University of Massachusetts Dartmouth

College of Engineering

Cross-View Action Recognition via Joint Dictionary and Transfer Learning

A Thesis in

Data Science

by

Deepak Kumar

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

May 2019

We approve the thesis of Deepak Kumar

Date of Signature

Ming Shao
Assistant Professor, Department of Computer and Information Science
Thesis Advisor

David Koop
Assistant Professor, Department of Computer and Information Science
Graduate Program Director, Data Science
Thesis Committee

Maoyuan Sun
Assistant Professor, Department of Computer and Information Science
Thesis Committee

Jean VanderGheynst
Dean, College of Engineering

Tesfay Meressi
Associate Provost for Graduate Studies

Abstract

Cross-View Action Recognition via Joint Dictionary and Transfer Learning

by Deepak Kumar

Human actions may be observed from multi-view, i.e., different cameras at the same time. Cross-view action recognition based on visual content is very challenging because the actions in different views may be internally related, but they look very different in the videos. With the help of improved dense trajectories (IDT), hand-crafted features satisfactory performance has been achieved on single-view, but these often fail for cross-view action recognition, due to non-discriminative features or feature space shift across the views. In this study, we proposed a novel method for cross-view action recognition by getting discriminative features for each view and bringing them into a common subspace via a novel joint dictionary and transfer learning framework. The dictionary learning method aims to learn a structured dictionary shared by all classes, and the learned dictionary shares the latent representation of data through a group of linear mappings and learns the sparse discriminative coefficients for each view. In the meanwhile, the encoded discriminative features are projected to a common feature space across different views through transfer learning. The dictionary and transfer learning are alternatively optimized to ensure discriminative and transferable features for cross-view action recognition. The proposed method has been extensively evaluated on PKU-MMD dataset to validate the effectiveness of the proposed approach.

Acknowledgements

Foremost I would like to express my sincere appreciation and gratitude to my Thesis Advisor, Dr. Ming Shao, for the patient guidance, encouragement, motivation and advice he has provided. His guidance helped me in all the time of research and writing of this thesis.

I have been with Dr. Shao for the past two years, and he would meet with me regularly on a weekly basis even on summer and winter breaks. He was extremely quick in response to emails even during vacations. He had a confidence in me, and he pushed me through my good and bad times to give my best. He is the main reason that I am proceeding with my PhD at University of Massachusetts Dartmouth. It is my honor to work under his guidance.

I would like to thank my Thesis Committee, Dr. David Koop and Dr. Maoyuan Sun for agreeing to be part of my thesis committee and their valuable feedback.

At last but not least, I would like to thank my family, parents, siblings, cousins, my uncle and friends, for all kind of support that they provided me, and without them I would not be going for my PhD today.

Contents

List of Figures

List of Tables

Abbreviations

IDT Improved Dense Trajectories

PCA Principle Component Analysis

HOF Histogram of Optical Flow

HOG Histogram of Oriented Gradient

MBH Motion Boundary Histogram

SVM Support Vector Machine

TCA Transfer Component Analysis

JDA Joint Distribution Adaptation

BDA Balanced Distribution Adaptation

TJM Transfer Joint Matching

GMM Gaussian Mixture Model

DL Dictionary Learning

SURF Speeded-up robust features

LRSDL Fast Low-Rank Shared Dictionary Learning

FDDL Fisher Discrimination Dictionary Learning

TL Transfer Learning

Chapter 1 Introduction

Action recognition is an active research field, which has received considerable attention in computer vision in the past decade. The major goal of action recognition is to automatically understand different types of actions to reduce human workload. The ever-growing interest in recognizing human actions is in part due to the wide range of applications, including, but not limited to the following:

Intelligent Visual Surveillance. There are number of cameras deployed for security surveillance to monitor criminal or suspicious activities, but these require constant human monitoring. To minimize the human workload, intelligent visual surveillance systems can be used to automatically detect and recognize suspicious activities and can generate alerts to security personnel [2] in order to prevent any dangerous situations.

Ambient Assisted Living. Ambient assisted living systems are used in health care [3] to automatically monitor and analyze patient activities, and also daily life activities especially for elderly people such as falling down, having a stroke or respiration issue, etc.

Human-computer Interaction. Human gestures can be used to operate different tasks in the computer such as changing presentation slides using hand gestures instead of using traditional methods such as keyboard and mouse [4].

Entertainment. The modern era of gaming uses the human body as a controller instead of using gaming controllers. These systems can be used to track a player's activity during his or her interaction with the gaming environment [5].

Considerable work is done for single view action recognition when videos are captured from a fixed view using one camera [6, 7, 8, 9, 10]. However, from the above-mentioned applications, there can be multiple cameras to capture a single action from different viewpoints or angles to get enriched information for action description as shown in Figure 1.1, where single object actions

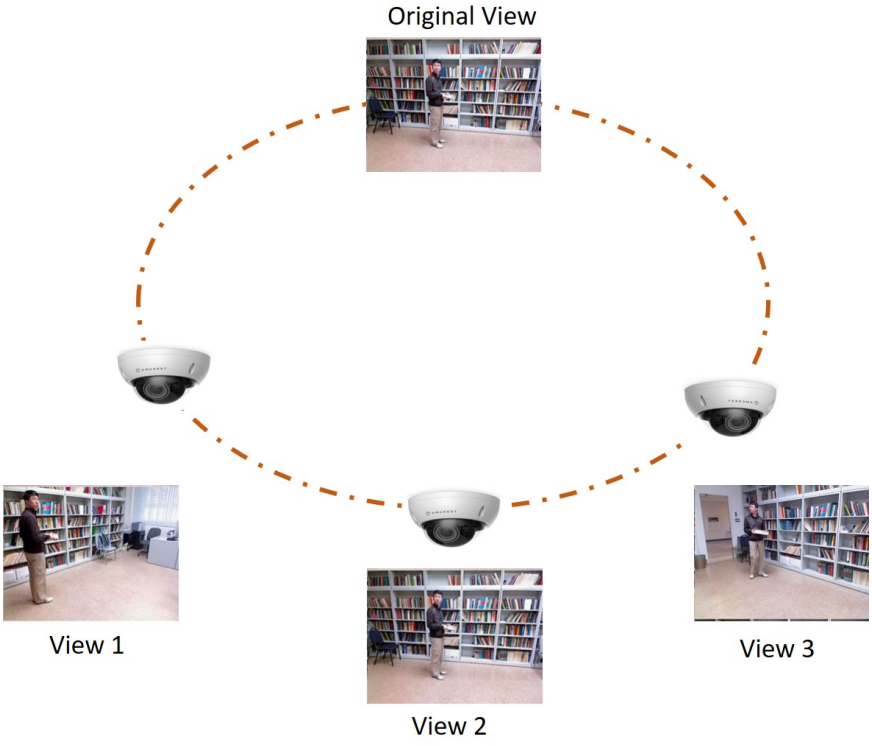


Figure 1.1: Illustration of cross-view action recognition in surveillance system.

are captured by three different cameras fixed at three different angles. Most of the proposed approaches for single view or multi-view action recognition consist of feature engineering which includes different feature extraction [11, 12, 13, 14, 15], dictionary learning [16, 17, 18], and deep learning [19] methods. However, it is still challenging to recognize action from different view points because the same action looks quite different across views. There are a couple of methods proposed for multi-view or cross-view action recognition, but multiple challenges still remain which need to be addressed [20, 21].

The first key step for action recognition is to extract features from video frames. Feature extraction can be understood as a process of retrieving values at the pixel level which is informative

and non-redundant. These values give us points of interest which on their own do not contribute to the action itself but extract most relevant points from the raw data. There are challenges in feature extraction, like cluttered images where it is hard to differentiate objects from one another, similarity in colors between objects, camera motion which can bring noise in the images and create blurred frames, large amount of viewpoint variations, different illumination conditions, occlusion, variance in human appearance, shapes, clothes, etc. Another major challenge in this process is selecting a suitable framework that can represent the extracted features accurately and also take into consideration the changes in the spatial and temporal information. Many feature representation methods are proposed for action recognition in video sequences, such as optical flow patterns [11, 22], shape feature [12, 13, 23, 24], space-time pattern templates [25, 26] dense trajectories [27], improved dense trajectories [14], and spatio-temporal interest points [15]. Although these handcrafted low-level feature methods are powerful to recognize actions in similar view or in controlled scenarios, the performance degrades with viewpoint changes. One reason is appearance difference when observed from different angles, due to this along with view change, the low-level features become less discriminative. One possible solution is brutal-force approach that trains a separate classifier for each view, but it becomes intractable as the number of action categories and views increase. In addition, it is computationally expensive or infeasible to train a separate classifier to address the action under novel views. Another approach is to build view invariant [28, 29, 18, 30, 31, 32, 33] representation for action recognition.

Although these low-level features achieved great success, designing the effective discriminative features requires the new domain knowledge and these hand-crafted features cannot simply be adapted to new conditions. Learning feature from data itself is considered a plausible way to overcome these limitations and successful examples are dictionary learning and deep learning. The idea of deep learning is to discover multiple levels of representation [19], that deeply learned features will represent the abstract semantics of data. It is expected from deep learning models,

that these will provide features which will be more invariant to intra-class variability. The issue with deep learning is that it requires the large amount of labeled data to train a model, and in some cases, it is impossible or not practical. Sometimes pre-trained models are used for feature extraction and then weights of these pre-trained models are fine-tuned. However, the effectiveness of the "transferred" features becomes worse when source and target become less similar.

In contrast, dictionary learning (DL) methods have demonstrated great success in image classification tasks on both small and large intra-class variation datasets [16]. The last few years have witnessed the fast development of DL methods under sparse representation theory, i.e., signals can be well-approximated by the linear combination of few columns of some basis or dictionary [17]. The dictionary that faithfully and discriminatively encodes signals plays an important role in the success of sparse representation. Conventional dictionary learning performs well for different classification and recognition tasks, but the performance deteriorates for cross-view action recognition. The primary reason is discriminative features for each view are not in common feature subspace. Notably, there are view-shared sparse representation based approaches proposed [30, 18]. In those approaches, they assumed that different views contribute equally in shared features, and they ignored view-private features, however, it is not always valid.

Another approach is to incorporate transfer learning to bring the source and target in the common subspace. Transfer learning aims to improve the learning of target predictive function using the source knowledge. Transfer learning tries to bring the distribution of source and target domain in the common subspace by learning knowledge from one or more source domains and transfer that to target domain in order to improve the performance [34]. The main purpose is to find the common representative for the similar objects which look different when they are in different views.

Transfer learning can be performed in supervised, semi-supervised and unsupervised ways. In supervised learning [35, 36] the source and target both contain labeled data, in semi-supervised [36,

37] source domain has labeled data, while target domain has limited or no labeled data, and in unsupervised [38] there is no labeled data for both source and target domains. In brief, transfer learning may help to improve the performance for cross-view action recognition, but with performance being compromised, because source and target may not be discriminative in common space.

1.1 Motivation

Considerable feature representation methods have been proposed for action recognition in videos, but these are only powerful for same view action recognition. In that sense, the performance degrades when the same action is performed from a different view. Recent learning approaches have cast a light on such issues. First, transfer learning or domain adaptation methods have shown good results for such scenarios, but this approach is largely limited by discriminant ability because the features brought up in the common subspace neither promotes the underlying sparse structure of data, nor preserve the best features. Second, shared dictionary learned for cross-view action recognition provides sparse representation, but it may ignore view private features. In some cases, e.g., when actions are performed from unseen views which are not part of the shared dictionary, the performance will degrade significantly.

Motivated by the above analysis and the limitation of transfer learning and dictionary learning for cross-view action recognition when these methods are incorporated separately, we proposed a novel approach called **Joint Dictionary and Transfer Learning (JDTL)** that helps to improve the performance of cross-view action recognition, where dictionary learning pursues the discriminative and robust representation for each view and transfer learning aims to bring those discriminative features of each view in the common subspace.

1.2 Contribution

The main technical contributions of this work are:

- We pursue discriminant dictionary learning in each view with $D^{(v)}$ as the v-th view dictionary and $X^{(v)}$ as the coding and we learned view independent dictionaries $D^{(v)} = [D_1^{(v)}, D_2^{(v)}, \cdots, D_C^{(v)}]$ to extract discriminative features for C different actions. A common feature space will be optimized through transfer learning across different views in a joint learning framework with dictionary learning.

- An iterative optimization is developed to approach our learning problems where the transfer learning projection matrix P, dictionary D and coefficients X are learned simultaneously. In each iteration, the coefficients X are updated by fixing P and D, dictionary D is refined by fixing the P and X, and P is learned by fixing D and X. The learning process will terminate until the objective converges.

- To evaluate the proposed method, extended experiments have been performed on PKU-MMD dataset [39], a benchmark multi-view action recognition dataset. Experimental results demonstrate that our proposed method has achieved promising results for cross-view action recognition.

1.3 Outline

This thesis consists of 6 chapters including the introduction. Chapter 2 reviews the recent work. Chapter 3 presents the proposed joint dictionary and transfer learning method. Dataset preprocessing and experimental setup are described in Chapter 4. Experimental results and evaluation of our method are discussed in Chapter 5 and we present future work and draw the conclusion in Chapter 6.

Chapter 2 Literature Review

A comprehensive review of the different types of visual handcrafted features for action recognition will be discussed in this chapter. What follows is recent learning techniques including dictionary and transfer learning.

2.1 Hand-Crafted Features

In handcrafted method, local features are extracted using space-time interest points [40], cuboids [41], dense trajectories [1] and improved dense trajectories [14]. Local features are calculated with the help of detectors and descriptors, which are introduced below.

2.1.1 Feature Detection

A simple definition for feature detection is to detect locations of interest points and to check if there are features available at those locations. A few commonly used methods are described below.

Harris3D Detector. Harris3D detector detects local structures in space-time, where there is significant local variation in the image space and time. This means that the Harris 3D detector can detect similar regions in images by using affine transformations which has different illuminations [40]. This is particularly useful when the same image is taken from different angles and similar regions need to be identified.

Cuboid Detector. The main task was to detect the spatio-temporal interest points. The main difference in this detector is that it makes use of two separate linear filters for both spatial and temporal dimensions instead of using a 3D filter [42].

Hessian Detector. The hessian detector was proposed in [43], and it is used for blob detection in images. It is an extension of the hessian saliency measure. The second-order Taylor expansion of the intensity surface is explored and especially the hessian matrix (containing the second order derivatives). The matrix determinant reaches a maximum in the image for blob-like structure [44].

$$H = \begin{bmatrix} I_{xx}(X, \sigma_D) & I_{xy}(X, \sigma_D) \\ I_{xy}(X, \sigma_D) & I_{yy}(X, \sigma_D) \end{bmatrix} \tag{2.1}$$

The 2×2 matrix given in equation 2.1 is the hessian matrix generated from the Taylor expansion of the image intensity function $I(X)$. where I_{xx} is the second-order gaussian smoothed image derivatives which encodes the shape information and captures the local image structure. The filters based on the determinant and the trace of this matrix are particularly interesting. The latter is often referred as the Laplacian. The blob-like image structure is detected by calculating the local maxima of both measurements. The drawback of Laplacian filter is that signal changes in one direction while local maxima is found close to contours or straight edges.

Dense Sampling. In dense sampling, video blocks are extracted at regular positions and scales in space and time [45]. The data/video is sampled from five dimensions x, y, t, σ, τ. Here σ and τ represent the spatial and temporal scale. Dense sampling covers the whole image patch and generates a large amount of features.

In brief, Cuboid is slower and extracts the densest features, whereas Hessian method extracts the sparsest features and it is most efficient. In dense sampling the more time spent of feature description because there is no feature detection as such, and it extracts more features points in comparison to interest point detectors [46].

2.1.2 Feature Description

Feature description is a process of describing the detected regions in a way that is not affected by clutter in the backgrounds, appearances, rotations, and occlusions. The size of the spatial and temporal patch is determined by the size of the interest points i.e. after the interest points of certain size are found, similar features are grouped and a descriptor is used to describe that patch so that it can be differentiated from other patches. It can be identified and represented in a compact form. Most of these descriptors were proposed to extract features from images but nowadays

these descriptors are used to extract feature from videos. Popular feature description methods are described below.

Histogram of Oriented Gradients (HOG). HOG method focus was to detect pedestrians in still images, and later on this method was extended to detect humans in videos and films. This method is used to describe the local appearance. In HOG, the distribution of direction of gradients is used as features. It counts the occurrences of gradient orientation by dividing the image into small regions and makes a histogram of gradient directions for each pixel in each region [47].

