

REPRESENTATION LEARNING FOR ACTION RECOGNITION

A THESIS

submitted by

DEBADITYA ROY

for the award of the degree

of

DOCTOR OF PHILOSOPHY



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY HYDERABAD

APRIL 2018

Declaration

I declare that this written submission represents my ideas in my own words, and where ideas or words of others have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources that have thus not been properly cited, or from whom proper permission has not been taken when needed.

Debaditya Roy

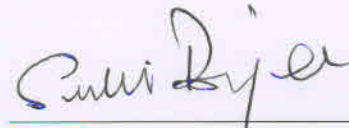
(Debaditya Roy)

CS13P1001

(Roll No.)

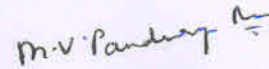
Approval Sheet

This thesis entitled "Representation Learning For Action Recognition" by Mr. Debaditya Roy (Roll No. CS13P1001) is approved for the degree of Doctor of Philosophy from IIT Hyderabad.



Prof. Subhashis Banerjee
IIT Delhi
Examiner 1

Prof. S. V. Rao
IIT Guwahati
Examiner 2



Dr. M. V. Panduranga Rao
IIT Hyderabad
Internal Examiner



Dr. C. Krishna Mohan
IIT Hyderabad
Adviser/Guide

-Name and affiliation-

Co-Adviser



Dr. Sumohana S. Channappayya
IIT Hyderabad
Chairman

Acknowledgements

This dissertation is the result of a long and arduous journey through the last five years. As all journeys, mine was filled with successes and failures both small and big. Holding my hand through the failures and buoying me on during my successes, my parents have been my source of perseverance and determination. My mother has been especially encouraging in times of crisis with hopes of a better tomorrow while my father has been constant in his unwavering support in my endeavours. I owe it all to my parents who have made me capable of charting my own course and making my own choices.

I consider myself fortunate to have Dr. C. Krishna Mohan as my adviser as he has been a constant source of wisdom and encouragement through all these years. He has been a guide in the truest sense of the word and brought out the best in me. Apart from giving me platform for showcasing my work at different avenues and allowing me to mentor students, he has shown immense trust in my abilities which has given me confidence and the zeal to become better everyday. As I leave IIT Hyderabad, I will definitely miss our daily conversations. I hope that one day I will be able to emulate a fraction of his humility and maturity.

I would also like to thank Dr. K. Sri Rama Murty for the countless hours of guidance. He has definitely helped in molding my thought process with his astute knowledge and sharp observational skills.

Finally, my time at IIT Hyderabad has flown by due to my friends who have made it memorable journey. I would like to thank Dr. Mettu Srinivas for all his support during the initial days of my research. A special thanks to my batch-mates, Rohit and Parag with whom I continue to share my thoughts to this day. I would like to mention Mukesh, Thomas, Shashwat, Ravi Sharan, Rashmi, Sherin and countless others who have been friends in more than one way. A special mention goes out to Shiraj whose company has made this journey even more worthwhile. Also, my colleagues at VIGIL, Dinesh, Nazil, and Sanjay, who have contributed to my growth as a researcher. A heartfelt thanks to all my lab-mates for maintaining a vibrant environment in our lab which has encouraged creativity and positive thinking.

Dedication

To my parents

Abstract

The objective of this research work is to develop discriminative representations for human actions. The motivation stems from the fact that there are many issues encountered while capturing actions in videos like intra-action variations (due to actors, viewpoints, and duration), inter-action similarity, background motion, and occlusion of actors. Hence, obtaining a representation which can address all the variations in the same action while maintaining discrimination with other actions is a challenging task. In literature, actions have been represented either using either low-level or high-level features. Low-level features describe the motion and appearance in small spatio-temporal volumes extracted from a video. Due to the limited space-time volume used for extracting low-level features, they are not able to account for viewpoint and actor variations or variable length actions. On the other hand, high-level features handle variations in actors, viewpoints, and duration but the resulting representation is often high-dimensional which introduces the curse of dimensionality. In this thesis, we propose new representations for describing actions by combining the advantages of both low-level and high-level features. Specifically, we investigate various linear and non-linear decomposition techniques to extract meaningful attributes in both high-level and low-level features.

In the first approach, the sparsity of high-level feature descriptors is leveraged to build action-specific dictionaries. Each dictionary retains only the discriminative information for a particular action and hence reduces inter-action similarity. Then, a sparsity-based classification method is proposed to classify the low-rank representation of clips obtained using these dictionaries. We show that this representation based on dictionary learning improves the classification performance across actions. Also, a few of the actions consist of rapid body deformations that hinder the extraction of local features from body movements. Hence, we propose to use a dictionary which is trained on convolutional neural network (CNN) features of the human body in various poses to reliably identify actors from the background. Particularly, we demonstrate the efficacy of sparse representation in the identification of the human body under rapid and substantial deformation.

In the first two approaches, sparsity-based representation is developed to improve discriminability using class-specific dictionaries that utilize action labels. However, developing an unsupervised representation of actions is more beneficial as it can be used to both recognize similar actions and localize actions. We propose to exploit inter-action similarity to train a universal attribute model (UAM) in order to learn action attributes (common and distinct) implicitly across all the actions. Using maximum *a posteriori* (MAP) adaptation, a high-dimensional super action-vector (SAV) for each clip is extracted. As this SAV contains redundant attributes of all other actions, we use factor analysis to extract a novel low-

dimensional action-vector representation for each clip. Action-vectors are shown to suppress background motion and highlight actions of interest in both trimmed and untrimmed clips that contributes to action recognition without the help of any classifiers.

It is observed during our experiments that action-vector cannot effectively discriminate between actions which are visually similar to each other. Hence, we subject action-vectors to supervised linear embedding using linear discriminant analysis (LDA) and probabilistic LDA (PLDA) to enforce discrimination. Particularly, we show that leveraging complimentary information across action-vectors using different local features followed by discriminative embedding provides the best classification performance. Further, we explore non-linear embedding of action-vectors using Siamese networks especially for fine-grained action recognition. A visualization of the hidden layer output in Siamese networks shows its ability to effectively separate visually similar actions. This leads to better classification performance than linear embedding on fine-grained action recognition.

All of the above approaches are presented on large unconstrained datasets with hundreds of examples per action. However, actions in surveillance videos like snatch thefts are difficult to model because of the diverse variety of scenarios in which they occur and very few labeled examples. Hence, we propose to utilize the universal attribute model (UAM) trained on large action datasets to represent such actions. Specifically, we show that there are similarities between certain actions in the large datasets with snatch thefts which help in extracting a representation for snatch thefts using the attributes from the UAM. This representation is shown to be effective in distinguishing snatch thefts from regular actions with high accuracy.

In summary, this thesis proposes both supervised and unsupervised approaches for representing actions which provide better discrimination than existing representations. The first approach presents a dictionary learning based sparse representation for effective discrimination of actions. Also, we propose a sparse representation for the human body based on dictionaries in order to recognize actions with rapid body deformations. In the next approach, a low-dimensional representation called action-vector for unsupervised action recognition is presented. Further, linear and non-linear embedding of action-vectors is proposed for addressing inter-action similarity and fine-grained action recognition, respectively. Finally, we propose a representation for locating snatch thefts among thousands of regular interactions in surveillance videos.

Contents

Declaration	ii
Approval Sheet	iii
Acknowledgements	iv
Abstract	vi
Nomenclature	x
LIST OF TABLES	2
LIST OF FIGURES	1
1 Introduction to action recognition	2
1.1 Human action representation	2
1.1.1 Single layered representation	3
1.1.2 Hierarchical representation	3
1.1.3 Challenges in representing human actions	3
1.2 Issues addressed in this thesis	6
1.3 Organization of the thesis	7
2 Review of representation learning for action recognition	8
2.1 Traditional approaches	9
2.1.1 Local feature extraction	9
2.1.2 Aggregated descriptors of local features	13
2.2 Deep features for action recognition	15
2.2.1 Spatio-temporal networks	16
2.2.2 Multi-stream networks	19
2.2.3 Generative networks	20
2.3 Observations from the review	21
2.4 Summary	22

3	Sparsity inducing dictionaries for action recognition	23
3.1	Action recognition based on sparse representation using dictionary learning	24
3.1.1	Exploring suitable features for sparsification	26
3.1.2	Dictionary based representation	26
3.1.3	Sparsity based classification	27
3.2	Experimental results	28
3.2.1	Datasets	28
3.2.2	Evaluation on different feature descriptors	29
3.2.3	Classification Performance vs. Dictionary Size	30
3.2.4	Visualization of dictionaries	31
3.2.5	Comparison with state-of-the-art	31
3.3	Summary	37
4	Recognition of actions with rapid body deformations using sparse representation	38
4.1	Review of human body representation	40
4.2	Dictionary-CNN hybrid approach for human body representation	42
4.2.1	Dictionary construction	44
4.2.2	Dictionary based representation	46
4.3	Experimental results	46
4.3.1	Dataset	47
4.3.2	Metrics for evaluation	49
4.3.3	Comparative analysis	49
4.4	Summary	53
5	Unsupervised action recognition using action-vectors	55
5.1	Unsupervised action-vector extraction	56
5.1.1	Universal attribute model (UAM)	57
5.1.2	Representation of actions as super action-vector (SAV)	59
5.1.3	Extraction of compact action-vectors using factor analysis	60
5.1.4	Unsupervised action-vector scoring	63
5.2	Experimental Results	64
5.2.1	Datasets	64
5.2.2	Experimental Settings	64
5.2.3	Super action-vector vs. Action-vector	65
5.2.4	Action-vectors vs. other aggregation methods	66
5.2.5	Analysis on untrimmed videos	66

