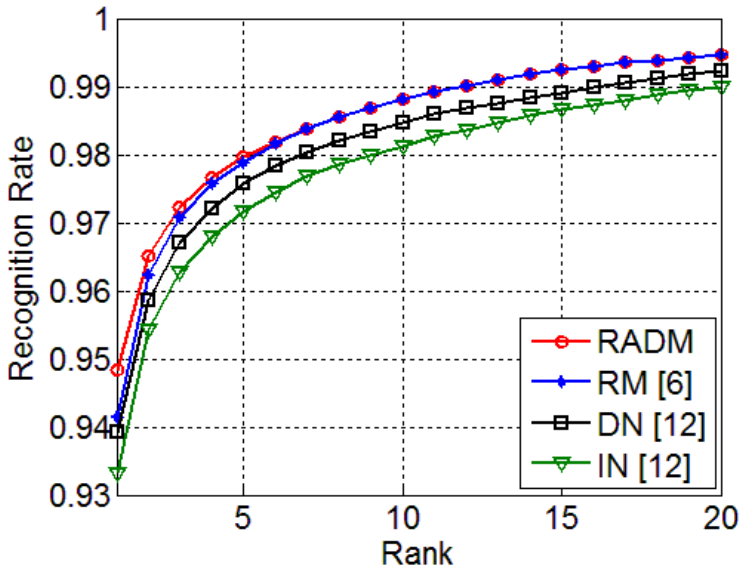
The mean accuracies and standard deviations on these two databases are reported in Table 5.3. Because GRLF requires to solve the singular value decomposition (SVD)

Method	CMU PIE	FERET
BestFea	88.85±2.93	83.80±12.51
Sum [2]	91.21±3.00	86.34±5.02
IN [13]	93.31±2.70	88.19±2.78
LP-B [14]	92.00±4.70	87.65±3.61
RM [6]	94.14±2.02	90.05±4.07
SSC [20]	90.66±2.87	84.26±5.78
GRLF [21]	—	84.03±4.52
DN [13]	93.91±2.29	87.73±3.50
LCDM	93.01±4.07	88.89±3.13
RADM	**94.83±1.94**	**92.13±1.75**

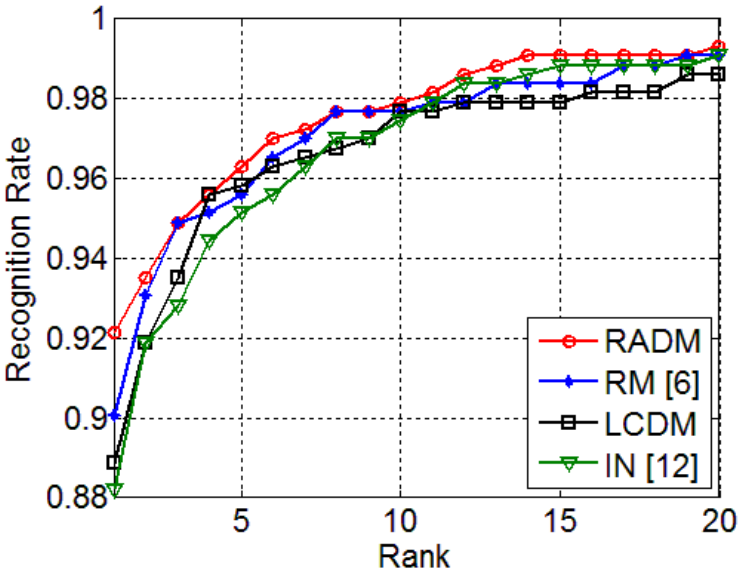
Table 5.3: Mean accuracy (%) and standard deviation on CMU PIE and FERET Face databases

in each iterative step [21], it is very time-consuming to get the optimal model when the number of class and the number of samples are large. Since there are 68 classes and around 7,000 testing samples on CMU PIE database for each validation, the result with GRLF is not available on CMU PIE database. On the other hand, GRLF only considers the intra-class relationship but does not take the inter-class relationship into account. Due to the problem of large number of class in face recognition task, GRLF gives poor performance on FERET database as shown in Table 5.3.

Comparing RADM with other methods, same conclusion can be drawn that RADM outperforms others on both two face databases. While results in Table 5.3 only show the rank-one accuracies, CMC curves of the top four methods on CMU PIE and FERET databases are plotted in Fig. 5.2(a) and Fig. 5.2(b), respectively, for detailed comparison. It can be seen that RADM outperforms IN and DN on CMU PIE, IN and LCDM on FERET database, and is slightly better than RM with different number of ranks. This indicates that RADM with dependency modeling



(a) CMC curve on CMU PIE



(b) CMC curve on FERET

Figure 5.2: CMC curves of the top four fusion methods on CMU PIE and FERET Face databases

Method \ Dataset	Weizmann	KTH
BestFea	82.22	78.70
Sum [2]	84.44	84.72
IN [13]	**85.56**	84.26
LP-B [14]	84.44	85.19
RM [6]	84.44	88.89
SSC [20]	84.44	83.33
GRLF [21]	83.33	83.80
DN [13]	84.44	83.80
LCDM	**85.56**	85.19
RADM	**85.56**	**90.28**

Table 5.4: Recognition accuracy (%) on Weizmann and KTH Human Action databases

gives the best performance for face recognition as well.

5.3.4 Results on Human Action Databases

In this section, we compare the classifier fusion methods on Weizmann [49] and KTH [55] human action databases. Table 5.4 shows the recognition rates of different methods on Weizmann and KTH human action databases. From Table 5.4, we can see that RADM, LCDM and IN get the highest recognition rate of 85.56% on Weizmann database, while RADM outperforms other methods on KTH database. We further compare the best three algorithms on Weizmann database by the receiver operating characteristic (ROC) measurement. The ROC curves in Fig. 5.3(a) show that RADM gives better performance when the false positive rate is larger than 10%. And the areas under curves (AUC) are 0.8524 for RADM, 0.8472 for LCDM and 0.8424 for IN. This also convinces that the proposed RADM is better than other classifier fusion methods for human action recognition.

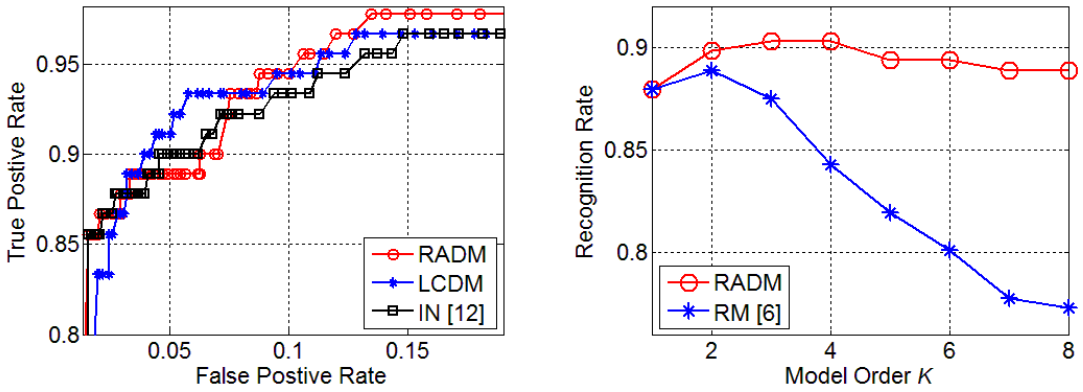
Method	VOC 2007	CCV
BestFea	42.63±**15.09**	50.81±16.71
Sum [2]	44.39±15.73	59.81±16.83
IN [13]	45.23±16.00	58.92±14.28
LP-B [14]	49.35±16.04	59.87±14.64
RM [6]	50.48±15.92	61.32±15.33
SSC [20]	44.50±15.60	59.61±15.80
GRLF [21]	46.00±16.17	60.61±14.31
DN [13]	46.59±16.48	58.52±**13.71**
LCDM	50.44±16.11	61.20±14.64
RADM	**52.03**±15.68	**62.99**±14.50

