Table of Contents

1 Introduction

1.1 Motivation

Human action and gesture recognition aims at recognizing an action from a given video clip. This is an attractive research topic, which has been extensively studied over the years due to its broad range of applications from video surveillance to human machine interaction [1, 2]. Within this scope, a very important demand is independence to viewpoint. However, different viewpoints result in various human pose, background, camera motions, lighting conditions and occlusions. Consequently, recognition performance could be dramatically degraded under viewpoint changes.

To overcome this problem, a number of methods have been proposed. View independence recognition such as [3, 4, 5] [6] generally require a careful multi-view camera setting for robust joint estimation. View invariance approach [7, 8] is usually limited by inherent structure of view-invariant features. Recently, knowledge transfer technique is widely deployed for cross-view action recognition, for instance bipartite graph that bridge the semantic gap across view dependent vocabularies [9], AND-OR graph (MSTAOG) for cross-view action recognition [10]. To increase discriminant and informative features, view private features and shared features are both incorporated in such frameworks to learn the common latent space [11, 12]. While existing works for human action and gesture recognition from common viewpoints explored different deep learning techniques and achieved impressive accuracy. In most of aforementioned multi-view action recognition techniques, the features extracted from each view are usually hand-crafted features (i.e improved dense trajectories) [13, 12, 11]. Deep learning techniques, if used, handle knowledge transfer among viewpoints. Deployment of deep features in such frameworks for cross-view scenario is under investigation.

In parallel with knowledge transfer techniques, building a common space from different views has been addressed in many other works using multi-view discriminant analysis techniques. The first work of this approach was initiated by Canonical Component Analysis (CCA) that tried to find two linear transformations for each view [14]. Various improvements of CCA have been made to take non-linear transformation into account (kernel CCA) or to extend to multiple viewpoints (Multi-view CCA) [15]. Henceforth, different improvements have been introduced such as MULDA [16], MvDA [17], MvMDA [18], MCCDA [18]. All of these techniques try to build a common space from different views by maximizing between-classes while minimizing within-classes distances among views. However, most of these works are still experimented with static images, none of them have been explored with videos. Particularly, their investigation for the case of human action recognition is still actively under taken.

1.2 Objective

The main goal of this thesis is to develop an effective approach for designing a noise robust small footprint phoneme classifier system which is also computationally efficient. This will be accomplished

through two contributions. The first contribution is designing a system that has the ability to replace a software or hardware denoiser by including noisy data in the training process of the machine learning technique. This reduces the cost, memory requirement and increases the robustness of the speech engine and the processing speed. This is achieved by applying multi-condition training of the recognizer. The targeted noises are represented by injecting the corresponding noise into the training data enabling the model to simulate the noisy environments as it is trained with the clean and noisy data. Hence the denoising aspect is no longer a separate module but rather incorporated in the deep learning model itself. The second is to reach a good compromise between the clean and noisy data training, replacing the need for multiple noise models with a single one that provides accurate and robust results for both clean and noisy conditions. This also serves the purpose of reducing memory footprint and decreasing the pre-processing delay caused by the model selection step.

1.3 Thesis Outline

Deo thich viet voi

2 Technical Background

Viet gi vao day bay gio?

3 Proposed method

3.1 General framework

We propose a framework for cross-view action recognition as illustrated in Fig.3.1. It composes of two phases: training phase and recognition phase.

- **Training phase**: Suppose we have v views. The training phase consists of three main steps:
  1. *Feature extraction at separated view*: All training samples from each separated view will be passed though a feature extraction step. Different deep techniques will be investigated, including 2D CNNs based feature extraction combined with aggregation techniques and 3D CNNs based feature extraction, to output the final descriptor for each video sample. We call the features extracted at this step as features at single views or separated view features.
  2. *Common space construction*: This step builds a common feature space so that samples belonging to the same class will be close to each other even they are captured from different viewpoints. It takes all separated view features extracted from training set from the previous step and find a set of v linear transformations $(\omega_1, \omega_2, ..., \omega_v)$ that minimize within-class variation while maximizing between-class variation of features in the projected space (common space).
  3. *Training classifier*: Once v transformations have been computed, the projected features in the common space of each view will be utilized to train a single predictive model F (i.e. k-Nearest Neighbors - kNN).
- **Recognition phase**. Multi-view recognition consists of two following steps:
  1. *Feature extraction* We extract features of the testing sample x_j from the view V_j. This feature will be projected in the pre-built common space by the corresponding transformation ω_j. The projected feature is denoted as $y_j = \omega_j^T * x_j$.
  2. *Class prediction*: The projected feature y_j will be passed into the classifier $F(y_j)$ that outputs the label of action.

3.2 Feature extraction at separated view using deep learning techniques

3.2.1 2D CNN based clip-level feature extraction

First, the ResNet-50 Convolution Neural Network [19] is used as a 2D CNN network to extract spatial features of each frame in video. Fig.3.2 illustrates the architecture of ResNet-50 which composes of

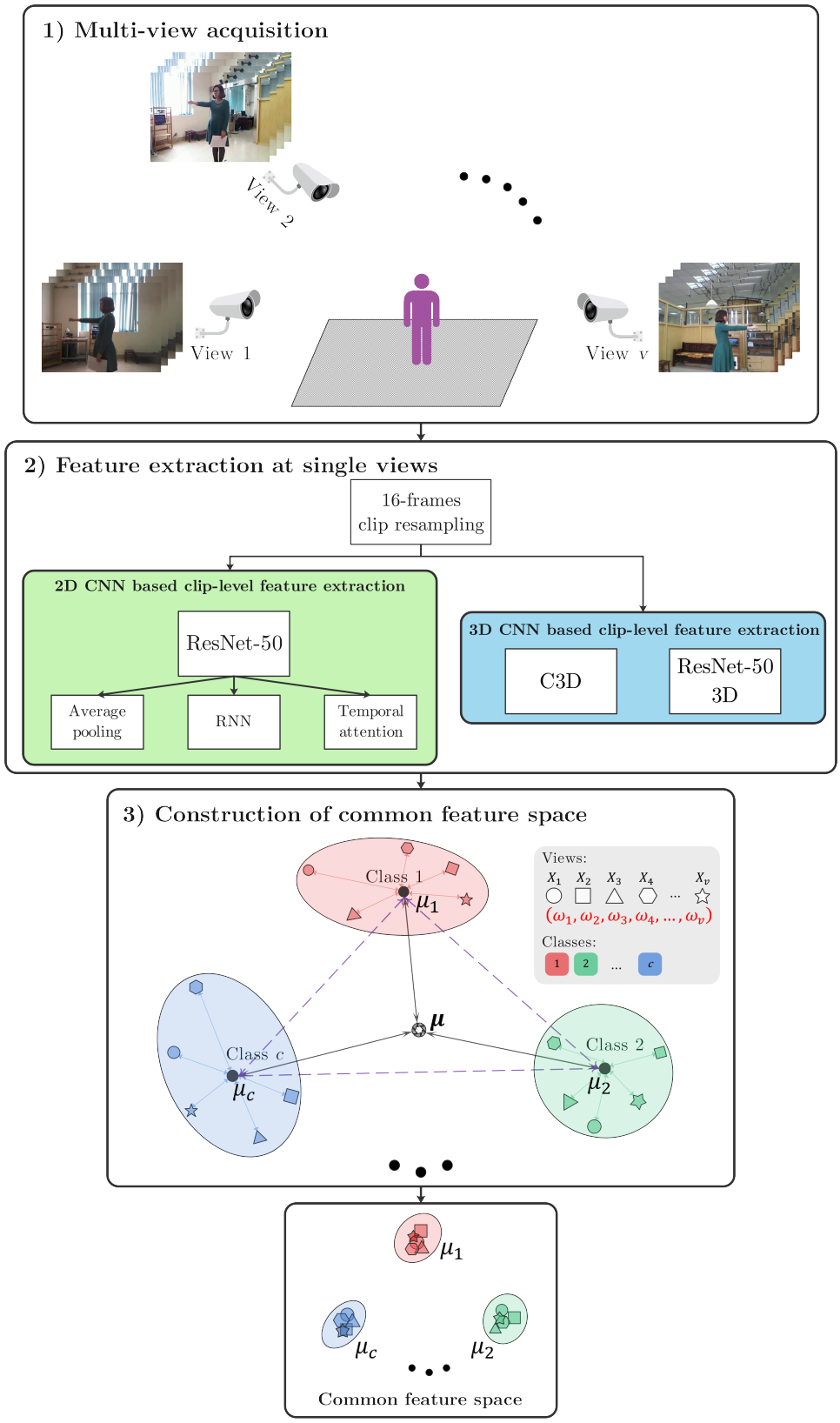


Figure 3.1: Proposed framework for building common feature space with pairwise-covariance multi-view discriminant analysis (pc-MvDA)

five convolutional blocks stacked on top of each other. The network is pre-trained on ImageNet then fine-tuned using training sets described in Section **??**. Deep residual features are extracted from the output of the last convolutional block of the network which is a 2048-D feature vector. We

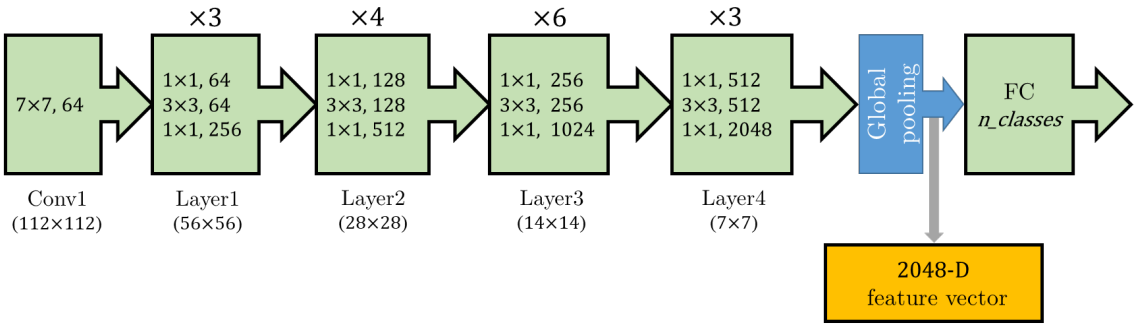


Figure 3.2: Architecture of ResNet-50 utilized in our work for feature extraction at each separated view

then aggregate frame-level features to create video-level features. In this work, we implement three temporal modeling techniques: 1) average pooling (AP); 2) recurrent neural network (RNN) and 3) temporal attention (TA). Fig.3.3 illustrates three techniques.