5.3	Summary	67
6	Supervised action recognition using linear and non-linear embedding of action-vectors	69
6.1	Linear decomposition using LDA and PLDA	70
6.1.1	Linear Discriminant Analysis (LDA)	70
6.1.2	Probabilistic Linear Discriminant Analysis (PLDA)	71
6.2	Non-linear decomposition using Siamese networks	73
6.2.1	Siamese network configuration	74
6.3	Experimental results	75
6.3.1	Datasets	75
6.3.2	Experimental settings	76
6.3.3	Performance of linear embedding techniques	76
6.3.4	Intermediate fusion techniques using PLDA	77
6.3.5	Linear embedding of action-vectors vs. state-of-the-art	78
6.3.6	Comparison of linear and non-linear embedding techniques	79
6.3.7	Non-linear embedding of action-vectors vs. state-of-the-art	80
6.4	Summary	81
7	Action recognition in surveillance videos	83
7.1	Related work in surveillance video analysis	86
7.2	Baseline: Rule-based snatch theft monitoring	88
7.2.1	Stage I: Detection and tracking of humans in crowded scenarios	88
7.2.2	Stage II: Victim and thief identification	88
7.2.3	Stage III: Snatch Theft Verification	89
7.3	Unsupervised modeling of snatch theft actions	90
7.4	Experimental results	91
7.4.1	Snatch 1.0 : Dataset Description	91
7.4.2	Effect of different feature descriptors and UAM mixtures	91
7.5	Summary	93
8	Summary and Conclusions	96
8.1	Contributions of the thesis	97
8.2	Directions for Further Research	97
	References	98

List of Tables

3.1	Performance comparison of sparsity-based dictionaries using different features on the HMDB51 action dataset	30
3.2	Effect of dictionary size on performance (in %)	31
3.3	Comparison of classification performance on the HMDB51 action dataset	36
3.4	Classification performance on the UCF50 dataset	37
5.1	Comparison of super action-vector & action-vector classification performance (%) using cosine scoring for varying UAM mixtures on UCF101 dataset.	65
5.2	Comparison of super action-vector & action-vector classification performance (%) using cosine scoring for varying UAM mixtures on HMDB51 dataset.	65
5.3	Comparison with other state-of-the-art aggregation methods	66
6.1	Performance (in %) of Siamese network with different configurations on HMDB51 dataset	75
6.2	Classification accuracy (%) for linear embedding techniques on UCF101.	76
6.3	Classification accuracy (%) for various scoring mechanisms on HMDB51. UAM is trained on HMDB51 training set.	77
6.4	Comparison of intermediate fusion and SVM (IF + SVM), PLDA-based intermediate latent decomposition fusion (ILDF), and score fusion (SF) on UCF101. The underlying UAM model has 2048 mixtures.	78
6.5	Comparison of classification accuracy of proposed approach with existing state-of-the-art methods	79
6.6	Performance comparison (%) of various discriminative embedding techniques on HMDB51	79
6.7	Classification accuracy (%) for various discriminative embedding techniques on MPII Cooking	80
6.8	Comparison of Siamese network based embedding of action-vectors with state-of-the-art on HMDB51	81
6.9	Comparison of Siamese network based embedding of action-vectors with state-of-the-art on MPII Cooking Activities	81
7.1	Ontology of snatching scenarios, types and victim reactions	85
7.2	Action-vector classification performance (in %) for Snatch 1.0.	92
7.3	Comparison with state-of-the-art feature descriptors using SVM classifier.	93

List of Figures

1.1	Inter-action similarity between (a) tennis smash and (b) volleyball spiking	4
1.2	Viewpoint variations for cricket batting	5
1.3	Variation in the duration of the action classes in UCF101. Adapted from [109]. Best viewed in color.	5
2.1	HOG, HOF, and MBH extraction from trajectories. Adapted from [117]. Best viewed in colour.	11
2.2	HOG features calculated on (a) running and (b) clapping. Notice that the gradients align along the outline of the body.	12
2.3	HOF features calculated on (a) running and (b) hand waving	12
2.4	MBH features on (a) running, (b) in x-direction, and (c) in y-direction	13
2.5	Action-bank extraction as adapted from [97]. Best viewed in color.	15
2.6	Spatiotemporal operations: 2D convolution (blue), 3D convolution on stacks of frames (red) as in [46], conventional spatial max-pooling (brown), and temporal max-pooling (yellow) as in [84]. Best viewed in color.	16
2.7	Foveated architecture of Karpathy et al. [49]. Green, red, and blue denote normalization, spatial-pooling, and convolutional layers, respectively. Best viewed in color.	17
2.8	Left: The recurrent structure of a RNN/LSTM network. Center: RNN cell structure as a linear dynamical system. Right: The LSTM cell includes additional gate controls. Time delay is indicated with a black square. Adapted from [36].	18
2.9	LRCN network structure [20]. Each group is a set of convolutional filters applied to a particular set of feature maps from the previous layer. Adapted from [36].	19
2.10	(a) The two-stream network by Simonyan and Zisserman [102] with RGB and stacked optical-flow frames as inputs. (b) An example of a two stream intermediate fusion network of Feichtenhofer et al. [24].	20
2.11	LSTM auto-encoder model by Srivastava et al. [120]. The internal states (represented by the circle inside) of the encoder LSTM capture a compressed version of the input sequence (frames 1,2, and 3). The states after that are copied into two decoder models, one of which is reconstructive and the other predictive. The decoder LSTM tries to reconstruct the input frames in the reverse order. The predictive LSTM is trained for predicting the future frames 4, 5 and 6. The colors on the state markers indicate the information from a particular frame. Best viewed in color	21

3.1	Flowchart of the proposed approach	25
3.2	Histogram of action bank features for all classes of HMDB51 dataset.	26
3.3	Sample actions from HMDB51 dataset	29
3.4	Visualization of dictionaries for selected classes in HMDB51. Best viewed in color	32
3.5	Visualization of dictionaries for selected classes in UCF50. Best viewed in color	33
3.6	Confusion Matrix for UCF50 dataset for dictionary of size 120. Classification performance : 72.46%	34
3.7	Confusion Matrix for HMDB51 dataset for dictionary of size 100. Classification performance : 99.87%	35
4.1	Scenarios depicting highly articulated motion leading to complex poses. Best viewed in color.	39
4.2	Dictionary-CNN hybrid framework for human tracking	40
4.3	CNN architecture adapted from [83]. The input size of 107×107 is designed to obtain 3×3 feature maps in <i>conv3</i> : $107 = 75$ (receptive field) + 2×16 (stride). The convolution layers obtained from VGG-16 [105] are not updated. The fully connected layers are learned originally from training videos and shared across videos. The binary classifier is trained from the first frame of the testing video. The fully connected layers and binary classifier are updated when target score drops below the threshold. The red and blue bounding boxes denote positive and negative samples, respectively.	43
4.4	<i>fc6</i> scores of MDNet [83] with the tracker output (in yellow) for few of the frames. Green represents the ground truth. Best viewed in color.	44
4.5	Sparsity scores of human patches (in red) and non-human patches (in blue) obtained from dictionaries trained using (a) HoG and (b) CNN features for the 3 classes : cliff-diving, platform-diving, and pole-vaulting. Best viewed in color.	45
4.6	Localization results of Hybrid dictionary-CNN and MDNet [83] for a few platform-diving scenarios. Each row represents a different video sequence. Blue represents ground truth, pink represents hybrid output, and red represents MDNet output. Best viewed in color.	50
4.7	Localization results of Hybrid dictionary-CNN and MDNet [83] for a few cliff-diving scenarios. Each row represents a different video sequence. Blue represents ground truth, pink represents hybrid output, and red represents MDNet output. Best viewed in color.	51
4.8	Comparison of accuracy (in %) between proposed hybrid framework and MDNet [83]	52
4.9	Failure cases of human body representation. Each row represents a different scenario. Blue represents ground truth, pink represents the output of the proposed framework, and red represents MDNet output. Best viewed in color.	53
5.1	Unsupervised attribute modelling and action-vector extraction	56

5.2	UAM posteriors (using 256 Gaussian mixtures) for two actions of UCF101: (a) <i>Hulahoops</i> and (b) <i>Benchpress</i> . Although the two clips of <i>Hulahoops</i> have variable number of frames, the sequence of Gaussian mixtures having the highest posterior probability (in black) is similar throughout the action. These mixtures represent the attributes which contribute to the action and the slight deviations may be caused by actor or viewpoint variability. Similar behavior can be observed in the clip of <i>Benchpress</i>	58
5.3	t-SNE visualization on selected classes of UCF101 (a) MBH features (b) MBH action-vectors. Best viewed in color.	63
5.4	Histogram of super action-vector for MBH features of an action class in HMDB51 dataset (reduced to 2 dimensions using t-SNE).	63
5.5	t-SNE plot shows similarity in action-vectors of UCF101 and THUMOS14 across two classes - WritingOnBoard (UCF101, ▼) (THUMOS14, ▼) and Wall-Pushups (UCF101, ●) (THUMOS14, ●)	67
5.6	Entropy plot for the number of UAM posteriors (using MBH features) for an untrimmed clip of <i>blowing candles</i> in THUMOS14.	68
6.1	Inter-action similarity shown with motion trajectory. Best viewed in color. .	70
6.2	t-SNE visualization of LDA projected MBH action-vectors on selected classes of UCF101. Best viewed in color.	72
6.3	Siamese networks for non-linear decomposition of action-vectors	74
6.4	t-SNE (stochastic neighbour embedding) plots for similar actions (see Figure 6.1) <i>baseball swing</i> (▼) and <i>sword exercise</i> (●). Best viewed in color. . .	75
7.1	Different snatching scenarios as captured in Snatch 1.0. Best viewed in colour.	84
7.2	Visual similarity in (a) <i>grabbing</i> of Snatch 1.0 and (b) <i>punching</i> of HMDB51 dataset.	86
7.3	Action-vectors for Snatch 1.0 using (a) HOF features and (b) MBH features, using 256 UAM mixtures. Best viewed in color.	92
7.4	ROC for performance comparison of action-vectors with state-of-the-art. Best viewed in colour.	93
7.5	Detection results of snatch theft using the proposed framework. For visualization, a YOLO [93] detector and a kernelised correlation filter [35] based tracker was applied on each of the detected snatch theft clips. Bounding boxes are drawn (in green) with the <i>id</i> (in blue) on top of the box for different persons. Best viewed in colour.	94
7.6	False positive cases where normal interactions detected as snatch thefts . .	95