Table 5.5: MAP (%) and standard derivation on PASCAL VOC 2007 and CCV databases

On the other hand, observing the results on KTH database, the recognition rates of RM and RADM clearly outperforms other methods, while their performances are close to each other. Since both RM and RADM take higher order terms into account comparing with the linear methods, these results indicate that the relationship between scores can be better modeled when considering terms with order higher than one. We further compare RADM and RM with changed model order K on this database. From Fig. 5.3(b), we can see that RADM outperforms RM when the model order is larger than one. And RADM is less sensitive to model order changed. This is another advantage of the proposed method.

5.3.5 Results on VOC 2007 Database

The mean average precisions (MAP) and standard derivations over the 20 object classes of the fusion methods are recorded in the second column of Table 5.5. Our method outperforms other fusion methods, and gives an remarkable MAP improvement 6.80% over the independent fusion methods, Sum and IN. This implies classi-



(a) ROC curves on Weizmann (b) Recognition rate on KTH

Figure 5.3: Results on Weizmann and KTH databases

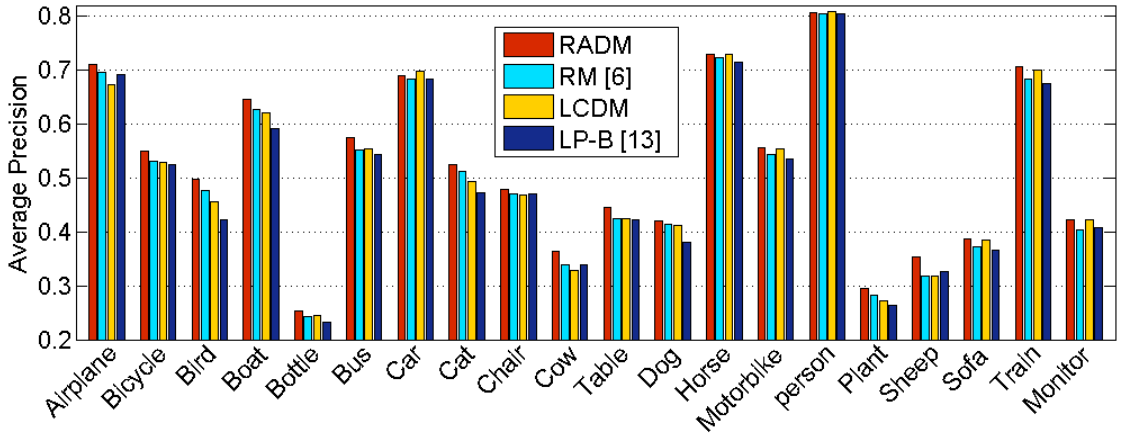


Figure 5.4: Per class average precisions of the top four methods on PASCAL VOC 2007 database

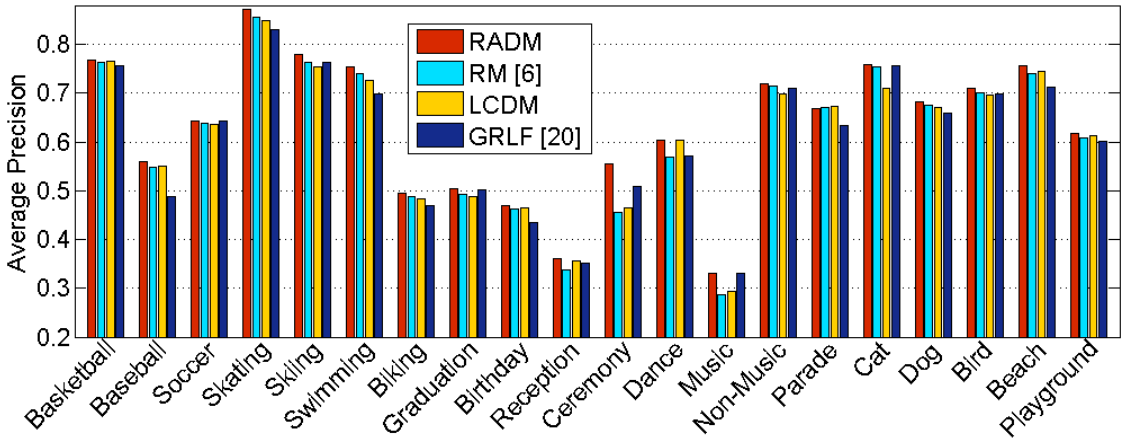


Figure 5.5: Per class average precisions of the top four methods on CCV database

fication performance can be greatly improved by modeling the relationship between scores, if features are dependent. On the other hand, we compare per-class average precisions (AP) of the top four fusion methods in Fig. 5.4. From Fig. 5.4, we can see that RADM clearly outperforms the other three fusion methods for categorization of some visual concepts, e.g. "bird", "boat" and "bus". Although LCDM achieves the highest AP for recognition of "car" and "person", RADM gives very close performance for classification of these two object classes. These results convince that RADM is still effective in the more challenging application of object categorization.

5.3.6 Results on Columbia Consumer Video Database

The third column of Table 5.5 presents the mean average precisions (MAP) and standard derivations over the 20 object classes of the fusion methods. From Table 5.5, we can see that the fusion methods with and without independent or other assumptions achieve close MAP performance around 60%. This indicates that it is easier to obtain comparable performance by combining information from different sensors or sources, e.g. video and audio in this experiment. However, RADM still outperforms the others due to the less demanding assumption. The per-class average precisions (AP) of the top four fusion methods are shown in Fig. 5.5. Our method outperforms others for categorization of most of 20 video concepts. These results show that RADM can better discover the score relationship reflecting feature dependency with the analytic function assumption, which is easier to be satisfied in many practical applications.

5.3.7 Comparing RADM with and without Marginal Distribution Constraint

In this experiment, we evaluate the classification performances of RADM with $\lambda > 0$ and $\lambda = 0$ in equation (5.2.26). Table 5.6 shows the comparing accuracies or MAPs

Database	RADM with $\lambda = 0$	RADM with $\lambda > 0$	Improvement
Digit	96.62	96.98	0.36
Flower	86.67	87.55	0.88
CMU PIE	94.59	94.83	0.24
FERET	91.67	92.13	0.46
Weizmann	84.44	85.56	1.12
KTH	90.28	90.28	0.00
VOC 2007	51.62	52.03	0.41
CCV	62.64	62.99	0.35

Table 5.6: Mean accuracy or mean average precision (%) on real databases

Database	Digit	Flower	CMU PIE	FERET
RADM with $\lambda = 0$	0.66	2.47	2.01	2.14
RADM with $\lambda > 0$	0.61	1.96	1.94	1.75
Improvement	0.06	0.51	0.07	0.39

Table 5.7: Standard derivation (%) on real databases

on the previous reported databases. On the other hand, since we have performed several runs on Digit, Flower, CMU PIE and FERET databases, the standard derivations on them are recorded in Table 5.7. From Table 5.6 and Table 5.7, we can see that RADM with $\lambda > 0$ achieves higher recognition rates or MAPs on most of the databases, while gives lower standard derivations under different splits of the data. This indicates that the probabilistic constraint (5.2.25) derived from marginal distribution properties not only improves the recognition performance, but also makes the fusion algorithm less sensitive to the selection of the training data.