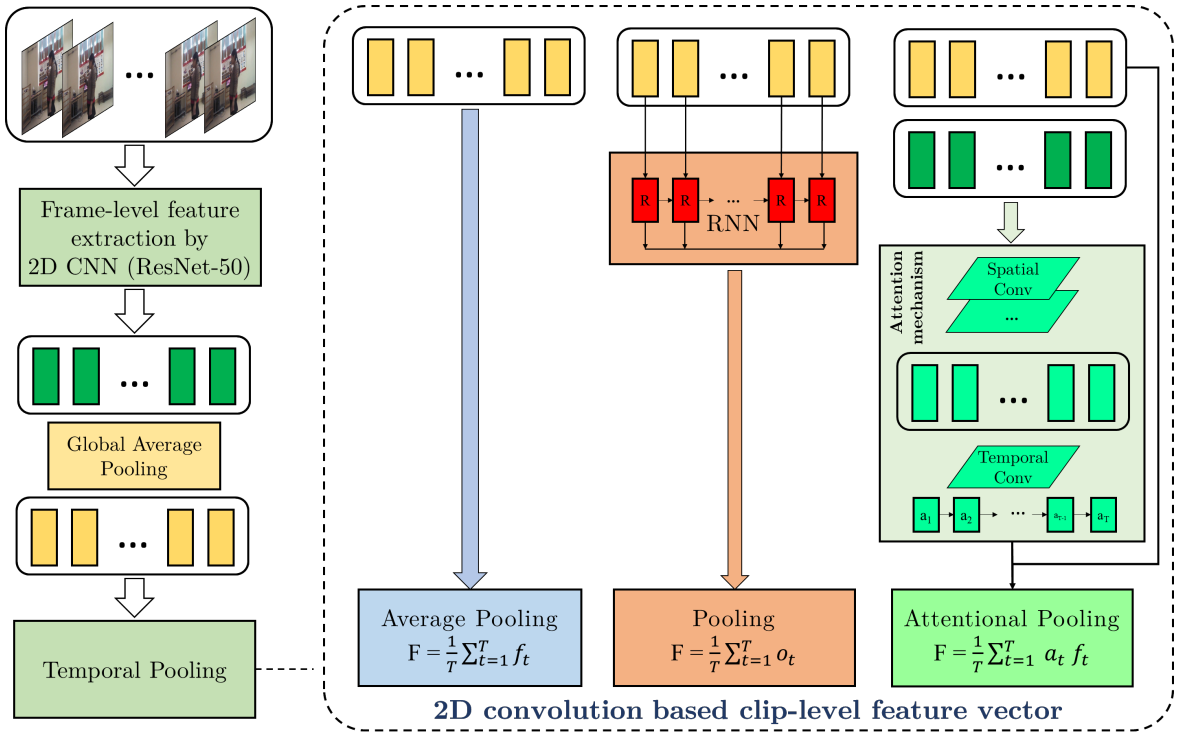


Figure 3.3: Three pooling techniques: Average Pooling (AP), Recurrent Neural Network (RNN) and Temporal Attention Pooling (TA)

Average Pooling - AP: Let f be the video-level feature, f_t be the frame-level feature at time t, T be the number of frames in video. Average pooling technique simply averages all frame-level features

uniformly to create the video-level feature:

$$f = \frac{1}{T}\sum_{t=1}^{T} f_t \tag{3.1}$$

As a frame-level feature is a 2048-D vector, the video-level feature is of the same dimension.

Recurrent Neural Network - RNN: An RNN cell encodes a t^{th} frame-level feature at time t of the sequence and passes the hidden state h_t into the next time step. In our work, a cell is a LSTM (Long Short Term Memory). The RNN is a single-layer with T cells. Each cell outputs a 512-D feature vector that contains information of the current frame and the previous ones. We aggregate a sequence of frame-level features into a video-level feature f by calculating the average of the RNN outputs $o_t = R(f_t, o_{t-1}); t \in [1; T]$.

$$f = \frac{1}{T}\sum_{t=1}^{T} o_t \tag{3.2}$$

Temporal Attention - TA: In the above pooling techniques, frame-level features are equally aggregated. In reality, some frames may have more important roles than remaining ones in recognizing an action. In temporal attention model, we learn a weight a_t for each frame f_t and apply an attention weighted average on the sequence of frame-level features as follows.

$$f = \frac{1}{T}\sum_{t=1}^{T} a_t f_t \tag{3.3}$$

To learn the weights $a_t, t \in [1, T]$, we adopt the attention generation network proposed by Jiyang Gao et al. [20]. The network takes a sequence of frame-level features $[T, w, h, 2048]$, each is tensor extracted from the last convolution layer of ResNet-50. The network architecture consists of two main components: Spatial Convolution and Temporal Convolution. First, a conv layer with shape $\{w, h, 2048, d_t\}$ is applied, then we get a d_t-dimensional feature for each frame of the clip (d_t = input channel). Then we apply a temporal conv layer $\{3, d_t, 1\}$ on these frame-level features to generate temporal attentions s_c^t. Once we have s_c^t, the final attention score a_t is computed by Softmax function:

$$a_t = \frac{e^{s_c^t}}{\sum_{k=1}^{T} e^{s_c^k}} \tag{3.4}$$

3.2.2 3D CNN based clip-level feature extraction

3D convolution network architectures is increasingly concerned with video based problems. In this paper, we deploy two 3D CNN architectures: C3D [21] and ResNet-50 3D [22].

ResNet 3D: ResNet-50 3D adopts 3D convolution kernels with ResNet-50 architecture. We apply transfer learning by using Kinetics pre-trained weights [23]. The architecture of ResNet-50 3D network is described in the Fig. 3.4. It has similar architecture as the ResNet-50 but the convolution is 3D operation.

C3D: 3D deep convolution neural network, which was introduced in [24] and has been shown to be very efficient for action recognition tasks. C3D takes input as an image sequence instead of a static

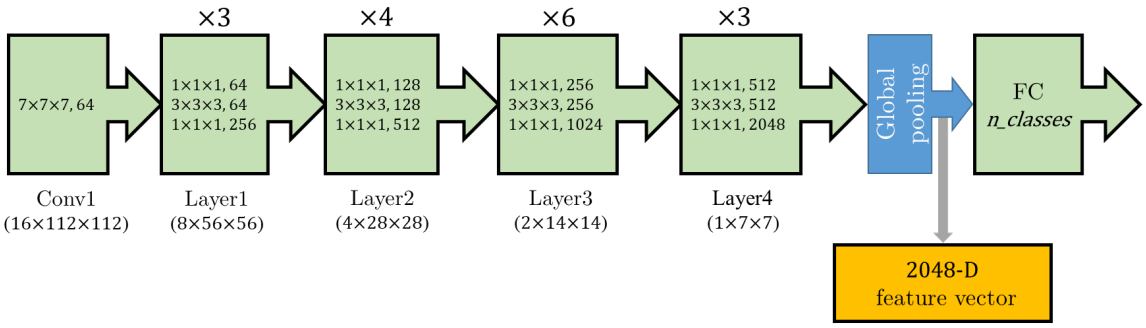


Figure 3.4: Architecture of ResNet-50 3D utilized in our work for feature extraction

image, computes the 3D convolution on each 3D cubes from video clip. By doing so, C3D captures both spatial and temporal characteristics of action at the same time. A C3D network contains 8

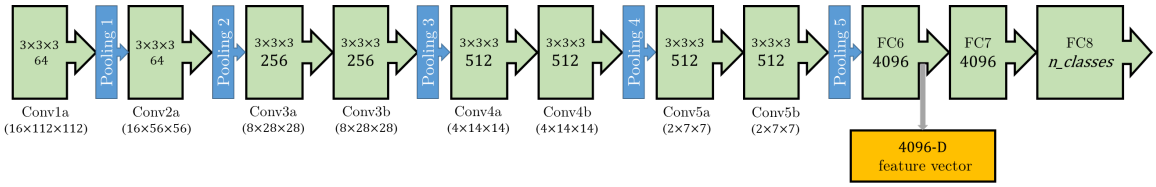


Figure 3.5: Architecture of C3D utilized in our work for feature extraction

convolution, 5 max-pooling and 2 fully connected layers as illustrated in Fig. 3.5. We utilize the network pre-trained on Sports-1M and fine-tuned on Kinetics dataset. To apply in our framework of hand gestures, we apply transfer learning technique on each data stream. Detail of training the network will be presented in section **??**. The feature vector of 4096 dimensions extracted from FC6 layer will be served for training and testing classifiers in further steps.

3.3 Construction of common feature space

3.3.1 Brief summary of Multi-view Discriminant Analysis

Suppose that actions belonging to c classes are observed from v views, the number of samples from the j^{th} view of the i^{th} class is n_{ij}. We define $X = \{\boldsymbol{x}_{ijk} | i = (1, .., c); j = (1, .., v); k = (1, .., n_{ij})\}$ as samples from v views where $\boldsymbol{x}_{ijk} \in R^{d_j}$ is the k^{th} sample from the j^{th} view of the i^{th} class, d_j is the dimensions of data at the j^{th} view. Here $\boldsymbol{x}_{ijk}$ is a feature vector extracted from the k^{th} sample from the j^{th} view of the i^{th} class. Different methods for features extraction from single view have been presented in previous sections.