Chapter 1

Introduction to action recognition

Human actions range from simple limb movements to complex coordinated movements of a group of limbs and the body. For instance, *throwing a ball* is a simple arm movement but *cricket bowling* involves the collective movement of legs, arms, and whole body. Due to these significant variations in what can be termed as an action, there are various definitions of actions. Recently, Wang et al. [127] have defined actions based on the transformation it brings to the environment. Using this definition, actions can be grouped into various categories [58] like: (a) general facial actions - e.g., smile, laugh, and chew, (b) facial actions with object manipulation - e.g., smoke, eat, and drink, (c) general body movements - e.g., cartwheel, clap hands, and climb, (d) body movements with object interaction - e.g., brush hair, catch, and draw sword, and (e) body movements for human interaction - e.g., fencing, and hug. Action recognition is the process of recognizing these different types of actions in real-world video streams. It lends itself to diverse applications like automated video indexing of huge online video repositories like YouTube and Vimeo, analyzing video surveillance systems in public places, human-computer interaction, sports analysis, etc. With such a diverse range of applications, action recognition is a very important area of research, and the efficacy of any framework depends on a robust action representation which can both adequately represent the variations in the same action and discriminate between similar instances of different actions.

1.1 Human action representation

Any human action can be described based on the detection of human and/or its body parts, and the subsequent tracking of the detected human/body part. For example, in an action like “shaking hand”, two person’s arms and hands are detected and tracked to generate a description of their movement. This description is then compared with patterns in the train-

ing data to relevant action type. As this process relies heavily on the accuracy of tracking, videos without a clear view of the action cannot be recognized. This has led to development of representations which can describe an action in greater detail using the spatio-temporal semantics of the different events involved in the action. These representations are broadly categorized as single-layered approaches and hierarchical approaches [1].

1.1.1 Single layered representation

Single layered approaches recognize actions directly from the raw frames of the video data instead of considering a logical breakdown of actions into primitive sub-actions or events. The image sequences from videos are regarded as being generated from a specific class of actions, and thus such approaches basically involve matching the representation of the videos to the correct action class. Specifically, single-layered approaches recognize simple actions that can be employed to detect more complex actions using hierarchical combinations.

1.1.2 Hierarchical representation

Hierarchical approaches recognize events of interest based on simpler sub-actions. Any high-level action can be decomposed into a sequence of several sub-actions like “hand shaking” can be understood as a sequence of two hands being extended, merging into a single object, and two hands being withdrawn. Sub-actions can be further considered as high-level actions until decomposed into atomic ones. The advantage of hierarchical approaches is the capability to model the complex structure of human activities and its flexibility for recognizing either individual activities, interaction between humans and/or objects or group activities. Hierarchical approaches are also related to single layer approaches. For example, non-hierarchical single layer approaches can be easily utilized for low-level or atomic action recognition such as gesture detection. Some non-hierarchical single layer approaches can also be extended to hierarchical models such as extended multi-layered hidden Markov models (HMM). Construction of HMM models requires features that are captured in smaller temporal regions and can be sequentially chained to form the entire action.

1.1.3 Challenges in representing human actions

The representations presented above are designed to counter the challenges in human action recognition arising due to the diversity in human actions and the unconstrained nature of

recorded videos. Some of the challenges are inter-action similarity, viewpoint variations, occlusion, variation in duration, actor-specific variation, background clutter, and camera motion.

Inter-action similarity

Few actions like *spiking a volleyball* and *smashing with a tennis racket* can appear visually similar as shown in Figure 1.1. With such similarity in their appearance, the features extracted from the videos of such actions are more challenging to discriminate.

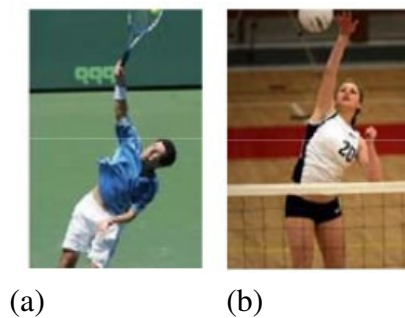


Figure 1.1: Inter-action similarity between (a) tennis smash and (b) volleyball spiking

Viewpoint variation

In real-life, actions are captured from different viewpoints and camera positions as shown in Figure 1.2. It can be observed that even though the same action, i.e., cricket batting is captured from different views, the posture of the person varies considerably in each view. Further, the action attributes or motion patterns appear different in each view which makes the task of action recognition even more challenging.

Occlusion

In surveillance videos, it is often difficult to obtain a clear recording of the action of interest because of the number of people in the field of view. Occlusions can also appear due to the parts of the actor's body performing the action being covered by other body parts or objects. This is particularly challenging as the key body parts acting may not be visible in the video sequence, and feature extraction from such occluded parts is not possible.

Variation in duration

As each person acts at his/her own pace, there may be large variations in the recorded duration of actions. Further, it is highly unlikely that a person will repeat the action at the



Figure 1.2: Viewpoint variations for cricket batting

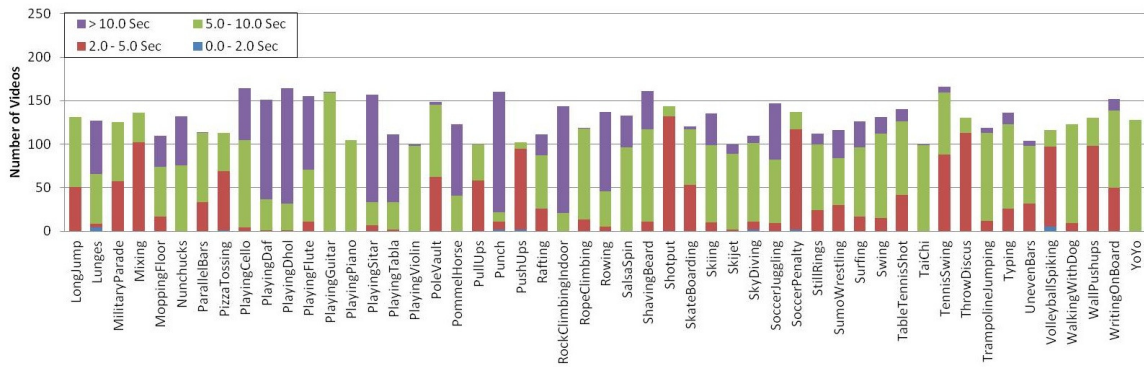


Figure 1.3: Variation in the duration of the action classes in UCF101. Adapted from [109]. Best viewed in color.

same speed. Figure 1.3 shows the variation in duration for all the actions in the UCF101 dataset [109]. It can be observed that for many of the actions like punch and shaving beard have large variations in the duration of the recorded actions. These variations in the rate of execution are challenging to handle when designing fixed length representations.

Actor-specific variations

Humans have differences in body size, proportion, and posture while performing an action. For example, one person might move his/her hand above the head when waving, but another person might just wave from shoulder height. In unconstrained datasets like HMDB51 and UCF101, such variations are rampant as the videos are curated mainly from YouTube where a large number of actors perform the same action.

Background clutter

Background clutter is motion patterns of background objects or persons which distract from the original action of interest. In such cases, the foreground needs to be reliably segmented, and the foreground object needs to be tracked. However, due to poor resolution and variable background motion, such accurate tracking may not always be possible.

Camera motion

Finally, there may be inherent motion in the camera while the action is being shot which severely affects motion features since erroneous motion patterns are introduced in the videos. In many cases, feature extraction using background subtraction methods may not be feasible as these methods are affected by moving cameras. Further, any motion compensation method might erroneously remove motion features that belong to the foreground object.

1.2 Issues addressed in this thesis

Ambiguities in action recognition are caused by one or more of the above issues. In this thesis, we explore various ways of building action representations that address one or more of these issues. Our first approach involves learning dictionaries for each action using a sparse representation of actions. We show that such dictionaries retain discriminative information for a particular action and address the issue of inter-action similarity. Further, training of these dictionaries involves the reconstructing the same action from different views which resolves viewpoint variations. In the next approach, we propose a tracking framework for actions where rapid deformations to the human body are encountered. A hybrid system combining the sparse representation of convolutional neural network (CNN) features helps in mitigating issues like occlusion, background clutter, and camera motion while tracking.

The next approach introduces a novel low-dimensional fixed dimensional representation for variable length actions called action-vector. Action-vectors suppress background motion by highlighting only the action of interest that is achieved through choosing action attributes from a universal attribute model. Action-vectors are shown to be invariant to actor-specific variations which result in a highly discriminative representation for each action. To improve inter-action discriminability and additionally improve viewpoint invariance, linear and non-linear embedding of action-vectors is proposed. Finally, we exploit inter-action similarity to represent snatch thefts from a pre-trained attribute model using a

specific subset of attributes. Despite encountering issues like background motion, view-point and actor variations which are common in surveillance footage, the obtained action-vectors for snatch thefts are distinguishable from regular actions.

1.3 Organization of the thesis

The thesis is organized as follows. In Chapter 2, we present a review of representation methods for action recognition and highlight the relevant research issues. The first approach for action representation using sparsity-based dictionaries is discussed in Chapter 3. In Chapter 4, we exploit sparsity to represent humans in actions with rapid body deformations. Then, in Chapter 5, an action attribute modeling based representation is proposed called action-vectors which can discriminate between actions without supervision. In Chapter 6, we subject action-vectors to linear embedding using linear discriminant analysis and non-linear embedding using deep Siamese networks to improve discriminability. The performance of the representations discussed in these chapters is demonstrated on benchmark unconstrained action datasets, and we extend our purview to highly challenging surveillance videos in Chapter 7. The performance of action-vector is demonstrated in recognition of snatch thefts in real surveillance footage collected from the city of Hyderabad in India. Finally, in Chapter 8, a summary of the research work carried out as part of this thesis is presented with directions for future work.