The results on KTH database show that RADM with $\lambda > 0$ and $\lambda = 0$ gives equal accuracy of 90.28%. We further compare them with different numbers of model orders in Fig. 5.6. From Fig. 5.6, we can see that the probabilistic constraint derived from marginal distribution properties help to improve the recognition rate when the model order is larger than three. This also shows that RADM with the

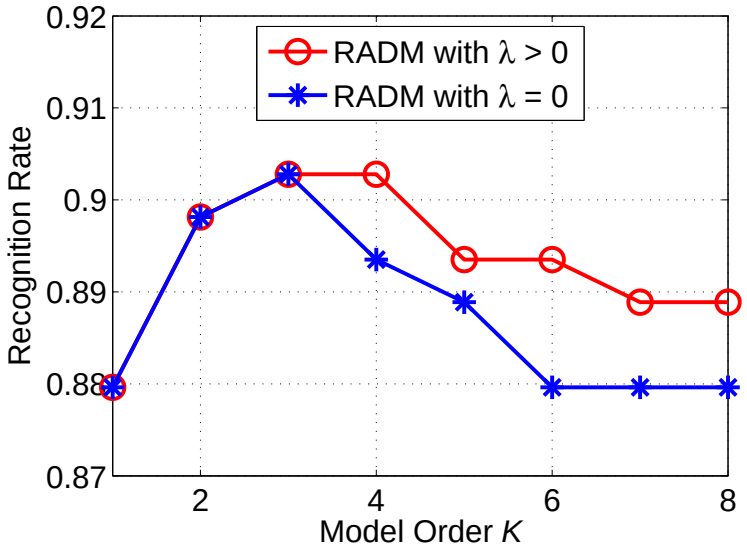
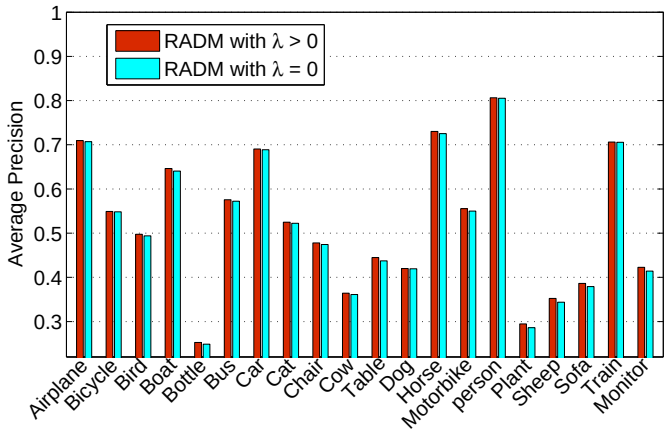


Figure 5.6: Recognition rate on KTH database

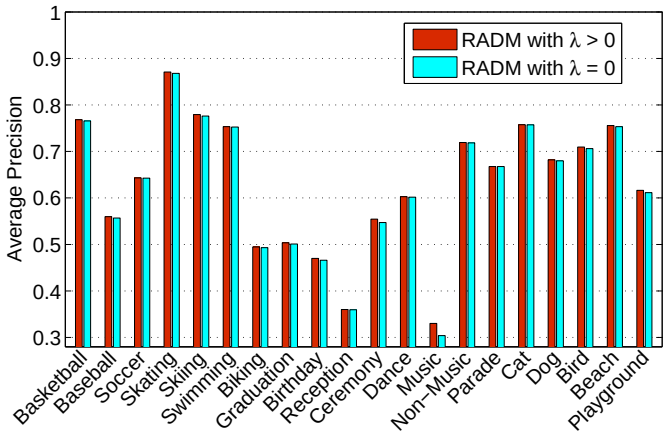
marginal distribution constraint (5.2.25) is less sensitive to model order changes. At last, we evaluate the per class AP on PASCAL VOC 2007 and CCV databases in Fig. 5.7(a) and Fig. 5.7(b), respectively. Since the fused score represents the posterior probability, these results convince that the fusion model taking advantages of probabilistic properties gives better performance in different recognition tasks.

5.3.8 Fusion with SSTNTL

In the last experiment, we evaluate the RADM fusion performance by combining the eight local features mentioned in Section 2.3.5 and the proposed SSTNTL for action recognition in HOHA database. The fusion performances with and without SSTNTL feature are reported in Table 5.8. From Table 5.8, we can see that all the fusion methods achieve improvements to different degrees by adding the SSTNTL feature. This indicates that fusion performance can be improved by fusing more discriminative features. Comparing the improvements in Table 5.8, RADM achieves the highest improvement of the average precision than others, while the improvements of LP-B, RM, LCDM and RADM are remarkably higher than others. This means some fusion methods can better discover the more discriminative features,



(a) Per class AP on PASCAL VOC 2007



(b) Per class AP on CCV

Figure 5.7: Per class average precision on PASCAL VOC 2007 and CCV databases

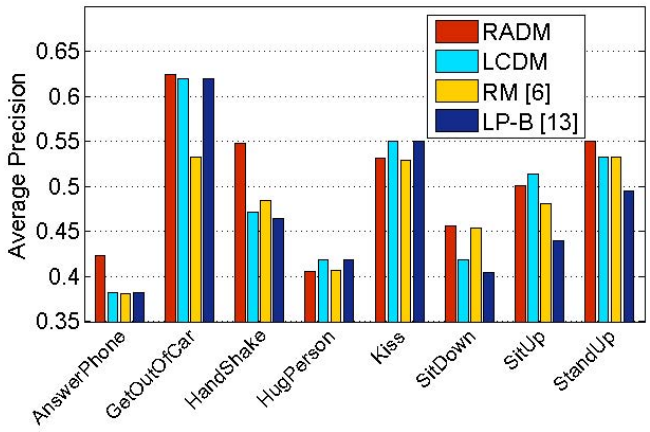


Figure 5.8: Per class precision on Hollywood Human Action database

Method	Without SSTNTL	With SSTNTL	Improvement
Sum [2]	42.58	42.88	0.30
IN [13]	42.74	43.07	0.36
LP-B [14]	42.90	47.17	4.27
RM [6]	43.97	47.53	3.56
SSC [20]	42.21	42.25	0.04
GRLF [21]	43.67	44.84	1.17
DN [13]	42.02	43.19	1.17
LCDM	43.64	48.85	5.21
RADM	**44.90**	**50.52**	**5.62**

Table 5.8: Average Precision (%) on Hollywood Human Action database

Method	AnP	GoC	HS	HP	Ki	SiD	SiU	StU	Avg
Best Local Feature	29.4	43.1	27.1	30.8	30.7	42.6	43.1	41.1	36.0
Local Feature Fusion	38.5	48.5	45.6	35.8	41.8	45.4	49.5	54.1	44.9
SSTNTL	40.0	**62.5**	44.4	38.1	44.2	30.2	44.4	52.4	44.5
Fusion with SSTNTL	**42.3**	62.4	**54.8**	**40.6**	**53.2**	**45.7**	**50.0**	**55.1**	**50.5**

Table 5.9: Per-class precision (%) comparison of RADM with and without SSTNTL feature on Hollywood Human Action database

but some cannot. The per-class precision comparison of the best four fusion methods with SSTNTL feature is presented in Fig. 5.8. From Fig. 5.8, we can see that RADM outperforms other fusion methods in classifying five out of the eight actions. These results convince that RADM is still effective when fusing more discriminative features.