Histogram of Optical Flow (HOF). HOF is a method which is used to describe motion by computing the histograms of optical flow by aggregating them in space-time neighborhoods of the interest points [40]. The blocks in the histograms are divided into bin sizes. These histograms are then normalized into HOF vectors.

Motion Boundary Histogram (MBH). Motion between two frames is achieved by the optical flow and the motion can be object motion in the foreground or it can be background motion due to camera motion. The camera motion is harmful for action classification, so [48] proposed the motion boundary histogram (MBH). MBH is a descriptor which is used for human detection where the horizontal and vertical components of the optical flow are computed separately. This descriptor separates the moving object from the still background and constructs histograms on the moving objects by encoding the relative motion between pixels [27].

Scale Invariant Feature Transform (SIFT). SIFT is a computer vision algorithm to detect and describe local features of the images. There are four major steps to find the key points and generate features based on the key points and these steps make sure that key points are stable to matching and recognition [49].

At the first stage, the interest points are selected with the help of the difference-of-gaussian function. These interest points are searched over different scales and different image locations and are invariant to scale and orientation. In second stage, key points location is refined with the help

of Taylor series expansion of space scale to get rid of bad key points. In third stage to make the key points more robust to orientation, an orientation is calculated for each key point. In final stage a descriptor is generated with the help of the list of features points which are described in the terms of location, scale, and orientation.

For the recognition task, the SIFT key points of training dataset are stored and matching keys for a new image are identified. The best match for each key point is found by identifying its nearest neighbor in the database of key points in the training image dataset.

Trajectory. The idea of trajectory is to track spatial interest points over time and based on tracked interest points corresponding trajectory is extracted. The generated trajectories in videos are clustered and a transformation matrix is calculated for each cluster [27]. Different methods have been proposed by using the different feature descriptor to extract trajectories. SIFT descriptors are matched between two frames to compute trajectories [50], by discarding matches that are not similar between frames. In [40] Harris3D interest points are tracked with the help of KLT (Kanade-Lucas-Tomasi) [51] tracker to extract trajectories.

Dense Trajectory. The dense trajectory is extracted for multiple spatial scales. Here the trajectories are obtained by tracking the densely sampled points using optical flow fields. Optical flow fields are used to find patterns of objects, surfaces, and edges in visual scenes which is the result of relative motion between the observer and the scene [45]. In dense trajectory, local descriptor HOG, HOF, MBH are calculated in space-time volume around the trajectory. HOG focuses on static appearance information, while HOF captures the local motion information. The MBH deals with the noise generated due to background motion.

In this study, we used Improved Dense Trajectory(IDT) for feature description. IDT is an extension of Dense Trajectory. We chose IDT because it captures spatial and temporal information and it also deals with noise that generated due to the camera or background motion.

2.2 Dictionary Learning

Sparse coding has been widely applied in a variety of computer vision problems and seeks to represent a signal as a sparse linear combination of few dictionary atoms. Dictionary plays an important role as it is expected to represent components of the query signal robustly. The dictionary is over-complete if the number of basis vector is greater than the number of input dimension. The over-complete basis tries to learn a natural representation of real data distribution. The higher dimension representation helps to model a complex distribution. The goal of DL is to learn a dictionary from an over-complete basis, where features which can yield a sparse representation of training samples. Sparse representation expresses a signal as a linear combination taken from the dictionary. The dictionary elements are called atoms [52, 53]. Sparsity constraints can be used when the dictionary is over-complete and atoms are not unique [54]. Efficient and sparse representation is obtained by relaxing the requirements. The main goal is to obtain a sparse vector that should have significant coefficients that are not close or equal to zero. This is an optimization problem and can be solved by different approaches, and these approaches are divided into two groups. The first group includes greedy algorithms, the greedy algorithms iteratively select locally optimal basis vectors such as Matching Pursuit (MP) [55] and Orthogonal Matching Pursuit (OMP) [56, 57]. The other group is based on convex relation methods such as Least Absolute Shrinkage and Selection Operator (LASSO).

Matching Pursuit (MP). Matching pursuit finds the best atoms for sparse representation construction on each iteration based on the similarity measurement from an over-complete dictionary. The process is iterative and finds the best atom until the stopping criterion is not satisfied.

Orthogonal Matching Pursuit Algorithm (OMP). Orthogonal matching pursuit algorithm is proposed to improve the matching pursuit algorithm. The OMP uses the method of orthogonalization to make sure that projection in each iteration is in the orthogonal direction.

However, based on label availability, existing dictionary learning and sparse representation approaches can be divided into two main categories: supervised learning and unsupervised learning. The difference between the supervised and unsupervised learning is that in supervised class label information is used to exploit the dictionary learning process while there is no need of labels in unsupervised dictionary learning.

A lot of work has been done in dictionary learning, and a few well-known methods are defined below.

K-SVD. K-SVD is an unsupervised shared dictionary learning algorithm and this method has achieved promising results in image restoration and image denoising, but these are not helpful for image classification. K-SVD is the generalization of K-means for dictionary learning. Each column of a dictionary is updated using the singular value decomposition after the sparse approximation achieved through OMP. This algorithm does not guarantee to converge in general [58].

FDDL. FDDL is the class-specific dictionary learning algorithm. For stronger discrimination, FDDL method combines the fisher discrimination criterion to learn the structured dictionary [59, 53]. Fisher discrimination criterion is a parametric criterion that maximizes the distance between the means of two classes and minimizes the variance within each class. Representation residual of FDDL method can be used to differentiate between different classes and additionally the representation coefficients have a smaller within-class scatter while between-class scatter is large.

COPAR. The COPAR is commonality and particularity dictionary learning algorithm. Some common patterns are usually shared by class-specific dictionaries from different categories which do not play any specific role for classification but are crucial for dictionary learning. Although these common or coherence patterns are essential, these can compromise classification performance when these are used interchangeably for representation of a query datum. So, COPAR method integrates the incoherence penalty term to learn a class-specific dictionary [60].

DLSI. Instead of learning class-specific dictionaries DLSI can learn the common features by making use of most coherent atoms as common features from different individual dictionaries [61].

JDL. JDL algorithm learns more discriminative dictionaries by using inter-category visual correlations. In this method, common-shared visual atoms and category-specific one's atoms are separated by jointly learning a shared dictionary and several category-specific dictionaries [62].

DFDL. DFDL is an extension of SRC [63]. The main goal of SRC is to express the test sample as a linear combination of the training set. DFDL method was proposed to learn features from histopathology images because it was difficult to get good enough discriminative features due to the geometric richness of tissue images. A discriminative dictionary is designed via optimization for each class to minimize intra-class differences while simultaneously emphasizing inter-class differences by imposing sparsity constraints [64].

CSDL. CSDL method was proposed to deal with fine-grained image categorization. Normal sparse coding learns a dictionary to minimize the reconstruction error, and it performs well for basic level classification because the difference between sparse representation for different categories images are large, but for fine-grained categorized images, most of the content is similar in all images, and it becomes hard to encode a category-specific dictionary. This method learns a category-specific and shared dictionary for each category and all categories respectively. The particular category dictionary can encode the differences between different categories, and the shared dictionary can encode common patterns [65].

LRSDL. Low-rank Shared Dictionary Learning (LRSDL) method is the extension of FDDL. DLSI does not learn shared features as these are still hidden in the sub-dictionaries. COPAR, JDL, and CSDL learn shared dictionaries but the shared dictionaries are not low-rank dictionaries. So, the class-specific dictionaries can be represented by shared dictionaries. LRSDL automatically extracts both discriminative and shared bases by alternatively optimizing the class-specific dictionary to

extract the class specific features and shared features for all classes [66]. In our study, we have used LRSDL for dictionary learning and it is discussed in detail in the methodology chapter.

2.3 Transfer Learning

Transfer Learning is a way to propagate what has been learned in one domain to another similar or different domain [38]. In conventional learning for each task everything is learned from scratch, while in transfer learning knowledge is transferred from existing task to the target task. Transfer learning has widely been adopted in the machine learning and data mining fields, and following surveys [67, 34] give detailed description about transfer learning. There are three main research issues in transfer learning i.e., 1) What to transfer 2) How to transfer 3) When to transfer. "What to transfer" sees which pieces of information can be transferred from one task or domain to another. Some information can be specific to a certain task or domain, and some information can be common across tasks or domains. After getting the information that can be transferred, different algorithms had been developed to deal with "how to transfer" the information. "When to transfer" figures out the criteria when transferring skills should be done. Transfer learning is categorized into three groups according to the labels availability in the source and target domains, and the tasks of the two domains. It is further discussed below.

2.3.1 Inductive Transfer Learning

In inductive transfer learning source and target tasks are different, and it does not matter source, and target domains are similar or different. In this case, labeled data in the target domain is required for induction. Based on the situation of labeled and unlabeled source domain, inductive transfer learning can further categorize in two cases.

Multi-Task Learning. In multi-task learning source and target tasks are learned simultaneously, it does not matter that the source and target domains are similar or different. The aim is to achieve high performance in the target task.

Self-Taught Learning. In self-taught-learning the source and target domains may have different label spaces, which show that we can not use directly side information of source domain.

2.3.2 Transductive Transfer Learning

The transductive transfer learning proposed by [68]. For transductive transfer learning source and target tasks should be similar, but the domains can be different. Domain adaptation is a type of transductive transfer learning. In domain adaptation, source domain contains abundant labeled data while the target domain contains no labeled data, and the source and target domains are different. Different transductive transfer learning had been proposed. Structural Correspondence Learning (SCL) is one of them. The SCL was proposed for speech tagging. It uses the unlabeled target domain for reducing the domain data differences. Based on unlabeled data in both domains, the set of pivotal features are defined. These defined pivotal features are used to learn the mapping between original feature space and shared low-dimension feature space.

2.3.3 Unsupervised Transfer Learning

Unsupervised transfer learning is more focused to clustering, dimension reduction, where there is no need for labeled data in both source and target domains, while it is similar to multi-task learning where the source and target tasks are different but related to each other.

Chapter 3 Methodology

In this section, we discuss our novel framework implementation and also discuss different methods which are used in our framework. Our framework is based on three steps. In the first step, visual descriptors are extracted from videos using Improved Dense Trajectory method (IDT) [14]. Second, Gaussian Mixture Model (GMM) and Fisher Vector are used for feature representation. In the last, we jointly learn the dictionary and projection matrix via transfer learning to get the discriminative and transferable features for cross-view action recognition. The Joint Dictionary and Transfer Learning (JDTL) framework is an iterative process. First, the new projection matrix is obtained by using different transfer learning methods, then a dictionary is learned by fixing the projection matrix and coefficients, and finally coefficients are learned by fixing the projection matrix and updated dictionary.

3.1 Feature Extraction

The first step of our framework is to extract the features from videos. Improved Dense Trajectories (IDT) is a state-of-the-art method used for feature extraction in our study. In brief, IDT was introduced by Wang et al. [14] and is extended from dense trajectories [27, 69]. Dense Trajectory densely sample the feature points in video frame and track those detected features points in upcoming video frames, while IDT improves dense trajectories by explicitly estimating camera motion.

Dense Trajectories (DT). It is difficult to determine the best scale to track feature points, and therefore dense trajectories are extracted from multiple spatial scales. Feature points are sampled in each frame of a video on a grid spaced by W pixels. To obtain enough dense trajectories and catch significant motion information, the parameter W is set to 5 pixels. Once feature points are

extracted, these are tracked in each video frame in each spatial scale separately. Eight spatial scales are used, and each spatial scale is increased by a factor of $\frac{1}{\sqrt{2}}$.

Dense optical flow field w_t is computed from frame I_t to the following frame I_{t+1}, using the Farneback algorithm [70]. Each point $P_t = (x_t, y_t)$ at frame t is tracked to the next frame $t + 1$ by median filtering in dense optical flow field $w_t = (u_t, v_t)$, where u_t, and v_t are the horizontal and vertical components of optical flow respectively.

$$P_{t+1} \ = \ (x_{t+1},\ y_{t+1}) \ = (x_t,\ y_t) \ + \ (M * \ w) \ |_{(\bar{x}_t,\ \ \bar{y}_t)} \tag{3.1}$$

DT Descriptors: Trajectory shape, HOG, HOF, and MBH. Wang et al. [14] proposed the trajectory shape descriptor to encode a form of the extracted dense trajectories. A sequence of displacement vectors normalized by the sum of displacement vectors magnitudes describe the trajectory shape.

In a space-time volume around a trajectory, three descriptors (HOG, HOF, and MBH) are calculated. Each local volume is subdivided into a grid with $n_x \times n_y \times n_t$ spatio-temporal cells in order to embed structure information. A histogram descriptor is calculated for each cell of the grid. The histograms are normalized with L_2 norm, and then normalized cells are concatenated into the final descriptors.

Similarly, the HOG and HOF descriptors are calculated for spatio-temporal interest points. In this case, edge and optical flow orientations are quantized into 8 bins using full orientations, with the HOF descriptor having an additional zero bin. The example of HOG is shown in Figure 3.1, where in (a) the image is the input image, (b) input image with extracted HOG features at different locations on input image, and in (c) extracted HOG features from input image are illustrated.

Dalal et al. [48] proposed the Motion Boundary Histogram (MBH) descriptor to detect human. Optical flow field $I_w = (I_x, I_y)$ is separated into its x and y components. Spatial derivatives for

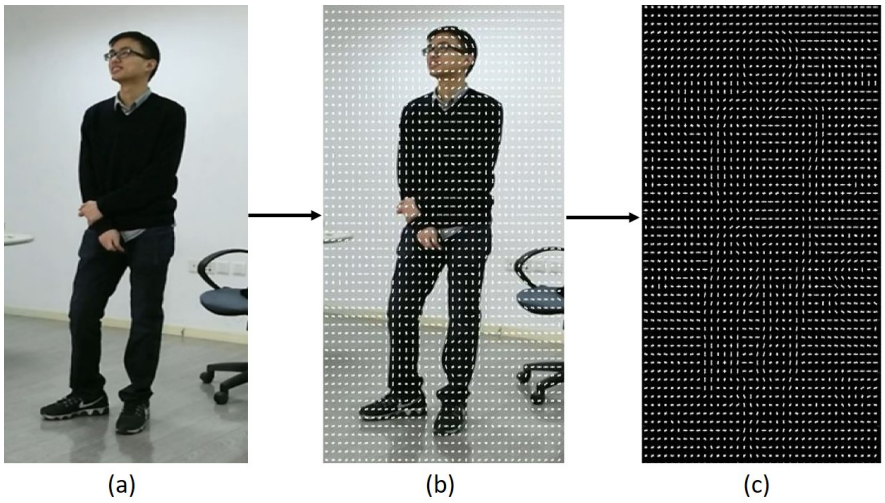


Figure 3.1: (a) Input image. (b) HOG features extracted from the image at different locations. (c) HOG features illustration.

the horizontal and vertical components of the optical flow field are computed separately, and the orientation information is quantized into histograms. Background motion is eliminated by suppressing the constant motion information and keeping the information about changes in the flow field.

IDT boosts the recognition performance of dense trajectories by taking camera motion into account. It assumes that background motion of two consecutive frames can be characterized by a homography matrix. To estimate the homography matrix, the first step is to find the correspondence between two consecutive frames. They resort to SURF [71] feature matching and optical flow based matching, as these two kinds of matching schemes are complementary to each other. Then RANSAC [72] algorithm is used to estimate the homography matrix. Based on the homography matrix, they rectify the frame image to remove camera motion and re-calculate the optical flow called *wrapped flow*. There are two major advantages for dense trajectories to cancel out camera

motion from optical flow. First, this brings advantages to the descriptor calculated from optical flows, in particular for HOF. Second, trajectories generated due to camera motion can be removed.

It describes the deviation of orignal set of features from average distribution of features modeled by parametric generative model,

3.2 Feature Representation

Once local features are extracted, these are used to represent actions in videos. As the length of each video is different, we refer to **Fisher Vector (FV)**, one of most efficient vector quantization method to represent each video and achieve a fixed length vector [73, 74]. FV is an extension of bag-of-visual word representation and its encoding represents differences between features and visual words. FV has achieved remarkable results as a global descriptor both for image classification and for image retrieval, outperforming the conventional bag-of-features approach. Given a large set of vectors, the FV method assembles them into a high dimensional representation. So, in the first stage diagonal covariance was used to train the Gaussian Mixture Model (GMM), and derivatives are considered with respect to gaussian mean and variance [75]. It gives the representation, which captures the average first and second order differences between the original feature and each of the GMM centers:

$$\begin{aligned}\Phi_k^{(1)} &= \frac{1}{N\sqrt{W_k}} \sum_{p=1}^{N} \alpha_p(k) \left(\frac{x_p - \mu_k}{\sigma_k} \right), \\ \Phi_k^{(2)} &= \frac{1}{N\sqrt{2W_k}} \sum_{p=1}^{N} \alpha_p(k) \left(\frac{(x_p - \mu_k)^2}{\sigma_k^2} - 1 \right),\end{aligned} \tag{3.2}$$

where $\{w_k, \mu_k, \sigma_k\}_k$ are mixture weights, means, and diagonal covariances of GMM. $\alpha_p(k)$ is the soft assignment weight of the p-th feature x_p to the k-th Gaussian. An FV Φ is obtained by stacking the differences: $\Phi = [\Phi_1^{(1)}, \Phi_1^{(2)}, ..., \Phi_K^{(1)}, \Phi_K^{(2)}]$. The final encoded vector shows the difference between distribution of video features and distribution fitted to the features of all training videos.

