Linear Dependency Modeling for Classifier Fusion and Feature Combination

Andy J Ma, Pong C Yuen, *Senior Member, IEEE,* and Jian-Huang Lai

Abstract—This paper addresses the independent assumption issue in fusion process. In the last decade, dependency modeling techniques were developed under a specific distribution of classifiers or by estimating the joint distribution of the posteriors. This paper proposes a new framework to model the dependency between features without any assumption on feature/classifier distribution, and overcome the difficulty in estimating the high-dimensional joint density. In this paper, we prove that feature dependency can be modeled by a linear combination of the posterior probabilities under some mild assumptions. Based on the linear combination property, two methods, namely Linear Classifier Dependency Modeling (LCDM) and Linear Feature Dependency Modeling (LFDM), are derived and developed for dependency modeling in classifier level and feature level, respectively. The optimal models for LCDM and LFDM are learned by maximizing the margin between the genuine and imposter posterior probabilities. Both synthetic data and real datasets are used for experiments. Experimental results show that LCDM and LFDM with dependency modeling outperform existing classifier level and feature level combination methods under non-normal distributions and on four real databases, respectively. Comparing the classifier level and feature level fusion methods, LFDM gives the best performance.

Index Terms—Linear dependency modeling, feature dependency, classifier level fusion, feature level fusion, multiple feature fusion.

1 Introduction

Many computer vision and pattern recognition applications face the challenges of complex scenes with clustered backgrounds, small inter-class variations and large intra-class variations. To solve this problem, many algorithms have been developed to extract local or global discriminative features such as SIFT [1], HOG [2], Laplacianfaces [3]. Instead of extracting a high discriminative feature, fusion has been proposed and the results are encouraging [4] [5]. 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 [6] [7] [8] 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 [9], instead of taking the advantage of conditionally independent assumption, the classifier fusion method is proposed by estimating the joint probability 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 [10]. On the other hand, Terrades *et al.* [11] 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 optimization problem. Nevertheless, if normal assumption is not valid, the combination will not be optimal. Apart from these methods, LPBoost [12] and its multi-class variants [5] aim at determining the correct weighting for linear classifier combination to improve the classification performance. In [13], 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 implicitly model dependency as well.

All the above-mentioned methods perform the fusion based on classifier scores and implicitly assume that the feature dependency can be reflected by the classifier dependency. However, the Data Processing Inequality (DPI) [14] 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 large and it is very hard to estimate the joint probability distribution in feature level. Multiple Kernel Learning (MKL) [15] 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 [16] [17] are proposed. Recently, variants of MKL have been developed and used for object recognition [18] [19] [20]. MKL can be considered as dependency modeling technique based on the kernel

- *A. J. Ma and P. C. Yuen are with the Department of Computer Science, Hong Kong Baptist University, Hong Kong.*
  E-mail: {jhma, pcyuen}@comp.hkbu.edu.hk
- *J.-H. Lai is with the School of Information Science and Technology, Sun Yat-Sen University, Guangzhou 510006, China. He is also with the Guangdong Province Key Laboratory of Information Security, Guangzhou 510006, China.*
  E-mail: stsljh@mail.sysu.edu.cn

matrices, but dependency information may lose when original features are converted into kernel matrices.

In this paper, we develop a novel framework for dependency modeling, and propose two methods, namely Linear Classifier Dependency Modeling (LCDM) and Linear Feature Dependency Modeling (LFDM), 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 is available in feature level for modeling dependency, 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 contributions of this paper are three-fold.

• We prove that feature dependency can be modeled by a linear combination of the posterior probabilities of each feature/classifier under mild assumptions. This conclusion is obtained by adding small dependency terms to the naive Bayesian probability, expanding the product formulation and neglecting the terms of order two and higher.

• We develop a novel framework for dependency modeling, and propose two novel LCDM and LFDM for classifier level and feature level fusions, respectively. LCDM is developed based on the linear classifier combination property, and LFDM is derived by generalizing LCDM to feature level. The optimization problems of LCDM and LFDM are formulated as standard linear programming problems, which therefore, have analytical solutions. LCDM and LFDM model dependency explicitly. Moreover, they are derived without any assumption on classifier and feature distribution and will not suffer from the difficulty in high-dimensional joint distribution estimation.

• We show that the feature level combination method, LFDM, outperforms the classifier level fusion method, LCDM, in two ways. First, we prove that feature level contains more information about the label than that in classifier level under the proposed linear dependency modeling framework. Thus, if dependency in feature level can be modeled, LFDM outperforms LCDM. Second, we analyze the sensitivity to the density estimation errors for these two methods, and induce that the upper bound of the error factor in LFDM is much smaller than that in LCDM. Consequently, LFDM is better than LCDM in the worst case.

The rest of this paper is organized as follows. We will first review the exiting works relative to our model. Section 3 will report our proposed method. Experimental results and conclusion are given in Sections 4 and Sections 5 respectively. The preliminary version of this paper has been reported in [21].

2 Related Works

In this section, we review the combination methods with and without independent assumption.

2.1 Review of Combination Models Under Independent Assumption

In [6], a theoretical framework was developed for combining classifiers. According to Bayesian theory, under the assumption that classifier scores are distributed independently, the posteriori probability is given by [6]

$$\begin{aligned}\Pr(\omega_l|\vec{x}_1,\cdots,\vec{x}_M) &= \frac{\Pr(\omega_l)\prod_{m=1}^{M}\Pr(\vec{x}_m|\omega_l)}{\Pr(\vec{x}_1,\cdots,\vec{x}_M)} \\ &= \frac{P_0}{\Pr(\omega_l)^{M-1}}\prod_{m=1}^{M}\Pr(\omega_l|\vec{x}_m)\end{aligned} \tag{1}$$

where ω_l denotes the label, M is number of features, $\vec{x}_m$ is the m feature, and $P_0 = \frac{\prod_{m=1}^{M}\Pr(\vec{x}_m)}{\Pr(\vec{x}_1,\cdots,\vec{x}_M)}$. Product rule was then derived by (1). Moreover, if the discriminatory information is highly ambiguous due to high levels of noise, it may be suitable to assume that posteriori probability of each classifier will not deviate dramatically from the prior probability [6]. In this situation, the posterior probabilities can be expressed as

$$\Pr(\omega_l|\vec{x}_m) = \Pr(\omega_l)(1+\delta_m^l) \tag{2}$$

where δ_m^l is a small term. Substituting $\Pr(\omega_l|\vec{x}_m)$ by (2) and expanding the product term in (1), Sum rule was induced [6], i.e.

$$\begin{aligned}&\Pr(\omega_l|\vec{x}_1,\cdots,\vec{x}_M) \\ &\quad = P_0[(1-M)\Pr(\omega_l) + \sum_{m=1}^{M}\Pr(\omega_l|\vec{x}_m)]\end{aligned} \tag{3}$$

Based on the Product rule and Sum rule, Kittler *et al.* [6] 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 [6], Nandakumar *et al.* [8] 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 [22], and the joint distribution is computed by taking advantage of the Product rule (1) under the conditionally independent assumption. Apart from the likelihood ratio based fusion method [8], Terrades *et al.* [11] 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 [11].