At last, the fusion performance with and without SSTNTL using RADM is presented in Table 5.9. From Table 5.9, we can see that the average precision of the SSTNTL feature is 8.5% higher that of the best local feature, and only 0.4% lower than that of the fusion of eight local features. These results show that SSTNTL is

more discriminative comparing with the local features extracted in this experiment. On the other hand, SSTNTL outperforms local feature fusion significantly by classifying the action "Get out of Car", while the per-class precision of the "Sit Down" by SSTNTL is much lower than that by the local feature fusion. This indicates that local features and SSTNTL can provide complementary information to each other. Thus, it can achieve a remarkable improvement by combining them with RAMD as shown in Table 5.9.

5.4 Summary

In this chapter, a new framework is designed and proposed for dependency modeling by analytic functions on posterior probabilities of each feature. It is shown that Product rule [2] (with independent assumption) and LCDM [35] (without independent assumption) can be unified by the proposed analytic dependency model (ADM). With the ADM, we give an equivalent condition to independent assumption from the properties of marginal distributions. Since the ADM may contain infinite number of undetermined coefficients, a reduced form is proposed based on the convergent properties of analytic functions. At last, the optimal Reduced Analytic Dependency Model (RADM) is learned by label information from training data and probabilistic constraint derived from marginal distribution properties.

Experimental results show that the proposed RADM outperforms existing score level fusion methods on Digit, Flower, Face, Human Action, Object Categorization, and Consumer Video databases. This indicates that the assumption in RADM is easier to be satisfied, so that RADM can better model dependency and help to improve the performance in many recognition problems. At last, comparing RADM with and without the marginal distribution constraint, it is shown that the fusion model gives extra advantages by taking probabilistic properties into account.

Chapter 6

Conclusions and Future Works

Visual recognition is a challenging task, especially in the presence of self/mutual occlusions, clustered backgrounds, illumination variations, etc. In this thesis, dependency modeling is proposed to fuse multiple features and improve the recognition performance.

Based on the Bayesian model, it is proved that the linear combination of posteriors can model dependency under the assumption that posteriors will not deviate dramatically from the priors. With this property, a new linear dependency modeling framework has been developed and two methods, namely Linear Classifier Dependency Modeling (LCDM) and Linear Feature Dependency Modeling (LFDM) are proposed for classifier level and feature level fusion, respectively. Experiments show that LCDM and LFDM with linear dependency modeling give convincing results compared with state-of-the-art classifier level and feature level fusion methods, respectively. This indicates that dependency modeling can improve the recognition performance and the proposed linear combination framework can model dependency. With the analysis on dependencies between classifiers/features, it is shown that statistics of classification scores cannot truly reflect the characteristics of feature level dependency. Consequently, LFDM, which models dependency in feature level explicitly, outperforms existing classifier level and feature level fusion methods on

the Weizmann and KTH human action databases. And dependency modeling in feature level should be performed for information fusion.

In order to remove the assumptions in existing dependency models, a novel framework has been proposed by analytic functions on posterior probabilities of each feature. With the analytic dependency model (ADM), an equivalent condition to independent assumption is given based on the structure of the solution to the equation system derived from properties of marginal distributions. Since the ADM may contain infinite number of undetermined coefficients, the Reduced Analytic Dependency Model (RADM) is proposed based on the convergent properties of analytic functions. Experimental results show that the proposed RADM outperforms existing score level fusion methods on Digit, Flower, Face, Human Action, Object Categorization, and Consumer Video databases. This indicates recognition performance can be improved by reducing the fusion assumptions and the analytic function assumption in the RADM is less demanding. Moreover, comparing RADM with and without the marginal distribution constraint, it is shown that the fusion model gives extra advantages by taking probabilistic properties into account.

Combining the data distribution information with class label and global constraint of temporal labels, Supervised Spatio-Temporal Neighborhood Topology Learning (SSTNTL) is proposed for feature extraction in video applications. Experimental results show that the proposed SSTNTL outperforms existing manifold learning methods for video classification by taking the global information of temporal orders in action sequences into account. On the other hand, with the segmentation of motion regions of interest, SSTNTL is better than the interest points or trajectories based methods. This indicates that global frame representation can be more discriminative than the local descriptors, if the video background is not so complicated and the motion regions of interest can be detected robustly. In addition, experiments fusing SSTNTL and interest points based local descriptors show that the recognition performance can be further improved by adding the more discriminative SSTNTL feature.

Although the proposed methods achieve convincing results in the visual recognition applications reported in this thesis, some issues may be further investigated in the future.

- For the spatio-temporal feature representation, SSTNTL takes advantage of the global constraint in temporal labels to improve the recognition performance. However, motion regions of interest need to be segmented before learning the spatio-temporal manifold. If the video background is complicated and the camera is moving unevenly, the segmentation results with automatic motion detection are not robust. And motion regions of interest need to be segmented manually to achieve comparable performance. Therefore, it is valuable to revise the SSTNTL algorithm, so that it is less sensitive to complex background with complex camera motion in the future. On the other hand, the global feature may not be robust to occlusion, so future works may focus on extending the manifold learning based feature extraction method for the occlusion problem in videos.

- For the linear dependency model, the feature level fusion method, LFDM, shows some superiorities to LCDM, but LFDM takes longer time for training compared to the classifier level combination methods as well as the fast multiple kernel learning methods. Thus, we will further study the efficient algorithm for LFDM and investigate the fusion process in feature level in future. Besides, it will be interesting to revisit the idea of dependency modeling for specific applications. And, the recognition performance could be further improved by incorporating some characteristics in the specific applications.

- For the analytic dependency model, RADM is derived with less demanding assumption comparing with LCDM. Nevertheless, the assumption in RADM may not be suitable in feature level, so RADM has not been extended to feature level like the extension of LCDM to LFDM. Thus, the assumption issue in feature level fusion need to be further investigated in the future.

Appendices

Proof of Proposition 1

Features $\vec{x}_1, \cdots, \vec{x}_M$ can be viewed as random vectors. And posterior probabilities $\Pr(\omega_l|\vec{x}_m)$ and $\Pr(\omega_l|x_{mn})$ can be considered as real-value continuous functions. According to measure theory [114], measurable functions on random variables are random variables, so $s_{lm} = \Pr(\omega_l|\vec{x}_m)$ and $f_{lmn} = \Pr(\omega_l|x_{mn})$ are random variables.