Table 3.1: Details of the variables used in our method

Variable	Description
P	Projection Matrix
Y	Training Sample
D_0	Shared Dictionary
D_j	Class-specific Dictionary
$\bar{D}$	Total Dictionary that includes class-specific and shared dictionary
X_0	Shared Coefficients
X_c	Class-Specific Coefficients
M	Each Dictionary Column Mean Vector
M_0	Shared Coefficient Mean Vector
M_c	Class-specific Coefficient Mean Vector
J	Input Data Matrix (Transfer Learning)
J_s	Source Task
J_t	Target Task
L_t	Target Label
L_s	Source Label
I	Identity Matrix
M_d	Maximum Mean Discrepancy (MMD) Matrix
M_{d_c}	Maximum Mean Discrepancy (MMD) Matrices involving class labels
K	Input Kernel Parameter
$\mathcal{P}$	Marginal Distribution
$\mathcal{Q}$	Conditional function
H	Centering Matrix
$\triangleq$	equal by definition
μ, λ	Trade-off Parameter/Regularize parameter

3.3 Transfer Learning

Transfer learning is employed in this chapter to map the source (one view) and target (another view) features into a common subspace. The transfer learning methods are categorized in supervised, semi-supervised and unsupervised as we described in related work, and we detail four methods below which will be incorporated into the joint learning framework. The variables and basic formulation in this chapter can be found in Table 3.1 and 3.2.

Table 3.2: Transfer learning methods

Method	Learning Objective	Constraints
Transfer Component Analysis	$\min_W \mathbf{tr}(W^\top W) + \mu\, \mathbf{tr}(W^\top KLKW)$	$W^\top KHKW = I$
Joint Distribution Adaptation	$\min\ \sum_C^{c=0} \mathbf{tr}(J^\top J M_{d_c} J^\top A)\ +\ \lambda \|\|A\|\|_F^2$	$A^\top JHJ^\top A = I$
Balance Distribution Adaptation	$\min\ \mathbf{tr}\Big(A^\top J\Big((1-\mu)M_{d_0}\ +\mu \sum_{c=1}^{C} M_{d_c}\Big)J^\top A\Big)\ +\lambda \|\|A\|\|_F^2$	$A^\top JHJ^\top A\ =\ I\ ,$ $0 \leq \mu \leq 1$
Transfer Joint Matching	$\min \mathbf{tr}(A^\top K M_d K^\top A)\ +\lambda\big(\|\|A_s\|\|_{2,1} + \|\|A_t\|\|_F^2\big)$	$A^\top KHK^\top A = I$

3.3.1 Transfer Component Analysis

The Transfer Component Analysis (TCA) algorithm reduces marginal distribution difference using a Reproducing Kernel Hilbert Space to discover common latent features [76, 77]. The Maximum Mean Discrepancy (MMD) measure is used to determine the marginal distribution difference between the source and target data. The TCA optimization equation is given in Table 3.2 where μ is the trade-off parameter, I is the identity matrix, H is the centering matrix, K is the kernel matrix. As transform matrix should not collapse at one point, an additional non-convex term constraint is added $W^\top KHKW = I$ to inflates the kernel. Therefore, the embedded data is preserved.

3.3.2 Transfer Joint Matching

The Transfer Joint Matching (TJM) algorithm aims to reduce the marginal distribution difference by first performing a dimensionality reduction step using Principal Component Analysis (PCA). Next, the MMD measure is employed in a Reproducing Kernel Hilbert Space to perform the feature matching method. At last, with the help of instance reweighing irrelevant instances are reduced and distribution difference is further minimized [78]. The TJM jointly match features and reweigh the source instances to reduce the domain differences. The formulation of TJM is given in Table 3.2,

where A, K, M_d are the adaptation matrix, kernel matrix, and MMD measure respectively, while $l_{2,1}$ sparsity constraint is added as a sparsity regularizer on the transformation matrix. Additional constraint $A^\top KHK^\top A = I$ is imposed to ensure that transform data should not collapse at one point.

3.3.3 Joint Distribution Adaptation

The Joint Distribution Adaptation (JDA) algorithm attempts to simultaneously correct the marginal and conditional distribution differences between the source and target data. The MMD distance measure is integrated into the PCA algorithm, which is used as a dimensionality reduction step to correct the marginal distribution differences. Pseudo labels are predicted by learning a classifier from the labeled source data, which are used to improve the conditional distribution differences [78]. The formulation of JDA algorithm is given in Table 3.2, where J is input data, A is transformation matrix, M_{d_c} is MMD matrices involving class labels, A is the adaptation matrix and λ is the regularization parameter to ensure that the optimization problem should be well-defined.

3.3.4 Balanced Distribution Adaptation

Balanced Distribution Adaptation (BDA) is proposed to balance the marginal and conditional distribution when marginal distribution becomes more dominant when datasets are very different and conditional distribution becomes more important for more similar datasets. Most of the transfer learning methods assume that data is balanced, but the imbalanced data widely exists in real issues. The class imbalance is handled by introducing Weighted Balanced Distribution Adaptation(W-BDA) algorithm, which adjusts the weight of each class in addition to the adaptation of distribution [79]. The formulation of BDA is given in Table 3.2. Where J is the input data, A represents transformation matrix, I denotes an identity matrix, H is the centering matrix and M_{d_0} and M_{d_c} are MMD matrices. The equation is divided into two terms, term 1 is the adaptation of marginal and conditional distribution with a balance factor, the second term is the regularization

term. Two constraints are involved in BDA equation, $A^\top J$ ensures that transformed matrix should preserve the original data properties. The second constraint denotes the range of balance factor μ.

3.4 Dictionary Learning

Before discussing our approach, first, we discuss the basic dictionary learning (DL) model. In many applications, the dictionary D is unknown and has to be learned from the training signals coming from the desired class. A DL algorithm finds an over-complete dictionary D that sparsely represents measurement Y such that $Y \simeq DX$. Each of these vectors x_i is the sparse representation of y_i with only s nonzero entries. Suppose training data from different categories denoted by $X_i \in \mathbb{R}^d (i = 1, ...N)$. In DL we want to find a dictionary $D \in \mathbb{R}^{d \times K}$ and a coefficient matrix $X = [x_i, ...x_N] \in \mathbb{R}^{K \times N}$, such that both D and X are minimized and the representations X are sparse enough. A non-traceable formulation of this problem is

$$\begin{aligned} &\min_{D,X} ||Y - DX||_F^2 \\ &\text{subject to } ||x_i||_0 \leqslant s, 1 \leqslant i \leqslant L, \end{aligned} \tag{3.3}$$

where $\|\|_0$ means vector ℓ_0 norm. Since both D and X are unknown, a common approach is to use alternating minimization in which we start with an initial guess of D and then obtain the solution iteratively by alternating between two stages, sparse coding and dictionary update. The flow of dictionary learning is shown in Figure 3.2.

3.5 Joint Transfer and Dictionary Learning Framework

The JDTL paradigm is shown in Figure 3.3. The left-hand side of the figure is the generic transfer learning modeling where different views features are brought into common subspace using different transfer learning methods. While on the right-hand side we aim to learn view independent dictionary and coefficients by using the LRSDL [66] method. More detail is given in following subsections.

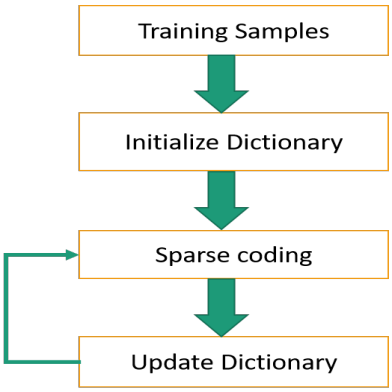


Figure 3.2: Pipeline of basic dictionary learning and sparse coding.

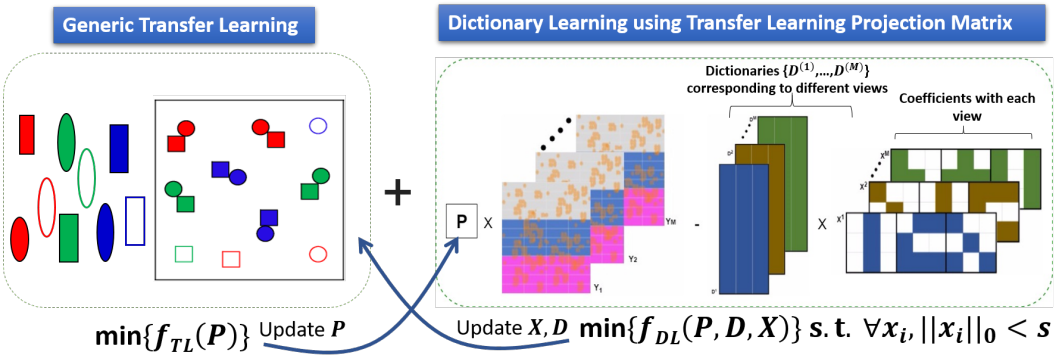


Figure 3.3: Joint dictionary and transfer learning paradigm.

3.5.1 Framework Overview

The framework of our proposed method is shown in Figure 3.4. Where first we extract features from videos of each view using IDT method, then different transfer learning methods are used to bring features from different views into common subspace, and transfer learning projection matrix is learned. After learning the projection matrix, the view independent dictionary is learned. In the last, our novel approach is used to jointly learn the dictionary and projection matrix in order to get

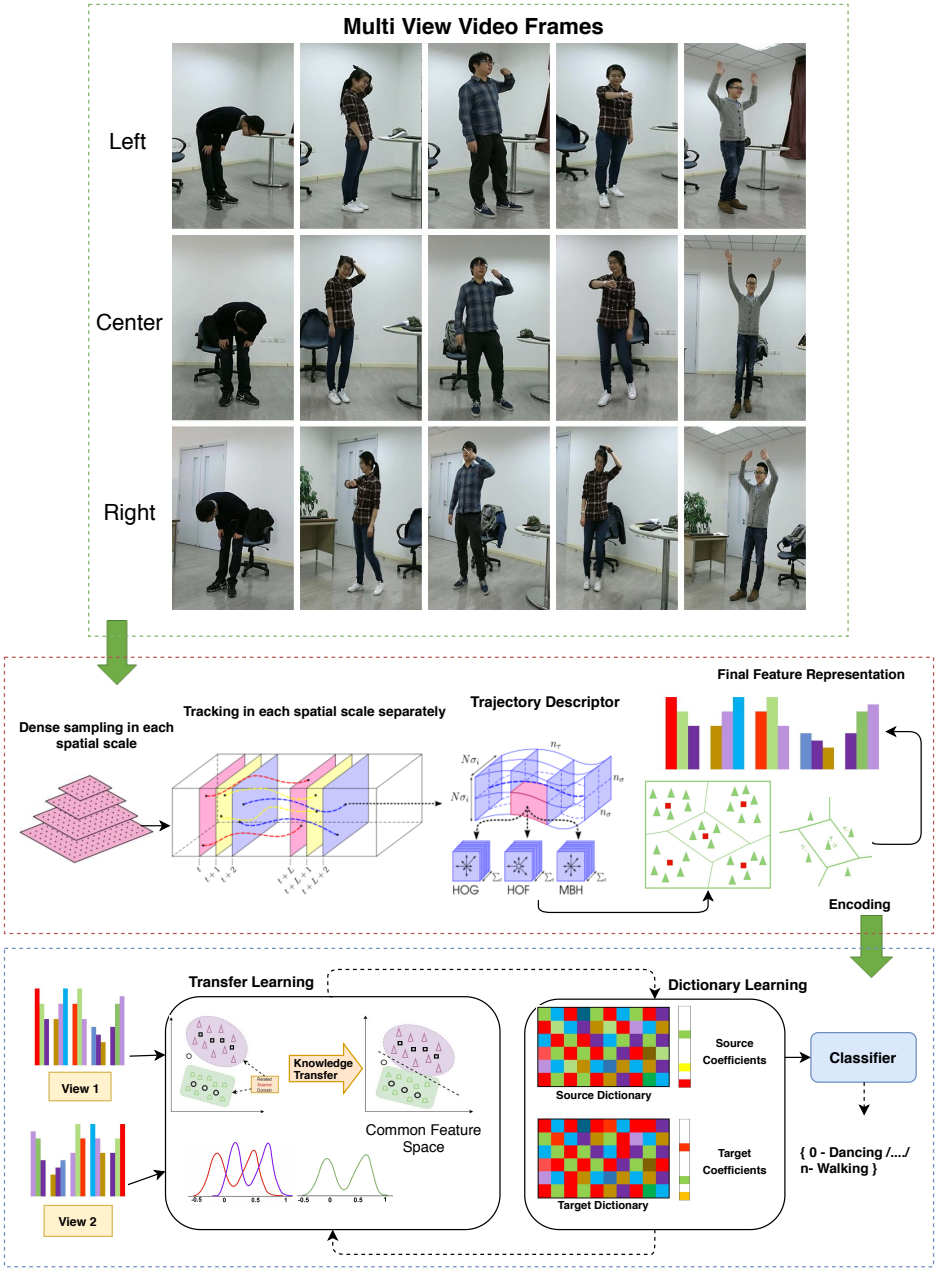


Figure 3.4: Overview of the proposed joint learning objective. Part of illustration is from [1].

the optimized dictionary, coefficients and transfer learning projection matrix. Finally, SVM is used for the classification.

3.5.2 Model Construction

Let Y be a set of m-dimensional training samples, i.e., $Y^{(v)} = [Y_1^{(v)}, Y_2^{(v)}, ..., Y_C^{(v)}]$ where Y_i denotes the training sample from class i and C denotes the number of classes for each view. The view independent dictionary is denoted by $D^{(v)} = [D_1^{(v)}, D_2^{(v)}, ..., D_C^{(v)}]$ where D_i is sub-dictionary associated with class i and v represents the views. In this approach we also learn the transfer learning projection matrix $P \in R^{m \times d}$, that projects the data into lower-dimensional space. $X^{(v)}$ is the sparse representation matrix of dimensionality reduced data $P^\top Y$ over dictionary $D^{(v)}$, $f_{\text{TL}}(\cdot)$ is the transfer learning objective function which we get from different transfer learning methods, $f_{\text{DL}}(\cdot)$ is the general dictionary learning function. Hence, our proposed novel approach is given in the following equation:

$$\begin{aligned} &\min_{P,D,X} f_{\text{TL}}(P) + \gamma f_{\text{DL}}(P, D, X), \\ &\text{s.t. } ||x_i||_0 < s \text{ and } x_i \text{ is a column of } X, \end{aligned} \tag{3.4}$$

where γ is the balancing parameter.

3.5.3 Optimization

The above equation can not be solved in closed form, so we refer to an iterative solution. The iterative process is shown in Figure 3.5 and Algorithm 1. Here our goal is to learn the projection matrix obtained via transfer learning and in the next step based on the projection matrix dictionary and sparse codes are updated. The objective function in Eq. (3.4) can be divided into three subproblems, to jointly learn projection P, dictionary D and coding coefficients X.

First, different transfer learning methods are used to get the projection matrix. The projection matrix is learned by calculating the gradient of the projection matrix. Fast Low-Rank Shared Dictionary Learning (LRSDL) framework is used for the dictionary and coefficient learning.

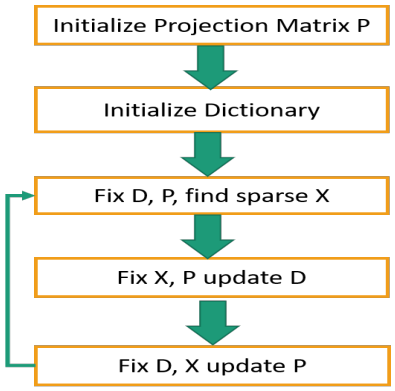


Figure 3.5: Pipeline of joint dictionary and transfer learning.