2.2 Review of Combination Models Without Independent Assumption

Without conditionally independent assumption, the posterior probability of classifiers can be computed by joint distribution estimation. For example, in [9], Parzen window density estimation [23] is used to estimate the joint density of the posterior probability by a selected set of classifiers. 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 [9]. While this may not be the case for many practical applications, the error in the estimated density could be very large. As a result, many other combination methods were developed based on linear model.

In [11], it is assumed that classifier scores follow multivariate normal distribution. When the match scores are not conditionally independent, the covariance matrix is not diagonal. In this case, the dependency modeling method [11] under dependent normal (DN) assumption was formulated into a constraint quadratic programming problem, and relies on a numerical method to solve it.

Besides the probabilistic fusion methods [9] [11], LPBoost and MKL can be used to combine different kinds of features to improve the recognition performance [5]. A multi-class variant of the two-class LPBoost [12] is presented in [5] and denoted as LP-B. This linear combination method determines the correct weights B^l_m by learning the following linear programming (LP) problem

$$\begin{aligned} &\min_{B,\rho,\xi} -\rho + \frac{1}{\nu J}\sum_{j=1}^{J}\xi_j \\ &s.t.\ \ i)\ \sum_{m=1}^{M} B^{y_j}_m h_{m,y_j}(\vec{x}^j) \\ &\qquad\qquad - \sum_{m=1}^{M} B^l_m h_{m,l}(\vec{x}^j) \geq \rho - \xi_j, \forall j, \omega_l \neq y_j \\ &\quad ii)\ \xi_j \geq 0, \forall j \\ &\quad iii)\ \sum_{m=1}^{M} B^l_m = 1, \forall l \end{aligned} \tag{4}$$

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. Different from LP-B, MKL 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 feature fusion problem is formulated as multiple kernel combination. Since both LPBoost and MKL are developed without independent assumption and follow the linear combination rule similar to DN, they can model feature dependency implicitly.

3 Linear Dependency Modeling

With conditionally independent assumption, the posteriori probability can be computed as in (1) or (3). However, as discussed in Section 1, this assumption may not be true in many practical applications. Thus, in this paper, 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.

3.1 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 (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 scores. 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 [6], δ^l_m, $m = 1, \cdots, M$ given by (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^l_m, prior probability $\Pr(\omega_l)$ and small value δ^l_m. In order to ensure that the dependency term $\alpha^l_m \Pr(\omega_l)\delta^l_m$ is still not too large and bounded by the fixed values of prior probability $\Pr(\omega_l)$ and small number δ^l_m, we restrict $|\alpha^l_m| \leq 1$, so that $|\alpha^l_m \Pr(\omega_l)\delta^l_m| \leq |\Pr(\omega_l)\delta^l_m|$.

Adding dependency term $\alpha^l_m \Pr(\omega_l)\delta^l_m$ to each $\Pr(\omega_l|\vec{x}_m)$, the posterior probability becomes

$$\begin{aligned} &\Pr(\omega_l|\vec{x}_1, \cdots, \vec{x}_M) \\ &\quad = \frac{P_0}{\Pr(\omega_l)^{M-1}} \prod_{m=1}^{M} (\Pr(\omega_l|\vec{x}_m) + \alpha^l_m \Pr(\omega_l)\delta^l_m) \end{aligned} \tag{5}$$

In the above equation (5), α^l_m, $m = 1, \cdots, M$ are the weights to model the dependency between classifiers. If all the classifiers are independent with each other, $\alpha^l_m = 0$ for $m = 1, \cdots, M$ and the posterior probability with dependency terms given by (5) becomes the Product model as in (1) with the independent assumption. On the other hand, when classifiers are dependent, $\alpha_m^l \neq 0$ and the posterior formulation (5) models dependency.

Since the product formulation (5) may suffer from the outlier problem, we further expand it as follows. With the definition of the small number δ_m^l in (2), dependency model (5) becomes

$$\Pr(\omega_l|\vec{x}_1, \cdots, \vec{x}_M) = P_0 * \Pr(\omega_l) \prod_{m=1}^{M} (1 + \beta_m^l \delta_m^l) \quad (6)$$

where $\beta_m^l = \alpha_m^l + 1$, $0 \leq \beta_m^l \leq 2$. Since the terms $\delta_1^l, \cdots, \delta_M^l$ defined by (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 (6), the posterior probability becomes

$$\Pr(\omega_l|\vec{x}_1, \cdots, \vec{x}_M) = P_0 * \Pr(\omega_l)(1 + \sum_{m=1}^{M} \beta_m^l \delta_m^l) \quad (7)$$

Substituting δ_m^l by (2) into (7), we obtain

$$\begin{aligned} &\Pr(\omega_l|\vec{x}_1, \cdots, \vec{x}_M) \\ &= P_0 * (\sum_{m=1}^{M} \beta_m^l [\Pr(\omega_l|\vec{x}_m) - \Pr(\omega_l)] + \Pr(\omega_l)) \end{aligned} \quad (8)$$

This gives the linear dependency model with summation instead of multiplication, which less suffers from outliers.

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

First, the general dependency model given by (8) includes the independent case. When the independent assumption holds, $\beta_m^l = 1$ for all m, which results in $Q = \sum_{m=1}^{M} \beta_m^l = M$.

Second, constraint $\sum_{m=1}^{M} \beta_m^l = M$ gives a balance between under-learning and over-learning, which is elaborated as follows. With constraint $0 \leq \beta_m^l \leq 2$ for the dependency model, Q can be chosen from 0 to $2M$. When $Q = 0$, all dependency weights β_m^l 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_m^l are all equal to two and need not to be determined by the distribution of scores. 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 $\sum_{m=1}^{M} \beta_m^l = M$ 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 (8) with constraints $0 \leq \beta_m^l \leq 2$ and $\sum_{m=1}^{M} \beta_m^l = M$. When classifiers are independent with each other, $\beta_m^l = 1$ for $m = 1, \cdots, M$ and LCDM becomes the Sum rule as in (3). On the other hand, when classifiers are dependent, $\beta_m^l \neq 1$ and LCDM models dependency.

3.2 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 the joint probability $\Pr(x_{m1}, \cdots, x_{mN_m}|\omega_l)$. Since $\vec{x}_1, \cdots, \vec{x}_M$ are normally high dimensional vectors, it needs numerous samples to estimate the joint density [10]. In general, the probability is hard to determine accurately.