On the other hand, similar to the derivation in Section 4.2.1 and Section 4.2.2, the relationship between s_{lm} and f_{lmn} is given by the following equation.

$$s_{lm} = P_m * (\sum_{n=1}^{N_m} \lambda_{lmn} f_{lmn} + \frac{1}{L}(1 - \sum_{n=1}^{N_m} \lambda_{lmn})) \tag{A1}$$

where $P_m = \frac{\prod_{n=1}^{N_m} \Pr(x_{mn})}{\Pr(x_{m1}, \cdots, x_{mN_m})}$. Equation (A1) implies that s_{lm} is a function on $f_{lm1}, \cdots, f_{lmN_m}$. This means $\mathcal{S}$ is a function on $\mathcal{F}$, i.e. $\mathcal{S} = g(\mathcal{F})$, so we get

$$\Pr(\mathcal{Y}|\mathcal{S}, \mathcal{F}) = \Pr(\mathcal{Y}|g(\mathcal{F}), \mathcal{F}) = \Pr(\mathcal{Y}|\mathcal{F}) \tag{A2}$$

where $\mathcal{Y}$ is the label viewed as a random variable and $\Pr(\cdot|\cdot)$ represents the conditional probability. This equation indicates that label $\mathcal{Y}$ is conditionally independent with scores $\mathcal{S}$ given features $\mathcal{F}$. Moreover, the Data Processing Inequality (DPI) [26] shows that, if random variables $\mathcal{Z}_1$ and $\mathcal{Z}_2$ are conditionally independent given $\mathcal{Z}_3$, then

$$I(\mathcal{Z}_1, \mathcal{Z}_3) \geq I(\mathcal{Z}_1, \mathcal{Z}_2) \text{ or } I(\mathcal{Z}_2, \mathcal{Z}_3) \geq I(\mathcal{Z}_1, \mathcal{Z}_2) \tag{A3}$$

where $I(\cdot,\cdot)$ represents the mutual information. Let $\mathcal{Z}_1 = \mathcal{Y}$, $\mathcal{Z}_2 = \mathcal{S}$ and $\mathcal{Z}_3 = \mathcal{F}$. Then, the first inequality in (A3) becomes $I(\mathcal{Y},\mathcal{F}) \geq I(\mathcal{Y},\mathcal{S})$, which indicates feature level contains more information about object label than score level does.

Proof of Proposition 2

The optimization problem for LFDM is given by (4.2.12) in Chapter 4. Since $p_{jlmn} = \Pr(\omega_l|x_{jmn}) - \Pr(\omega_l)$, the inequality condition $\Pr(\omega_{l_0}|x_{j_0mn}) < \Pr(\omega_{l_0})(1-\frac{1}{K})$ is equivalent to

$$p_{j_0l_0mn} + \frac{1}{KL} < 0 \tag{A4}$$

Multiplying (A4) with $\gamma_{mn}^{l_0}$ and computing the summation over m and n, we get

$$\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{l_0mn}p_{j_0l_0mn} + \frac{1}{KL}\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{l_0mn} < 0 \tag{A5}$$

According to (A5), if constraints iv) in (4.2.12) is valid, i.e. $\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{l_0mn} = K$, $\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{l_0mn}p_{j_0l_0mn} + \frac{1}{L} < 0$. And if constraints ii) in (4.2.12) is true, equation (A5) implies $\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{l_0mn} < K$. Thus, constraints ii) and iv) in (4.2.12) cannot be be hold at the same time.

Proof of Proposition 3

We first show that conditionally independent condition implies the solution to the equation system (5.2.14) is trivial, i.e. $\vec{\alpha}_{lm0} = \vec{0}, \vec{\alpha}_{lm2} = \vec{0}, \vec{\alpha}_{lm3} = \vec{0}, \cdots$ is a trivial solution to equation system (5.2.14) for $m = 1, \cdots, M$. In equation system (5.2.14),

$$G_{lmr}(\vec{\alpha}_{lmr}) = \int \prod_{i\neq m} \Pr(\vec{x}_i) g_{lmr}(\tilde{s}_{lm};\vec{\alpha}_{lmr}) d\vec{x}_1 \cdots d\vec{x}_{m-1} d\vec{x}_{m+1} \cdots d\vec{x}_M \cdots$$

where g_{lmr} is an analytic function defined on $\tilde{s}_{lm} = (s_{l1}, \cdots, s_{l(m-1)}, s_{l(m+1)}, \cdots s_{lM})^T$ with coefficient vector $\vec{\alpha}_{lmr}$. If feature representations are independent with each

other given class label ω_l, the ADM function $h_l(\vec{s}_l;\vec{\alpha}_l)$ becomes

$$h_l(\vec{s}_l;\vec{\alpha}_l) = c_l^{1-M}\prod_{m=1}^{M} s_{lm} \tag{A6}$$

Rewriting the ADM function (A6) according to the order of s_{lm}, we get

$$h_l(\vec{s}_l;\vec{\alpha}_l) = g_{lm1}(\tilde{s}_{lm};\vec{\alpha}_{lm1})s_{lm} \tag{A7}$$

where $g_{lm1}(\tilde{s}_{lm};\vec{\alpha}_{lm1}) = c_l^{1-M}\prod_{i\neq m} s_{li}$. Equation (A7) means that $g_{lmr}(\tilde{s}_{lm};\vec{\alpha}_{lmr}) \equiv 0$ or equivalently $\vec{\alpha}_{lmr} = \vec{0}$ for $r \neq 1$, i.e. the solution to equation system (5.2.14) is trivial.

On the other hand, given the solution to equation system (5.2.14) is trivial, we need to show that the ADM function $h_l(\vec{s}_l;\vec{\alpha}_l)$ is equal to equation (A6). If $\vec{\alpha}_{lmr} = \vec{0}$ for $r \neq 1$, then the ADM function $h_l(\vec{s}_l;\vec{\alpha}_l)$ can be rewritten as equation (A7) for $1 \leq m \leq M$. This implies each term in the power series h_l contains all variables $s_{l1}, \cdots, s_{lM}$ and the order of each s_{lm} cannot be larger than one. In this case, there is only one non-zero term $\prod_{m=1}^{M} s_{lm}$ in the analytic function h_l. In addition, according to the normalization equation (5.2.13), the non-zero term $\prod_{m=1}^{M} s_{lm}$ is normalized by the prior. And the ADM function becomes equation (A6). This complete the proof of this proposition.

Proof of Proposition 4

With matrix representation, the error function $E_{\mathrm{LS}}(\vec{\alpha}, b) - \rho b$ can be rewritten as

$$\begin{aligned} & E_\rho(\tilde{\alpha}) = \frac{1}{2}\tilde{\alpha}^T\mathcal{H}\tilde{\alpha} - \rho\tilde{\alpha}^T\tilde{u} \\ \text{s.t. } & \tilde{u} = (\vec{0};1), \\ & \mathcal{H} = \frac{1}{JL}\begin{pmatrix} \mathcal{Z}_1\mathcal{Z}_1^T & & & -\mathcal{Z}_1\vec{\delta}_1 \\ & \ddots & & \vdots \\ & & \mathcal{Z}_L\mathcal{Z}_L^T & -\mathcal{Z}_L\vec{\delta}_L \\ -\vec{\delta}_1^T\mathcal{Z}_1^T & \cdots & -\vec{\delta}_L^T\mathcal{Z}_L^T & \sum_{l=1}^{L}\vec{\delta}_l^T\vec{\delta}_l \end{pmatrix} \end{aligned} \tag{A8}$$

where $\mathcal{Z}_l = (\vec{z}_{1l}, \cdots, \vec{z}_{Jl})$ and $\vec{\delta}_l = (\delta_{1l}, \cdots, \delta_{Jl})^T$. We minimize the error function (A8) by computing the first derivative respect to $\tilde{\alpha}$. And the optimal solution is obtained by $\tilde{\alpha}^*_\rho = \rho\mathcal{H}^{-1}\tilde{u}$. Since $\tilde{\alpha}^*$ denotes the solution when $\rho = 1$, it has $\tilde{\alpha}^* = \mathcal{H}^{-1}\tilde{u}$ and $\tilde{\alpha}^*_\rho = \rho\tilde{\alpha}^*$, which proves this proposition.