Chapter 2

Review of representation learning for action recognition

The challenges in action recognition presented in detail in the previous chapter have been studied with great interest in the computer vision community. The main challenges encountered in action recognition are inter-action similarity, viewpoint variations, occlusion, variation in duration, actor-specific variation, background clutter, and camera motion. In literature, both spatial and temporal information have been shown to be important for action representation [37]. Broadly, these representations are categorized based on the level of granularity of the described spatio-temporal volume. The level of granularity gives rise to representations extracted from either a local or a global context in an action video.

Local features can address issues like camera motion, background clutter, and occlusion because the granular localization of motion and appearance can be used to segregate the background from the foreground motion patterns. Also, local features can be used to match short-term patterns between actions which provides more robust recognition even if the action is partially occluded. However, local features are able to effectively address actor-specific and duration based variations which are based on the appearance and execution rate of the actor. As the number of features extracted is strictly dependent on the length of the video, there is a high mismatch if the duration varies widely for the same action which is often the case in unconstrained videos. Further, in case of large variations across actors in terms of body size and posture, the spatio-temporal locations for extraction of feature change substantially which results in different local features. On the other hand, global features or holistic representations can deal with actor-specific and duration changes as the motion patterns are aggregated in fixed-dimensional representation. This aggregation normalizes the effect of variations in appearance and duration which enhances the similarity across the same action in unconstrained videos. However, inter-action similarity

is not suitably addressed by global features because of the inability to highlight minute local variations that separate such actions. Also, global features are high dimensional which introduces the curse of dimensionality while training classifiers on these representations. Still, challenges like viewpoint variations are difficult to address using representation alone and classification approaches like support vector machines (SVM) are used to define discriminant surfaces for each action that separates it from other actions. These decision surfaces account for dissimilar representations within the same action arising out of viewpoint differences.

As we have seen, a typical action recognition framework consists of feature extraction, aggregation, and classification. Through recent advances in deep learning, convolutional neural network (CNN) based approaches have been used extensively for action recognition. These CNN architectures perform local feature extraction through the various filters in the first few layers and combine them to form a global descriptor in the last fully connected layers [113, 19] which is followed by *softmax* layer based classification. Hence, a single deep network can be used to replace the entire framework.

The rest of the chapter is organized as follows. Traditional approaches like local and aggregation features are discussed in Section 2.1. We describe the various deep architectures for action representation in Section 2.2. Finally, the observations arising from the literature review are presented in Section 2.3 and the summary is presented in Section 2.4.

2.1 Traditional approaches

The most effective traditional approaches for action recognition use local/low-level/short-term features. A low-level feature describes a small space-time volume in the entire video. Typically, action recognition frameworks perform local feature extraction followed by aggregated descriptors of local features.

2.1.1 Local feature extraction

The first local feature for actions in videos was proposed by Laptev [66] in the form of space-time interest points (STIP). Every STIP feature descriptor is computed using 3D-Harris corner detectors to discover interest points in the spatio-temporal volume with spatial variations and non-constant motion. However, camera motion is a recurring problem while capturing actions and can lead to irrelevant interest points. In [76], such interest points are discarded using statistical analysis. Further, an interesting observation was presented in [76] which shows that static features obtained from the background helped in action recog-

dition. This is because the background (e.g., the table of billiards) can supply contextual information that aids the action recognition. A related idea was presented in Solmaz et al. [108] in the form of gist which is a video descriptor computed using discrete 3D discrete Fourier transform with 68 different Gabor filters placed in different orientations.

Gradient-based representations like the 2D histogram of oriented gradients (HOG) [15] have been used for objects and shapes within an image based on the distribution of intensity gradients or edge directions. This idea was extended in [53], where spatio-temporal gradient variations are captured as a collection of quantized 2D histograms from each frame of the video to form histograms of oriented 3D gradients (HOG3D). It was shown in [53] that instead using motion patterns described by optical flow fields lead to better action representation than spatio-temporal gradients. The histogram of optical flow (HOF) proposed by Laptev et al. [67] describes the optical flow over a local region as a spatio-temporal descriptor. To compensate for camera motion that unfortunately contributes to erroneous optical flows, motion boundary histogram (MBH) was introduced in [16]. MBH is computed as the spatial derivative of optical flow fields.

Local descriptors like HOG3D and 3D scale-invariant feature transform (3DSIFT) [98] are usually computed in a 3D video volume around interest points which ignore the fundamental dynamic structures in the video. Instead, a fluid 3D trajectory tracks the movement of an interest point more accurately. Especially, human-human actions can be more easily identified by the relative motion between trajectories [47]. Further, discrepancies due to camera motion can be compensated from original trajectories by subtraction as shown in [117].

Improved dense trajectories

Improved dense trajectories (iDT) proposed by Wang et al. [118] describe the absolute movement, position, and relative movement of a particle within a space-time volume of size $N \times N$ pixels and L frames aligned with a trajectory as shown in Figure 2.1. To obtain local structural information, this volume is subdivided into cells of size $n_h \times n_w \times n_t$, where n_h , n_w , and n_t are height, width, and temporal segment lengths, respectively.

At first, feature points are densely sampled on a grid spaced by W pixels (set to 5 as per [117]) in different spatial scales. There are at most eight spatial scales increasing by a factor of $1/\sqrt{2}$ and the actual number of scales used depends on the resolution of the video. Feature points are tracked on each spatial scale separately using dense optical flow. For each frame I_t , its dense optical flow field $\omega_t = (u_t, v_t)$ is computed w.r.t. the next frame I_{t+1} , where u_t and v_t are the horizontal and vertical components of the optical flow, respectively. Given a point $P_t = (x_t, y_t)$ in frame I_t , its tracked position in frame I_{t+1} is

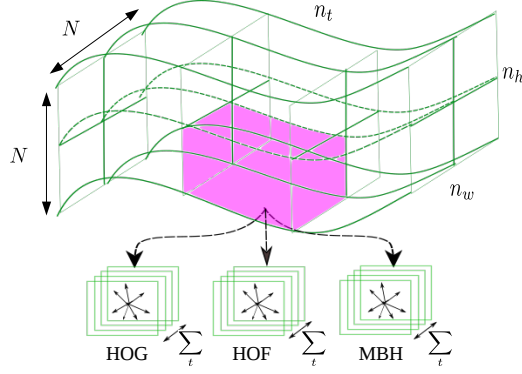


Figure 2.1: HOG, HOF, and MBH extraction from trajectories. Adapted from [117]. Best viewed in colour.

smoothed by applying a median filter on ω_t :

$$P_{t+1} = (x_{t+1}, y_{t+1}) = (x_t, y_t) + (\mathbf{K} * \omega_t)|_{(x_t, y_t)}, \quad (2.1)$$

where \mathbf{K} is the median filtering kernel of size 3×3 pixels. Once the dense optical flow field is computed, points can be tracked without additional cost. The points in subsequent frames are concatenated to form trajectories $(P_t, P_{t+1}, P_{t+2}, \dots)$. However, if no tracked point is found in a $W \times W$ neighborhood, a new point is sampled and added to the tracking process so that a dense coverage of trajectories is ensured. As trajectories tend to drift from their initial locations during the tracking process, their length is limited to L frames to overcome this problem (set to 15 frames as per [117]). Also, trajectories with sudden large displacements are most likely to be erroneous. Hence, if the consecutive frame displacement $> 70\%$ of the overall displacement of the trajectory, the trajectory is removed. From each trajectory, the following descriptors are extracted.

- **Histogram of gradient (HOG)** features localize the particle in the video frame. HOG has been shown to be excellent human detectors [15] and describe a particle as a measure of the dominant gradients in its neighborhood. In Figure 2.2, the HOG features detected for a person in a frame while clapping and walking are shown. In case of a particle, the HOG features describe the dominant shape of the particle like the shoulder or hands.
- **Histogram of optical flow (HOF)** features [10] form the next 96 dimensions which describe the movement of a particle in subsequent frames in a small neighborhood (3×3 or 5×5). Due to the change in scale of the actions a pyramidal approach is

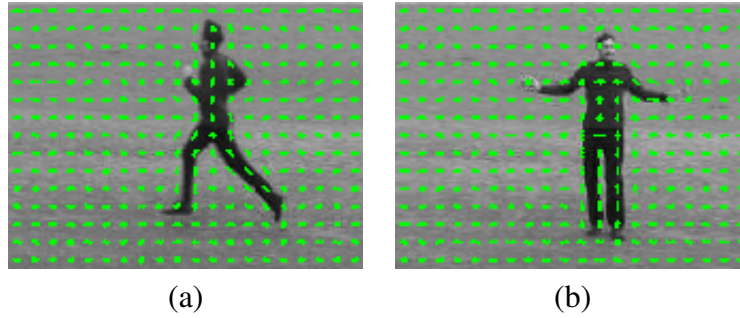


Figure 2.2: HOG features calculated on (a) running and (b) clapping. Notice that the gradients align along the outline of the body.