Multi-view discriminant analysis (MvDA) MvDA was an extension of LDA for multi-view scenario [17]. It tries to determine a set of v linear transformations to project all action samples from each view $j = (1, .., v)$ to a common space. The projection results of X on the common space is denoted by $Y = \{\boldsymbol{y}_{ijk} = w_j^T \boldsymbol{x}_{ijk} | i = (1, .., c); j = (1, .., v); k = (1, ..., n_{ij})\}$. The common space is built by maximizing the between-class variation $\boldsymbol{S}_B^y$ while minimizing the within-class variation $\boldsymbol{S}_W^y$ from all

views. $\boldsymbol{S}_B^y$ and $\boldsymbol{S}_W^y$ are computed as follows:

$$\boldsymbol{S}_W^y = \sum_{i=1}^{c}\sum_{j=1}^{v}\sum_{k=1}^{n_{ij}}(y_{ijk}-\boldsymbol{\mu}_i)(y_{ijk}-\boldsymbol{\mu}_i)^T \tag{3.5}$$

$$\boldsymbol{S}_B^y = \sum_{i=1}^{c} n_i(\boldsymbol{\mu}_i-\boldsymbol{\mu})(\boldsymbol{\mu}_i-\boldsymbol{\mu})^T \tag{3.6}$$

where $\boldsymbol{\mu}_i = \frac{1}{n_i}\sum_{j=1}^{v}\sum_{k=1}^{n_{ij}}\boldsymbol{y}_{ijk}$ is the mean of all samples of the i^{th} class from all views in the common space; $\boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{c}\sum_{j=1}^{v}\sum_{k=1}^{n_{ij}}\boldsymbol{y}_{ijk}$ is the mean of all samples of all classes from all views in the common space; $n = \sum_{i=1}^{c} n_i$ is the total data samples from all views. Then the objective function is formulated by a Rayleigh quotient:

$$(\boldsymbol{\omega}_1^*, \boldsymbol{\omega}_2^*, ..., \boldsymbol{\omega}_v^*) = \underset{\boldsymbol{\omega}_1,\boldsymbol{\omega}_2,...,\boldsymbol{\omega}_v}{\operatorname{argmax}} \frac{trace(S_B^y)}{trace(S_W^y)} \tag{3.7}$$

According to [25], the optimization problem could be analytically solved through generalized eigenvalue decomposition.

In [25], the authors observed that as multiple views correspond to the same objects, there should be some correspondence between multiple views. They then introduce a view consistency constraint into the objective function. The method is called as MvDA-vc. In this paper, we will also compare with MvDA-vc but our method improves the MvDA, so we will not detail MvDA-vc in this section.

3.3.2 Pairwise-covariance Multi-view Discriminant Analysis pc-MvDA

MvDA emphasizes on finding a common space with minimal within-class variation while the distances between class means and global mean are jointly maximized. However, the distances between some pairs of classes can be disregarded. In this paper, to obtain this property, we modified MvDA with reformulated between and with-in class scatter matrices terms in a pairwise manner. First, let us define a new inter-class scatter matrix formula that takes paired distance into account:

$$\boldsymbol{S}_B^y = \sum_{a=1}^{c}\sum_{b=a+1}^{c}(\mu_a-\mu_b)(\mu_a-\mu_b)^T \tag{3.8}$$

where a, b are two distinct classes. For each pairs of class a and class b, the between covariance is calculated as:

$$\boldsymbol{S}_{Bab}^y = (\mu_a-\mu_b)(\mu_a-\mu_b)^T \tag{3.9}$$

The intra-class scatter matrix $\boldsymbol{S}_W^y$ of MvDA is calculated as simple summation of all covariance matrices of each i^{th} class, $i = 1, ..., c$:

$$\boldsymbol{S}_{W\,i}^y = \sum_{j=1}^{v}\sum_{k=1}^{n_{ij}}(y_{ijk}-\mu_i)(y_{ijk}-\mu_i)^T \tag{3.10}$$

Mathematically, it assumes data samples from each class among all views are identically Gaussian distributed, and contribute evenly to the minimization of intra-class scatter. In reality, it is hardly

the case as data variations of samples from different classes and different view points are usually drastically diverse and may also have different dimensions.

To better represent the distribution of data, pc-MvDA uses a paired intra-scatter matrix which is denoted as

$$\boldsymbol{S}_{W\,ab}^{y} = \beta \frac{n_a \boldsymbol{S}_{W\,a}^{y} + n_b \boldsymbol{S}_{W\,b}^{y}}{n_a + n_b} + (1-\beta)\,\boldsymbol{S}_{W}^{y} \tag{3.11}$$

where $0 \leq \beta \leq 1$ is a hyper-parameter for regularization between the original global intra-covariance $\boldsymbol{S}_{W}^{y}$ and the novel two local class covariances $\boldsymbol{S}_{W\,a}^{y}$ and $\boldsymbol{S}_{W\,b}^{y}$. This formulation is closer to the value of covariances of both classes than the standard intra-covariance.

Instead of solving the vanilla generalized eigen value problem for the whole dataset, we split it into sub-problems for each pairs of class a and b, in which we define the objective distance to be minimized between two corresponding classes. **Difference of our proposed method compared to pairwise LDA (pc-LDA)** In pc-LDA [26], the pairs of a and b classes are regarded as two Gaussian distributions $\mathcal{N}_a(\mu_a, \boldsymbol{S}_{W\,a}^{y}), \mathcal{N}_b(\mu_b, \boldsymbol{S}_{W\,b}^{y})$ and the objective distance between two classes is defined as their Kullback-Leibler divergence [27]:

$$D_{KL}\left(\mathcal{N}_a \parallel \mathcal{N}_b\right) = \frac{1}{2}\left(\mu_a - \mu_b\right)^T \left(\boldsymbol{S}_{W\,ab}^{y}\right)^{-1} \left(\mu_a - \mu_b\right), \tag{3.12}$$

Then the final objective is properly weighted to focus on classes with more samples:

$$\min_{\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \ldots, \boldsymbol{\omega}_v} J = \sum_{a=1}^{c} \sum_{b=a+1}^{c} \frac{n_a n_b}{\left[2 D_{KL}\left(\mathcal{N}_a \parallel \mathcal{N}_b\right)\right]^q} \tag{3.13}$$

here $q \geq 1$ is a hyper-parameter that controls how much the pairs of classes with smaller objective distances are biased over the others.

In our observation, the minimization of KL divergence for each pairs of classes a and b can be substituted by a generalized eigenvalue problem with the pairwise $\boldsymbol{S}_{W\,ab}^{y}$ and the $\boldsymbol{S}_{B\,ab}^{y}$ we defined. Although these sub-problems do not have an unified analytical solution to be solved concurrently, the criteria can be formulated in various differentiable ways like in [28] so we can use gradient descent algorithm:

$$J_1 = \frac{tr(S_B)}{tr(S_W)}; \quad J_2 = tr\left(\frac{S_B}{S_W}\right); \quad J_3 = tr\left|\left|\frac{S_B}{S_W}\right|\right|; \quad J_4 = \frac{det(S_B)}{det(S_W)} \tag{3.14}$$

We choosed the former Fisher loss as the ratio of scalar values is computationally cheaper. Comparing to the objective of pc-LDA in Eq.(3.13), our proposed model is obviously more efficient since it contains only simple operators and the need to inverse a singular-prone matrix is negated. Therefore, it is better fit in the scenario where we want to train the model multiple iterations over multi-view high dimensional output.

Our final objective function of pairwise-covariance multi-view discriminant analysis is sum of all pairwise Fisher criteria with normalized weights:

$$\min_{\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \ldots, \boldsymbol{\omega}_v} J = \sum_{a=1}^{c} \sum_{b=a+1}^{c} \frac{n_a n_b}{n_{cc}^2} \left[\frac{trace\left(\boldsymbol{S}_{W\,ab}^{y}\right)}{trace\left(\boldsymbol{S}_{B\,ab}^{y}\right)}\right]^q \tag{3.15}$$

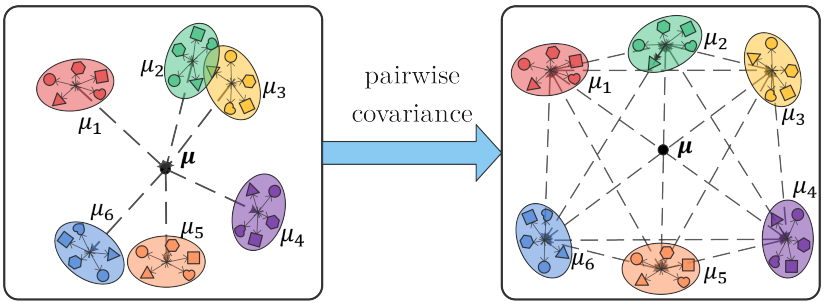
where n_{cc} is the number of common components as coined in [18] or simply number of samples in each view. This added normalization term assures that the objective does not depend on size of dataset.

Let's look at the Fig.3.1, the nominator of MvDA(-vc) $\boldsymbol{S}_B^y$ is the sum of all distances from mean of every class μ_i, $i = 1, ..., c$ to the mean of all classes μ. This helps to push class far away from the mean of all classes (the dotted green lines in Fig.3.1). However, it may not ensure that the classes will be far from each other (Fig.3.6a). In this work, we take the pairwise distance into account and integrate them in an multi-view framework (the dotted violet lines in Fig.3.1). Thanks to this, the classes will be more discriminated (Fig. 3.6b). The algorithm for solving pc-MvDA is presented in supplemental materials.

Illustrative examples: We illustrate the advantage of pc-MvDA on artificial datasets. Firstly, using function provided by scikit-learn library, we generate several isotropic Gaussian blobs equal to desired number of classes as data samples from one single view. To generate other views, we clone and apply translation, point-wise noise and rotation with random orthogonal group multiplication to the first view.

In Fig.3.7, MvDA and pc-MvDA results on 2D synthetic dataset of 180 data points in 2-D space. There are three classes, observed at three viewpoints. We show data distribution and 1-D projection results using MvDA and pc-MvDA respectively. We notice that data points of two classes (the first and the third classes) are close together in original space. When we project in common space by MvDA, these classes are still close together. In contrast, these classes are more discriminant when projected by pc-MvDA.

Similar to the first example, in the second example, we generate 300 data points of 5 classes observed from 3 different views (Fig.3.8). We observe that the five clusters of pc-MvDa common space contract strongly and become more separated as compared to the MvDA results (especially the first and the fourth classes).

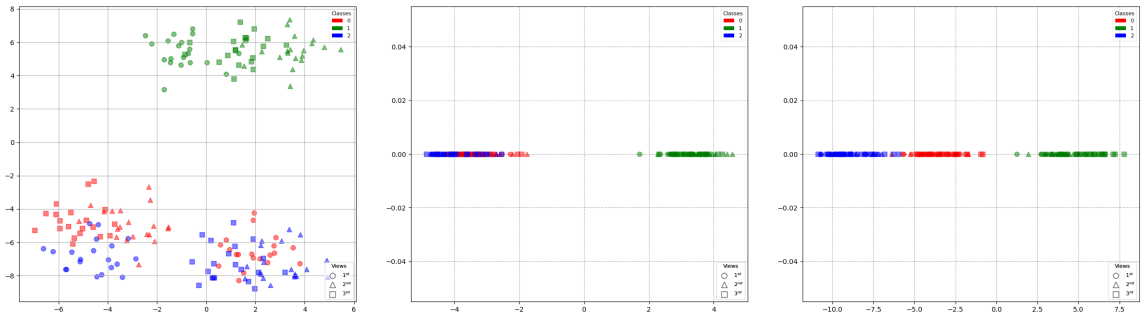


a) MvDA b) pc-MvDA

Figure 3.6: a) MvDA does not optimize the distance between paired classes in common space. b) pc-MvDA takes pairwise distances into account then distinguish better the classes.