Proof of Proposition 5

Let us consider solving the optimization problem (5.2.26) in an alternative way. For fixed b, we minimize the objective function respect to $\vec{\alpha}$ and denote the solution as a function of b, i.e.

$$F(b) = \min_{\vec{\alpha}}(E(\vec{\alpha}, b) + \lambda E_{\text{Marg}}(\vec{\alpha}) + \frac{1}{2}\mu\vec{\alpha}^T\vec{\alpha}) \tag{A9}$$

With the matrix representation, the objective function in equation (A9) can be rewritten as

$$\begin{aligned} &\frac{1}{2}\sum_{l=1}^{L}[\frac{1}{LJ}(\vec{\alpha}_l^T\mathcal{Z}_l - b\vec{\delta}_l^T)(\mathcal{Z}_l^T\vec{\alpha}_l - b\vec{\delta}_l) + \mu\vec{\alpha}_l^T\vec{\alpha}_l \\ &+\lambda\frac{1}{LMJ_l}\sum_{m=1}^{M}(\vec{\alpha}_l^T\mathcal{Q}_{lm} - \vec{s}_{lm}^T)(\mathcal{Q}_{lm}^T\vec{\alpha}_l - \vec{s}_{lm})] - b \end{aligned} \tag{A10}$$

where $\mathcal{Z}_l$, $\vec{\delta}_l$, $\mathcal{Q}_{lm}$ and $\vec{s}_{lm}$ are defined in Section 5.2.3. Based on the objective function in equation (A10), the optimization problem in (A9) can be decomposed into L subproblem respect to $\vec{\alpha}_l$. And the solution which minimizes the objective function (A10) for each $\vec{\alpha}_l$ is given as follow,

$$\begin{aligned} &\vec{\alpha}_l^* = \mathcal{H}_l^{-1}(b\vec{\phi}_l + \vec{\psi}_l), \\ \text{s.t. } &\mathcal{H}_l = \frac{1}{LJ}\mathcal{Z}_l\mathcal{Z}_l^T + \frac{\lambda}{LMJ_l}\sum_{m=1}^{M}\mathcal{Q}_{lm}\mathcal{Q}_{lm}^T + \mu I, \\ &\vec{\phi}_l = \frac{1}{LJ}\mathcal{Z}_l\vec{\delta}_l,\ \vec{\psi}_l = \frac{\lambda}{LMJ_l}\sum_{m=1}^{M}\mathcal{Q}_{lm}\vec{s}_{lm} \end{aligned} \tag{A11}$$

Substituting the optimal solution (A11) about $\vec{\alpha}_l$ into the objective function (A10), the function of b becomes

$$F(b) = \frac{1}{2}[\sum_{l=1}^{L}(\frac{1}{LJ}\vec{\delta}_l^T\vec{\delta}_l - \vec{\phi}_l^T\mathcal{H}_l^{-1}\vec{\phi}_l)]b^2 - (1 + \vec{\phi}_l^T\mathcal{H}_l^{-1}\vec{\psi}_l)b + c_0' \tag{A12}$$

where c_0' is a constant respect to b. This means, we obtain a quadratic function after minimizing the objective function respect to $\vec{\alpha}$.

Since matrix $\mathcal{H}_l$ is positive definite for positive regularization parameter μ, the inverse of $\mathcal{H}_l$ is positive definite as well. While the elements in vectors $\vec{\phi}_l$ and $\vec{\psi}_l$ are positive, $\vec{\phi}_l^T \mathcal{H}_l^{-1} \vec{\psi}_l$ is positive. Thus, the first order coefficient in equation (A12) is negative. On the other hand, according to equation (A10), $F(b) + b$ is non-negative for any b. This means the second order coefficient in $F(b) + b$ is positive. Otherwise, $F(b)+b$ is negative for some b. Since the second order coefficient does not change by adding the first order term b to quadratic function $F(b)$, the second order coefficient in equation (A12) is positive. Therefore, b^* corresponding to minimum value $F(b^*)$ is positive. Moreover, as discussed in Section 3.3.3 in the manuscript, equation (5.2.30) also gives the solution to optimization problem (5.2.26). Therefore, b^* given by equation (5.2.30) is also positive, which prove this proposition.

Bibliography

[1] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Pattern Recognition*, vol. 55, no. 1, pp. 119–139, 1997.

[2] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 226–239, 1998.

[3] N. Ueda, "Optimal linear combination of neural networks for improving classification performance," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 2, pp. 207–215, 2000.

[4] A. Demiriz, K. P. Bennett, and J. Shawe-Taylor, "Linear programming boosting via column generation," *J. Machine Learning Research*, vol. 46, no. 1-3, pp. 225–254, 2002.

[5] S. Prabhakar and A. K. Jain, "Decision-level fusion in fingerprint verification," *Pattern Recognition*, vol. 35, no. 4, pp. 861–874, 2002.

[6] K.-A. Toh, W.-Y. Yau, and X. Jiang, "A reduced multivariate polynomial model for multimodal biometrics and classifiers fusion," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 14, no. 2, pp. 224–233, 2004.

[7] S. C. Dass, K. Nandakumar, and A. K. Jain, "A principled approach to score level fusion in multimodal biometric systems," *Proc. Int'l Conf. Audio- and Video-Based Biometric Person Authentication*, pp. 1049–1058, 2005.

[8] A. Jain, K. Nandakumar, and A. Ross, "Score normalization in multimodal biometric systems," *Pattern Recognition*, vol. 38, no. 12, pp. 2270–2285, 2005.

[9] A. Ross and R. Govindarajan, "Feature level fusion using hand and face biometrics," *Proc. SPIE Conf. Biometric Technology for Human Identification II*, pp. 196–204, 2005.

[10] C. G. M. Snoek, M. Worring, and A. W. M. Smeulders, "Early versus late fusion in semantic video analysis," *Proc. ACM Int'l Conf. Multimedia*, pp. 399–402, 2005.

[11] A. Ross, K. Nandakumar, and A. K. Jain, *Handbook of Multibiometrics*. Springer, 2006.

[12] K. Nandakumar, Y. Chen, S. C. Dass, and A. K. Jain, "Likelihood ratio based biometric score fusion," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 342–347, 2008.

[13] O. R. Terrades, E. Valveny, and S. Tabbone, "Optimal classifier fusion in a non-bayesian probabilistic framework," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 31, no. 9, pp. 1630–1644, 2009.

[14] P. Gehler and S. Nowozin, "On feature combination for multiclass object classification," *Proc. IEEE Int'l Conf. Computer Vision*, pp. 221–228, 2009.

[15] M. He, S.-J. Horng, P. Fan, R.-S. Run, R.-J. Chen, J.-L. Lai, M. K. Khan, and K. O. Sentosa, "Performance evaluation of score level fusion in multimodal biometric systems," *Pattern Recognition*, vol. 43, no. 5, pp. 1789–1800, 2010.

[16] W. Scheirer, A. Rocha, R. Micheals, and T. Boult, "Robust fusion: Extreme value theory for recognition score normalization," *Proc. European Conf. Computer Vision*, pp. 481–495, 2010.

[17] A. Mittal, A. Zisserman, and P. Torr, "Hand detection using multiple proposals," *Proc. British Machine Vision Conference*, 2011.

[18] M. Awais, F. Yan, K. Mikolajczyk, and J. Kittler, "Augmented kernel matrix vs classifier fusion for object recognition," *Proc. British Machine Vision Conference*, 2011.