Algorithm 1 Joint Dictionary and Transfer Learning

Input
  Initialize P_i as an output from transfer learning method.
  Initialize D_i and X_i as a view independent Dictionary and Coefficients respectively
while convergence or maximal iteration step is not reached **do**
    $t \leftarrow t + 1$
    Fix P_i and D_i, update X_i using **Equation:** (3.7)
    Fix P_i and X_i, update D_i using **Equation:** (3.17)
    Fix D_i and X_i, update P_i using **Equation:** (3.26)
output P, X and D

LRSDL method automatically extracts both discriminative and shared bases. LRSDL framework is a supervised method which learns a dictionary to extract the discriminative features for each class independently and also learns the shared features [66]. In our method shared dictionary and coefficients are used. FDDL has been broadly used for exploiting both structured dictionary and learning discriminative coefficient in LRSDL [59]. Specifically, the discriminative dictionary D and the sparse coefficient matrix X is learned based on the following equation.

$$J_Y(D, X) = \frac{1}{2} f_Y(D, X) + \lambda_1 ||X||_1 + \frac{\lambda_2}{2} g(X), \tag{3.5}$$

where $f_Y(D,X) = \sum_{c=1}^{C} r_{Y_c}(D,X_c)$ is a discriminative fidelity with:

$$r_{Y_c}(D,X_c) = ||Y_c - DX_c||_F^2 + ||Y_c - D_c X_c^c||_F^2 + \sum_{j \neq c} ||D_j X_c^j||_F^2. \tag{3.6}$$

For $c = 1, \cdots, C$; $i = 0, 1, \cdots, C$. Assume that $Y_c \in \mathbb{R}^{d \times n_c}$ and $Y \in \mathbb{R}^{d \times N}$ with $N = \sum_{c=1}^{C} n_c$; $D_i \in \mathbb{R}^{d \times k_i}, D \in \mathbb{R}^{d \times K}$ with $K = \sum_{c=1}^{C} k_c$; and $X \in \mathbb{R}^{K \times N}$. Notation X^i means the sparse coefficient of Y on D_i, by $X_c \in \mathbb{R}^{K \times N_c}$ means the sparse coefficient of Y_c on D and by X_c^i means the sparse coefficient of Y_c on D_i.

The last term in $r_{Y_c}(D,X_c)$ means that D_j has small contribution to the representation of Y_c for all $j \neq c$, and $g(X) = \sum_{c=1}^{C}(||X_c - M_c||_F^2 - ||M_c - M||_F^2) + ||X||_F^2$. In $g(X)$, $M_c = \mu(X_c)$ and $M = \mu(X,n)$ is the mean matrices. n is the number of columns depending on the context. The last term $||X||_F^2$ in $g(X)$ makes the cost function convex with respect to X. These subproblems are optimized alternatively by updating one variable and fixing the others, through an iterative process. Our proposed method is discussed in the following sections.

Updating Coding Coefficients X

Assuming D and P are fixed, X is solved by following equation:

$$\min_X h(X) + \lambda ||X||_1, \tag{3.7}$$

where $h(X) = \frac{1}{2} f_{Y,D}(X) + \frac{\lambda_2}{2} g(X)$. This problem can be solved by FISTA [80]. Where gradient of $f(\cdot)$ and $g(\cdot)$ are calculated with respect to X. The gradient of $h(X)$ is calculated in FDDL method by the following equation:

$$\frac{\partial \frac{1}{2} f_{Y,D}(X)}{\partial X} = \mathcal{M}(D^\top D)X - \mathcal{M}(D^\top Y), \tag{3.8}$$

$$\frac{\partial \frac{1}{2} g(X)}{\partial X} = 2X + M - 2\underbrace{[M_1, M_2, ..., M_c]}_{\widehat{M}}, \tag{3.9}$$

Where $\mathcal{M}(\cdot)$ doubles the diagonal block of the given matrix. The row and column partition of the matrix are inferred from the given matrix and it is also computationally inexpensive function of the given matrix . $\widehat{M} = [M_1, M_2, ..., M_c] = [\mu(X_1), \mu(X_2), ..., \mu(X_c)]$ is the mean of each class specific coefficients.

Using Eq. (3.8) and Eq. (3.9) we obtain:

$$\frac{\partial h(X)}{\partial X} = (\mathcal{M}(D^\top D) + 2\lambda_2 I)X - \mathcal{M}(D^\top Y) + \lambda_2(M - 2\widehat{M}). \tag{3.10}$$

As the LRSDL is the extension of FDDL, class-specific coefficients X and shared coefficients X^0 are solved alternatively, Here X and X^0 are combined into one and find $\bar{X}$ by solving the following optimization problem:

$$\bar{X} = \arg\min_{\bar{X}} \bar{h}(\bar{X}) + \lambda_1 ||\bar{X}||_1. \tag{3.11}$$

where $\bar{h}(\bar{X}) = \frac{1}{2}\bar{f}_{Y,\bar{D}}(\bar{X}) + \frac{\lambda_2}{2}\bar{g}(\bar{X})$. This problem is solved using FISTA with the gradient of $\bar{h}(\bar{X})$:

$$\bigtriangledown \bar{h}(\bar{X}) = \begin{bmatrix} \frac{\partial \bar{h}_{X^0}(X)}{\partial X} \\ \frac{\partial \bar{h}_X(X^0)}{\partial X^0} \end{bmatrix}. \tag{3.12}$$

In the upper term, by combining the observations

$$\begin{aligned} \bar{h}_{X^0}(X) &= \frac{1}{2}\bar{f}_{Y,\bar{D},X^0}(X) + \frac{\lambda_2}{2}\bar{g}_{X^0}(X) \\ &= \frac{1}{2}\bar{f}_{\bar{Y},D}(X) + \frac{\lambda_2}{2}g(X) + \text{constant}, \end{aligned} \tag{3.13}$$

where $\bar{D} = [DD_0]$ is total dictionary and $\bar{Y} = Y - D_0X^0$ and using the equation, we have:

$$\frac{\partial \bar{h}_{X^0}(X)}{\partial X} = (\mathcal{M}(D^\top D) + 2\lambda_2 I)X - \mathcal{M}(D^\top \bar{Y}) + \lambda_2(M - 2\widehat{M}). \tag{3.14}$$

the lower term is solved by

$$\bar{h}_X(X^0) = ||V - D_0X^0||_F^2 + \frac{\lambda_2}{2}||X^0 - M^0||_F^2 + \text{constant}, \tag{3.15}$$

$$\begin{aligned} \Rightarrow \frac{\partial \bar{h}_X(X^0)}{\partial X^0} &= 2D_0^T D_0X^0 - 2D_0^T V + \lambda_2(X^0 - M^0), \\ &= (2D_0^T D_0 + \lambda_2 I)X^0 - 2D_0^T V - \lambda_2 M^0, \end{aligned} \tag{3.16}$$

where $V = Y - \frac{1}{2}D\mathcal{M}(X)$ and X^0 is the shared coefficients and M^0 is the mean vector of shared coefficients. The Eq. (3.12) can be solved by combining Eq. (3.16) and Eq. (3.14). After solving the equation, $\bigtriangledown\bar{h}(\bar{X})$, $\bar{X}$ can be updated by FISTA algorithm. We also need to compute the Lipschitz coefficient L of $\bigtriangledown\bar{h}(\bar{X})$.

Updating Dictionary D

D is updated by fixing P and X. The most important part of the shared dictionary is to represent samples from all classes. In other words, it is expected that the collaboration of the particular dictionary D_c and the shared dictionary D_0 should well represent Y_c. The discriminative fidelity term $f_Y(D, X)$ in Eq. (3.5) can be extended to $\bar{f}_Y(\bar{D}, \bar{X}) = \sum_{c=1}^{C} \bar{r}_{Y_c}(\bar{D}, \bar{X}_c)$ with $\bar{r}_{Y_c}(\bar{D}, \bar{X}_c)$ is defined as:

$$||Y_c - \bar{D}\bar{X}_c||_F^2 + ||Y_c - D_cX_c^c - D_0X_c^0||_F^2 + \sum_{j=1, j\neq c} ||D_jX_c^j||_F^2. \tag{3.17}$$

Since $\bar{r}_{Y_c}(\bar{D}, \bar{X}_c) = r_{\bar{Y}_c}(D, X_c)$ with $\bar{Y}_c = Y_c - D_0X_c^0$, we have $\bar{f}_Y(\bar{D}, \bar{X}) = f_{\bar{Y}}(D, X)$ with $\bar{Y} = Y - D_0X^0$

The atoms that are the part of shared dictionary D_0 should represent the samples from all

classes. The nuclear norm regularization $||D||_*$ is used as a convex relaxation of D_0. It forces that shared dictionary should have low-rank to prevent the dictionary from absorbing the discriminative atoms and it handles the worst case, where all atoms are the part of shared dictionary and there is no discriminative features remaining for class-specific atoms.

The contribution of each class-specific feature might be different, even the shared dictionary has low-rank. The contribution of each class-specific feature is measure by shared coefficients X_0, which they aim to avoid. A regularization term $||X^0 - M^0||$ is added that forces each x^0 to be close to mean vector m^0 for all X^0. By adding this constraint, the fisher-based discriminative coefficient term $g(x)$ can be extended to $\bar{g}(\bar{x})$ and it is defined as:

$$\bar{g}(\bar{x}) = g(X) + ||X^0 - M^0||_F^2, \tag{3.18}$$

In total, the cost function $\bar{J}_Y(\bar{D}, \bar{X})$ of the LRSDL method is:

$$\bar{J}_Y(\bar{D}, \bar{X}) = \frac{1}{2}\bar{f}_Y(\bar{D}, \bar{X}) + \lambda_1||\bar{X}||_1 + \frac{\lambda_2}{2}\bar{g}(\bar{X}) + \eta||D^0||_*. \tag{3.19}$$

The shared dictionary and class-specific dictionary can be found by minimizing the objective function. $\bar{D}, \bar{X}$ become D, X respectively, $\bar{J}_Y(\bar{D}, \bar{X})$ becomes $J_Y(D, X)$ and LRSDL reduces to FDDL in case if there is no shared dictionary.

The FDDL method updates each class-specific dictionary by fixing the X. This process takes time for convergence, so in LRSDL the total dictionary D is optimized while fixing X instead of updating the class-specific dictionaries, so this problem is given in below equation:

$$\min_D \ f_{Y,X}(D) \Rightarrow \min_D \big\{ -2\mathbf{tr}(ED^\top) + \mathbf{tr}(FD^\top D)\big\}, \tag{3.20}$$

where $E = Y\mathcal{M}(X^\top)$, **tr(·)** means the trace of a matrix, and $F = \mathcal{M}(XX^\top)$.

The LRSDL method finds total dictionary $\bar{D} = [D, D_0]$. So, the method solves the D and D_0 separately. First, to update the D in Eq. (3.17), assuming $\bar{f}_y(\bar{D}, \bar{X}) = f_Y(D, X)$ with $\bar{Y} \triangleq Y_{D_0} X^0$, the E-FDDL-D algorithm for D is similar to Eq. (3.20). Second, D_0 is optimized through the following equation when D, X in Eq. (3.19) are fixed,

$$\begin{aligned} \bar{J}_{Y,D,X}(D_0, X_0) =& ||V - D_0 X_0||_F^2 + \frac{\lambda_2}{2}||X^0 - M^0||_F^2 + \\ & \eta||D_0||_* + \lambda_1||X^0||_1 + \text{constant}, \end{aligned} \tag{3.21}$$

and $V = Y - \frac{1}{2} D\mathcal{M}(X)$. The D_0 is updated by solving following equation based on the above equation:

$$\min_{D_0} \ \mathbf{tr}(F D_0^\top D_0) - 2\mathbf{tr}(E D_0^\top) + \eta||D_0||_*, \tag{3.22}$$

where $E = V(X^0)^T$; $F = X^0(X^0)^T$ using the ADMM [81] method and singular value thresholding algorithm [82]. In ADMM process we choose ρ that should be greater than 0 and $Z = U = D_0$ is initialized, then the following problems are solved alternatively until they meet convergence.

$$\begin{aligned} &\min_{D_0} -2\mathbf{tr}(\bar{E} D_0^\top) + \mathbf{tr}(\bar{F} D_0^\top D_0), \\ &\text{with } \bar{E} = E + \frac{\rho}{2}(Z - U); \bar{F} = F + \frac{\rho}{2} I, \end{aligned} \tag{3.23}$$

$$Z = \mathcal{D}_{\eta/\rho}(D_c + U), \tag{3.24}$$

where $\mathcal{D}$ is shrinkage thresholding operator [82].

$$U = U + D_0 - Z. \tag{3.25}$$

The Eq. (3.23) can be solved by ODL [83] and the Eq. (3.24) and Eq. (3.25) are computationally inexpensive.

Updating Projection P

First different transfer learning methods are used to get the projection matrix. Then, we minimize the projection matrix by calculating the gradient of the projection matrix. The projection matrix is optimized by using the similar way as sparse coefficients X are optimized. Here, we used the FDDL method to optimize the projection matrix.

$$\min_{P} \frac{1}{2} f_{Y,D,X}(P) + \frac{\lambda_2}{2} g(P). \tag{3.26}$$

This problem can be solved by FISTA, where we need to calculate gradient with respect to P. The gradients of the first and second terms can be calculated by FDDL method using the equations given below.

$$\frac{\partial \frac{1}{2} f_{Y,D}(P)}{\partial P} = \mathcal{M}(D^\top D)P - \mathcal{M}(D^\top (P^\top Y)), \tag{3.27}$$

$$\frac{\partial \frac{1}{2} g(P)}{\partial P} = 2P + M - 2\underbrace{[M_1, M_2, ..., M_c]}_{\widehat{M}}. \tag{3.28}$$

Using Eq. (3.27) and (3.28) we obtain:

$$\frac{\partial h(P)}{\partial P} = (\mathcal{M}(D^\top D) + 2\lambda_2 I)P - \mathcal{M}(D^\top (P^\top Y)) + \lambda_2 (M - 2\widehat{M}). \tag{3.29}$$

Chapter 4 Experimental Setup

The performance of our method is evaluated on the PKU-MMD [39], a multi-view dataset. The dataset consists of 51 different classes and each action is captured from three different angles. In our experiment, we used only five action classes and multi-view videos. In PKU-MMD dataset each video contains different number of actions that requires video segmentation to segment them into individual action. The video contains multiple objects in the background and those objects are not required in our study, so a human detector is used to crop video frames. The detail description of the dataset, data preprocessing and experimental setup is given in this chapter.

4.1 PKU-MMD Dataset

PKU-MMD is a large-scale dataset. This dataset mainly focuses on long continuous sequences action detection and multi-modality analysis. This dataset contains 51 action classes in total. These classes are divided into two parts: 41 daily routine actions (drinking, brushing your teeth, waving hand, etc.) and 10 interaction actions (hugging, shaking hands, giving something to another person, kicking another person, etc.).

The dataset was captured in a daily life indoor environment through Kinect sensor. Videos are captured from three different horizontal views on fixed height and angle. Horizontal angles of each camera is -45°, 0°, and +45° with the height of 120 cm. The dataset was collected by 66 distinct subjects, each subject took part in 4 daily-life actions and 2 interactive actions. Figure 4.1 represents PKU-MMD dataset where different actions are performed in different views by different subjects.

4.2 Data Preprocessing

The data preprocessing consist of two steps. In the first step, large videos containing multiple actions are segmented into smaller videos which contain only a single action. Then an object

Figure 4.1: PKU-MMD video sample frames from three different angles.

detector is used to detect humans and other objects in video frames. The object detector also gives the bounding box information and based on that information video frames are cropped to remove the background objects, which are not required for the study.

4.2.1 Data Segmentation

The dataset has 1000+ long action sequences, each of which lasts about 3$\sim$4 minutes and has about 20 action instances. For each long action sequence, the data label file is provided that contains the label information for each action in video and the starting and ending frame of an action. Based on the label file information, each video is segmented. After segmenting each view, the dataset has 3,000 minutes video with 20,000+ temporally localized actions.

4.2.2 Video Frame Cropping

The PKU-MMD dataset is collected through Kinect V2 sensor with a resolution of 1920×1080 pixels. While the action setup area is 180 cm as length and 120 cm as width. To crop the images, we have to first detect the person in the frame and based on that we have to crop the frames. For the person detection, we used YOLO (You Only Live Once) [84] the state-of-the-art, real-time object detection system that can detect over 9000 objects.