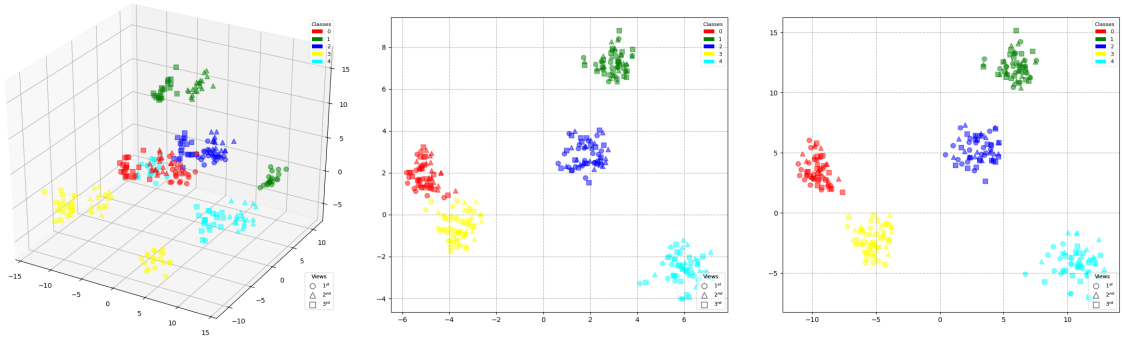
3.3.3 Algorithm to solve pc-MvDA

As our objective does not match any optimization form of generalized eigenvalue problems, we use gradient descent algorithm to solve pc-MvDA. This approach also allow the algorithm to update incre-



a) Original distribution b) MvDA c) pc-MvDA

Figure 3.7: A synthetic dataset of 180 data points, evenly distributed to 3 classes among 3 different views; a) 2-D original distribution; b) 1-D projection of MvDA; c) 1-D projection of pc-MvDA



a) Original distribution b) MvDA c) pc-MvDA

Figure 3.8: A synthetic dataset of 300 data points, evenly distributed to 5 classes among 3 different views; a) 3-D original distribution; b) 2-D projection of MvDA; c) 2-D projection of pc-MvDA

mentally even in higher dimensional data where the computations of matrix inversion and eigenvectors are not conducive. The derivative of Eq.(3.15) is:

$$\frac{\partial J\left(W\right)}{\partial W} = -\sum_{a<b}^{c} \frac{qn_a n_b}{{n_{cc}}^2 J_{ab}\left(W\right)^{q+1}} \frac{\partial J_{ab}\left(W\right)}{\partial W} \tag{3.16}$$

where $J_{ab}\left(W\right) = tr\left(\boldsymbol{S}^y_{B\,ab}\right)/tr\left(\boldsymbol{S}^y_{W\,ab}\right)$ is the Fisher loss of class pair a and b. Its gradient is computed using the funky trace derivative and quotient rule:

$$\frac{\partial J_{ab}\left(W\right)}{\partial W} = \frac{tr\left(\boldsymbol{S}^y_{W\,ab}\right) W^T \left(\boldsymbol{S}^x_{B\,ab} + {\boldsymbol{S}^x_{B\,ab}}^T\right) - tr\left(\boldsymbol{S}^y_{B\,ab}\right) W^T \left(\boldsymbol{S}^x_{W\,ab} + {\boldsymbol{S}^x_{W\,ab}}^T\right)}{tr\left(\boldsymbol{S}^y_{W\,ab}\right)^2} \tag{3.17}$$

The superscript x replacing y in scatter matrices notations means that they denote the pre-transformed version. With simple algebra, we can rewrite the formulas in matrix multiplication form, separating transformation vectors ω_j in a concatenated matrix W and a multi-view covariance matrix constructed from $v \times v$ cells, each cell S_{jk} stands for the covariance between view j and view k.

$$\boldsymbol{S}^y = W^T \boldsymbol{S}^x W = \begin{bmatrix} \omega_1^T & \omega_2^T & \cdots & \omega_v^T \end{bmatrix} \begin{bmatrix} S_{11} & S_{12} & \cdots & S_{1v} \\ S_{21} & S_{22} & \cdots & S_{2v} \\ \vdots & \vdots & \ddots & \vdots \\ S_{v1} & S_{v2} & \cdots & S_{vv} \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \vdots \\ \omega_v \end{bmatrix} \tag{3.18}$$

The complete algorithm to compute ∇J is given in Algorithm 1.

Algorithm 1: Computation of $\nabla J\left(W\right)$ (i.e. gradient of Eq.(3.15))

Input : W, $\{X,\ \mu\}$, q
Output: $\nabla J\left(W\right)$

$\boldsymbol{F} = 0$
Compute $\boldsymbol{S}^x_W$ according to Eq.(3.5)
for $a = 1$ **to** c **do**
  for $b = a + 1$ **to** c **do**
    Compute $\boldsymbol{S}^x_{B\,ab}$ according to Eq.(3.9)
    Compute $\boldsymbol{S}^x_{W\,ab}$ according to Eq.(3.11)
    Compute $\boldsymbol{S}^y_{B\,ab} = W^T \boldsymbol{S}^x_{B\,ab} W$
    Compute $\boldsymbol{S}^y_{W\,ab} = W^T \boldsymbol{S}^x_{W\,ab} W$
    Compute $J_{ab} = tr\left(\boldsymbol{S}^y_{B\,ab}\right)/tr\left(\boldsymbol{S}^y_{W\,ab}\right)$
    Compute ∇J_{ab} according to Eq.(3.17)
    $\boldsymbol{F} = \boldsymbol{F} + n_a n_b \nabla J_{ab} / J_{ab}^{q+1}$
  end
end
$\nabla J = -q\boldsymbol{F}/n_{cc}^2$

Output: ∇J

We do not impose any further constraint to the model. In our implementation of pc-MvDA in Pytorch, the computation of gradient is in fact handled by automatic differentiation. We choose Adam as our optimizer with learning rate set to 0.01. To have better starting point and mitigate variances, we initialize pc-MvDA with transformation learnt by MvDA.

4 Experiments

4.1 Datasets

In this work, we utilize three datasets for evaluating our algorithm. Two of which are benchmark datasets IXMAS [29], MuHAVi [30] and one dataset (namely MICAGes) is recorded by ourselves.

4.1.1 IXMAS dataset

The IXMAS dataset was built by Perception project. It contains 12 action classes recorded by 4 side view cameras and 1 top view camera. Each action is performed 3 times by 10 actors (5 males

Figure 4.1: Illustration of a frame extracted from an action observed by five camera viewpoints [29]

/ 5 females). During the time of writing this paper, the dataset is not publicly provided due to the privacy issues. We have utilized a subset containing samples from four side camera views (excluding the top down view) that we have downloaded previously with agreement of the data-set's authors. The comparison with SOTA methods on this dataset will be only taken from the four side view cameras.

4.1.2 MuHAVi dataset

The dataset MuHAVi is constructed and introduced by [30]. It is usually referred as MuHAVi-uncut because it contains full, raw videos of 17 actions, asynchronously captured by eight cameras that provide completely overlapping coverage of a rectangular action zone from different viewing directions. The actions are performed several times by 7 actors (5 males / 2 females). The 8^{th} camera whose data is not included in experiments of this paper due to absence of annotation.

4.1.3 MICAGes dataset

There does not exist a dataset dedicated to the evaluation of robustness of hand gesture recognition w.r.t. viewpoint changes. Therefore, in our work, we carefully design a dataset collected from multiple viewpoints in indoor environment with complex background. Our dataset consists of nine dynamic hand gestures which correspond to controlling commands of electronic home appliances. Each gesture

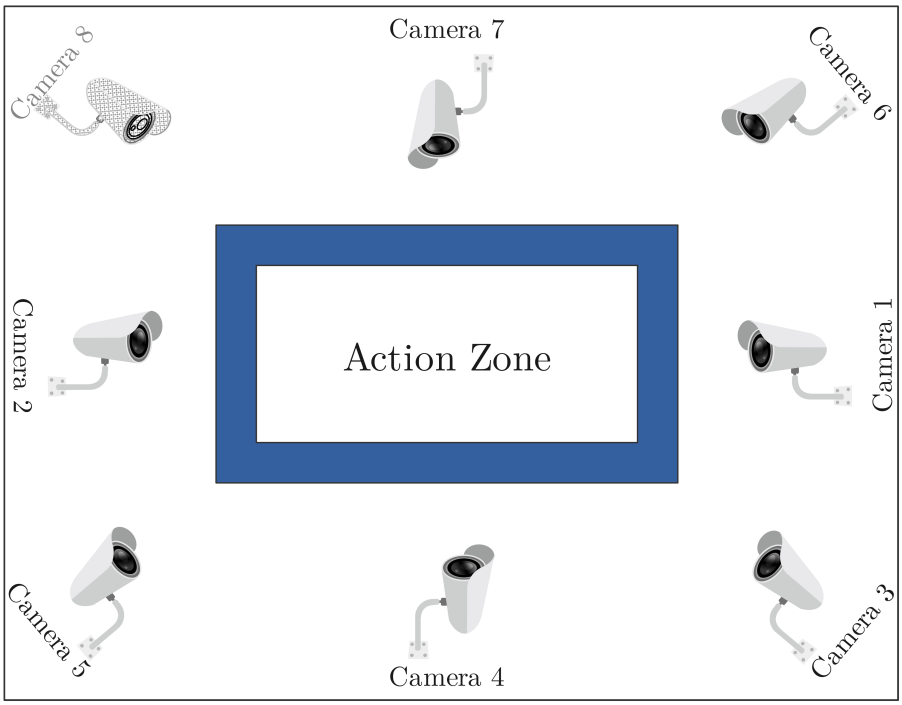


Figure 4.2: Environment setup to collect action sequences from 8 views [30].