In classifier combination methods [6] [11], 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 $\mathcal{S} = (s_1^1, \cdots, s_M^L)$ and $\mathcal{F} = (f_{11}^1, \cdots, f_{MN_M}^L)$, where $s_m^l = \Pr(\omega_l|\vec{x}_m)$ and $f_{mn}^l = \Pr(\omega_l|x_{mn})$. In the classification problems, information from $\mathcal{S}$ in classifier level or $\mathcal{F}$ in feature level is used to infer the class label $\mathcal{L}$. From the aspect of information quantity of $\mathcal{S}$ and $\mathcal{F}$ about $\mathcal{L}$, we have the following proposition. (Please refer to Appendix A in the supplemental material for the proof of this proposition.)

Proposition 1. *Under the framework of linear dependency modeling by* (8), $I(\mathcal{S}, \mathcal{L}) \leq I(\mathcal{F}, \mathcal{L})$, *where* $\mathcal{S}$ *and* $\mathcal{F}$ *are the random vectors of the classification scores and features respectively,* $\mathcal{L}$ *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

$$\begin{aligned} &\Pr(\omega_l|\vec{x}_1, \cdots, \vec{x}_M) \\ &= \Pr(\omega_l|x_{11}, \cdots, x_{1N_1}, \cdots, x_{M1}, \cdots, x_{MN_M}) \end{aligned} \quad (9)$$

With the representation (9), each element in the feature vector can be viewed as a single classifier. Similar to the analysis in Section 3.1, the dependency term for each

x_{mn} can be written as $\alpha^l_{mn}\Pr(\omega_l)\delta^l_{mn}$, where α^l_{mn} satisfies $|\alpha^l_{mn}| \leq 1$ and each δ^l_{mn} is a small number, s.t.

$$\Pr(\omega_l|x_{mn}) = \Pr(\omega_l)(1+\delta^l_{mn}) \tag{10}$$

Then, the posterior probability analogous to (5) is given by

$$\begin{aligned}&\Pr(\omega_l|x_{11},\cdots,x_{1N_1},\cdots,x_{M1},\cdots,x_{MN_M})\\ &= \frac{P'_0}{\Pr(\omega_l)^{M-1}}\prod_{m=1}^{M}\prod_{n=1}^{N_m}(\Pr(\omega_l|x_{mn})+\alpha^l_{mn}\Pr(\omega_l)\delta^l_{mn})\end{aligned} \tag{11}$$

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 (10), the above equation can be rewritten as

$$\begin{aligned}&\Pr(\omega_l|x_{11},\cdots,x_{1N_1},\cdots,x_{M1},\cdots,x_{MN_M})\\ &= P'_0 * \Pr(\omega_l)\prod_{m=1}^{M}\prod_{n=1}^{N_m}(1+\gamma^l_{mn}\delta^l_{mn})\end{aligned} \tag{12}$$

where $\gamma^l_{mn} = \alpha^l_{mn}+1$. As δ^l_{mn} defined in (10) measures the difference between posterior $\Pr(\omega_l|x_{mn})$ and prior $\Pr(\omega_l)$, δ^l_{mn} is assumed to be small. By expanding the product in (12) and neglecting the terms of second and higher orders, it has

$$\begin{aligned}&\Pr(\omega_l|x_{11},\cdots,x_{1N_1},\cdots,x_{M1},\cdots,x_{MN_M})\\ &= P'_0 * \Pr(\omega_l)(1+\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma^l_{mn}\delta^l_{mn})\end{aligned} \tag{13}$$

Substituting δ^l_{mn} by (10) into (13), the posterior probability with dependency modeling in feature level is given as

$$\begin{aligned}&\Pr(\omega_l|x_{11},\cdots,x_{1N_1},\cdots,x_{M1},\cdots,x_{MN_M})\\ &= P'_0 * (\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma^l_{mn}[\Pr(\omega_l|x_{mn})-\Pr(\omega_l)]+\Pr(\omega_l))\end{aligned} \tag{14}$$

At last, applying the analysis for the LCDM constraints in Section 3.1 to feature level, we have two constraints, $\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma^l_{mn} = \sum_{m=1}^{M}N_m$ and $0 \leq \gamma^l_{mn} \leq 2$, for the proposed Linear Feature Dependency Model (LFDM).

3.3 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 $\vec{x}^j = (x^j_{11},\cdots,x^j_{1N_1},\cdots,x^j_{M1},\cdots,x^j_{MN_M})$ and y_j, $j=1,\cdots,J$, respectively. If all the training samples are correctly classified, $\Pr(y_j|x^j) > \max_{\omega_l\neq y_j}\Pr(\omega_l|x^j)$. Under this condition, the difference between the genuine and imposter probabilities $\Pr(y_j|x^j) - \max_{\omega_l\neq y_j}\Pr(\omega_l|x^j)$ is large. In turn, the performance will be good. Considering the marginal criterion, and inspired by LPBoost [12] and its multiclass generalization [5], the objective function to learn the best model is defined as

$$\begin{aligned}&\min_{\gamma,\rho,\xi} -\rho+\frac{1}{\nu J}\sum_{j=1}^{J}\sum_{\omega_l\neq y_j}\xi^l_j\\ &s.t.\ \ i)\ \Pr(y_j|x^j)-\Pr(\omega_l|x^j) \geq \rho-\xi^l_j, \forall j, \omega_l\neq y_j\\ &\quad ii)\ \Pr(\omega_l|x^j) \geq 0, \forall j, \omega_l\\ &\quad iii)\ \xi^l_j \geq 0, \forall j, \omega_l \neq y_j\end{aligned} \tag{15}$$

where ν is a constant parameter, $\nu \in (0,1)$. Let us denote $\Pr(\omega_l|x^j_{mn}) - \Pr(\omega_l)$ in (14) as $p^{j,l}_{mn}$ 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^l_{mn} as mentioned in Section 3.2, substituting (14) into (15), and ignoring P'_0 as a constant respective to class label, the optimization problem (15) becomes

$$\begin{aligned}&\min_{\gamma,\rho,\xi} -\rho+\frac{1}{\nu J}\sum_{j=1}^{J}\sum_{\omega_l\neq y_j}\xi^l_j\\ &s.t.\ \ i)\ \sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma^{y_j}_{mn}p^{j,y_j}_{mn}\\ &\qquad\qquad -\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma^l_{mn}p^{j,l}_{mn} \geq \rho-\xi^l_j, \forall j, \omega_l\neq y_j\\ &\quad ii)\ \sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma^l_{mn}p^{j,l}_{mn}+\frac{1}{L} \geq 0, \forall j, \omega_l\\ &\quad iii)\ \xi^l_j \geq 0, \forall j, \omega_l\neq y_j\\ &\quad iv)\ \sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma^l_{mn} = K, \forall l\\ &\quad v)\ 0 \leq \gamma^l_{mn} \leq 2, \forall l, m, n\end{aligned} \tag{16}$$

Optimization problem (16) is theoretically sound and purely derived from (15) mathematically. However, according to the following proposition, optimization problem (16) may not have feasible solution in practice. (Please refer to Appendix B in the supplemental material for the proof of this proposition.)