The YOLO does not only detect objects in the video frame, but also labels them as shown in Figure 4.2. It outputs the bounding box positions for each frame. So, in our method, we first used YOLO to detect the person in videos and get the bounding box values for the detected person in each frame. Then, single maximum bounding box value is chosen from all video frames and based on the maximum bounding box size for a person of all videos.

4.3 Fisher Vector Encoding

To evaluate the performance of IDT, we used a standard FV approach. A codebook for each descriptor is constructed (trajectory, HOG, HOF, MBH) and a codebook is constructed for the combination of all descriptors. The different number of centers were used which varies from 32 to 256. We fixed it to 64 in our experiments because the performance is very stable compared to other number of centers. Note, the FV encoded descriptor dimension for each video is 1×54528. To avoid the curse of dimensionality and reduce the computational burden, feature dimensions are further reduced to 400 using PCA. SVM and LibSVM with Linear Kernel are employed for classification.

4.4 Experimental Setting

The IDT code was compiled using the same setting as described in [1]. We complied this code on 16GB Linux (Ubuntu 16.04) machine with i7-7700k (4.2 GHz). In order to compile the IDT code,

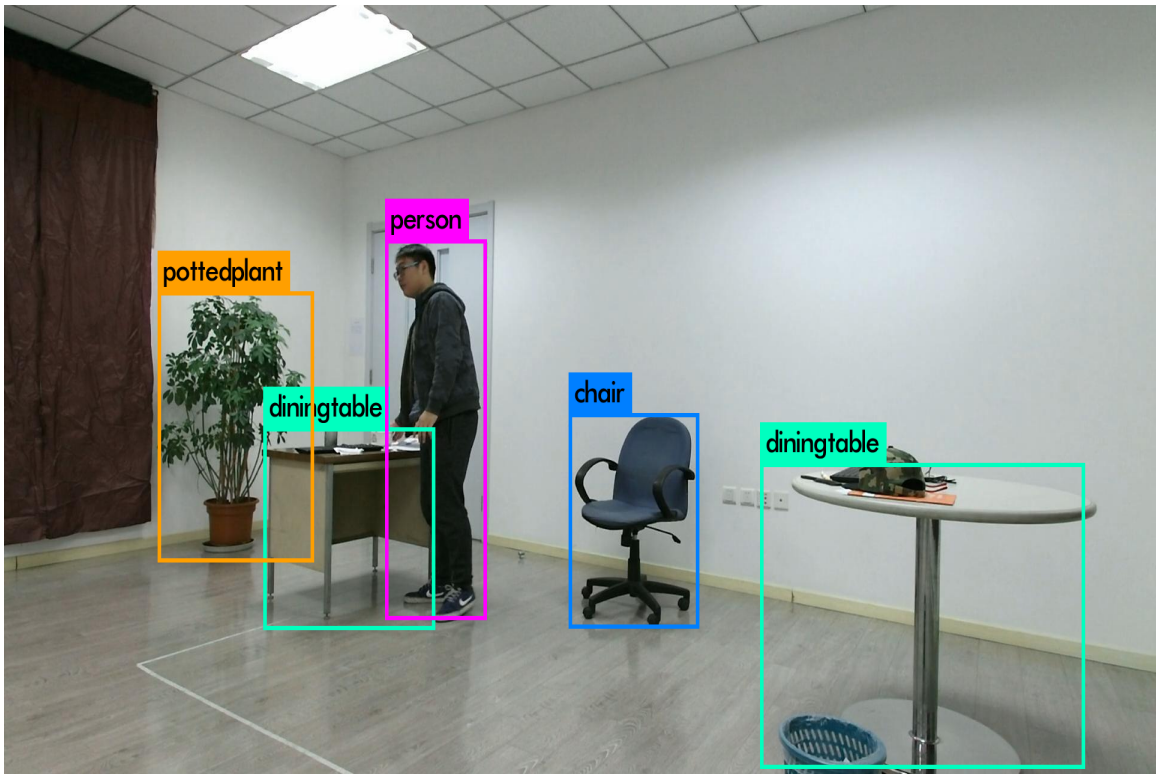


Figure 4.2: Detection on PKU-MMD data frame.

Opencv 2.4.11 and ffmpeg-0.11.1 were installed on the machine. The rest of the experiments were performed using MATLAB R2017a.

Chapter 5 Experimental Results

In this chapter, we will evaluate:

- Different IDT descriptors for the single view action recognition
- Cross-view action recognition results with/without transfer learning
- Cross-view action recognition results with joint dictionary and transfer learning

Detailed results and model parameters analysis will be found in the following sections.

5.1 Comparison of Different Descriptors

Different IDT descriptors are compared to check the performance of each individual descriptor (i.e., HOG, HOF, MBH, TRA) and their combination of all descriptors for each view. The results of MBH are the combination of MBHx, and MBHy channel. The results in Table 5.1 show that IDT descriptor performance is better than other descriptors for the single view recognition, so this motivates us to apply IDT descriptor for later action recognition experiments.

Table 5.1: Comparison of IDT descriptors on single view classification.

Data	HOF	HOG	MBH	TRA	IDT
View1	94.25%	93.10%	95.40%	81.60%	93.10%
View2	96.55%	89.65%	64.58%	77.61%	100%
View3	93.10%	93.10%	94.20%	80.06%	97.70%

5.2 Cross-View Action Recognition without TL

To conduct ablation study and show the effectiveness of our joint modeling, first, cross-view action recognition is carried out without using any transfer learning method. The model is trained on

Fisher Vector encoded video descriptors after applying dimension reduction with PCA. Basically, the model is trained on one view and tested on another view. In the meanwhile, the single view recognition result is also shown to compare the cross-view results. The results are shown in Table 5.2. We can see that the single view accuracy (on the diagonal) is between 95% and 100% while the cross-view accuracy for different cross-view settings are between 40% and 55%, which is much more challenging.

Table 5.2: Cross-view action recognition without TL.

	Training Data		
Testing Data	View1	View2	View3
View1	96%	44.93%	45.26%
View2	41.93%	100%	48.87%
View3	45.55%	52.82%	98.85%

5.3 Cross-View Action Recognition by TL Methods.

In this section, different transfer learning methods are applied for cross-view settings, and compared with results without using the transfer learning method under the same setting. The evaluation is done using the four state-of-the-art methods, i.e., TCA, TJM, JDA, and BDA, and their details can be found from the methodology section.

Parameters Setting. Note we do not apply PCA for dimension reduction as all transfer learning methods implement dimension reduction by themselves. For all methods λ is set to 1, γ is set to 0.1 as given in [78], and number of dimensions are set to 400. The iteration number is set to 5 due to its performance for all methods except TCA since it is not an iterative process. The BDA requires μ value which is set to 0 as it provided by default in their package and the linear kernel SVM is used for all methods.

Results for cross-view action recognition using different transfer learning methods are listed in Table 5.3. From the table, we can see JDA renders higher accuracy in comparison to all other methods, while TJM results are close to JDA results for cross-view action recognition from View2 to View3 and the results for TCA and BDA are between 45% and 60%. This indicates that Transfer Learning methods are performing better as compared to the normal approach without knowledge transfer.

Table 5.3: Cross-view action recognition by TL methods.

Source	Target	TCA	TJM	JDA	BDA
View1	View2	47.04%	56.22%	71.53%	53.22%
View2	View1	59.47%	72.81%	71.82%	59.37%
View1	View3	62.18%	64.58%	69.83%	42.92%
View3	View1	47.38%	63.09%	72.39%	45.92%
View2	View3	52.97%	71.67%	72%	49.43%
View3	View2	55%	74.96%	75.67%	57.99%

5.4 Cross-View Action Recognition by DL Method

In this section, cross-view action recognition is carried out on the dictionary learning features. The model is trained on Fisher Vector encoded video descriptors after applying PCA. The model is trained on one view and tested on another view and the single view recognition result is also shown. The results given in Table 5.4 demonstrate that the single view accuracy is between 95% and 100% while cross-view accuracies are between 40% and 55%.

5.5 Cross-View Action Recognition by JDTL

In this section, we will evaluate our proposed joint learning approach for cross-view action recognition where the model is trained on one view and tested on another view. Table 5.5 summarizes the highest accuracy achieved by our novel approach using different cross-view

Table 5.4: Cross-view action recognition by DL.

	Training Data		
Testing Data	View1	View2	View3
View1	85.05%	42.18%	34.13%
View2	42.18%	94.25%	49.08%
View3	42.03%	45.06%	91.95%

settings and different transfer learning methods. From the table, we can see that TCA is performing well, as accuracy is greater than 90% for all cross-view combinations. The highest accuracy we got using JDA is 92%, while BDA and TJM accuracy is between 85% and 95%.

Table 5.5: Cross-view action recognition by JDTL.

Training Data	Testing Data	TCA	JDA	BDA	TJM
View1	View2	92.8%	92.01%	94.51%	91.77%
View1	View3	94.02%	89.98%	93.38%	83.84%
View2	View1	91.34%	91.50%	92.48%	90.69%
View2	View3	96.61%	92.41%	90.47%	92.89%
View3	View1	91.83%	88.56%	92.32%	88.24%
View3	View2	90.32%	86.94%	88.55%	85.65%

Next, we will lead detailed discussions on three different cross-view settings. As our method works in an iterative way to optimize the features, we are allowed to observe the accuracy change in each iteration for different transfer learning methods. The following sections also discuss the confusion matrix where we are allowed for further diagnosis for our cross-view model.

5.5.1 Cross-View Action Recognition by JDTL on View1 and View2

Table 5.6 and Figure 5.1 show the results of cross-view action recognition by training the model on View1 and testing on View2 using our novel proposed approach. Table 5.7 and Figure 5.2 show the results of cross-view action recognition by training on View2 and testing on View1. From these cross-view settings, we can see that BDA and JDA (supervised learning mode) performance is increasing in each iteration and BDA also handles class imbalance. TCA performance in the first six iterations is increasing, but after that there is a small decrease in accuracy of each iteration.

Table 5.6: Cross-view action recognition by JDTL trained on View1 and tested on View2.

Iteration No.	TCA	JDA	BDA	TJM
1	79.83%	69.03%	82.41%	85.16%
2	92.25%	73.38%	90.32%	91.77%
3	90.8%	70.48%	88.38%	85.8%
4	89.67%	80.96%	92.58%	79.51%
5	93.54%	87.74%	94.03%	75.84%
6	92.25%	90.32%	92.51%	78.81%
7	90.96%	90.01%	94.51%	81.77%
8	87.74%	92.01%	92.1%	84.7%
9	88.06%	91.93%	91.61%	82.6%
10	86.45%	91.61%	92.1%	87.41%

Confusion matrices are illustrated in Figure 5.3 and 5.4 to identify which actions cause confusion with other actions. Each column represents the ground truth, while each row represents the predicted results. As seen from Figure 5.3, our model presents appealing results using different transfer learning methods for most action classes in cross-view action recognition such as “Brushing Teeth”, “Check Time” and “Cheer Up”. The accuracy of JDA and BDA is relatively worse, because for most of the cases, one of the action classes are not recognized properly, e.g., “Bow”. Figure 5.4 shows the confusion matrices when the model is trained on View2 and tested

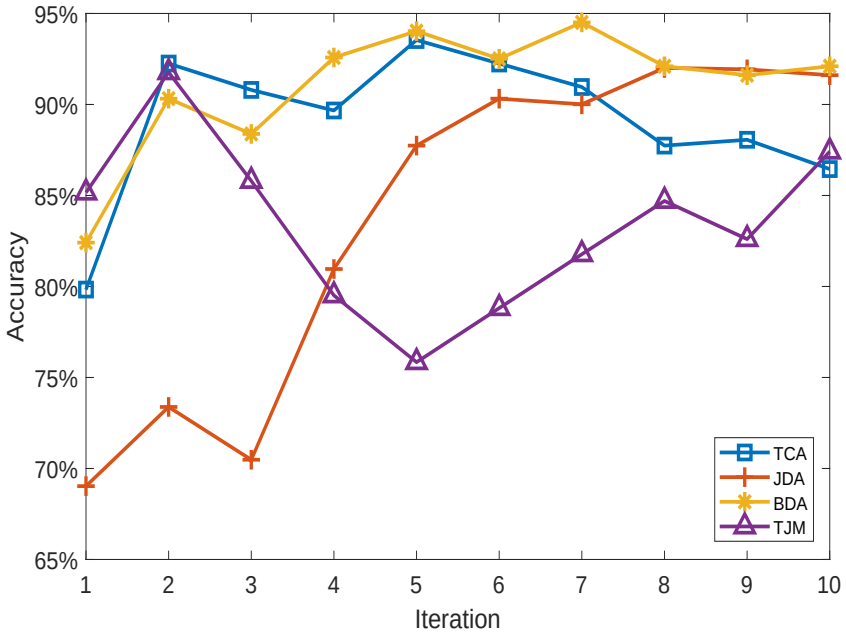


Figure 5.1: Cross-view action recognition by JDTL trained on View1 and tested on View2.

on View1. All classes are properly classified in BDA except “Brushing Teeth” which is mostly confused with “Bow” action. The “Bow” action is creating confusion for other transfer learning methods, such as in TCA, it is confused with “Cheer Up” action, while in JDA and TJM, it is recognized as “Brushing Hair”.

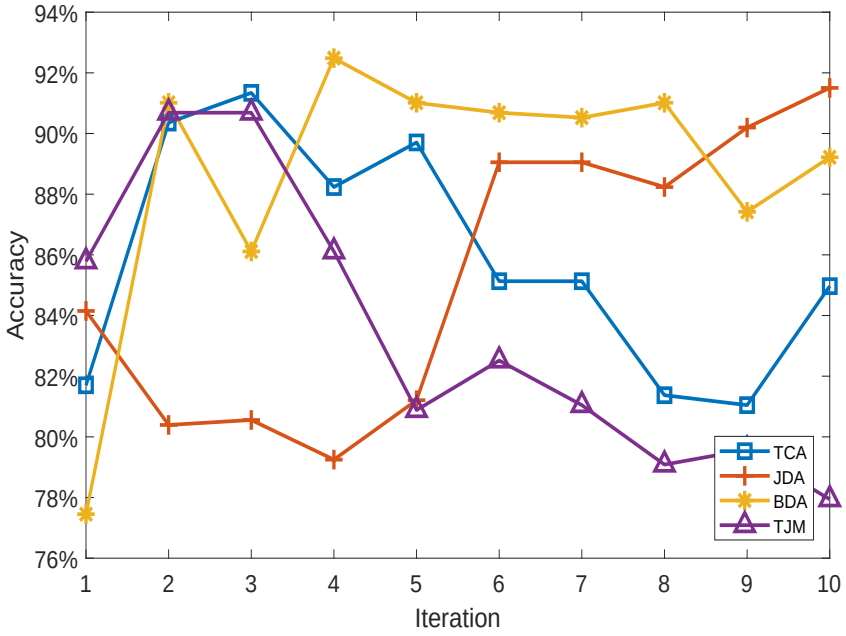
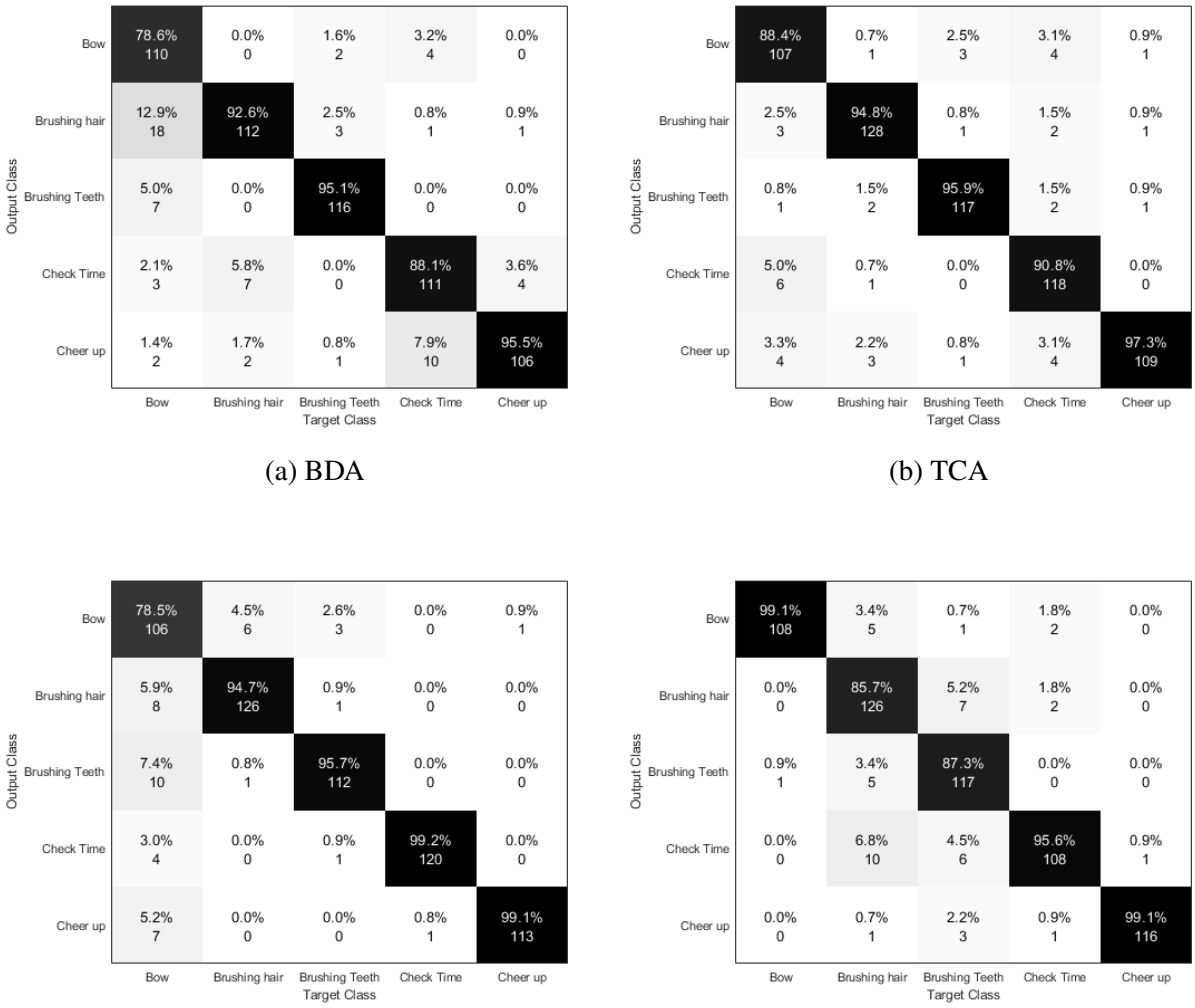


Figure 5.2: Cross-view action recognition by JDTL trained on View2 and tested on View1.