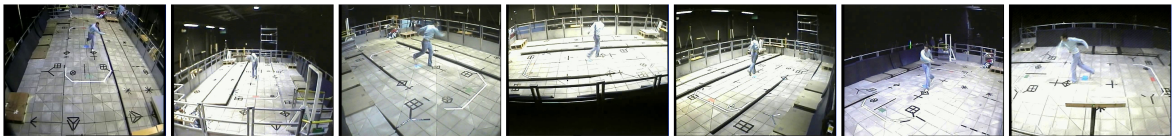


Figure 4.3: Illustration of a frame extracted from an action *punch* observed from Camera 1 to Camera 7.

is a combination of hand movement following a pre-defined direction and changing of hand shape in a natural manner.

Five Kinect sensors K1, K2, K3, K4, K5 are setup at five positions in a square simulation room of $16m^2$ (Fig. 4.4). This work aims to capture hand gestures under multiple viewpoints synchronously. The subjects are invited to stand at a fixed position approximately 2 meters in front of the center view. The Kinect sensors provide both RGB and Depth data captured at frame rate of 20 fps and resolution of 640 × 480, which allows us to capture a multi-view and multi-modal dataset.

Twelve participants (08 males and 04 females) are voluntary to perform gestures one after another, each gesture three times. Totally, the dataset contains 1620 (5 views × 09 gestures × 12 subjects × 3 times) dynamic hand gestures.

4.2 Evaluation protocol

We employ the leave-one-actor out cross validation strategy represented in [31] for small datasets and compute average accuracy (%) for comparison. In order to evaluate our framework in terms of cross-view performance, we further combine with cross view validation. That means at a moment we only

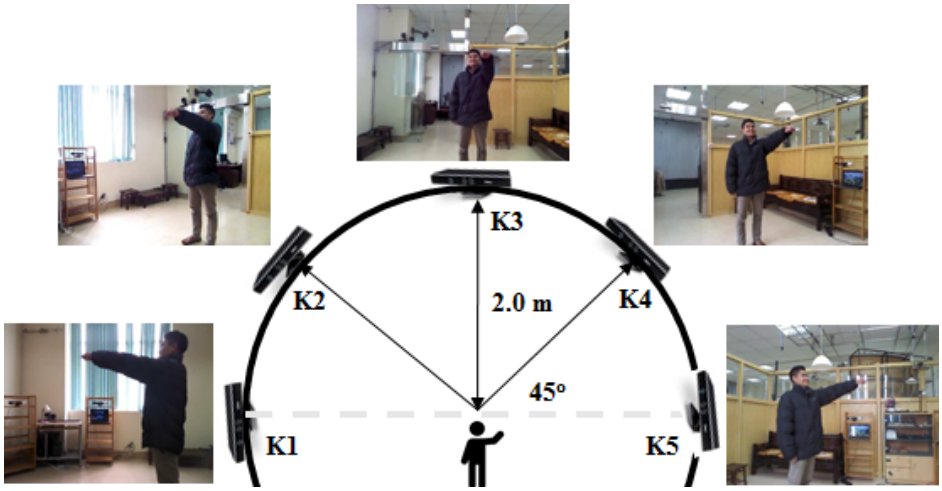


Figure 4.4: Environment setup to capture MICAGes dataset.

Figure 4.5: Illustration of a gesture belonging to the 6^{th} class observed from five different views.

pick two views to evaluate, each consists of a train and a test part defined by the leave-one-actor out strategy. Apart from our evaluation protocol, from now named "*protocol 1*", we also assess another "*protocol 2*" that is regularly used in the literature. The main difference is in second protocol, the notation of train or test is omitted after feature extraction stage, every features samples from single view are included to train the multi-view metrics, then the classifier is fit on common features of one view and tested on common features of another view. It is noticed that the first evaluation protocol is more challenging because the in the second protocol, the constructed common space is already fit on the whole multi-view dataset. As a result, the later predictive models are trained and evaluated on features generated by a severely over-fitted projector. The difference between two protocols are illustrated in Fig. 4.6.

The overall cross-view evaluation score is computed as average of all evaluation scores of each split, whose method of computation is described specifically in Fig.4.6:

$$acc_{ij} = \frac{\sum_{k=1}^{M} acc_{ij}^{(k)}}{M} \tag{4.1}$$

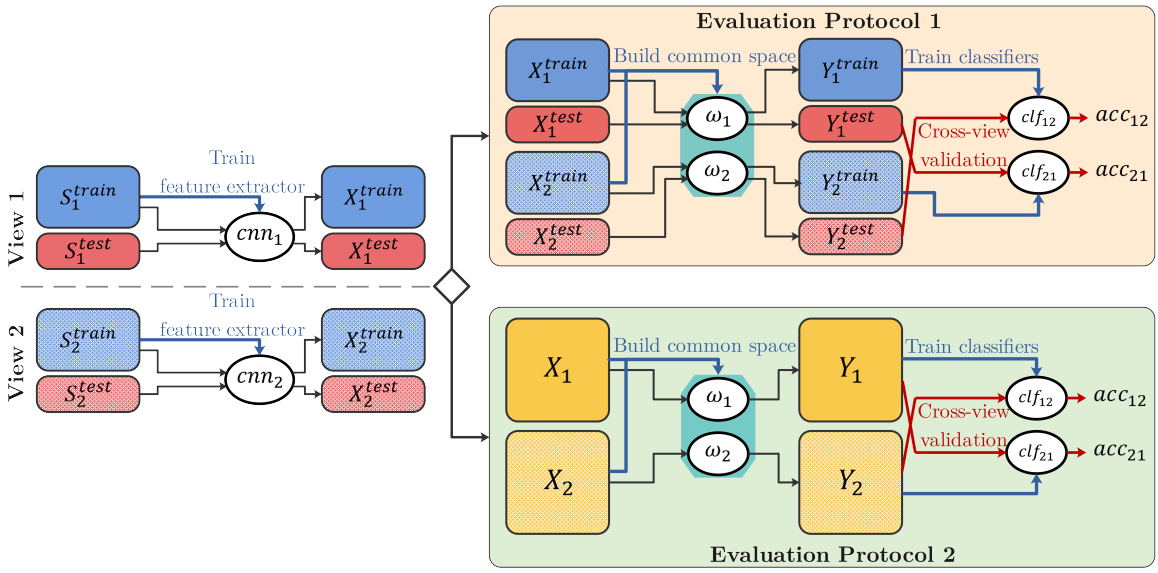


Figure 4.6: Two evaluation protocols used in our experiments.

where M is the number of actors. Both protocols finally generate a cross-view accuracy matrix as follows:

$$\begin{bmatrix} acc_{11} & acc_{12} & \cdots & acc_{1v} \\ acc_{21} & acc_{22} & \cdots & acc_{2v} \\ \vdots & \vdots & \ddots & \vdots \\ acc_{v1} & acc_{v2} & \cdots & acc_{vv} \end{bmatrix} \tag{4.2}$$

For eager expression, we define function accuracy $\left(Y, \tilde{Y}\right)$ as the accuracy score when fitting a classifier on Y features and evaluating its prediction on $\tilde{Y}$ features. Each cell can be expressed accordingly:

$$score_{jr}^{\textbf{protocol1}} = \left\langle \left\{ \text{accuracy} \left(Y_j^{train}, Y_r^{test}\right) \middle|\ j, r = (1, .., v) \right\} \right\rangle \tag{4.3}$$

$$score_{jr}^{\textbf{protocol2}} = \left\langle \left\{ \text{accuracy} \left(Y_j, Y_r\right) \middle|\ j, r = (1, .., v) \right\} \right\rangle \tag{4.4}$$

Multi-view strategy: As MvDA and its variants can theoretically scale for an arbitrary number of views, in order to fairly compare our algorithm with others that only apply for 2 views, we also introduce multi-view strategy. The key idea is that each strategy restricts the number of views participated in the training process of MvL algorithms.

In "*multi-view*" strategy, only one set of $W^* = \{\omega_1, \omega_2, ..., \omega_v\}$ is learnt and all separated view features are transformed $Y_j = \omega_j^T X_j,\ j = (1, ..., v)$ before computation of accuracy.

In "*cross-view*" strategy, we restrict each cell $score_{jr}$ of (4.2) must be calculated exclusively with the inclusion of only separated view features of those views j and r and not information of other views. Therefore, for each distinct combination jk, $W_{jr}^* = \{\omega_j, \omega_r\}$ is learnt to transform X_j and X_r. The diagonal of (4.2) is also ignored in this strategy.

Since there are many scores generated by our extra complicated cross validation process when we cycle through every actors, the final reported in this paper is average of all. Further, it is noticed that in existing works, the evaluation have been done mostly similar to our second evaluation protocol. To

be comparable with state of the art techniques, we take scores resulted from the second evaluation protocol.

4.3 Experimental results and discussions

4.3.1 Experimental results on IXMAS dataset

1) Cross-view validation: In this experiment, we train on data from one view (training view) and testing on data from another view (testing view). We compute accuracies for two evaluation protocols. Table. 4.1 shows comparative results of using different deep features (C3D, ResNet-50 3D, ResNet-50 RNN, ResNet-50 TA, ResNet-50 AP) and multi-view discriminant analysis techniques (MvDA, MvDA-vc and our proposed pc-MvDA).

With the first evaluation protocol, our proposed pc-MvDA, when combined with deep features, gives mostly best accuracy for both evaluation protocols. With C3D features, accuracy by pc-MvDA (88.43%) is 6.6% higher than MvDA (81.84%). pc-MvDA is even better than MvDA-vc (2.87%). pc-MvDA achieved higher accuracy (about 6% higher) than MvDA and MvDA-vc with ResNet-50 TA and ResNet-50 AP features. In average, pc-MvDA is 3.09% higher than MvDA and 1.4% higher than MvDA-vc.

Table 4.1: Cross-view recognition comparison on IXMAS dataset

Deep features	Protocol 1			Protocol 2		
	MvDA	MvDA-vc	pc-MvDA	MvDA	MvDA-vc	pc-MvDA
C3D	81.84	85.56	**88.43**	96.54	99.29	**99.98**
ResNet-50 3D	91.25	92.19	**92.89**	97.30	**99.73**	99.65
ResNet-50 RNN	75.51	76.12	**76.54**	99.99	99.71	**100**
ResNet-50 TA	79.44	80.58	**82.05**	99.33	99.34	**99.84**
ResNet-50 AP	77.00	79.04	**80.58**	99.68	99.81	**99.92**

For the second evaluation protocol, pc-MvDA almost always keeps better performance compared to MvDA and MvDA-vc. The recognition accuracy scores obtained by both pc-MvDA and MvDA-vc are nearly 100% for every kind of deep features. With C3D features and ResNet-50 3D features, pc-MvDA is 99.98% and 99.65%, which is 3.44% and 2.35% higher than the orginal MvDA (96.54% and 97.3%) respectively. In average, pc-MvDA is 1.31% better than MvDA and 0.3% better than MvDA-vc.