Proposition 2. *If there exist j_0 and l_0, s.t. $Pr(\omega_{l_0}|x^{j_0}_{mn}) < Pr(\omega_{l_0})(1-\frac{1}{KL}), \forall m, n$, then constraints ii) and iv) in* (16) *cannot be hold at the same time.*

When the fusion is performed in feature level, K will be a very large number, i.e. $\frac{1}{KL}$ is very small. In this situation, The condition in the above proposition is likely to be satisfied. Thus, optimization problem (16) will not have feasible solution, and one of the two constraints should be removed.

Constraint *ii)* in (15) is to ensure that the posterior probability is non-negative. From (15) to (16), the posterior probability is substituted with (14). Let us check the derivation from (12) to (14). We can see that equation (14) 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 (16) cannot guarantee that the posterior probability is non-negative. On the other hand, since γ_{mn}^l is less than or equal to 2 and δ_{mn}^l is a small number, e.g. $|\delta_{mn}^l| \leq \frac{1}{2}$, the posterior probability in product formulation is always non-negative according to (12). Thus, we remove the constraint $ii)$ in (16). Consequently, the final optimization problem becomes

$$\begin{aligned} &\min_{\gamma,\rho,\xi} -\rho + \frac{1}{\nu J}\sum_{j=1}^{J}\sum_{\omega_l \neq y_j} \xi_j^l \\ &s.t.\ \ i)\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{mn}^{y_j}p_{mn}^{j,y_j} \\ &\qquad -\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{mn}^{l}p_{mn}^{j,l} \geq \rho - \xi_j^l, \forall j, \omega_l \neq y_j \\ &\quad ii)\ \xi_j^l \geq 0, \forall j, \omega_l \neq y_j \\ &\quad iii)\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{mn}^l = K, \forall l \\ &\quad iv)\ 0 \leq \gamma_{mn}^l \leq 2, \forall l, m, n \end{aligned} \tag{17}$$

The optimal LFDM is learned by (17). Similarly, the optimal LCDM can be obtained by applying (8) into (15) and following the same derivation procedures from (15) to (17). The optimization problem for LCDM is given as

$$\begin{aligned} &\min_{\beta,\rho,\xi} -\rho + \frac{1}{\nu J}\sum_{j=1}^{J}\sum_{\omega_l \neq y_j} \xi_j^l \\ &s.t.\ \ i)\sum_{m=1}^{M}\beta_m^{y_j}h_m^{j,y_j} \\ &\qquad -\sum_{m=1}^{M}\beta_m^{l}h_m^{j,l} \geq \rho - \xi_j^l, \forall j, \omega_l \neq y_j \\ &\quad ii)\ \xi_j^l \geq 0, \forall j, \omega_l \neq y_j \\ &\quad iii)\sum_{m=1}^{M}\beta_m^l = M, \forall l \\ &\quad iv)\ 0 \leq \beta_m^l \leq 2, \forall l, m, n \end{aligned} \tag{18}$$

where $h_m^{j,l} = \Pr(\omega_l|\vec{x}_m^j) - \Pr(\omega_l)$. The optimization problems (18) and (17) for LCDM and LFDM are standard linear programming problems. Thus, the solutions can be determined by any off the shelf techniques, e.g. [24]. Since our experiments are performed in the Matlab environment, a Matlab build-in function with the interior point method is employed.

3.4 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

$$e_m^l = \widehat{\Pr}(\omega_l|\vec{x}_m) - \Pr(\omega_l|\vec{x}_m) \tag{19}$$

In practice, the estimated probabilities are used instead of the true ones. Consequently, according to (8), the LCDM model is changed to

$$\begin{aligned} &\widehat{\Pr}(\omega_l|\vec{x}_1,\cdots,\vec{x}_M) \\ &\quad = P_0 * [\Pr(\omega_l)(1-M) + \sum_{m=1}^{M}\beta_m^l\widehat{\Pr}(\omega_l|\vec{x}_m)] \end{aligned} \tag{20}$$

Substituting $\widehat{\Pr}(\omega_l|\vec{x}_m)$ with (19) into (20), it has

$$\begin{aligned} \widehat{\Pr}(\omega_l|\vec{x}_1,\cdots,\vec{x}_M) = &P_0 * [\Pr(\omega_l)(1-M) \\ &+ (1+E_c)\sum_{m=1}^{M}\beta_m^l\Pr(\omega_l|\vec{x}_m)] \end{aligned} \tag{21}$$

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

$$E_c = \frac{\sum_{m=1}^{M}\beta_m^l e_m^l}{\sum_{m=1}^{M}\beta_m^l\Pr(\omega_l|\vec{x}_m)} \tag{22}$$

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

$$E_f = \frac{\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{mn}^l\epsilon_{mn}^l}{\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{mn}^l\Pr(\omega_l|x_{mn})} \tag{23}$$

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

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 $\mathcal{B}_c$ and $\mathcal{B}_f$ as the upper bounds of E_c and E_f respectively. We have the following equations

$$E_c \leq \frac{\sum_{m=1}^{M}\beta_m^l|e_m^l|}{\sum_{m=1}^{M}\beta_m^l\Pr(\omega_l|\vec{x}_m)} = \mathcal{B}_c \tag{24}$$

$$E_f \leq \frac{\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{mn}^l|\epsilon_{mn}^l|}{\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{mn}^l\Pr(\omega_l|x_{mn})} = \mathcal{B}_f \tag{25}$$

By analyzing the denominators and numerators of $\mathcal{B}_c$ in (24) and $\mathcal{B}_f$ in (25) respectively, it can be shown that $\mathcal{B}_f$ is smaller than $\mathcal{B}_c$, which is elaborated as follows.