[19] M. Awais, F. Yan, K. Mikolajczyk, and J. Kittler, "Novel fusion methods for pattern recognition," *Proc. European conf. Machine Learning and Knowledge Discovery in Databases*, vol. 1, 2011.

[20] H. He and Y. Cao, "SSC: A classifier combination method based on signal strength," *IEEE Trans. Neural Networks and Learning Systems*, vol. 23, no. 7, pp. 1100–1117, 2012.

[21] G. Ye, D. Liu, I.-H. Jhuo, and S.-F. Chang, "Robust late fusion with rank minimization," *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pp. 3021–3028, 2012.

[22] J. Liu, S. McCloskey, and Y. Liu, "Local expert forest of score fusion for video event classification," *Proc. European Conf. Computer Vision*, 2012.

[23] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int'l J. Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[24] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 886–8930, 2005.

[25] X. He, S. Yan, Y. Hu, P. Niyogi, and H. J. Zhang, "Face recognition using laplacianfaces," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 3, pp. 328–340, 2005.

[26] T. M. Cover and J. A. Thomas, *Elements of information theory.* Wiley-Interscience, 2006.

[27] B. W. Silverman, *Density Estimation for Statistics and Data Analysis.* Chapman and Hall, 1986.

[28] M. Sugiyama, "Local fisher discriminant analysis for supervised dimensionality reduction," *Proc. Int'l Conf. Machine learning*, 2006.

[29] D. Cai, X. He, K. Zhou, J. Han, and H. Bao, "Locality sensitive discriminant analysis," *Proc. Int'l Joint Conf. Artificial intelligence*, 2007.

[30] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin, "Graph embedding and extensions: A general framework for dimensionality reduction," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 1, pp. 40–51, 2007.

[31] Z. Zheng, F. Yang, W. Tan, J. Jia, and J. Yang, "Gabor feature-based face recognition using supervised locality preserving projection," *Signal Processing*, vol. 87, pp. 2473–2483, 2007.

[32] K. Jia and D.-Y. Yeung, "Human action recognition using local spatio-temporal discriminant embedding," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

[33] A. J. Ma and P. C. Yuen, "Linear dependency modeling for feature fusion," *Proc. IEEE Int'l Conf. Computer Vision*, pp. 2041–2048, 2011.

[34] A. J. Ma, P. C. Yuen, and J.-H. Lai, "Linear dependency modeling for classifier fusion and feature combination," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 35, no. 5, pp. 1135–1148, 2013.

[35] A. J. Ma and P. C. Yuen, "Reduced analytical dependency modeling for classifier fusion," *Proc. European Conf. Computer Vision*, 2012.

[36] A. J. Ma and P. C. Yuen, "Reduced analytical dependency modeling for score level fusion," *submittd to Int'l J. Computer Vision*, 2013.

[37] A. J. Ma, P. C. Yuen, W. W. Zou, and J.-H. Lai, "Supervised spatio-temporal neighborhood topology learning for action recognition," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 23, no. 8, pp. 1447–1460, 2013.

[38] M. A. T. Figueiredo and A. K. Jain, "Unsupervised learning of finite mixture models," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 381–396, 2002.

[39] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Wiley, 2000.

[40] D. G. Luenberger and Y. Ye, *Linear and Nonlinear Programming, Third Edition*. Springer, 2008.

[41] M. V. Breukelen, R. P. W. Duin, D. M. J. Tax, and J. E. D. Hartog, "Handwritten digit recognition by combined classifiers," *Kybernetika*, vol. 34, no. 4, pp. 381–386, 1998.

[42] S. Canu, Y. Grandvalet, V. Guigue, and A. Rakotomamonjy, "SVM and kernel methods matlab toolbox." Perception Systèmes et Information, INSA de Rouen, Rouen, France, 2005.

[43] M.-E. Nilsback and A. Zisserman, "A visual vocabulary for flower classification," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2006.

[44] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," *Proc. IEEE Indian Conf. Computer Vision, Graphics and Image Processing*, 2008.

[45] T. Sim, S. Baker, and M. Bsat, "The CMU pose, illumination, and expression database," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, no. 12, pp. 1615–1618, 2003.

[46] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. Fisherfaces: recognition using class specific linear projection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 711–720, 1997.

[47] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face recognition with local binary patterns," *Proc. European Conf. Computer Vision*, pp. 469–481, 2004.

[48] P. J. Phillips, S. A. R. H. Moon, and P. J. Rauss, "The FERET evaluation methodology for face recognition algorithms," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, pp. 1090–1104, 2000.

[49] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2247–2253, 2007.

[50] C. Liu, P. C. Yuen, and G. Qiu, "Object motion detection using information theoretic spatio-temporal saliency," *Pattern Recognition*, vol. 42, no. 11, pp. 2897–2906, 2009.

[51] C. Liu and P. C. Yuen, "Human action recognition using boosted EigenActions," *Image and Vision Computing*, vol. 28, no. 5, pp. 825–835, 2010.

[52] C. Liu and P. C. Yuen, "A boosted co-training algorithm for human action recognition," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 21, no. 9, pp. 1203–1213, 2011.

[53] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," *Joint IEEE Int'l Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 2005.

[54] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[55] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local svm approach," *Proc. IEEE Int'l Conf. Pattern Recognition*, 2004.

[56] A. Oliva and A. Torralba, "Modeling the shape of the scene: a holistic representation of the spatial envelope," *Int'l J. Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001.

[57] Z. Gao, M. yu Chen, A. G. Hauptmann, and A. Cai, "Comparing evaluation protocols on the KTH dataset," *Human Behavior Understanding*, pp. 88–100, 2010.

[58] M. D. Rodriguez, J. Ahmed, and M. Shah, "Action mach a spatio-temporal maximum average correlation height filter for action recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[59] T.-K. Kim and R. Cipolla, "Canonical correlation analysis of video volume tensors for action categorization and detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 31, no. 8, pp. 1415–1428, 2009.

[60] Y. M. Lui, J. R. Beveridge, and M. Kirby, "Action classification on product manifolds," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 833–839, 2010.

[61] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results." http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[62] M. Guillaumin, J. Verbeek, and C. Schmid, "Multimodal semi-supervised learning for image classication," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 902–909, 2010.

[63] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes (VOC) challenge," *Int'l J. Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[64] Y.-G. Jiang, G. Ye, S.-F. Chang, D. Ellis, and A. C. Loui, "Consumer video understanding: A benchmark database and an evaluation of human and machine performance," *Proc. ACM Int'l Conf. Multimedia Retrieval*, 2011.

[65] A. Elgammal and C.-S. Lee, "Nonlinear manifold learning for dynamic shape and dynamic appearance," *Computer Vision and Image Understanding*, vol. 106, pp. 31–46, 2007.

[66] G. Guo, Y. Fu, C. Dyer, and T. Huang, "Image-based human age estimation by manifold learning and locally adjusted robust regression," *IEEE Trans. Image Processing*, vol. 17, no. 7, pp. 1178–1188, 2008.

[67] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, pp. 2319–2323, 2000.

[68] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, pp. 2323–2326, 2000.

[69] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," *Advances in Neural Information Processing Systems*, 2001.

[70] O. C. Jenkins and M. J. Matarić, "A spatio-temporal extension to Isomap nonlinear dimension reduction," *Proc. Int'l Conf. Machine Learning*, pp. 441–448, 2004.