In terms of deep features, with the first evaluation protocol, ResNet-50 3D gives the best accuracy (92.89%) following by C3D (88.43%), ResNet-50 TA (82.05%), ResNet-50 AP (80.58%). The lowest accuracy obtained by ResNet-50 RNN is only 76.54%. It shows that ResNet-50 3D produces the most discriminative feature space.

2) Multi-view validation Table.4.3 shows multi-view recognition results. The conclusion is consistent with the case of cross-view evaluation: ResNet-50 3D is the best feature extractor. When it is combined with our proposed pc-MvDA, the framework gives the highest accuracy for the first protocol (92.82%), following by C3D (88.19%), ResNet-50 TA (81.49%), ResNet-50 AP(80.43%), and ResNet-50 RNN (76.47%). Both MvL algorithms give similar accuracies for the second evaluation protocols (nearly 100%). Again, our proposed pc-MvDA enhances the performance of MvDA by 2.43% for the

Table 4.2: Cross-view recognition results of different features on IXMAS dataset with pc-MvDA method. The result in the bracket are accuracies of using features C3D, ResNet-50 3D, ResNet-50 RNN, ResNet-50 TA, Restnet-50 AP respectively. Each row corresponds to training view (from view C0 to view C3). Each column corresponds to testing view (from view C0 to view C3)

Training \ Testing	C0	C1	C2	C3
C0	N/A	(90.7, **91.9**, 81.6, 84.6, 83.6)	(86.9, **93.9**, 78.0, 79.6, 78.1)	(89.9, **93.4**, 76.3, 82.8, 82.1)
C1	(86.6, **91.9**, 71.5, 81.1, 77.5)	N/A	(85.9, **94.2**, 78.6, 79.3, 80.3)	(88.9, **93.7**, 74.8, 82.3, 81.3)
C2	(87.6, **92.4**, 72.8, 81.8, 78.5)	(90.7, **91.7**, 80.6, 83.6, 82.6)	N/A	(89.4, **93.7**, 75.5, 82.3, 80.6)
C3	(87.6, **92.9**, 70.2, 82.1, 78.8)	(90.7, **91.2**, 81.6, 84.1, 82.8)	(86.4, **93.7**, 77.3, 80.6, 80.8)	N/A

first protocol and 0.83% for the second protocol. It only gives slightly better result than MvDA-vc (0.83% for the first protocol and 0.74% for the second protocol).

Table 4.3: Multi-view recognition comparison on IXMAS dataset

Deep features	Protocol 1			Protocol2		
	MvDA	MvDA-vc	pc-MvDA	MvDA	MvDA-vc	pc-MvDA
C3D	86.93	87.04	**88.19**	**99.99**	99.44	**99.98**
ResNet-50 3D	91.84	92.33	**92.82**	**100**	99.80	99,67
ResNet-50 RNN	72.44	75.95	**76.47**	99.34	**99.97**	**99.96**
ResNet-50 TA	76.74	81.01	**81.49**	95.80	98.25	**99.79**
ResNet-50 AP	79.28	78.93	**80.43**	**100**	98.11	99.85

Figure.4.7 compares the performance of feature extractors regarding each action class in multi-view evaluation scheme combined with the first evaluation protocol. There are clear margins between the performance of ResNet-50 3D and followed by C3D with other types of deep features, especially in harder actions (the first class to the third class and the eighth class to the eleventh class). These actions (check watch, cross arm, scratch head, wave, punch, kick, point) involve the most part static pose of the body and only movement of limbs, whereas other actions 4 - sit down, 5 - get up, 6 - turn around, 7 - walk and 12 - pickup include movement of the whole body and are easier to be recognized. This suggests that 3D convolution can better deal with small difference of movement in action images, which in turn generates a much more classification-ready feature space for latter stage of recognition.

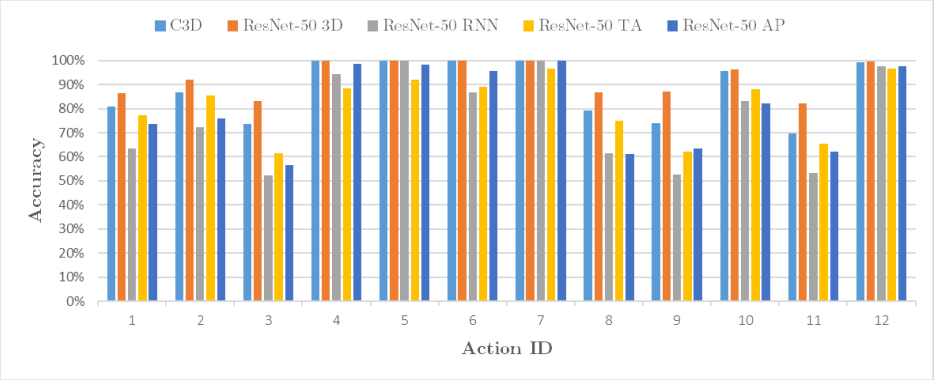


Figure 4.7: Comparison of accuracy on each action class using different deep features combined with pc-MvDA on IXMAS dataset

Table.4.4 compares our best combination of ResNet-50 3D and pc-MvDA with state-of-the-art frameworks. The comparison is for reference only because in other works, low-level hand-crafted

video representation is used as private features and train-test strategy is leave-one-class out, which is not applicable to our supervised deep feature extractors. However, the results are still commensurable.

Table 4.4: Comparison of our proposed methods with SOTA methods on IXMAS dataset according to the second evaluation protocol

Methods	Cross-view	Multi-view
Liu et al. [9]	76.32	N/A
Zheng et al. [32]	95.1	99.32
Zheng et al. [33]	97.8	99.4
Kong et al. [11]	99.92	100
Ulhaq et al. [34]	66.82	92.47
Zhang et al. [35]	84.1	N/A
Liu et al. [36]	N/A	90.3
Liu et al. [12]	99.95	99.67
Our proposed method (ResNet-50 3D + pc-MvDA)	99.67	99.65

4.3.2 Experimental results on MuHAVi dataset

1) Cross-view validation: Table.4.5 illustrates cross-view recognition results. Our proposed pc-MvDA consistently produces a better average accuracy of around 96.19% for protocol 1, approximately 4.55% and 1.51% higher than that of MvDA (91.64%) and MvDA-vc (94.67%) respectively.

Table 4.5: Cross-view recognition comparison on MuHAVi dataset

Deep features	Protocol 1			Protocol 2		
	MvDA	MvDA-vc	pc-MvDA	MvDA	MvDA-vc	pc-MvDA
C3D	92.85	95.15	**97.65**	98.95	99.76	**100**
ResNet-50 3D	97.94	99.15	**99.18**	96.77	**99.94**	**99.94**
ResNet-50 RNN	95.54	95.14	**96.18**	98.70	99.05	**99.99**
ResNet-50 TA	85.80	88.95	**91.12**	96.57	97.03	**99.88**
ResNet-50 AP	86.09	94.98	**96.82**	89.04	98.62	**99.93**

Table 4.6: Cross-view recognition results of different features on MuHAVi dataset with pc-MvDA method. The result in the bracket are accuracies of using features C3D, ResNet-50 3D, ResNet-50 RNN, ResNet-50 TA, ResNet-50 AP respectively. Each row corresponds to training view (from view C1 to view C7). Each column corresponds to testing view (from view C1 to view C7)

Training \ Testing	C1	C2	C3	C4	C5	C6	C7
C1	N/A	(96.1, **100**, 96.8, 88.5, 98.5)	(98.6, **99.6**, 96.6, 88.9, 96.3)	(98.6, **99.6**, 98.0, 93.1, 98.2)	(97.7, **99.6**, 95.4, 91.3, 96.1)	(97.7, **98.4**, 92.3, 93.1, 96.3)	(98.2, **98.5**, 97.2, 89.8, 97.1)
C2	(95.6, **99.0**, 95.2, 91.3, 95.5)	N/A	(98.9, **99.6**, 96.9, 90.3, 96.1)	(98.4, **99.6**, 97.6, 92.6, 98.2)	(97.7, **99.6**, 95.4, 90.9, 96.1)	(97.8, **98.4**, 92.7, 93.6, 96.5)	(98.6, **99.0**, 97.4, 90.6, 96.9)
C3	(96.1, **99.0**, 95.6, 90.3, 95.1)	(97.0, **100**, 96.7, 88.4, 97.8)	N/A	(98.1, **99.2**, 98.2, 92.8, 98.2)	(97.9, **99.6**, 95.4, 90.2, 95.6)	(97.9, **98.2**, 93.0, 94.2, 96.5)	(97.8, **98.7**, 97.4, 89.4, 97.6)
C4	(96.4, **98.8**, 95.4, 89.6, 95.3)	(96.3, **100**, 96.6, 91.0, 98.5)	(98.6, **99.6**, 97.3, 89.5, 96.1)	N/A	(97.7, **99.4**, 95.6, 91.3, 96.1)	(**98.4**, 98.0, 93.2, 93.8, 95.6)	(97.7, **98.7**, 97.4, 91.3, 96.7)
C5	(95.9, **99.0**, 95.6, 91.3, 95.5)	(96.6, **100**, 97.1, 89.9, 98.3)	(98.8, **99.4**, 96.8, 90.7, 96.1)	(98.4, **99.6**, 98.0, 92.1, 97.6)	N/A	(97.9, **98.2**, 93.4, 94.4, 96.8)	(98.0, **98.7**, 96.9, 89.0, 97.8)
C6	(95.8, **99.0**, 95.8, 90.3, 95.0)	(97.0, **100**, 97.6, 89.7, 98.5)	(98.8, **99.6**, 96.9, 88.7, 96.1)	(98.4, **99.4**, 98.0, 92.8, 98.2)	(97.5, **99.6**, 95.9, 91.2, 95.5)	N/A	(98.2, **99.0**, 97.2, 90.7, 97.4)
C7	(96.2, **98.8**, 95.7, 91.5, 96.2)	(96.4, **100**, 97.5, 90.3, 98.5)	(98.4, **99.0**, 97.3, 90.0, 96.7)	(**98.8**, 98.8, 98.2, 92.8, 98.0)	(97.5, **98.8**, 95.9, 91.5, 95.9)	(**98.9**, 98.2, 92.8, 93.9, 97.1)	N/A

2) Multi-view validation : Table.4.7 shows multi-view recognition results. The first protocol shows very close capability of MvDA-vc and pc-MvDA at 95.54% and 95.83% whereas MvDA is about 3% behind at 92.7%. For the second protocol, our variant exceeds MvDA by 2.21% and MvDA-vc by 1.49%. The near perfect results of the second protocol give us the same indication that it is not an inequitable method of comparison of MvL algorithms.