We first consider the denominators of $\mathcal{B}_c$ in (24) and $\mathcal{B}_f$ in (25). According to (2) and the normalization constraint, the denominator in (24) can be approximated by

$$\begin{aligned} \sum_{m=1}^{M}\beta_m^l\Pr(\omega_l|\vec{x}_m) &= \Pr(\omega_l)\sum_{m=1}^{M}\beta_m^l(1+\delta_m^l) \\ &\approx \Pr(\omega_l) * M \end{aligned} \tag{26}$$

Similarly, the denominator in (25) can be approximated by

$$\sum_{m=1}^{M}\sum_{n=1}^{N_m}\gamma_{mn}^l\Pr(\omega_l|x_{mn}) \approx \Pr(\omega_l) * \sum_{m=1}^{M}N_m \tag{27}$$

```
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 $\mathcal{C}_1^1, \cdots, \mathcal{C}_M^L$;
         Classifier dependency weight $\beta$;

for $m = 1, \cdots, M$
    for $j = 1, \cdots, J$
        Get feature vector $\vec{x}_m^j = \mathcal{D}_m(\mathcal{O}_j)$;
    endfor;
endfor;
Train classifiers for the $M$ descriptors, giving confidence
on the $L$ classes, $\mathcal{C}_m^l$;
for $l = 1, \cdots, L$
    for $m = 1, \cdots, M$
        for $j = 1, \cdots, J$
            Compute hypothesis $h_m^{j,l} = \mathcal{C}_m^l(\vec{x}_m^j) - \frac{1}{L}$;
        endfor;
    endfor;
endfor;
Solve the optimization problem (16) to obtain $\beta$;
return $\beta$ and $\mathcal{C}_1^1, \cdots, \mathcal{C}_M^L$.
```

Fig. 1. Algorithm 1: The training procedure of LCDM.

Since $1 \ll N_m$, we get

$$\sum_{m=1}^{M} \beta_m^l \Pr(\omega_l|\vec{x}_m) \ll \sum_{m=1}^{M} \sum_{n=1}^{N_m} \gamma_{mn}^l \Pr(\omega_l|x_{mn}) \qquad (28)$$

This equation shows that the denominator in $\mathcal{B}_c$ is much smaller than that in $\mathcal{B}_f$. On the other hand, we compare the numerators of $\mathcal{B}_c$ in (24) and $\mathcal{B}_f$ in (25). It is reported in [10] 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. $|e_t^k| \gg |\epsilon_{mn}^l|$. If $|e_t^k| > 2N_m|\epsilon_{mn}^l|$, according to the range and normalization constraint, the numerator in $\mathcal{B}_c$ is larger than that in $\mathcal{B}_f$, i.e.

$$\sum_{m=1}^{M} \beta_m^l |e_m^l| > \sum_{m=1}^{M} \sum_{n=1}^{N_m} \gamma_{mn}^l |\epsilon_{mn}^l| \qquad (29)$$

With (28) and (29), we have $\mathcal{B}_f \ll \mathcal{B}_c$. 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.

3.5 Remarks

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

```
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 $\mathcal{P}_{11}^1, \cdots, \mathcal{P}_{MN}^L$;
         Feature dependency weight $\gamma$;

for $m = 1, \cdots, M$
    for $j = 1, \cdots, J$
        Get feature vector $\vec{x}_m^j = \mathcal{D}_m(\mathcal{O}_j)$;
    endfor;
endfor;
Estimate the posterior distributions $\mathcal{P}_{mn}^l$ of $L$ classes
given $x_{mn}$;
for $l = 1, \cdots, L$
    for $j = 1, \cdots, J$
        for $m = 1, \cdots, M$
            for $n = 1, \cdots, N_m$
                Compute $p_{mn}^{j,l} = \mathcal{P}_{mn}^l(x_{mn}^j) - \frac{1}{L}$;
            endfor;
        endfor;
    endfor;
endfor;
Solve the optimization problem (15) to obtain $\gamma$;
return $\gamma$ and $\mathcal{P}_{11}^1, \cdots, \mathcal{P}_{MN}^L$.
```

Fig. 2. Algorithm 2: The training procedure of LFDM.

- LCDM is an "optimal" classifier-level fusion method in the sense of classifier dependency modeling given in Section 3.1. The training procedure of LCDM is described in Fig. 1.
- LFDM is an "optimal" feature-level combination method in the sense of feature dependency modeling given in Section 3.2. The algorithmic procedure of LFDM is shown in Fig. 2.
- Considering $h_m^{j,l}$ in (18) and $p_{mn}^{j,l}$ in (17) as classification scores, the derived formulations (18) and (17) look like the objective function of LP-B in [5]. Though the goal and the starting points of this paper are different from boosting methods, we come up with a similar optimization problem with different constraints ($0 \leq \beta_m^l \leq 2$ in classifier level or $0 \leq \gamma_{mn}^l \leq 2$ in feature level). Boosting methods aim at finding the correct weighting for classifier combination, while our objective is to model the feature dependency. Without the range constraint $0 \leq \beta_m^l \leq 2$ or $0 \leq \gamma_{mn}^l \leq 2$, the dependency weight β_m^l or γ_{mn}^l can be a large value, so that the derivation from (6) to (7), or from (12) to (14) is invalid. On the other hand, this causes the posterior probability in product form given as (6) or (12) be negative. Thus, the constraint $0 \leq \gamma_{mn}^l \leq 2$ is very important for dependency model. This can be observed from the experimental results.
- Compared to the dependency modeling technique by

estimating the joint probability [9], 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 the joint distribution estimation in high dimensions.

- Compared to the combination rule under dependent normal (DN) assumption [11], LCDM and LFDM are derived without any assumption on classifier and feature distribution. In many practical applications, data may not follow normal distribution, so our methods could have better performance in the non-normal cases.
- Compared to the linear combination methods, LP-Boost [5] [12] and MKL [15] [16] [17], our methods are derived from a probabilistic framework and model dependency explicitly.
- Compared to feature level fusion method, MKL [15] [16] [17], 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, but the proposed LFDM will not suffer from this problem.

4 Experiments

In this section, we evaluate the two proposed methods, namely Linear Classifier Dependency Modeling (LCDM) and Linear Feature Dependency Modeling (LFDM), with synthetic data as well as four real image/video databases. The real image/video databases are Oxford Flower [25], Digit [26], Weizmann [27] and KTH [28]. First, we compare the proposed LCDM with state-of-art classifier combination methods with synthetic data and the Flower database in Sections 4.1 and 4.2 respectively. Then, we evaluate LCDM with different types of classifiers and LFDM with different probability estimation methods on the Digit database in Section 4.3. After that, we compare the best performances with LCDM and LFDM on Weizmann and KTH action databases in Section 4.4. Finally, analysis on the dependencies between classifiers or features and their relationship with the learning results is presented in Section 4.5.

4.1 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 [11] [29]. In the experiments, four types of classifier distributions, namely Independent Normal (IndNormal), Dependent Normal (DepNormal), Independent Non-Normal (IndNonNor), Dependent Normal (DepNonNor) are used to evaluate 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 [30], i.e. $\Pr(+|s_m)$ and $\Pr(-|s_m)$, where s_m represents the score.