[71] M. Lewandowski, J. M. del Rincon, D. Makris, and J.-C. Nebel, "Temporal extension of laplacian eigenmaps for unsupervised dimensionality reduction of time series," *Proc. IEEE Int'l Conf. Pattern Recognition*, pp. 161–164, 2010.

[72] A. J. Ma, P. C. Yuen, W. Zou, and J.-H. Lai, "Supervised neighborhood topology learning for human action recognition," *IEEE Int'l Conf. Computer Vision Workshops*, pp. 476–481, 2009.

[73] M. P. do Carmo, *Riemannian geometry*. Birkhauser, 1993.

[74] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea, "Machine recognition of human activities: A survey," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 18, no. 11, pp. 1473–1488, 2008.

[75] R. Poppe, "A survey on vision-based human action recognition," *Image and Vision Computing*, vol. 28, no. 6, pp. 976–990, 2010.

[76] J. K. Aggarwal and M. S. Ryoo, "Human activity analysis: A review," *ACM Computing Surveys*, vol. 43, no. 3, pp. 16:1–16:43, 2011.

[77] D. Weinland and E. Boyer, "Action recognition using exemplar-based embedding," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 1–7, 2008.

[78] S. Baysal, M. C. Kurt, and P. Duygulu, "Recognizing human actions using key poses," *Proc. IEEE Int'l Conf. Pattern Recognition*, pp. 1727–1730, 2010.

[79] T. Zhang, J. Liu, C. X. Si Liu, and H. Lu, "Boosted exemplar learning for action recognition and annotation," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 21, no. 7, pp. 853–866, 2011.

[80] J. C. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised learning of human action categories using spatial-temporal words," *Int'l J. Computer Vision*, vol. 79, no. 3, pp. 299–318, 2008.

[81] A. Kläser, M. Marszałek, and C. Schmid, "A spatio-temporal descriptor based on 3d-gradients," *Proc. British Machine Vision Conference*, 2008.

[82] R. Messing, C. Pal, and H. Kautz, "Activity recognition using the velocity histories of tracked keypoints," *Proc. IEEE Int'l Conf. Computer Vision*, pp. 104–111, 2009.

[83] M. Raptis and S. Soatto, "Tracklet descriptors for action modeling and video analysis," *Proc. European Conf. Computer Vision*, vol. 6311/2010, pp. 577–590, 2010.

[84] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Action recognition by dense trajectories," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 3169–3176, 2011.

[85] L. Wang and D. Suter, "Learning and matching of dynamic shape manifolds for human action recognition," *IEEE Trans. Image Processing*, vol. 16, no. 6, pp. 1646–1661, 2007.

[86] C. Sminchisescu and A. Jepson, "Generative modeling for continuous non-linearly embedded visual inference," *Proc. Int'l Conf. Machine learning*, 2004.

[87] L. Wang and D. Suter, "Visual learning and recognition of sequential data manifolds with applications to human movement analysis," *Computer Vision and Image Understanding*, vol. 110, no. 2, pp. 153–172, 2008.

[88] X. He and P. Niyogi, "Locality preserving projections," *Advances in Neural Information Processing Systems*, 2003.

[89] J. L. Kelley, *General topology.* Birkhauser, 1975.

[90] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. 2, no. 1, pp. 43–49, 1978.

[91] L. Ballan, M. Bertini, A. D. Bimbo, L. Seidenari, and G. Serra, "Effective codebooks for human action categorization," *IEEE Int'l Conf. Computer Vision Workshops*, 2009.

[92] T. Guha and R. K. Ward, "Learning sparse representations for human action recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 34, no. 8, pp. 1576–1588, 2012.

[93] L. Yeffet and L. Wolf, "Local trinary patterns for human action recognition," *Proc. IEEE Int'l Conf. Computer Vision*, pp. 492–497, 2009.

[94] A. Kovashka and K. Grauman, "Learning a hierarchy of discriminative space-time neighborhood features for human action recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 2046–2053, 2010.

[95] D. Han, L. Bo, and C. Sminchisescu, "Selection and context for action recognition," *Proc. IEEE Int'l Conf. Computer Vision*, pp. 1933–1940, 2009.

[96] H. Wang, M. M. Ullah, A. Kläser, I. Laptev, and C. Schmid, "Evaluation of local spatio-temporal features for action recognition," *Proc. British Machine Vision Conference*, 2009.

[97] G. Yu, A. Norberto, J. Yuan, and Z. Liu, "Fast action detection via discriminative random forest voting and top-k subvolume search," *IEEE Trans. Multimedia*, vol. 13, no. 3, pp. 507–517, 2011.

[98] X. Wu, D. Xu, L. Duan, and J. Luo, "Action recognition using context and appearance distribution features," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 489–496, 2011.

[99] H. Ning, T. X. Han, D. B. Walther, M. Liu, and T. S. Huang, "Hierarchical space-time model enabling efficient search for human actions," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 19, no. 6, pp. 808–820, 2009.

[100] M.-P. Dubuisson and A. K. Jain, "A modified hausdorff distance for object matching," *Proc. IAPR Int'l Conf. Pattern Recognition*, vol. 1, pp. 566–568, 1994.

[101] A. Saffari, M. Godec, T. Pock, C. Leistner, and H. Bischof, "Online multi-class LPBoost," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 3570–3577, 2010.

[102] F. Bach, G. R. G. Lanckriet, and M. I. Jordan, "Multiple kernel learning, conic duality, and the smo algorithm," *Proc. Int'l Conf. Machine Learning*, 2004.

[103] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf, "Large scale multiple kernel learning," *J. Machine Learning Research*, vol. 7, pp. 1531–1565, 2006.

[104] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet, "SimpleMKL," *J. Machine Learning Research*, vol. 9, pp. 2491–2521, 2008.

[105] J. Yang, Y. Li, Y. Tian, L. Duan, and W. Gao, "Group-sensitive multiple kernel learning for object categorization," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 436–443, 2009.

[106] F. Orabona, J. Luo, and B. Caputo, "Online-batch strongly convex multi kernel learning," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 787–794, 2010.

[107] F. M. Alkoot and J. Kittler, "Experimental evaluation of expert fusion strategies," *Pattern Recognition Letters*, vol. 20, no. 11-3, pp. 1361–1369, 1999.

[108] Z. I. Botev, J. F. Grotowski, and D. P. Kroese, "Kernel density estimation via diffusion," *Annals of Statistics*, vol. 38, no. 5, pp. 2916–2957, 2010.

[109] A. W. Bowman and A. Azzalini, *Applied smoothing techniques for data analysis : the kernel approach with S-Plus illustrations*. Oxford University Press, 1997.

[110] W. Feller, *An Introduction to Probability Theory and Its Applications*, vol. 1. Wiley, 1968.

[111] G. J. Székely, M. L. Rizzo, and N. K. Bakirov, "Measuring and testing dependence by correlation of distances," *Annals of Statistics*, vol. 35, no. 6, pp. 2769–2794, 2007.

[112] S. G. Krantz and H. R. Parks, *A Primer of Real Analytic Functions*. Birkhäuser, 2002.

[113] W. Rudin, *Principles of mathematical analysis*. McGraw-Hill, 1976.

[114] P. R. Halmos, *Measure Theory*. Springer, 1974.

Curriculum Vitae

Academic qualifications of the thesis author, Mr. MA Jinhua:

- Received the degree of Bachelor of Mathematics and Applied Mathematics from Sun Yat-Sen University, July 2007.
- Received the degree of Master of Applied Mathematics from Sun Yat-Sen University, July 2009.

August 2013