Table 4.7: Multi-view recognition comparison on MuHAVi dataset

Deep features	Protocol 1			Protocol2		
	MvDa	MvDA-vc	pc-MvDa	MvDa	MvDA-vc	pc-MvDa
C3D	81.80	96.05	**97.37**	88.39	99.61	**100**
ResNet-50 3D	99.07	99.12	**99.15**	**99.98**	99.96	99.89
ResNet-50 RNN	**96.18**	95.55	95.97	**100**	99.00	99.98
ResNet-50 TA	**90.45**	89.73	90.22	**99.92**	98.17	99.72
ResNet-50 AP	96.01	**96.75**	96.42	**100**	95.13	99.68

Figure.4.8 compares the performance of feature extractors regarding each action class in multi-view evaluation scheme combined with protocol 1. For this dataset, ResNet-50 3D persistently yields out-standing performance while ResNet-50 TA has the worst recognition rates in all action classes.

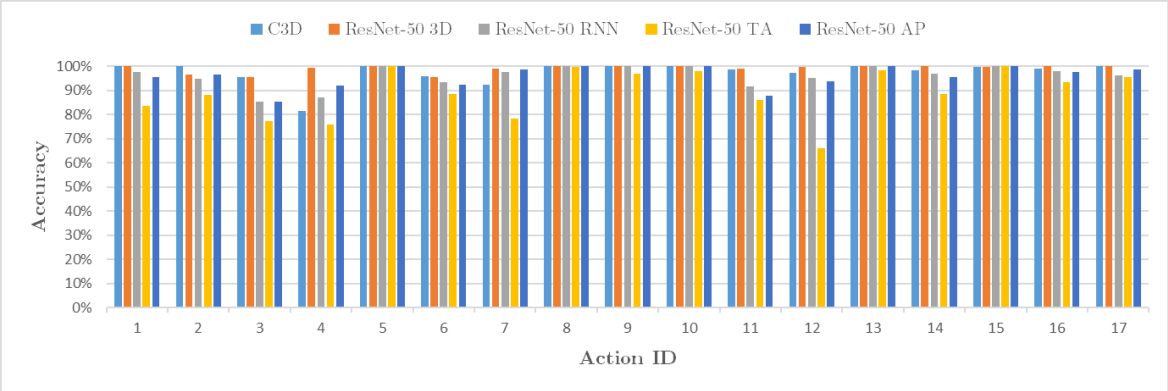


Figure 4.8: Comparison of accuracy on each action class using different deep features combined with pc-MvDA.

Table.4.8 compares our best combination of ResNet-50 3D and pc-MvDA with state-of-the-art frameworks. The comparison is for reference only because of aforementioned difference in setup of number of views and cross validation scheme.

Table 4.8: Comparison of our proposed methods with SOTA methods on MuHAVi dataset according to the second evaluation protocol.

Methods	Cross-view	Multi-view
Wu et al. [37]	N/A	94.5
Zheng et al. [33]	94.88	99.8
Liu et al. [36]	N/A	91.2
Liu et al. [12]	99.91	99.8
Our proposed method (ResNet-50 3D + pc-MvDA) [1]	99.90	99.88

[1] Our results when choosing only 4 views Camera 1, Camera 3, Camera 4 and Camera 6 following the other works.

4.3.3 Experimental results on MICAGes dataset

1) Cross-view validation: It can be seen in Table.4.9 that pc-MvDA outperforms in almost every cases. With the first protocol, our norm accuracy of 74.52% surpasses that of MvDA-vc (72.12%) by 2.4% and that of MvDA (64.94%) by a large margin of 9.58%. Especially in case of C3D features, our algorithm achieved 90.2% recognition rate whereas MvDA returns only 67.13%.

Table 4.9: Cross-view recognition comparison on MICAGes dataset

Deep features	Protocol 1			Protocol 2		
	MvDA	MvDA-vc	pc-MvDA	MvDA	MvDA-vc	pc-MvDA
C3D	67.13	85.98	**90.20**	93.74	99.80	**100**
ResNet-50 3D	94.91	95.25	**95.62**	97.43	**99.97**	99.79
ResNet-50 RNN	58.01	61.64	**64.13**	100	100	100
ResNet-50 TA	53.58	**59.55**	58.62	96.70	99.48	**99.83**
ResNet-50 AP	51.07	58.19	**64.04**	99.73	**99.99**	99.89

The top ranking of features applies identically and radically proves the superiority of 3D convolution based clip-level feature extraction in video recognition: ResNet-50 3D at 95.62% and C3D at 90.2%. The rest are ResNet-50 RNN (64.13%), ResNet-50 AP (64.04%) and ResNet-50 TA (58.62%). For the second protocol, MvDA-vc (99.85%) and pc-MvDA (99.9%) are nearly even in performance while MvDA achieved 97.52%.

Table 4.10: Cross-view recognition results of different features on MICAGes dataset with pc-MvDA method. The result in the bracket are accuracies of using features C3D, ResNet-50 3D, ResNet-50 RNN, ResNet-50 TA, RestNet-50 AP respectively. Each row corresponds to training view (from view K1 to view K5). Each column corresponds to testing view (from view K1 to view K5)

Training \ Testing	K1	K2	K3	K4	K5
K1	N/A	(92.9, **95.5**, 69.4, 63.9, 65.1)	(93.7, **98.0**, 79.1, 75.8, 74.7)	(90.5, **97.6**, 64.5, 68.8, 70.8)	(89.2, **92.6**, 56.0, 43.7, 58.2)
K2	(84.7, **94.6**, 51.3, 41.4, 53.7)	N/A	(94.6, **97.6**, 78.9, 75.4, 77.5)	(91.3, **97.6**, 64.4, 68.5, 68.0)	(87.4, **92.7**, 56.1, 42.0, 55.6)
K3	(87.5, **94.9**, 51.9, 43.6, 52.7)	(93.0, **95.5**, 70.1, 64.7, 63.4)	N/A	(90.0, **97.6**, 63.2, 62.6, 67.5)	(88.9, **92.2**, 57.2, 41.8, 56.6)
K4	(86.7, **94.7**, 50.8, 48.9, 54.9)	(90.2, **95.5**, 71.5, 68.2, 61.8)	(92.5, **98.0**, 81.0, 76.0, 74.4)	N/A	(89.7, **92.3**, 54.3, 44.9, 56.6)
K5	(84.9, **94.9**, 48.4, 41.5, 57.9)	(90.4, **95.5**, 72.6, 64.1, 65.7)	(94.0, **97.8**, 79.1, 72.5, 76.3)	(91.9, **97.3**, 62.7, 64.1, 69.0)	N/A

2) Multi-view validation : The multi-view recognition results in Table.4.11 shows an almost alike trend for both protocols. In general, pc-MvDA is 8.95% and 0.72% higher in accuracy for protocol 1, and 2.75% and 0.04% for protocol 2, in comparison with MvDA and MvDA-pc respectively.

In virtually every experiments of MICAGes and two earlier benchmark datasets, our proposed extension is superior. The average accuracies of pc-MvDA is, in comparison on protocol 1 (and protocol 2 resp.), 5.29% (1.21% resp.) higher than MvDA, 1.21% (0.62% resp.) better than MvDA-vc. Despite not being as intuitive and straightly intelligible as pairwise-covariance, the multi-view resemblance added in MvDA-vc achieved nearly the performance of pc-MvDA. However, this view-

consistency can be easily splitted into pairwise terms and intergrated into our objective in future works for further analysis.

Table 4.11: Multi-view recognition comparison on MICAGes dataset

Deep features	Protocol 1			Protocol2		
	MvDA	MvDA-vc	pc-MvDA	MvDA	MvDA-vc	pc-MvDA
C3D	68.31	87.70	**89.99**	88.79	99.54	**100**
ResNet-50 3D	95.32	95.26	**95.54**	**100**	99.96	99.80
ResNet-50 RNN	57.50	63.05	**63.83**	99.90	**100**	99.97
ResNet-50 TA	54.43	**61.46**	58.25	96.72	99.49	**99.69**
ResNet-50 AP	50.93	60.20	**63.66**	**99.99**	99.93	99.67

To better study the behavior of our algorithm, t-SNE embedding of original private spaces, MvDA and pc-MvDA common space generated by protocol 1 are plotted in Fig.4.9. In these scatter plots, we denote action classes by colors, views by shapes and train/test data are distinguished by border type. They explicitly denotes the surpassing capability of our algorithms in finding a common space with prominent extra-class discrepancy. The data samples involved in training process are generally clustered in compact and separated blobs while testing data samples dissolve nearby. The better convergence of pc-MvDA compared to the baseline is usually only visible for harder features, whereas with the inputs of ResNet-50 3D features, which are already highly discriminated in private spaces, the improvement of pc-MvDA is negligible. Note that these illustrations of t-SNE embedding do not strictly depict an identical distribution of data.

MICAGes also reveals the robustness of 3D convolution to generate fine clustered private feature space because our dataset contains exclusively hand gestures, which take part in a relatively small region of the captured videos. ResNet-50 3D and C3D predominantly distances other features (Fig.4.10).