Table 5.7: Cross-view action recognition by JDTL trained on View2 and tested on View1.

Iteration No.	TCA	JDA	BDA	TJM
1	81.70%	84.15%	77.45%	85.78%
2	90.36%	80.39%	91.01%	90.69%
3	91.34%	80.56%	86.11%	90.69%
4	88.24%	79.25%	92.48%	86.11%
5	89.71%	81.21%	91.01%	80.88%
6	85.13%	89.05%	90.69%	82.52%
7	85.13%	89.05%	90.52%	81.05%
8	81.37%	88.24%	91.01%	79.08%
9	81.05%	90.20%	87.42%	79.58%
10	84.97%	91.50%	89.22%	77.94%



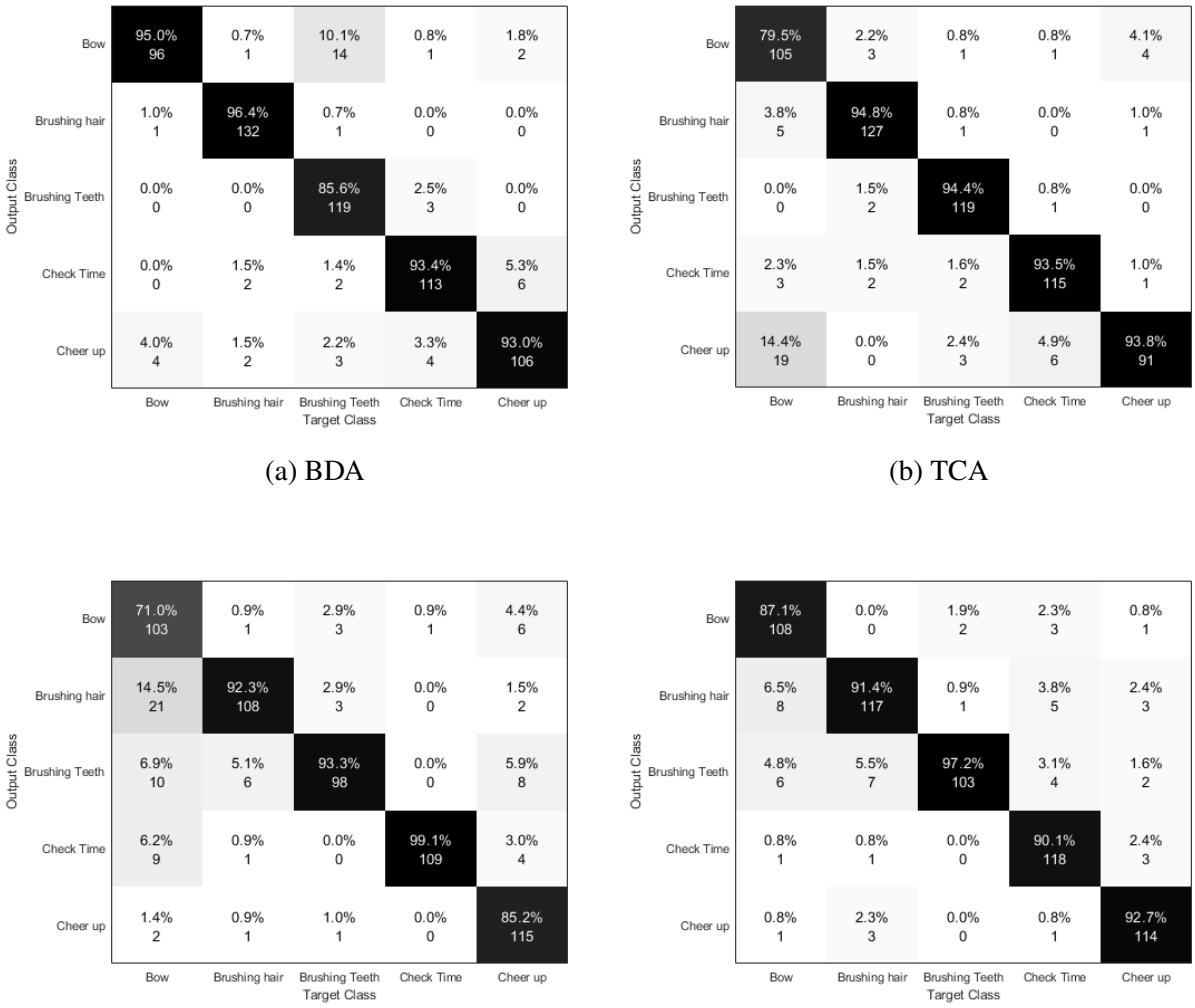
(a) BDA

(b) TCA

(c) JDA

(d) TJM

Figure 5.3: Confusion matrix trained on View1 and tested on View2



(a) BDA (b) TCA

(c) JDA (d) TJM

Figure 5.4: Confusion matrix trained on View2 and tested on View1.

5.5.2 Cross-View Action Recognition by JDTL on View1 and View3

Here, we trained the model on View1 and tested on View3 and vice versa. Table 5.6 and Figure 5.5 summarize the results for View1 to View3 while Table 5.7 and Figure 5.6 show the results for View3 to View1. View1 and View3 have mirror videos as those are captured in exact opposite

angles. From Figure 5.5, we can see that TCA and BDA methods are performing well while JDA and TJM do not have such good results. But in Figure 5.6, we can see BDA is performing well as compared to the other methods. The accuracy is increasing for the first five iterations in BDA, and then it is varying with each iteration. In TCA, performance of the first three iteration is increasing, and then it is decreasing with each iteration. The accuracy of JDA and TJM methods does not change too much with each iteration.

Table 5.8: Cross-view action recognition by JDTL trained on View1 and tested on View3.

Iteration No.	TCA	JDA	BDA	TJM
1	86.75%	80.29%	86.27%	70.44%
2	90.31%	77.38%	89.98%	67.53%
3	91.11%	73.02%	93.38%	67.04%
4	92.41%	70.60%	93.05%	69.79%
5	94.02%	80.45%	91.92%	72.54%
6	91.76%	82.07%	87.40%	73.51%
7	93.70%	79.97%	88.85%	78.35%
8	92.73%	83.52%	86.59%	77.71%
9	92.25%	85.62%	85.46%	83.68%
10	93.38%	89.98%	89.34%	83.84%

Confusion matrices for these cross-view combinations are given in Figure 5.7 and Figure 5.8 for View1 to View3 and View3 to View1, respectively. Figure 5.7 shows that in BDA and TJM "Brushing Teeth" is misclassified as "Check Time" or "Cheer Up", while in TCA "Brushing Hair" and "Check Time" are mostly confused with each other. From Figure 5.8, it can be seen that "Bow" action is misclassified with all other actions in BDA, JDA and TJM methods, while in JDA and BDA methods "Cheer Up" action is recognized correctly. In TCA "Brushing Hair" is misclassified with "Check Time" and "Cheer Up" actions.

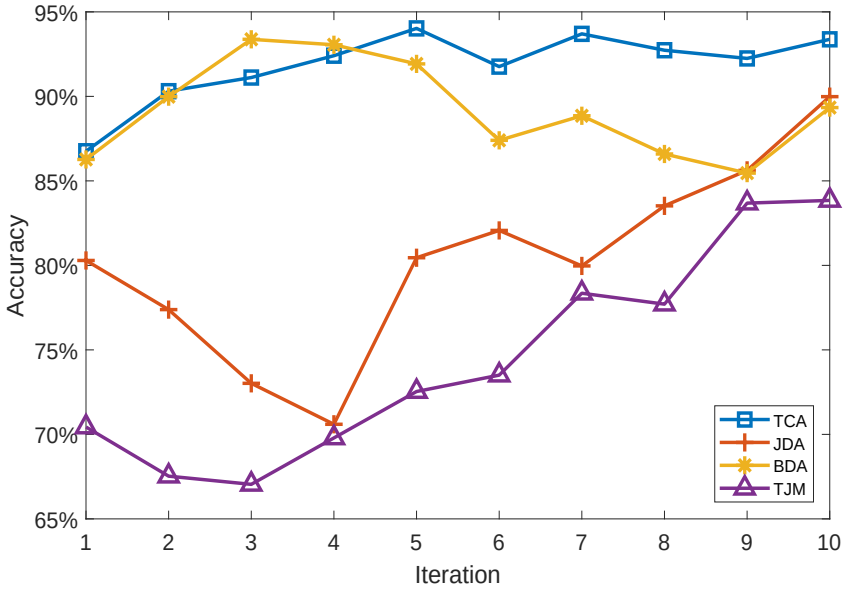


Figure 5.5: Cross-view action recognition by JDTL trained on View1 and tested on View3.

Table 5.9: Cross-view action recognition by JDTL trained on View3 and tested on View1.

Iteration No.	TCA	JDA	BDA	TJM
1	70.42%	84.31%	72.22%	79.90%
2	87.25%	86.93%	88.07%	84.48%
3	91.83%	86.93%	90.52%	86.44%
4	87.58%	84.31%	87.91%	86.11%
5	85.46%	88.56%	92.32%	85.29%
6	85.62%	82.35%	88.40%	85.13%
7	84.80%	84.31%	89.22%	85.46%
8	84.64%	86.60%	88.07%	86.76%
9	84.15%	85.95%	90.52%	88.24%
10	83.33%	85.29%	90.36%	86.44%

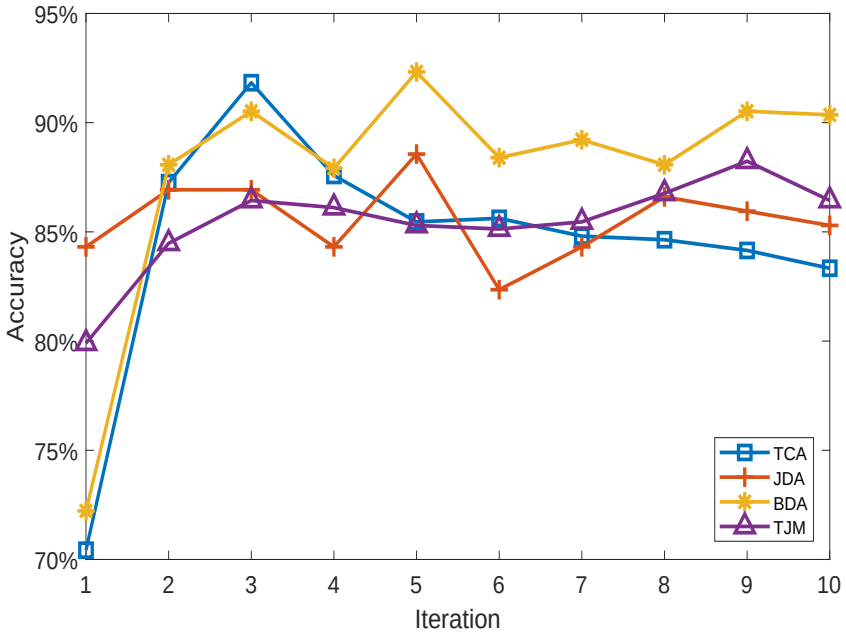
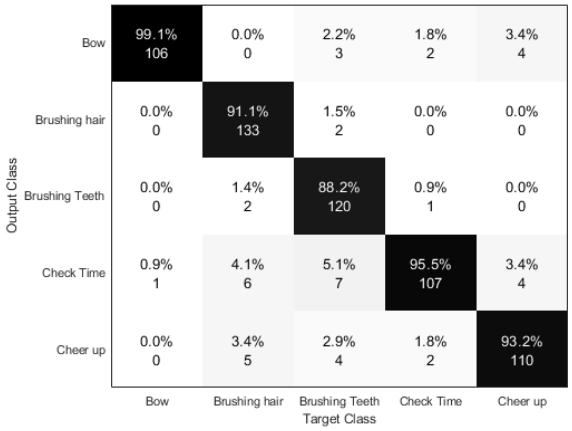


Figure 5.6: Cross-view action recognition by JDTL trained on View3 and tested on View1.

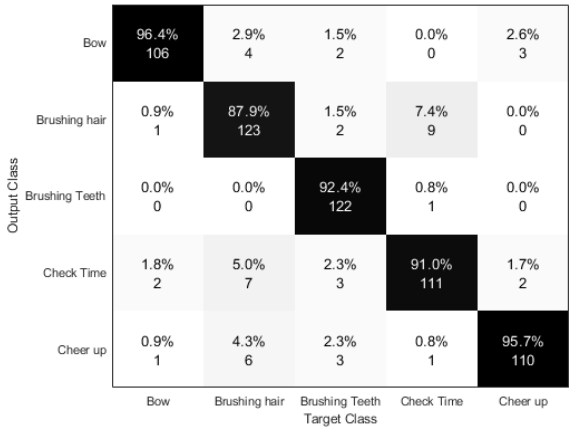
5.5.3 Cross-View Action Recognition by JDTL on View2 and View3

Table 5.10 and Figure 5.9 show the results for cross-view action recognition when the model is trained on View2 and tested on View3 and Table 5.11 and Figure 5.10 show the results when the model is trained on View3 and tested on View2. We can see in Figure 5.9, TCA accuracy is increasing until the seventh iteration and then it is decreasing. In BDA, accuracy increases until sixth iteration and in TJM it increases until fourth iteration. Then performance decreases in both methods. In JDA, accuracy increases until fifth iteration and then it is changing with different iterations.