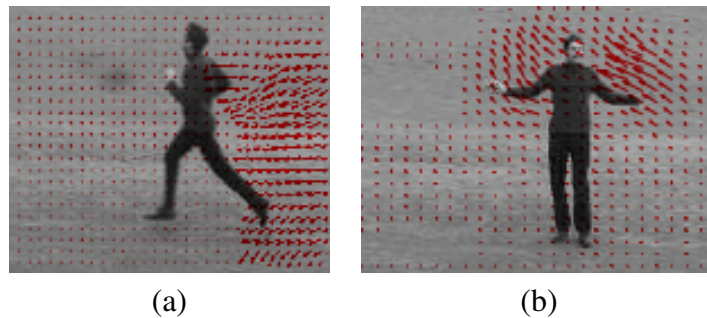


Figure 2.3: HOF features calculated on (a) running and (b) hand waving

considered. For instance, in figure 2.3 the optical flow or HOF features are shown for running and hand waving actions. Note that in case the background is static, as in this case, HOF portrays the motion signature as in the hand motion for hand waving and the entire body motion for running.

- **Motion boundary histogram (MBH)** features are computed in both horizontal and vertical direction denoted as MBHx and MBHy (96 dimensions each). It quantifies the relative motion between two particles as shown in Figure 2.4 where MBHy barely captures the predominantly horizontal movement of running captured by MBHx. The MBH descriptor separates optical flow into horizontal and vertical directions using spatial derivatives in each direction separately. This results in distinct orientation information for each direction which is quantized into histograms, and the magnitude is used for weighting.

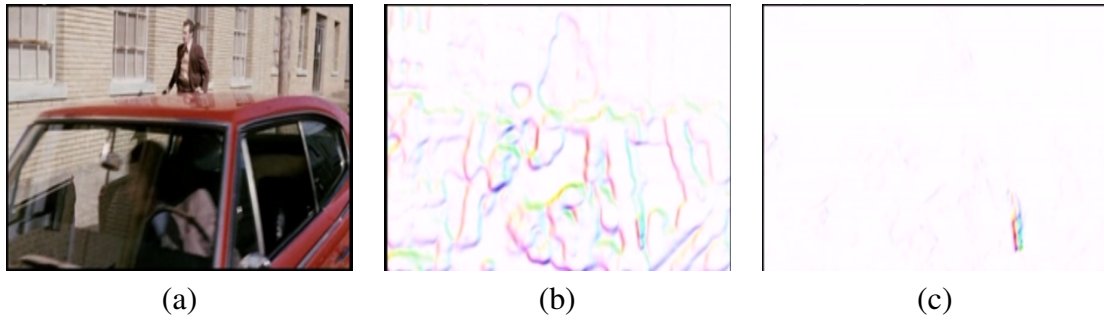


Figure 2.4: MBH features on (a) running, (b) in x-direction, and (c) in y-direction

2.1.2 Aggregated descriptors of local features

The biggest challenge in using local features is that the number of features is dictated by the duration of the clip, leading to varying length patterns. As classifiers like support vector machines (SVM) require a fixed dimensional representation, a mechanism to represent sets of local features into fixed-size descriptors is required, and this is fulfilled by aggregation.

Aggregation using bag-of-words

There are many aggregation based frameworks like bag-of-words (BoW), Fisher vector, and vector of locally aggregated descriptors (VLAD) which have been used predominantly for action representation [118]. To derive either of the descriptors mentioned above, the short-term features are clustered using k -means or Gaussian mixture models (GMM). Then, three different types of global descriptors can be obtained: a) BoW - using the zeroth order statistics of the clusters, b) VLAD - using the first-order statistics of the clusters, and c) Fisher vector - using both first and second-order statistics. In [126], a BoW model was created using HOG and HOF features which were utilized for action classification. Similarly, Fisher vectors have also been extensively used as a feature for standard classifiers such as SVM [118] and feed-forward neural networks [17] to perform action recognition. Notably, Fisher vectors calculated with the motion descriptors such as HOF and MBH have shown good classification performance on large action datasets[117, 60]. A recent improvement in VLAD termed as VLAD³ was used to provide a video-based representation [73] and was shown to perform better than Fisher vector on action datasets. In [42], an alternative descriptor called a super vector was computed using the maximum *a posteriori* (MAP) adaptation of the means of the GMM. However, the resultant super vector is quite high-dimensional as the GMM contains many mixture components to accommodate all the actions and is also computationally expensive.

Aggregation using sequential models

Some aggregation models consider an action as a sequence of features. Gaidon et al. [28] proposed that each action can be decomposed as a sequence of atomic units called actoms. A histogram of visual features was extracted from each actom, and a sequence of these features was used to model the action. While an actom is annotated manually during training, it is obtained automatically in testing. In another approach [70], actions were decomposed into actionlets and represented using Markov dependencies between these actionlets. Although this method can capture long-term dependencies among actionlets, it requires manual identification of actionlets and explicit modeling. In [133], every action was divided into a set of events, and the start of each event was associated with a latent variable. The action detection problem was then posed as quadratic programming (QP) problem using these latent variables. However, this approach demands precise manual identification of start and end points of the events, and also requires fixed length representation of each event for formulating the QP. A similar problem is encountered in temporal clustering-based methods relying mainly on change detection for identifying and clustering motion segments [143]. Finally, the methods mentioned above do not model fluid actions like *blowing candles* and *playing flute* where marking the start and end of events within the action becomes challenging.

Aggregation using dictionary learning and sparse coding

Sparse activation has been observed in the hippocampus where at most one-third of the neurons get activated for any memory operation [3]. Similarly, in the motor cortex which is responsible for higher-order motion planning and execution, very few neuron activation spikes are discovered when performing certain locomotion activities [7]. These two works show that at higher levels of sensory information processing, the feature representation is highly sparse. For action recognition, Zhu et al. [148] use sparse coding to aggregate HOG3D descriptors obtained from uniformly distributed spatio-temporal regions. The final video descriptor is found using max-pooling on the sparse codes. The dictionary is learned using transfer learning from unlabeled video data. Guha and Ward [31] propose a class-agnostic dictionary for representing all action classes using local motion patterns. However, due to limitations of such a dictionary in admitting new action classes, class-specific dictionaries are suggested for better representation.

Inspired by the sparse representation of objects [72] in a low-dimensional space, Sadanand and Corso [97] propose an *action bank*, where actions are described based on a large set of detectors acting as dictionary elements of a high-dimensional action-space. There are 205

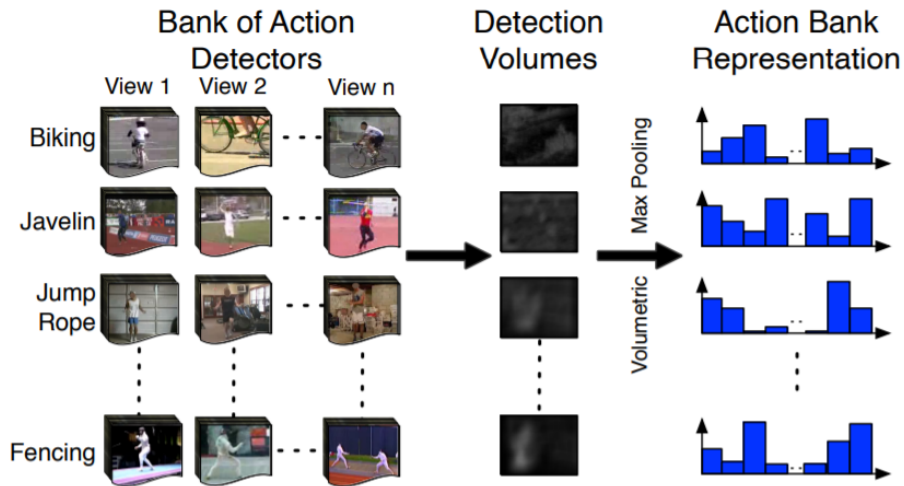


Figure 2.5: Action-bank extraction as adapted from [97]. Best viewed in color.

action templates chosen from different actions in the UCF101 and HMDB51 datasets with multiple examples of actions with high variability and a single example of actions with low variability. Each action template having an average spatial resolution of approximately 50×120 pixels and a temporal length of 40–50 frames in the action-bank. Spatio-temporal volume detectors are applied at three scales (1, 2, and 4) on a video clip. The number of spatio-temporal descriptors is set to 73 ($1^3 + 2^3 + 4^3$) that is based on volumetric max-pooling and considering three levels in the octree. The detectors decompose each input video using 3D Gaussian third derivative filtering and match the decomposed spatio-temporal volumes to the templates in the bank using Bhattacharya distance. This results in a 14965-dimensional (73×205) feature vector for each video clip under consideration as shown in Figure 2.5.

2.2 Deep features for action recognition

With the ever-growing action datasets and increased computation power at disposal, deep neural networks like convolutional neural networks (CNN) have been extensively used for action recognition [49]. Deep architectures are composed of multiple levels of nonlinearities applied on linear combinations of inputs and network weights. Training deep networks is critical because of the non-convexity of the decision surface and gradient algorithms have been most successful when a large number of annotated examples are available [34, 56]. Broadly, deep architectures used for action recognition can be categorized as (a) spatio-temporal networks, (b) multi-stream networks, and (c) generative networks.

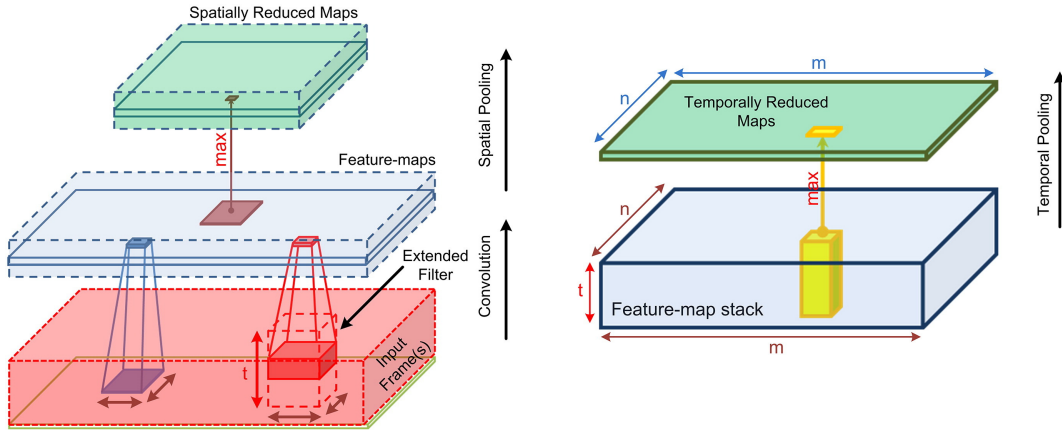


Figure 2.6: Spatiotemporal operations: 2D convolution (blue), 3D convolution on stacks of frames (red) as in [46], conventional spatial max-pooling (brown), and temporal max-pooling (yellow) as in [84]. Best viewed in color.