Table 1 shows the mean recognition rates and standard deviations of the six combination methods with different data distributions. From Table 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 IN and DN are derived under these specific assumptions. For the non-normal distribution, the proposed LCDM outperforms all other methods.

4.2 Results on Oxford Flower Database

Results with synthetic data show that the proposed dependency modeling method, LCDM is effective. In this section, We evaluate the classifier combination methods on a real database.

Oxford Flower database contains 17 different types of flowers with 80 images per category [25]. Some example images are shown in Fig. 3. Seven different types of features including shape, color, texture, HSV, HoG, SIFT internal, and SIFT boundary, are extracted using the method reported in [25] [31]. Distance matrices of these features and three predefined splits of the database (17×40 for training, 17×20 for validation, and 17×20 for testing) are available on their website[1]. We follow the experiment settings in [5]. 5-fold cross validation (CV) in the training set is used to select the best kernel SVM classifier [32] for each feature. The parameter introduced in the soft margin SVM is selected from $C \in \{0.01, 0.1, 1, 10, 100, 1000\}$. The CV outputs of the SVMs are used to train the weights for classifier combination. The best parameter ν is selected from $\{0.05, 0.1, \ldots, 0.95\}$ on the validation set with the Kernel matrices $\exp\left(-d(x, x')/\lambda\right)$, where d is the distance and λ is the mean of pairwise distances.

Table 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 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

1. http://www.robots.ox.ac.uk/~vgg/data/flowers/17/index.html

Method / Test	Sum [6]	LPBoost [12]	LP-B [5]	IN [11]	DN [11]	LCDM
IndNormal	95.34±0.71	96.81±0.48	97.51±0.48	**97.67±0.40**	97.56±0.42	97.66±0.46
DepNormal	86.44±1.16	95.29±0.85	93.17±1.30	92.52±0.98	**95.64±0.89**	93.88±0.93
IndNonNor	74.67±1.50	90.10±1.61	89.00±0.08	84.80±1.65	91.41±1.38	**93.00±0.07**
DepNonNor	63.33±0.89	68.95±1.30	69.37±1.90	65.46±0.92	69.84±1.35	**72.14±1.52**

TABLE 1
Mean accuracy (%) and standard deviation on synthetic data.

Fig. 3. Example images from Flower database [25]. Images from the same columns are from the same classes.

Single feature		Classifier Combination	
feature	accuracy	method	accuracy
Color	61.14±1.81	Sum [6]	85.39±3.14
Shape	70.16±1.28	IN [11]	85.49±1.72
Texture	62.90±2.44	DN [11]	84.22±1.91
HSV	61.44±0.47	LPBoost [12]	82.65±0.82
HoG	58.94±4.14	LP-β [5]	85.50±3.00
SIFTint	70.40±1.43	LP-B [5]	85.52±2.37
SIFTbdy	59.37±3.40	LCDM	**86.27±2.43**

TABLE 2
Mean accuracy (%) and standard deviation on Oxford Flower database.

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 classifiers.

4.3 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 [26].

This database 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 [26] and available on the website[2]. Some samples of the pixel average feature are visualized in

2. http://archive.ics.uci.edu/ml/datasets/Multiple+Features

Fig. 4. Visualization of digits from 0 to 9 of the pixel average feature [26].

Fig. 4. In this experiment, we use 20 samples of each digit for train and retain the rest for testing. And this experiment has been repeated ten times.

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 on the left hand side in Table 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 LCDM can be used with different classifiers and improve the performance in most cases.

In the proposed LFDM for feature dependency modeling, the one-dimensional probability $\Pr(\omega_l|x^j_{mn})$ needs to be estimated. So we evaluate LFDM with different probability estimation methods, including kernel density estimation (KDE) via diffusion [30] and kernel smoothing density estimation (KSD) [33]. 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 3. It can be seen that the proposed LFDM outperforms the others and models feature dependency with different probability estimations. Another observation is that DN-F in feature level gets an obvious improvement compared to the other methods without explicit dependency modeling, while it is not the case with the classifier combination methods as shown on the left hand side in Table 3. This implies that dependency information is useful and important for the fusion in feature level.

4.4 Results on Human Action Databases

In this section, we give a detailed evaluation of the proposed methods on the two standard human action databases. It is important to point out that the main objective of this experiment is to evaluate the performance

Classifier Level Fusion					Feature Level Fusion		
	1-NN	3-NN	SVM C=0.1	SVM C=100		KDE [30]	KSD [33]
Sum [6]	93.76±0.72	**94.76±0.70**	94.82±0.60	96.23±0.66	Sum-F [6]	92.91±0.80	93.63±0.59
IN [11]	95.06±0.71	93.40±0.84	94.75±0.61	95.63±1.08	IN-F [11]	93.47±0.82	94.37±0.86
DN [11]	94.77±0.55	93.50±0.89	94.82±0.59	94.93±1.09	DN-F [11]	97.00±0.62	96.87±0.89
LPBoost [12]	94.88±0.64	94.18±0.71	89.71±2.31	96.41±0.41	LPBoost-F [12]	95.79±1.03	95.34±1.28
LP-B [5]	94.13±1.09	93.53±0.81	94.95±0.55	96.57±0.35	LP-B-F [5]	94.36±1.13	95.44±1.15
LCDM	**95.18±0.56**	94.46±0.71	**95.34±0.66**	**96.79±0.60**	LFDM	**97.05±0.60**	**97.09±0.62**

TABLE 3
Mean accuracy (%) and standard deviation on Multiple Feature Digit database.

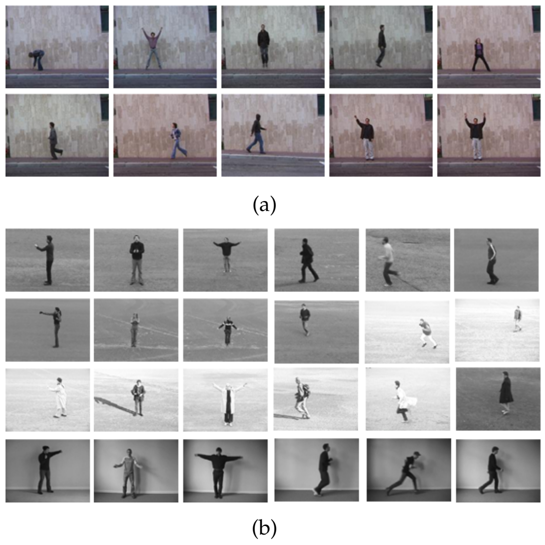


Fig. 5. (a) Example images from videos representing all the ten actions in Weizmann [27]. (b) Example images from videos in KTH [28]. All six actions and four scenarios are presented.

of the classifier level and feature level combination methods given a set of action features, but not state-of-art human action recognition algorithms.