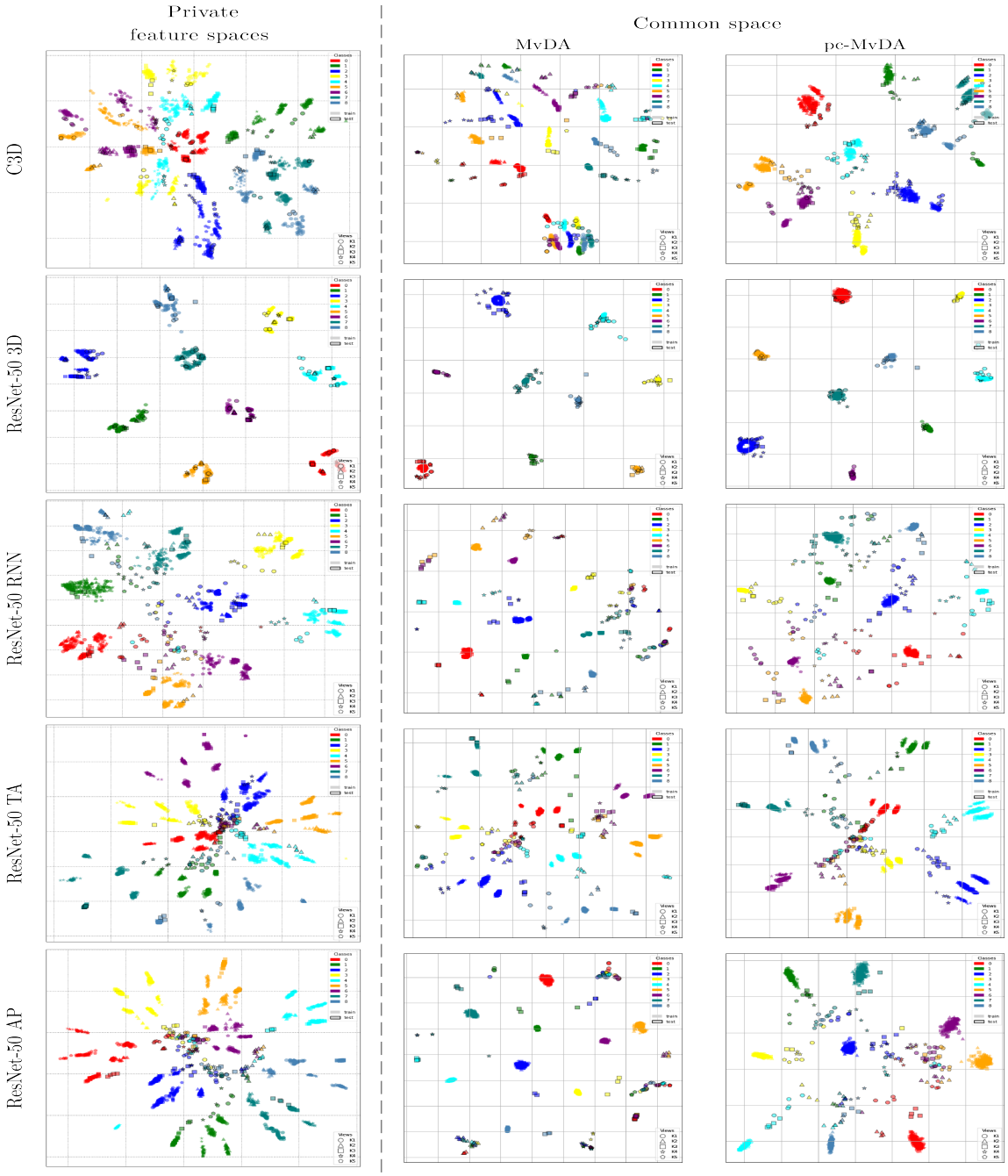


Figure 4.9: First column: private feature spaces stacked and embedded together in a same coordinate system; Second column: MvDA common space; Third column: pc-MvDA common space

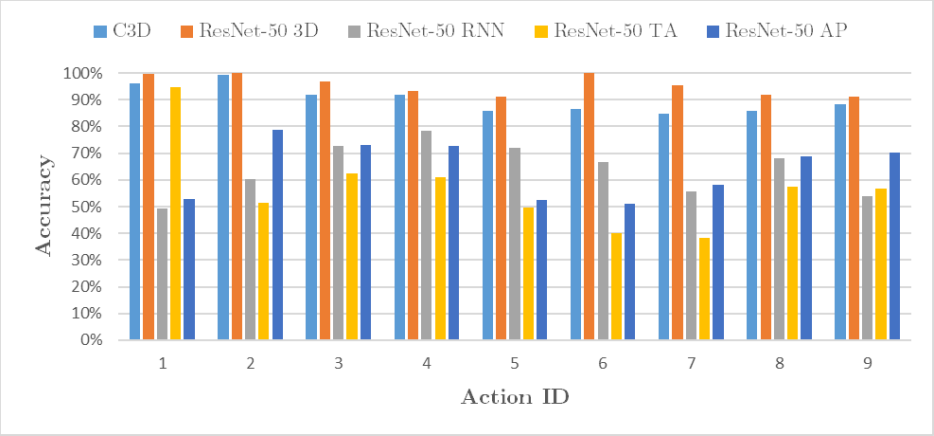


Figure 4.10: Comparison of accuracy on each action class using different deep features combined with pc-MvDA on MICAGes dataset

5 Conclusion

5.1 Accomplishments

5.2 Drawbacks

5.3 Future Works

Bibliography

[1] S. Herath, M. Harandi, and F. Porikli, "Going deeper into action recognition: A survey," *Image and vision computing*, vol. 60, pp. 4–21, 2017.

[2] H.-B. Zhang, Y.-X. Zhang, B. Zhong, Q. Lei, L. Yang, J.-X. Du, and D.-S. Chen, "A comprehensive survey of vision-based human action recognition methods," *Sensors*, vol. 19, no. 5, pp. 1005, 2019.

[3] D. M. Gavrila and L. S. Davis, "3-d model-based tracking of humans in action: a multi-view approach," in *Proceedings cvpr ieee computer society conference on computer vision and pattern recognition*. IEEE, 1996.

[4] F. Lv and R. Nevatia, "Single view human action recognition using key pose matching and viterbi path searching," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2007.

[5] D. Weinland, E. Boyer, and R. Ronfard, "Action recognition from arbitrary views using 3d exemplars," in *2007 IEEE 11th International Conference on Computer Vision*. IEEE, 2007.

[6] D. Weinland, R. Ronfard, and E. Boyer, "A survey of vision-based methods for action representation, segmentation and recognition," *Computer vision and image understanding*, vol. 115, no. 2, pp. 224–241, 2011.

[7] I. N. Junejo, E. Dexter, I. Laptev, and P. PÚrez, "Cross-view action recognition from temporal self-similarities," in *European Conference on Computer Vision*. Springer, 2008.

[8] B. Li, O. I. Camps, and M. Sznaier, "Cross-view activity recognition using hankelets," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012.

[9] J. Liu, M. Shah, B. Kuipers, and S. Savarese, "Cross-view action recognition via view knowledge transfer," in *CVPR 2011*. IEEE, 2011.

[10] J. Wang, X. Nie, Y. Xia, Y. Wu, and S.-C. Zhu, "Cross-view action modeling, learning and recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[11] Y. Kong, Z. Ding, J. Li, and Y. Fu, "Deeply learned view-invariant features for cross-view action recognition," *IEEE Transactions on Image Processing*, vol. 26, no. 6, pp. 3028–3037, 2017.

[12] Y. Liu, Z. Lu, J. Li, and T. Yang, "Hierarchically learned view-invariant representations for cross-view action recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 8, pp. 2416–2430, 2018.

[13] H. Rahmani, A. Mian, and M. Shah, "Learning a deep model for human action recognition from novel viewpoints," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 667–681, 2017.

[14] B. Thompson, *Canonical correlation analysis: Uses and interpretation*, Number 47. Sage, 1984.

[15] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural computation*, vol. 16, no. 12, pp. 2639–2664, 2004.

[16] M. Yang and S. Sun, "Multi-view uncorrelated linear discriminant analysis with applications to handwritten digit recognition," in *2014 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2014.

[17] M. Kan, S. Shan, H. Zhang, S. Lao, and X. Chen, "Multi-view discriminant analysis," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 1, pp. 188–194, 2015.

[18] X. You, J. Xu, W. Yuan, X.-Y. Jing, D. Tao, and T. Zhang, "Multi-view common component discriminant analysis for cross-view classification," *Pattern Recognition*, vol. 92, pp. 37–51, 2019.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[20] J. Gao and R. Nevatia, "Revisiting temporal modeling for video-based person reid," *arXiv preprint arXiv:1805.02104*, 2018.

[21] I. C. Duta, B. Ionescu, K. Aizawa, and N. Sebe, "Spatio-temporal vector of locally max pooled features for action recognition in videos," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017.

[22] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018.

[23] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, et al., "The kinetics human action video dataset," *arXiv preprint arXiv:1705.06950*, 2017.

[24] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015.

[25] M. Kan, S. Shan, and X. Chen, "Multi-view deep network for cross-view classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[26] D. Kong and C. Ding, "Pairwise-covariance linear discriminant analysis," in *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[27] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 03 1951.

[28] K. Fukunaga, "Chapter 10 - feature extraction and linear mapping for classification," in *Introduction to Statistical Pattern Recognition (Second Edition)*, K. Fukunaga, Ed., pp. 441 – 507. Academic Press, second edition edition, 1990.

[29] D. Weinland, R. Ronfard, and E. Boyer, "Free viewpoint action recognition using motion history volumes," *Computer vision and image understanding*, vol. 104, no. 2-3, pp. 249–257, 2006.

[30] F. Murtaza, M. H. Yousaf, and S. A. Velastin, "Multi-view human action recognition using 2d motion templates based on mhis and their hog description," *IET Computer Vision*, vol. 10, no. 7, pp. 758–767, 2016.

[31] M. Stone, "Cross-validatory choice and assessment of statistical predictions," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 36, no. 2, pp. 111–147, 1974.

[32] J. Zheng, Z. Jiang, P. J. Phillips, and R. Chellappa, "Cross-view action recognition via a transferable dictionary pair," in *bmvc*, 2012, vol. 1.

[33] J. Zheng, Z. Jiang, and R. Chellappa, "Cross-view action recognition via transferable dictionary learning," *IEEE Transactions on Image Processing*, vol. 25, no. 6, pp. 2542–2556, 2016.

[34] A. Ulhaq, X. Yin, J. He, and Y. Zhang, "On space-time filtering framework for matching human actions across different viewpoints," *IEEE Transactions on Image Processing*, vol. 27, no. 3, pp. 1230–1242, 2017.

[35] C. Zhang, H. Zheng, and J. Lai, "Cross-view action recognition based on hierarchical view-shared dictionary learning," *IEEE Access*, vol. 6, pp. 16855–16868, 2018.

[36] C. Liu, Z. Li, X. Shi, and C. Du, "Learning a mid-level representation for multiview action recognition," *Advances in Multimedia*, vol. 2018, pp. 1–10, 2018.

[37] X. Wu and Y. Jia, "View-invariant action recognition using latent kernelized structural svm," 10 2012.