2.2.1 Spatio-temporal networks

Each convolutional layer in a CNN utilizes the image structure to reduce the search space of the network by *pooling* and *weight-sharing* as shown in Figure 2.6. Pooling and weight-sharing also help in achieving scale and spatial invariance. Visualizing the filters of the first convolutional layer shows that they learn low-level features (e.g., edges) while the top layers capture high-level semantics [138]. This establishes CNNs as generic feature extractors which can be used for the spatial description of actions.

The first approach to action recognition using deep networks was proposed by Ji et al. [46] where spatial convolutional was augmented with temporal information in the form of 3D convolutional networks. A 3D convolution network uses 3D filters extended along space and time to extract spatiotemporal information and motion encoded in consecutive stacks of frames as shown in Figure 2.6. Though the network was provided with supplementary information like optical flow to facilitate the training, a noticeable improvement in classification accuracy was shown for 3D CNNs over 2D CNNs in [46].

One of the drawbacks of 3D convolutional networks is the rigid temporal structure which fails to address the variability in duration for the same action. As the network presented in Ji et al. [46] considers the input of only seven frames, there is no consensus on the right number of input frames for actions with different speeds. This led to various fusion schemes being investigated for the appropriate amount of temporal information to be supplied into convolutional networks. In [84], max-pooling in the temporal domain was found to be preferable. In [49], slow fusion was proposed where a convolutional network is given several consecutive parts of a video, and these parts are processed by the same

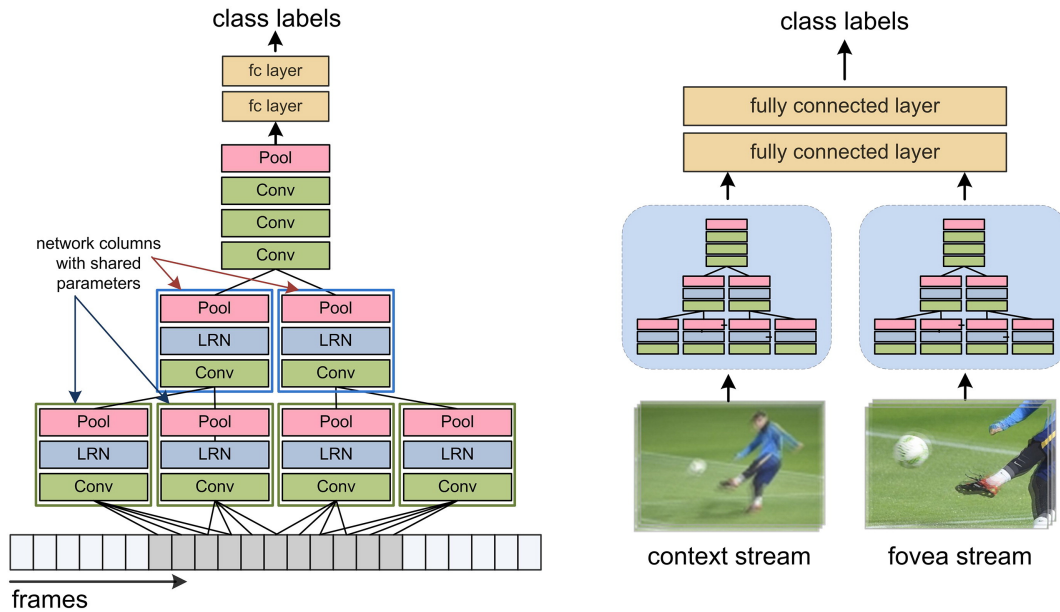


Figure 2.7: Foveated architecture of Karpathy et al. [49]. Green, red, and blue denote normalization, spatial-pooling, and convolutional layers, respectively. Best viewed in color.

layers to produce different temporal responses. These responses are then fused in the fully connected layers to generate a final video descriptor as shown in Figure 2.7. It was also shown in [49] that using two separate networks, i.e., foveal and context streams are more beneficial as shown in Figure 2.7. As the foveal stream focuses on the central region of a frame, the object of interest can be captured in more detail as it often occupies the central region. However, this assumption does not hold for surveillance scenarios where the subject can be anywhere in the frame.

Tran et al.[113] reintroduced 3DCNNs to produce a generic video descriptor called C3D which could be used for object, scene, and action representation. The feature extraction network was trained on a large action dataset Sports-1M [49]. It was shown that a network with homogeneous $3 \times 3 \times 3$ filters performs better than varying the temporal depth of filters. Temporal flexibility is obtained with 3D pooling layers, and the C3D descriptor is obtained by averaging the fully connected layers. Varol et al. [114] demonstrated the effect of 3D convolutions over longer temporal duration by extending the temporal depth of the input and combining the output of networks with varying temporal duration. Though 3D convolutions capture both spatio-temporal information at once, the number of parameters of the network increase substantially. To overcome 3D filters, Sun et al. [111] proposed the factorization of the 3D filter into a combination of 2D and 1D filters which produced similar results as slow fusion in [49].

Another class of temporal structures are called recurrent neural networks (RNN) which

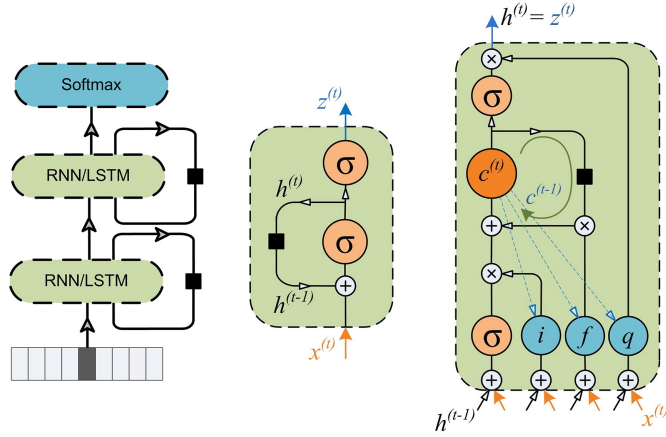


Figure 2.8: Left: The recurrent structure of a RNN/LSTM network. Center: RNN cell structure as a linear dynamical system. Right: The LSTM cell includes additional gate controls. Time delay is indicated with a black square. Adapted from [36].

can model variable length sequences. A recurrent neural network models the dynamics using a feedback loop as shown in Figure 2.8. The typical form of a RNN block accepts an input signal $\mathbf{x}^{(t)} \in \mathbb{R}^d$ and produces an output $\mathbf{z}^{(t)} \in \mathbb{R}^m$ based on its hidden-state $\mathbf{h}^{(t)} \in \mathbb{R}^r$ by

$$\mathbf{h}^{(t)} = \sigma(\mathbf{W}_x \mathbf{x}^{(t)} + \mathbf{W}_h \mathbf{h}^{(t-1)}),$$

and

$$\mathbf{z}^{(t)} = \sigma(\mathbf{W}_z \mathbf{h}^{(t)}),$$

where $\mathbf{W}_x \in \mathbb{R}^{r \times d}$, $\mathbf{W}_h \in \mathbb{R}^{r \times r}$, and $\mathbf{W}_z \in \mathbb{R}^{m \times r}$. However, training a RNN is not easy due to the issue of vanishing (or exploding) gradient [4]. A long short-term memory (LSTM) cell (Figure 2.8) was proposed in [38] to solve this issue by constraining the states and outputs of an RNN cell through control gates.

Donahue et al. [20] use LSTMs for end-to-end training of a composite network over variable length action videos as shown in Figure 2.9. The motion dynamics from a whole clip are captured using an LSTM where each frame is represented as an output of a CNN. The resulting structure named long-term recurrent convolutional network (LRCN) has been shown to be successful not only in recognizing actions but also in captioning images and videos. With the end-to-end learning and CNN-LSTM convolution, the spatiotemporal receptive filter parameters are computed in a data-driven fashion. In [124], temporal segment networks were used to sample entire videos to produce a single feature vector representation. Though the long-term features can summarize an entire video, it is computationally expensive to compute such features for very long videos. Especially, in the case of temporally untrimmed videos which contain background movement, obtaining an adequate

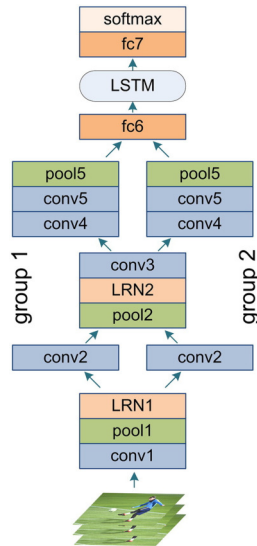


Figure 2.9: LRCN network structure [20]. Each group is a set of convolutional filters applied to a particular set of feature maps from the previous layer. Adapted from [36].

representation for the desired action is challenging.

2.2.2 Multi-stream networks

Multi-stream networks are inspired by visual perception where the motion of an object and its location are handled by separate streams of neurons. Simonyan and Zisserman [102] introduced multi-stream deep convolutional networks for action recognition as shown in Figure 2.10(a). The input for one of the streams called the spatial stream is raw video frames while the temporal stream network uses optical flow fields as input. The spatial stream network is pre-trained on the ILSVRC-2012 image dataset [57] and fine-tuned with the videos from action datasets. The optical flow fields are stacked (early fusion) at the input of the temporal stream network and are trained only from the available video data. The temporal stream is modified to have more than one classification layer where each of those layers operates on a specific dataset (e.g., one for the HMDB51 and the other for the UCF-101 dataset). This realizes multi-task learning, as there are common layers shared between different classification tasks across datasets.