Weizmann database contains 93 videos from nine persons, each performing ten actions [27]. Example images from the videos are shown in Fig. 5(a). Since the size of the database is small, we use nine-fold cross validation to evaluate the proposed method. In each validation, data from eight persons are used to train classifiers and the best combinations, as well as estimate the posterior probability.

There are 25 subjects performing six actions under four scenarios [28] in KTH database. Example images from the videos are shown in Fig. 5(b). We follow the common setting [28] to separate the video set into training (eight persons), validation (eight persons), and testing (nine persons) sets. Each classifier and combining classifiers 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 bag-of-features approach is adopted for these two databases to extract eight types of features for the videos. First, space-time interest points are detected by [34] from 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 current spatial blocks in the local cuboids are subtracted by those of the following spatial blocks over time. 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 [2], 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 [1], 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 [35] 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 [35] to form the feature vectors for classifier combination and feature fusion. Training data in Weizmann, and training and validation data in KTH are used to construct the vocabulary. The number of clusters for Weizmann and KTH databases is set as 100 and 600 respectively.

For the classifier combination methods, we select the best SVM classifier from $C \in \{0.01, 0.1, 1, 10, 100, 1000\}$ for each feature. The best parameter in MKL is also selected from the same set. 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.

The highest recognition accuracy of each feature is recorded in Table 4. Similar patterns on the two databases can be found in Table 4. The second feature with intensity difference over time gives the highest accuracies

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
Recognition accuracy (%) of each feature on Weizmann and KTH database.

	Weizmann		KTH	
	KDE [30]	KSD [33]	KDE [30]	KSD [33]
Sum-F [6]	57.78	48.89	78.70	50.00
IN-F [11]	68.89	67.78	69.91	77.31
DN-F [11]	77.78	76.67	–	–
LPBoost-F [12]	67.78	68.89	75.93	75.00
LP-B-F [5]	70.00	65.56	76.56	75.46
LFDM	**86.67**	**86.67**	**87.96**	**88.43**

TABLE 5
Recognition accuracies (%) of feature level fusion methods with different density estimation techniques on Weizmann and KTH database.

Method \ Dataset	Weizmann	KTH
Sum [6]	84.44	84.72
IN [11]	85.56	84.26
DN [11]	84.44	83.80
LPBoost [12]	83.33	83.33
LP-B [5]	84.44	85.19
LCDM	85.56	85.19
MKL [16] [17]	81.11	82.41
Sum-F [6]	57.78	78.70
IN-F [11]	68.89	77.31
DN-F [11]	77.78	–
LPBoost-F [12]	68.89	75.93
LP-B-F [5]	70.00	76.56
LFDM	**86.67**	**88.43**

TABLE 6
Recognition accuracy (%) of the best performance on Weizmann and KTH database.

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 top row in Table 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.

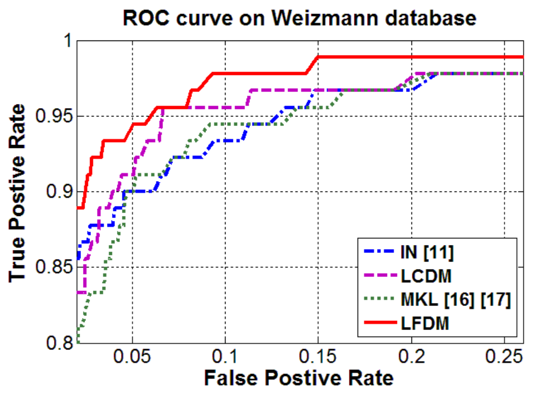
Similar to the experiments in Section 4.3, we use different density estimation techniques to evaluate the feature level fusion methods. The recognition accuracies on the two databases are shown in Table 5. Since the combination method developed under the dependent normal assumption [11] requires to solve a constraint quadratic problem numerically which has high dimensionality problem, and the feature dimension is high on KTH database, the result with DN-F is not available on this database. From Table 5, we can see that LFDM outperforms the other fusion methods much more, compared to the results on the Digit database. This convinces that LFDM works well even when the estimated probabilities deviate from the true ones.

The highest accuracies of all the fusion methods mentioned above as well as MKL[3] are summarized in Table 6. From Table 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 other hand, comparing DN-F and LFDM on Weizmann database and digit database in the previous section, 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 validates that LFDM works well under different distributions.

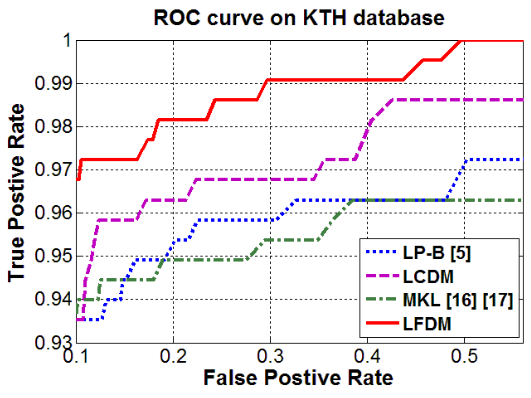
3. The results with SILP MKL [16] and simple MKL [17] are the same.

We also perform additional experiments for verification on Weizmann and KTH databases with best two classifier level methods and feature level fusion methods, respectively. The ROC curves are recorded and plotted in Fig. 6(a) and Fig. 6(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.

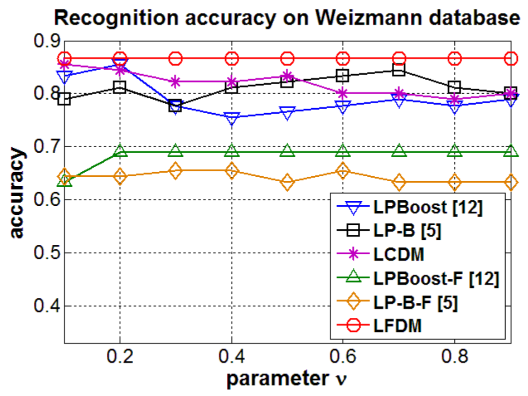
In the last experiment, we evaluate the sensitivity of parameter ν in linear programming based methods. Fig. 6(c) and Fig. 6(d) show the recognition accuracy of these methods with different values of ν. It can be observed that the feature level fusion methods are less sensitive to parameter changed on these two databases. The reason is as follows. Parameter ν is related to the tradeoff between misclassification error and the margin maximization [12]. Since the feasible solution set in feature level is much larger than that in classifier level, the misclassification error is more likely to be zero in feature level, and the tradeoff between misclassification error and the margin maximization becomes less important. On the other hand, the proposed LFDM is insensitive to parameter and obtains much better results, compared to the other feature level fusion methods. This is another advantage of the proposed model.