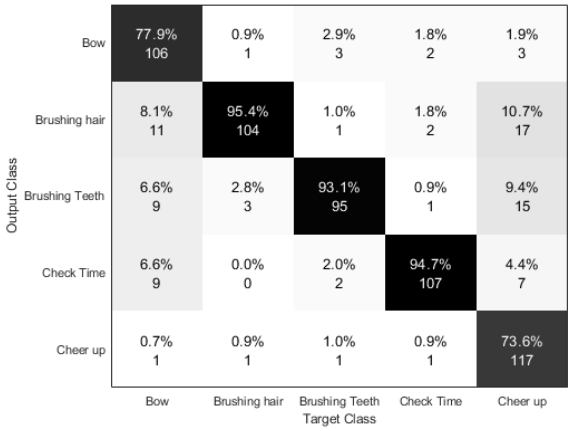
The Figure 5.11 shows which action classes are creating confusion when the model is trained on View2 and tested on View3. From the confusion matrices, we can see that “Bow” action is creating



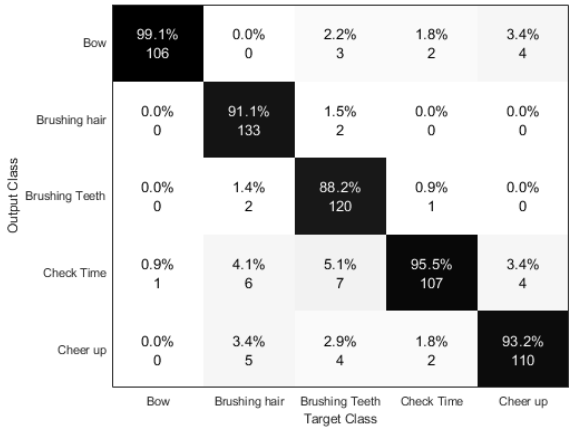
(a) BDA



(b) TCA

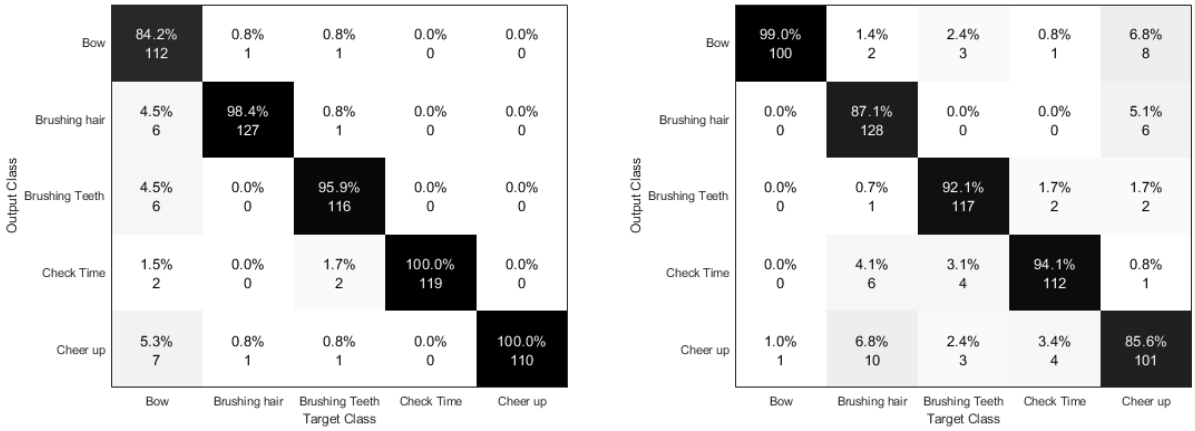


(c) JDA



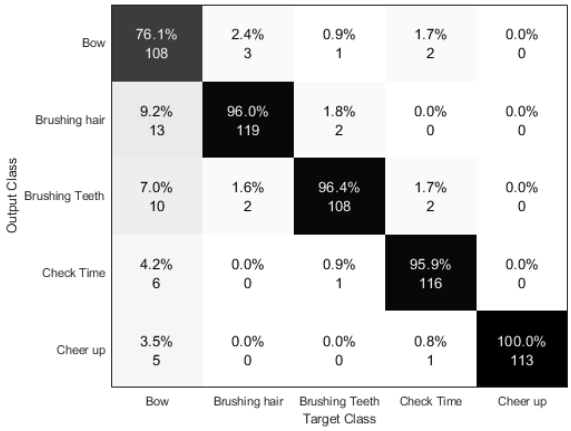
(d) TJM

Figure 5.7: Confusion matrix trained on View1 and tested on View3

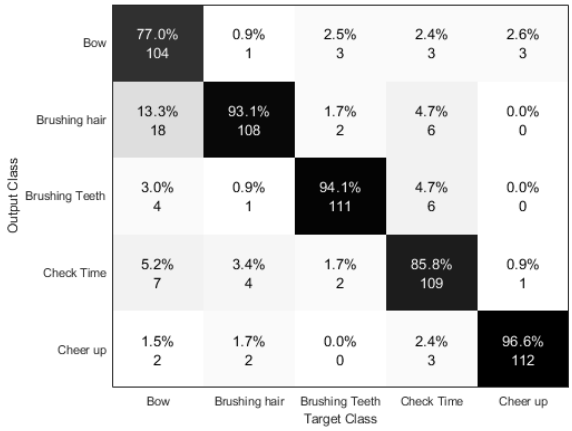


(a) BDA

(b) TCA



(c) JDA



(d) TJM

Figure 5.8: Confusion matrix trained on View3 and tested on View1.

the confusion in BDA, JDA, and TJM transfer learning methods. While in TCA “Brushing Teeth” is recognized as the “Cheer Up” or “Bow” actions.

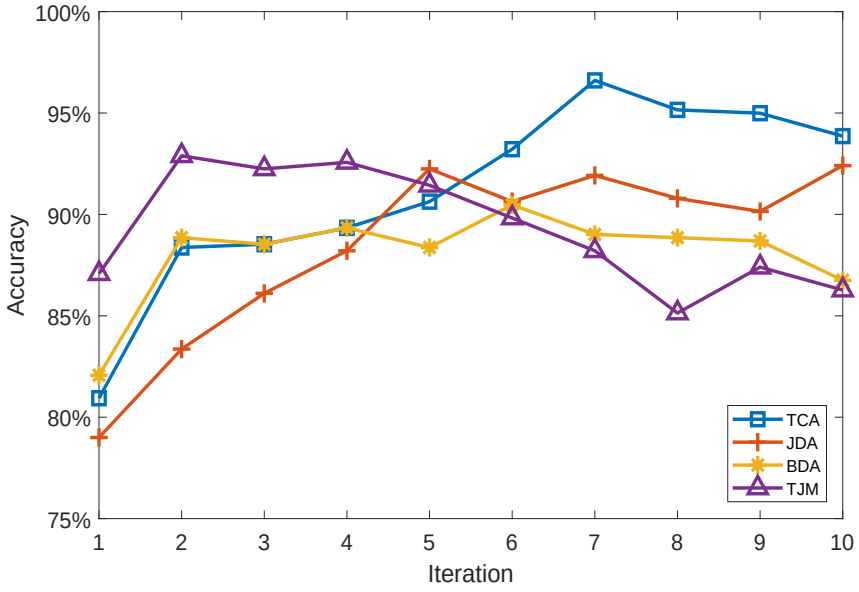


Figure 5.9: Cross-view action recognition by JDTL trained on View2 and tested on View3.

From Figure 5.10, we can see that the highest accuracy for TCA, JDA, and TJM is achieved in the second iteration, while in JDA it is achieved in sixth and eighth iteration. As we can see in Figure 5.10 that JDA accuracy seems to be increasing after third iteration while BDA accuracy is decreasing after second iteration. The accuracy of TCA is stabilized over the iteration while TJM is very unstable. The confusion matrix for this cross-view combination is shown in Figure 5.12. “Bow” and “Cheer Up” actions are mostly misclassified with different actions in all transfer learning methods. In TCA and BDA “Cheer Up” and “Check Time” are misclassified with each other.

Table 5.10: Cross-view action recognition by JDTL trained on View2 and tested on View3.

Iteration No.	TCA	JDA	BDA	TJM
1	80.94%	79.00%	82.07%	87.08%
2	88.37%	83.36%	88.85%	92.89%
3	88.53%	86.11%	88.53%	92.25%
4	89.34%	88.21%	89.34%	92.57%
5	90.63%	92.25%	88.37%	91.44%
6	93.21%	90.63%	90.47%	89.82%
7	96.61%	91.92%	89.01%	88.21%
8	95.15%	90.79%	88.85%	85.14%
9	94.99%	90.15%	88.69%	87.40%
10	93.86%	92.41%	86.75%	86.27%

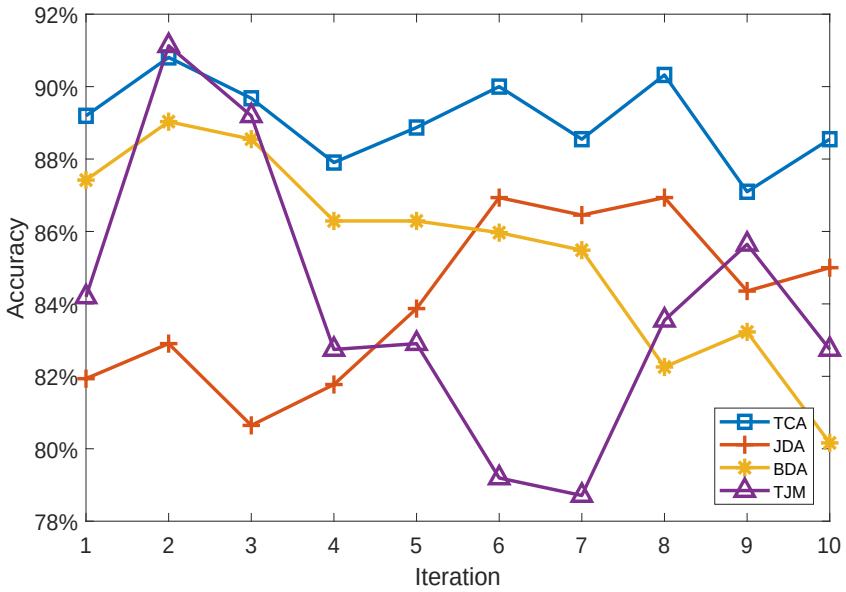


Figure 5.10: Cross-view action recognition by JDTL trained on View3 and tested on View2.

Table 5.11: Cross-view action recognition by JDTL trained on View3 and tested on View2.

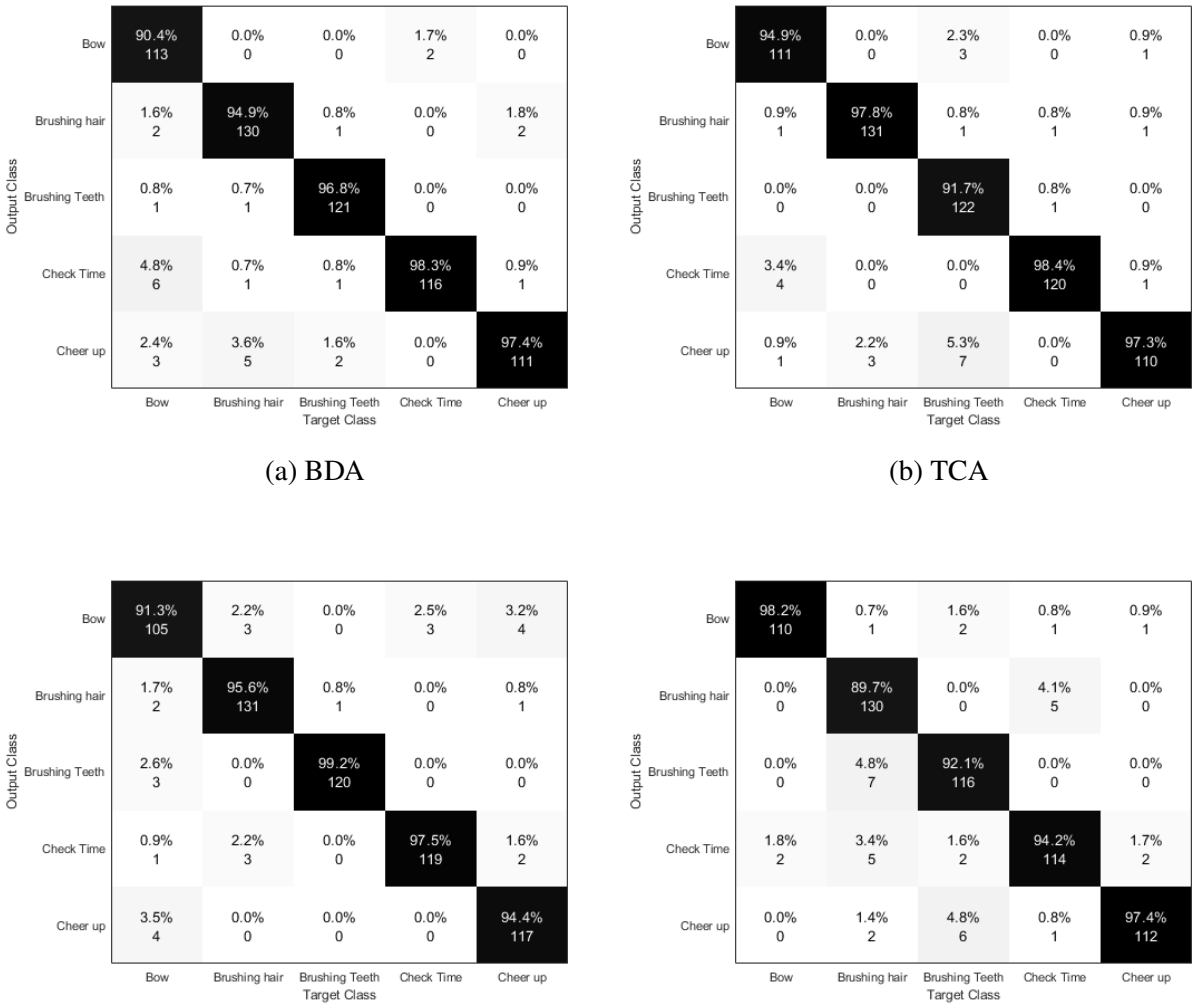
Iteration No.	TCA	JDA	BDA	TJM
1	89.19%	81.94%	87.42%	84.19%
2	90.81%	82.90%	89.03%	91.13%
3	89.68%	80.65%	88.55%	89.19%
4	87.90%	81.77%	86.29%	82.74%
5	88.87%	83.87%	86.29%	82.90%
6	90.00%	86.94%	85.97%	79.19%
7	88.55%	86.45%	85.48%	78.71%
8	90.32%	86.94%	82.26%	83.55%
9	87.10%	84.35%	83.23%	85.65%
10	88.55%	85.00%	80.16%	82.74%

5.6 Discussion

From the experimental results, it can be concluded that we obtained significant results from different cross-view settings by training the model on one view and recognizing the same action from different views using our joint dictionary and transfer learning method (JDTL). The effectiveness of our proposed approach has been demonstrated by comparing our models with those trained with, or without transfer learning and dictionary learning methods.

As we have discussed earlier that our method works in an iterative manner, the accuracy of most of the transfer learning methods is increasing until fifth iteration. After that, the performance becomes different, which may be caused by the confusion between two actions. For example, from the confusion matrices, we can learn that mostly “Bow” action is confused with “Brushing Hair” action.