The streams are fused using the softmax scores at a fully connected layer. A softmax function converts a k -dimensional vector of arbitrary real values to a range of k values in the range of $(0, 1)$ that add up to 1. In [24], it is shown that fusion at an intermediate layer improves the performance and reduces the number of parameters as shown in Figure 2.10(b). Notably, fusion at the last convolutional layer produces better classification performance than late fusion in [102]. This alleviates the need for fully connected layers in

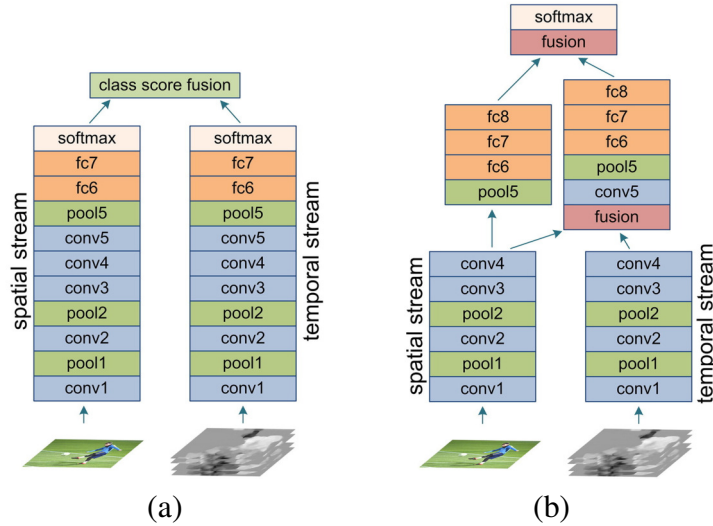


Figure 2.10: (a) The two-stream network by Simonyan and Zisserman [102] with RGB and stacked optical-flow frames as inputs. (b) An example of a two stream intermediate fusion network of Feichtenhofer et al. [24].

both streams and effectively reduces the number of trainable parameters by half.

Traditional descriptors like improved dense trajectories [118] have been used with multi-stream networks in [123]. The iDT features traced using convolutional feature maps of the two-stream network are aggregated using the Fisher Vector. Such a combination yields state-of-the-art performance but not far off from the handcrafted iDT features. This poses a question on whether the deep networks can capture optical flow dynamics better than HOF descriptors in iDT.

2.2.3 Generative networks

As the number of videos on the Web keeps on increasing exponentially, it is pertinent to have a model that can learn in an unsupervised manner as it is impossible to annotate the vast majority of videos. Such a model is called a generative model, and it has been used for sequence analysis to predict the future of a sequence. Given a sequence $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_t$, a generative model may be trained to predict the next instance, i.e. \mathbf{x}_{t+1} . However, accurate predictions are only achieved if contents and dynamics of the sequence are modeled faithfully. Deep-generative architectures like generative adversarial networks [29] aim to learn from temporal data in an unsupervised matter.

Particularly for action recognition, generative models need to discover long-term dynamics for better representation of actions. Hence, in [110] an LSTM auto-encoder model was proposed that consists of RNNs as both encoder and decoder [110] which is shown

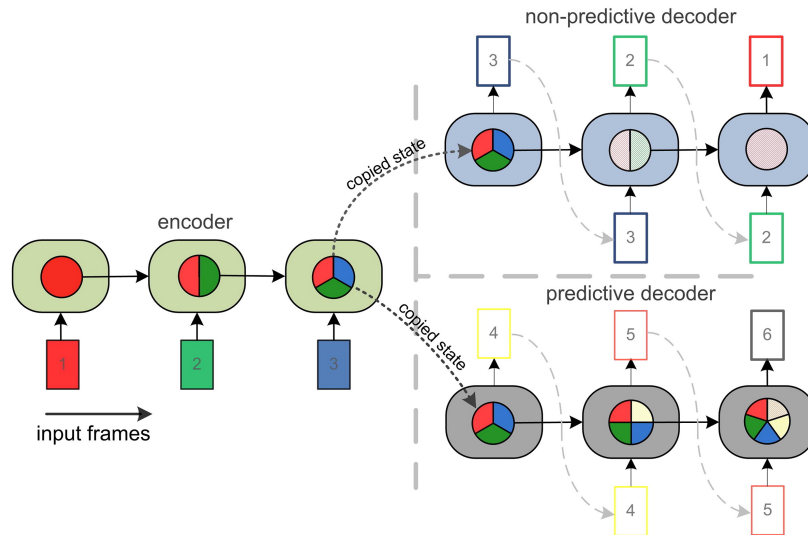


Figure 2.11: LSTM auto-encoder model by Srivastava et al. [120]. The internal states (represented by the circle inside) of the encoder LSTM capture a compressed version of the input sequence (frames 1,2, and 3). The states after that are copied into two decoder models, one of which is reconstructive and the other predictive. The decoder LSTM tries to reconstruct the input frames in the reverse order. The predictive LSTM is trained for predicting the future frames 4, 5 and 6. The colors on the state markers indicate the information from a particular frame. Best viewed in color

in Figure 2.11. The encoder LSTM accepts an input sequence and the states learn both the appearance and dynamics of the sequence. The states are chosen to be the compact representation of the input sequence, and the decoder LSTM tries to reconstruct the input sequence from the compact representation. The LSTM auto-encoder has the added capability to predict the future of a sequence. However, training such a model assumes coherence in subsequent frames of an action video that may not always be present with the sudden appearance/disappearance of actor body parts as in actions with rapid deformations.

2.3 Observations from the review

The entire review of representation learning for action recognition presents two diverse set of approaches - traditional and deep learning based. Within traditional approaches, aggregated descriptors using sparse representations were shown to be effective for representing actions as the subspace of human actions is sparse because of the limited degrees of freedom of human limbs. In order to apply sparse representation, existing approaches use local features extracted using sparse sampling in constrained settings. Instead, we hypothesize that there is inherent sparsity in high-level aggregate descriptors which can be exploited

directly to obtain sparse representations. In Chapter 3, we propose a sparse representation based approach for obtaining discriminative dictionaries for different action classes. The final classification is also based on the sparsest representation of an aggregated descriptor given an action-specific dictionary.

Many of the recognition approaches presented in the review like [46] rely on human body representation for extraction of local features. In many actions, due to rapid body deformations, identification of humans becomes challenging which results in erroneous recognition of actions. In Chapter 4, a sparse representation based approach is presented to describe the different postures of the human body. We show that with such a representation, the identification of the human body under rapid and heavy deformation is much more reliable than CNN based representations.

In our review, most of the observed traditional and deep learning based approaches employ a classification technique to recognize the actions. However, real-world video data is not annotated and obtaining manual annotations is both expensive and unfeasible. Hence, we propose an unsupervised approach for representing and recognizing actions. We build a universal attribute model for learning the atomic motion patterns across actions implicitly. An implicit model is essential because an explicit breakdown of actions into its attributes is subjective and unreliable because of no strict definition of action attributes. Finally, we obtain a fixed-dimensional action-vector for a varying length action clip which contains only the attributes of the action in the clip. Action-vectors are also low-dimensional and address the curse of dimensionality that is demonstrated by most of the aggregate descriptors.

2.4 Summary

In this chapter, the existing literature on the representation of actions was reviewed covering both traditional and deep representations. Most traditional representations were based on aggregated descriptors of local features that holistically describe actions. However, such representations are high-dimensional which introduces the curse of dimensionality when training classification algorithms like support vector machines. Further, deep representations rely on supervised training that requires a large number of labeled examples. In this thesis, we propose low-dimensional representation that can identify subtle differences in actions with minimal or no supervision. Notably, we show that such representations have more discriminative ability than the existing representations as only the unique information is retained in the proposed representations. Finally, we show that even in challenging scenarios like surveillance video footage, the proposed representations can easily identify anomalous actions.

Chapter 3

Sparsity inducing dictionaries for action recognition

The aggregated representations discussed in the last chapter were high-dimensional and in this chapter, a linear decomposition method for low-dimensional embedding is explored. Specifically, we design a sparse dictionary based representation which highlights discriminative information about various action classes. Actions are composed of limb movements and limbs have limited degrees of freedom which means that limb movements cover a sparse subspace of the total space of movements possible. This leads us to conclusion that global representation of actions which capture all movements for a particular action should be effectively sparse. In order to extract this sparsity but retain the essential information that is unique to each action, a reconstruction mechanism like dictionary is used. Any input representation can be built using the sparse linear combination of dictionary atoms. However, such dictionaries can be learned only from the data and this process is termed as dictionary learning.

Dictionaries have been previously used in literature for action classification. In [91], information maximization was used for building discriminative dictionaries. These dictionaries were used to represent action attributes to classify images representing human actions. Sparse modeling for motion analysis was proposed by Castrodad et al. [9] where using highly redundant features, a two-level pipeline was built to distinguish human actions. In [31], three different dictionary were trained - shared (one dictionary for all classes), class-specific, and concatenated (class-specific dictionaries concatenated to form a single dictionary). It was found that class-specific dictionaries perform better on average than the shared and concatenated types. In [33], a sparse dictionary was constructed in an on-line manner for each incoming frame. In case of normal activity, consequent frames are related to each other and dictionary update is minimal. However, any abnormal activity would

cause a major change in the dictionary. A new descriptor known as locally weighted word context was introduced in [121] which is a context-aware spatio-temporal descriptor. A sparse dictionary based on the descriptor was constructed using the joint $L_{2,1}$ -norm where each action category share similar atoms in the dictionary.