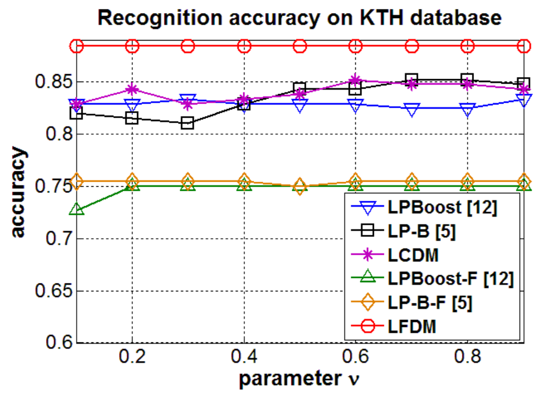
(a)



(b)



(c)



(d)

Fig. 6. (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.

4.5 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 [14], product-moment correlation coefficient [36] and distance correlation [37], 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 [37] 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 [37], we employ it to measure the dependencies between classifiers or features for the four real databases in this experiment. (Definition of the distance correlation $\mathcal{R}(\mathcal{Z}_1, \mathcal{Z}_2)$ between any tow random vectors $\mathcal{Z}_1$ and $\mathcal{Z}_2$ is given in Appendix C in the supplemental material.)

Since distance correlation $\mathcal{R}(\mathcal{Z}_1, \mathcal{Z}_2)$ is only for two random vectors, 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.

$$D_c = \frac{2}{M(M-1)} \sum_{l=1}^{L} \sum_{m_1=1}^{M} \sum_{m_2=m_1+1}^{M} \mathcal{R}(s^l_{m_1}, s^l_{m_2}) \quad (30)$$

$$D_f = \frac{2}{M(M-1)} \sum_{l=1}^{L} \sum_{m_1=1}^{M} \sum_{m_2=m_1+1}^{M} \mathcal{R}(\vec{x}^l_{m_1}, \vec{x}^l_{m_2}) \quad (31)$$

where random variables s^l_m and $\vec{x}^l_m$ represent the classifier score and feature vector for class l and feature m. Table 7 shows the dependency scores computed by (30) and (31) for the four real databases. For the Flower database, since distance matrices (instead of original features) are 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 8.

Comparing the classifier dependencies (D_c) in Table 7, we can see that score in the Flower database has the smallest dependency. This is in line with the recognition

	Flower	Digit	Weizmann	KTH
D_c	0.2253	0.4485	0.4740	0.4926
D_f	–	0.5360	0.9373	0.9108

TABLE 7
Dependency indicators in classifier level and feature level for real databases.

	1-NN	3-NN	SVM C=0.1	SVM C=100
D_c	0.2253	0.4485	0.4740	0.4926

TABLE 8
Dependency indicators with different classifiers in Digit database.

performance in Section 4.2 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 7 that both Weizmann and KTH action databases have large feature dependency scores (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. From Table 7 and Table 8, 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 R2 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 classifier level. Therefore, LFDM which models dependency in feature level, outperforms LCDM, a classifier level dependency modeling method.

5 Conclusion & Future Works

In this paper, we have designed and proposed a new framework for dependency modeling between features by linear combination 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 without any assumption on the data distribution. Experimental results demonstrate that IN and DN [11] 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 or/and 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. This indicates that feature-level fusion should be performed.

Although LFDM gives the best performances in our experiments, 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.

Acknowledgments

This project was partially supported by the Science Faculty Research Grant of Hong Kong Baptist University, National Natural Science Foundation of China Research Grants 61128009 and 61172136, and NSFC-GuangDong Research Grant U0835005. The authors would like to thank the reviewers for their helpful comments which improved the quality of this paper.

References

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

[2] 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.

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

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

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

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

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

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

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

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

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

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

[13] 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.

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

[15] 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.

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

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

[18] 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.

[19] 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.

[20] 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.

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

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

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

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

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

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

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

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

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

[30] 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.

[31] 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.

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

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

[34] 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.

[35] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

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

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

Andy J Ma received the B.Sc. degree in mathematics and applied mathematics, and the M.Sc. degree in applied mathematics from Sun Yat-Sen University, Guangzhou, China, in 2007 and 2009, respectively. He is currently working toward the Ph.D. degree in the Department of Computer Science, Hong Kong Baptist University, Kowloon, Hong Kong.

His research interests include pattern recognition, computer vision and machine learning. And he is now focusing on classifier fusion, feature combination and action recognition.

Pong C Yuen (M'93-SM'11) received his B.Sc. degree in Electronic Engineering with first class honours in 1989 from City Polytechnic of Hong Kong, and his Ph.D. degree in Electrical and Electronic Engineering in 1993 from The University of Hong Kong. He joined the Hong Kong Baptist University in 1993 and, currently is a Professor and Head of the Department of Computer Science.

Dr. Yuen was a recipient of the University Fellowship to visit The University of Sydney in 1996. He was associated with the Laboratory of Imaging Science and Engineering, Department of Electrical Engineering. In 1998, Dr. Yuen spent a 6-month sabbatical leave in The University of Maryland Institute for Advanced Computer Studies (UMIACS), University of Maryland at college park. He was associated with the Computer Vision Laboratory, CFAR. From June 2005 to January 2006, he was a visiting professor in GRAVIR laboratory (GRAphics, VIsion and Robotics) of INRIA Rhone Alpes, France. He was associated with PRIMA Group. Dr. Yuen was the director of Croucher Advanced Study Institute (ASI) on biometric authentication in 2004 and the director of Croucher ASI on Biometric Security and Privacy in 2007.

Dr. Yuen has been actively involved in many international conferences as an organizing committee and/or technical program committee member. He was the track co-chair of International Conference on Pattern Recognition (ICPR) 2006 and is the program co-chair of IEEE Fifth International Conference on Biometrics: Theory, Applications and Systems (BTAS) 2012. Currently, Dr. Yuen is an editorial board member of Pattern Recognition.

Dr. Yuen's current research interests include human face recognition, biometric security and privacy, and human activity recognition.

Jian-Huang Lai received his M.Sc. degree in applied mathematics in 1989 and his Ph.D. in mathematics in 1999 from SUN YAT-SEN University, China. He joined Sun Yat-sen University in 1989 as an Assistant Professor, where currently, he is a Professor with the Department of Automation of School of Information Science and Technology and vice dean of School of Information Science and Technology. His current research interests are in the areas of digital image processing, pattern recognition, multimedia communication, wavelet and its applications. He has published over 100 scientific papers in the international journals and conferences on image processing and pattern recognition, e.g. IEEE TNN, IEEE TIP, IEEE TSMC (Part B), Pattern Recognition, ICCV, CVPR and ICDM. Prof. Lai serves as a standing member of the Image and Graphics Association of China and also serves as a standing director of the Image and Graphics Association of Guangdong.