The JDTL method is efficient as compared to other cross-view or multi-view action recognition methods because in other methods common feature representation is extracted by constructing a



(a) BDA

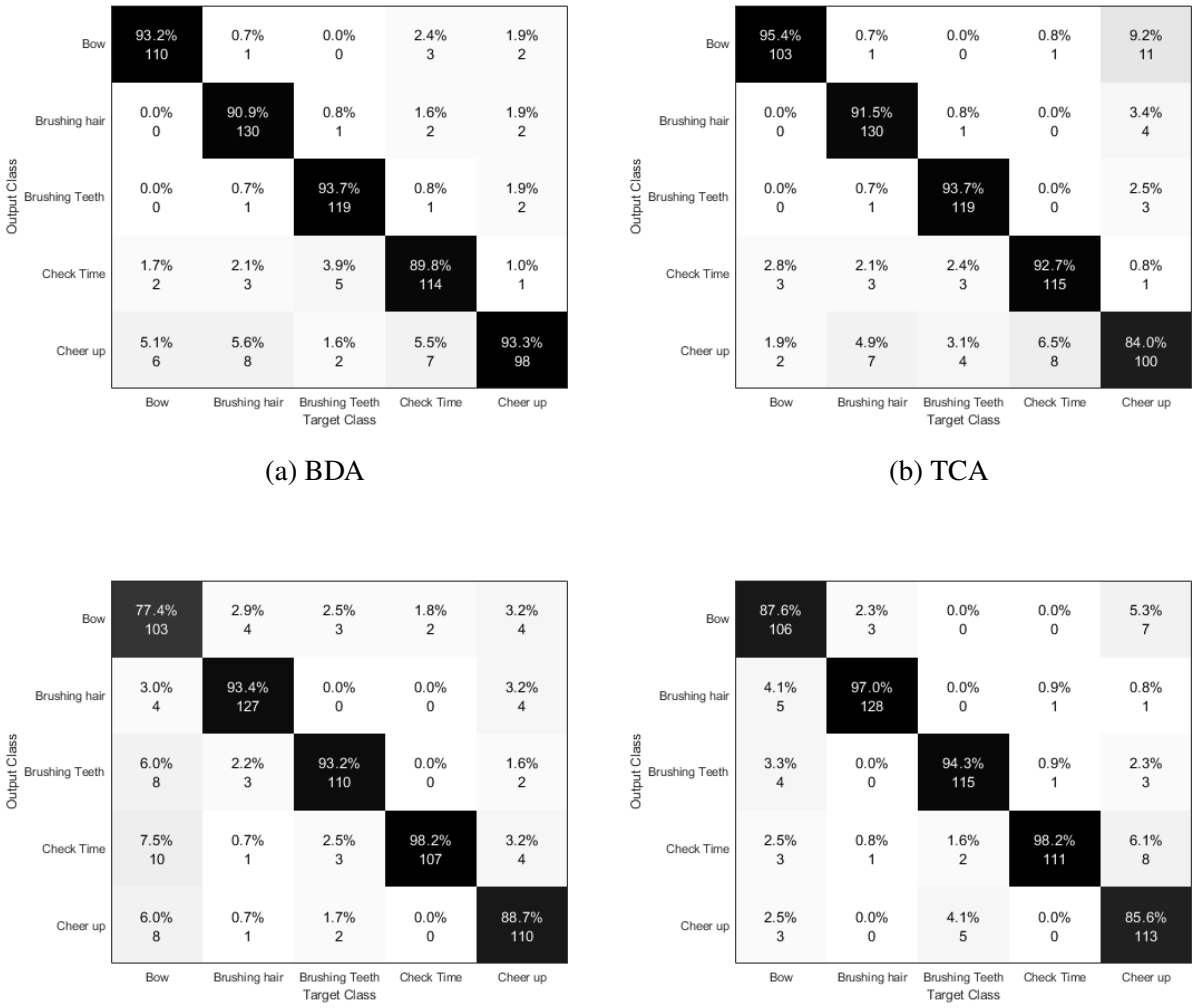
(b) TCA

(c) JDA

(d) TJM

Figure 5.11: Confusion matrix trained on View2 and tested on View3.

connection or mapping between both views. To that end, videos need to be aligned well based on the frame-to-frame, video-to-video, or codebook-to-codebook criteria. In our method, however, we did not construct any mapping, as dictionary learning is used to extract discriminative features



(a) BDA (b) TCA

(c) JDA (d) TJM

Figure 5.12: Confusion matrix trained on View3 and tested on View2.

and transfer learning is used to project those discriminative features into common subspace, which provides better time and memory efficacy.

In this study, we evaluated our method on the action classes with only a single person performing some actions in a closed and fixed environment. It will become more complex but

attractive in recognizing actions of an individual person when multiple people exist and act differently. This will reflect most real-world scenarios for action recognition. Also, in our study, we considered five action classes for the evaluations, while we may further explore other classes to testify the robustness of our model, and its real-world values.

Chapter 6 Conclusions

In this study, we proposed a new joint learning model to address problem of cross-view action recognition where we trained the model in one view and tested the model in a different view. For this purpose, we developed a joint dictionary and transfer learning framework. The dictionary generated the view-independent discriminative features while the transfer learning projected those discriminative features into a common subspace. Specifically, Improve Dense Trajectory (IDT) visual descriptor is applied for robust action feature extraction. State-of-the-arts transfer learning methods were applied and integrated in our model for comparison. Extended evaluations were done on PKU-MMD multi-view dataset, and our proposed approach showed significant results compared to existing methods.

For the future work, the short-term plan is to extend this work by considering recent deep learning features such as two-stream deep model. It learns the semantic representation from raw data and considers both spatial and temporal information. In addition to this, our proposed approach is only evaluated on five action classes, but we are planning to add more action classes which include interactions between two individuals.

In a long-term plan for the improvement and extension of proposed approach, we may consider zero-shot learning and action prediction problem in multi-view setting. In zero-shot learning, a model is constructed on the labeled training set of seen classes to discover the semantic relationship between seen and newly available unseen classes in test data. While the motive for multi-view action prediction problem is to recognize the action (future action) from incomplete action execution where multi-view information can be utilized to avoid the scene loss and recover the blocked and cluttered views. In addition, we will consider improving our approach by recognizing actions of multiple people in a single video.

References

[1] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3169–3176, 2011.

[2] Tao Xiang and Shaogang Gong. Video behavior profiling for anomaly detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):893–908, 2008.

[3] Muhammad Mubashir, Ling Shao, and Luke Seed. A survey on fall detection: Principles and approaches. *Neurocomputing*, 100:144–152, 2013.

[4] Hyeon-Kyu Lee and Jin-Hyung Kim. An hmm-based threshold model approach for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):961–973, 1999.

[5] Rosane M. M. Vallim, José A. Andrade Filho, Rodrigo F. De Mello, and André C. P. L. F. De Carvalho. Online behavior change detection in computer games. *Expert Systems with Applications*, 40(16):6258–6265, 2013.

[6] Gaurav Sharma, Frédéric Jurie, and Cordelia Schmid. Expanded parts model for human attribute and action recognition in still images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–659, 2013.

[7] Xiaodan Liang, Liang Lin, and Liangliang Cao. Learning latent spatio-temporal compositional model for human action recognition. In *Proceedings of the 21st ACM International Conference on Multimedia*, pages 263–272. ACM, 2013.

[8] Di Wu and Ling Shao. Multi-max-margin support vector machine for multi-source human action recognition. *Neurocomputing*, 127:98–103, 2014.

[9] Rachid Benmokhtar. Robust human action recognition scheme based on high-level feature fusion. *Multimedia Tools and Applications*, 69(2):253–275, 2014.

[10] Georgios Th Papadopoulos, Apostolos Axenopoulos, and Petros Daras. Real-time skeleton-tracking-based human action recognition using kinect data. In *International Conference on Multimedia Modeling*, pages 473–483. Springer, 2014.

[11] Alexei A. Efros, Alexander C. Berg, Greg Mori, and Jitendra Malik. Recognizing action at a distance. In *Proceedings of the 9th IEEE International Conference on Computer Vision*, pages 726–733, 2003.

[12] Fengjun Lv and Ramakant Nevatia. Single view human action recognition using key pose matching and viterbi path searching. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[13] Zhe Lin, Zhuolin Jiang, and Larry S. Davis. Recognizing actions by shape-motion prototype trees. In *IEEE 12th International Conference on Computer Vision*, pages 444–451, 2009.

[14] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3551–3558, 2013.

[15] Laptev and Lindeberg. Space-time interest points. In *Proceedings of the 9th IEEE International Conference on Computer Vision*, pages 432–439, 2003.

[16] Homa Foroughi, Nilanjan Ray, and Hong Zhang. Object classification with joint projection and low-rank dictionary learning. *IEEE Transactions on Image Processing*, 27:806–821, 2018.

[17] John Wright, Allen Y. Yang, Arvind Ganesh, S. Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009.

[18] Jingjing Zheng, Zhuolin Jiang, and Rama Chellappa. Cross-view action recognition via transferable dictionary learning. *IEEE Transactions on Image Processing*, 25(6):2542–2556, 2016.

[19] Li Deng, Dong Yu, et al. Deep learning: methods and applications. *Foundations and Trends® in Signal Processing*, 7(3-4):197–387, 2014.

[20] Xiaofei Ji and Honghai Liu. Advances in view-invariant human motion analysis: A review. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(1):13–24, 2010.

[21] Michael B. Holte, Cuong Tran, Mohan M. Trivedi, and Thomas B. Moeslund. Human action recognition using multiple views: a comparative perspective on recent developments. In *Proceedings of the Joint ACM Workshop on Human Gesture and Behavior Understanding*, pages 47–52. ACM, 2011.

[22] Greg Mori, Xiaofeng Ren, Alexei A. Efros, and Jitendra Malik. Recovering human body configurations: Combining segmentation and recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages II–II, 2004.

[23] K. M. G. Cheung, Simon Baker, and Takeo Kanade. Shape-from-silhouette of articulated objects and its use for human body kinematics estimation and motion capture. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–I, 2003.

[24] Jingen Liu, Saad Ali, and Mubarak Shah. Recognizing human actions using multiple features. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[25] Alper Yilmaz and Mubarak Shah. Actions sketch: A novel action representation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005.

[26] Moshe Blank, Lena Gorelick, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. In *IEEE 10th International Conference on Computer Vision Volume 1*, volume 2, pages 1395–1402, 2005.

[27] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Dense trajectories and motion boundary descriptors for action recognition. *International Journal of Computer Vision*, 103:60–79, 2013.

[28] Imran N. Junejo, Emilie Dexter, Ivan Laptev, and Patrick Pérez. Cross-view action recognition from temporal self-similarities. In *European Conference on Computer Vision*, pages 293–306. Springer Berlin Heidelberg, 2008.

[29] Yan Yan, Elisa Ricci, Ramanathan Subramanian, Gaowen Liu, and Nicu Sebe. Multitask linear discriminant analysis for view invariant action recognition. *IEEE Transactions on Image Processing*, 23:5599–5611, 2014.

[30] Jingjing Zheng and Zhuolin Jiang. Learning view-invariant sparse representations for cross-view action recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3176–3183, 2013.

[31] Wanqi Yang, Yang Gao, Yinghuan Shi, and Longbing Cao. Mrm-lasso: A sparse multiview feature selection method via low-rank analysis. *IEEE Transactions on Neural Networks and Learning Systems*, 26:2801–2815, 2015.

[32] Yu Kong, Zhengming Ding, Jun Li, and Yun Fu. Deeply learned view-invariant features for cross-view action recognition. *IEEE Transactions on Image Processing*, 26:3028–3037, 2017.

[33] Hong Lin, Lekha Chaisorn, Yongkang Wong, An-An Liu, Yu-Ting Su, and Mohan S. Kankanhalli. View-invariant feature discovering for multi-camera human action recognition. In *IEEE 16th International Workshop on Multimedia Signal Processing*, pages 1–6, 2014.

[34] Karl Weiss, Taghi M. Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big Data*, 2016.

[35] Hal Daumé III. Frustratingly easy domain adaptation. *arXiv preprint arXiv:0907.1815*, 2009.

[36] Rita Chattopadhyay, Qian Sun, Wei Fan, Ian Davidson, Sethuraman Panchanathan, and Jieping Ye. Multisource domain adaptation and its application to early detection of fatigue. *ACM Transactions on Knowledge Discovery from Data*, 6(4):18, 2012.

[37] John Blitzer, Ryan McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 120–128, 2006.

[38] Diane Cook, Kyle D. Feuz, and Narayanan C. Krishnan. Transfer learning for activity recognition: A survey. *Knowledge and Information Systems*, 36(3):537–556, 2013.

[39] Chunhui Liu, Yueyu Hu, Yanghao Li, Sijie Song, and Jiaying Liu. Pku-mmd: A large scale benchmark for continuous multi-modal human action understanding. *arXiv preprint arXiv:1703.07475*, 2017.

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

[41] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *IEEE International Conference on Computer Vision*, pages 2556–2563, 2011.

[42] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. Behavior recognition via sparse spatio-temporal features. In *IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 65–72, 2005.

[43] Geert Willems, Tinne Tuytelaars, and Luc Van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In *European Conference on Computer Vision*, pages 650–663. Springer, Berlin,Heidelberg, 2008.

[44] Tinne Tuytelaars, Krystian Mikolajczyk, et al. Local invariant feature detectors: A survey. *Foundations and Trends® in Computer Graphics and Vision*, 3(3):177–280, 2008.

[45] Amir H. Shabani, David A. Clausi, and John S. Zelek. Evaluation of local spatio-temporal salient feature detectors for human action recognition. In *9th Conference on Computer and Robot Vision*, pages 468–475, 2012.

[46] Heng Wang, Muhammad Muneeb Ullah, Alexander Klaser, Ivan Laptev, and Cordelia Schmid. Evaluation of local spatio-temporal features for action recognition. In *British Machine Vision Conference*, pages 124–1. BMVA Press, 2009.

[47] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 886–893, 2005.

[48] Navneet Dalal, Bill Triggs, and Cordelia Schmid. Human detection using oriented histograms of flow and appearance. In *European Conference on Computer Vision*, pages 428–441, 2006.

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

[50] Ju Sun, Yadong Mu, Shuicheng Yan, and Loong-Fah Cheong. Activity recognition using dense long-duration trajectories. In *IEEE International Conference on Multimedia and Expo*, pages 322–327, 2010.

[51] Bruce D. Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, pages 674–679, 1981.

[52] Ivana Tosic and Pascal Frossard. Dictionary learning. *IEEE Signal Processing Magazine*, 28(2):27–38, 2011.

[53] Yong Xu, Zhengming Li, Jian Yang, and David Zhang. A survey of dictionary learning algorithms for face recognition. *IEEE Access*, 5:8502–8514, 2017.

[54] Zheng Zhang, Yong Xu, Jian Yang, Xuelong Li, and David Zhang. A survey of sparse representation: algorithms and applications. *IEEE Access*, 3:490–530, 2015.

[55] Stéphane Mallat and Zhifeng Zhang. Matching pursuit with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.

[56] Joel A. Tropp and Anna C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53(12):4655–4666, 2007.

[57] Yagyensh Chandra Pati, Ramin Rezaiifar, and Perinkulam Sambamurthy Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Proceedings of the 27th Asilomar Conference on Signals, Systems and Computers*, pages 40–44, 1993.

[58] Michal Aharon, Michael Elad, and Alfred Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311, 2006.

[59] Meng Yang, Lei Zhang, Xiangchu Feng, and David Zhang. Fisher discrimination dictionary learning for sparse representation. In *IEEE International Conference on Computer Vision*, pages 543–550, 2011.

[60] Shu Kong and Donghui Wang. A dictionary learning approach for classification: separating the particularity and the commonality. In *European Conference on Computer Vision*, pages 186–199. Springer, Berlin, Heidelberg, 2012.

[61] Ignacio Ramirez, Pablo Sprechmann, and Guillermo Sapiro. Classification and clustering via dictionary learning with structured incoherence and shared features. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3501–3508, 2010.

[62] Ning Zhou and Jianping Fan. Jointly learning visually correlated dictionaries for large-scale visual recognition applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(4):715–730, 2014.

[63] John Wright, Allen Y Yang, Arvind Ganesh, S Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009.

[64] Tiep H. Vu, Hojjat S. Mousavi, Vishal Monga, U. K. Arvind Rao, and Ganesh Rao. Dfdl: Discriminative feature-oriented dictionary learning for histopathological image classification. In *IEEE 12th International Symposium on Biomedical Imaging*, pages 990–994, 2015.

[65] Shenghua Gao, Ivor Wai-Hung Tsang, and Yi Ma. Learning category-specific dictionary and shared dictionary for fine-grained image categorization. *IEEE Transactions on Image Processing*, 23(2):623–634, 2014.

[66] Tiep Huu Vu and Vishal Monga. Fast low-rank shared dictionary learning for image classification. *IEEE Transactions on Image Processing*, 26(11):5160–5175, 2017.

[67] Sinno Jialin Pan, Qiang Yang, et al. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 2010.

[68] Andrew Arnold, Ramesh Nallapati, and William W. Cohen. A comparative study of methods for transductive transfer learning. In *ICDM Workshops*, pages 77–82, 2007.

[69] Piotr Tadeusz Bilíński. *Human action recognition in videos*. PhD thesis, Université Nice Sophia Antipolis, 2014.

[70] Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Scandinavian Conference on Image Analysis*, pages 363–370, 2003.

[71] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110:346–359, 2008.

[72] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24:381–395, 1981.

[73] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *European Conference on Computer Vision*, pages 143–156. Springer, Berlin, Heidelberg, 2010.

[74] Florent Perronnin and Christopher Dance. Fisher kernels on visual vocabularies for image categorization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[75] Karen Simonyan, Omkar M. Parkhi, Andrea Vedaldi, and Andrew Zisserman. Fisher vector faces in the wild. In *British Machine Vision Conference*, volume 2, page 4, 2013.

[76] Tsang Ivor W. Pan, Sinno Jialin, James T. Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22:199–210, 2011.

[77] Deepak Kumar, Chetan Kumar, and Ming Shao. Cross-database mammographic image analysis through unsupervised domain adaptation. In *IEEE International Conference on Big Data*, pages 4035–4042, 2017.

[78] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S. Yu. Transfer feature learning with joint distribution adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2200–2207, 2013.

[79] Jindong Wang, Yiqiang Chen, Shuji Hao, Wenjie Feng, and Zhiqi Shen. Balanced distribution adaptation for transfer learning. In *IEEE International Conference on Data Mining*, pages 1129–1134, 2017.

[80] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *Siam Journal on Imaging Sciences*, 2009.

[81] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[82] Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

[83] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11:19–60, 2010.

[84] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7263–7271, 2017.