In [87], feature encoding methods like vector quantization (VQ), Fisher vector (FV), locality-constrained linear coding (LLC) and soft assignment (SA) were evaluated in the context of sparse coding. Fisher vector was found to be the most suitable representation to form sparse dictionaries using improved dense trajectory (iDT) features [117] on HMDB51 and UCF101 datasets. Lu et al. [77] proposed a new sparse coding scheme in which optimized local pooling was used to form discriminative dictionaries. A multilevel branch-and-bound approach was developed to achieve action localization on videos. This extensive review of sparsity-based dictionary learning methods for action recognition showed that dictionaries can be effectively used for action classification. In [88], the dictionary learning phase and feature encoding phase (e.g. fisher vector with GMM) were studied separately for action recognition. Various features like spatio-temporal interest points (STIP), cuboids, and iDT were used to construct discriminative dictionaries. These dictionaries were formed using GMM, k -means, orthogonal matching pursuit, and sparse coding. They found that the efficacy of dictionaries was not dependant on different feature encoding techniques. In [129], the authors proposed a representation for action recognition based on high-order statistics of the interaction among regions of interest in actions called action-gons. These action-gons were extracted using iDT features and served as discriminative dictionaries. Hence, it can be observed from the literature that dictionaries are able to provide a robust representation of actions on different kinds of features.

The rest of the chapter is organized as follows. The framework for action representation based on dictionary learning is detailed in Section 3.1. In Section 3.2, experimental results are presented for sparse representation on various benchmark datasets which is followed by the summary in Section 3.3.

3.1 Action recognition based on sparse representation using dictionary learning

In this section, a detailed discussion of the proposed method is presented. The classification scheme in typical dictionary learning consists of two phases - dictionary construction from training examples (training) and sparsity based evaluation of test clip (testing). The detailed block diagram of the proposed approach is given in Figure 3.1. In the training phase,

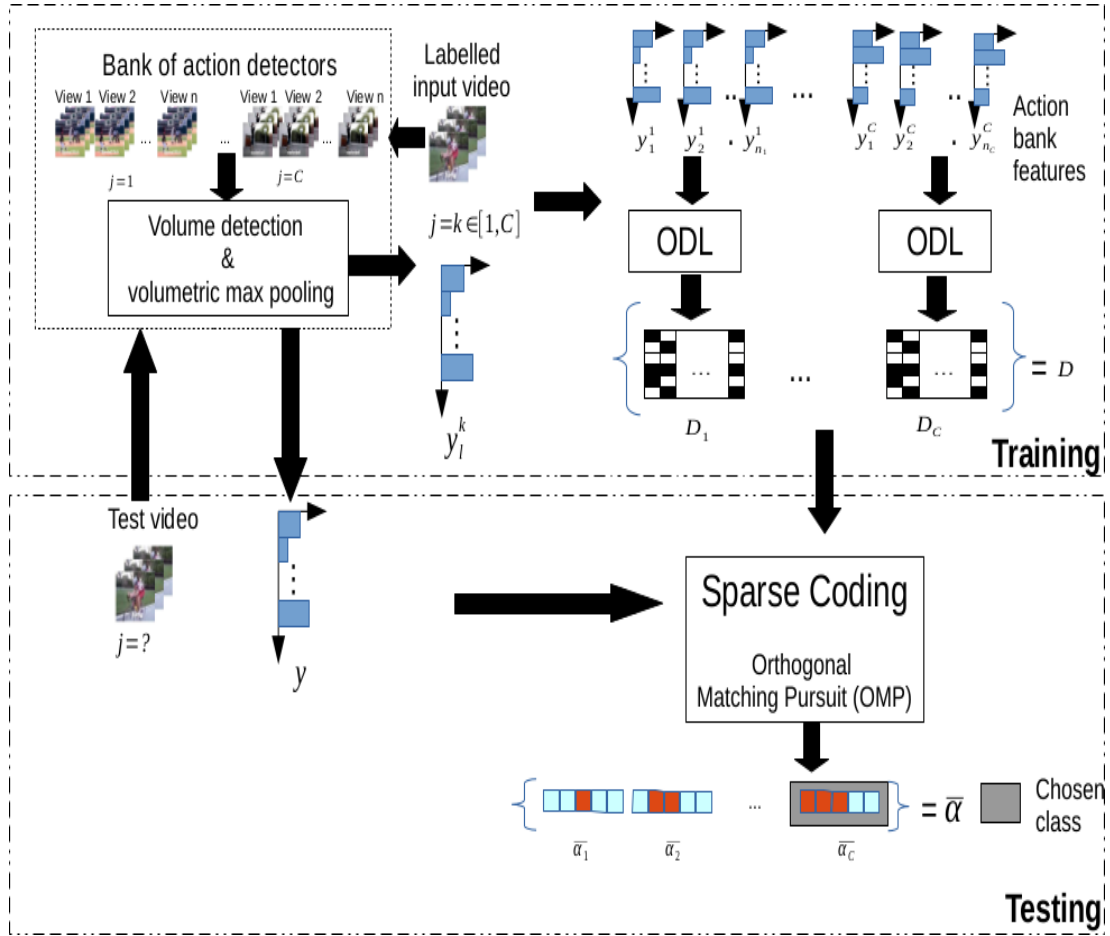


Figure 3.1: Flowchart of the proposed approach

features are extracted from the videos of each action class and dictionaries are constructed using online dictionary learning (ODL). Then, these dictionaries are concatenated to form a single dictionary and each atom in this concatenated dictionary is labelled based on its corresponding class dictionary. In the testing phase, feature descriptors are extracted from a test clip and its sparse representation is calculated using orthogonal matching pursuit (OMP) based on the concatenated dictionary. Finally, the sparse representation is segmented into class specific sparse vectors corresponding to the labels of the atoms in the concatenated dictionary. The L_1 -norm for each of these sparse vectors is calculated and the one with the highest L_1 -norm is considered as the sparsest representation for the test clip. The test clip is then assigned to the action class corresponding to this sparse vector.

3.1.1 Exploring suitable features for sparsification

Action bank features are useful for semantic representation of videos proposed by Sadanand and Corso [96]. A single feature vector is obtained for an entire video clip which is much larger (14965×1) as compared to the number of video clips per class in any of the standard datasets (≈ 100). The resultant matrix is a low-rank rectangular matrix (14965×100) which gives rise to an under-complete dictionary learning setting. From Figure 3.2, it can be observed that the action bank features follow a Laplacian distribution with most of the feature values being zero. This shows that action bank features are indeed suitable for sparsification using dictionaries and in this work, we explore sparsity-inducing dictionaries to achieve a discriminative representation of human actions.

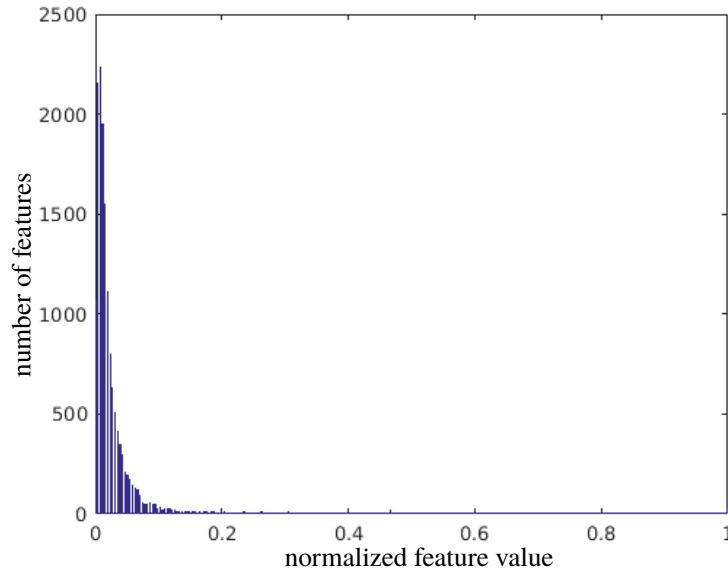


Figure 3.2: Histogram of action bank features for all classes of HMDB51 dataset.

3.1.2 Dictionary based representation

The aim of dictionary learning is to extract a sparse formulation for a set of dense features while retaining the information contained in the feature. Given a set of n m -dimensional features $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^n$, the K -SVD based dictionary learning method [2] finds an optimal dictionary $\mathbf{D}_{m \times k}$ consisting of k atoms and a sparse matrix $\mathbf{X}_{k \times n}$ which best represents the features, as follows:

$$\arg \min_{\mathbf{D}, \mathbf{X}} \|\mathbf{Y} - \mathbf{DX}\|_F^2 \quad (3.1)$$

subject to

$$\|\mathbf{x}_i\|_0 \leq \tau \text{ for } i = 1, \dots, n, \quad (3.2)$$

where \mathbf{x}_i represents i^{th} column of the sparse matrix \mathbf{X} , \mathbf{Y} is the matrix whose columns are \mathbf{y}_i , τ is the sparsity parameter which is empirically determined (0.3 in our experiments), and $\|\cdot\|_F$ denotes the Fröbenius norm. The K -SVD algorithm alternates between sparse coding (finding \mathbf{X}) and dictionary update (finding \mathbf{D}) steps.

On-line dictionary learning (ODL) is an on-line version of k -SVD algorithm proposed by Mairal et al. [81]. The sparse stage in ODL is a Cholesky-based implementation of LARS-lasso algorithm which is similar to k -SVD (equation 3.1) but with a different sparsity constraint based on the L_1 -norm of \mathbf{x} as given in equation 3.3. The sparse vector for the t^{th} incoming feature, \mathbf{y}_t is found using the optimization function :

$$\arg \min_{\mathbf{D}, \mathbf{X}} \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_2^2 + \lambda \|\mathbf{x}_t\|_1 \quad (3.3)$$

In the dictionary update stage, to avoid tuning the learning rate, block coordinate descent is used. It learns one example at a time giving the on-line nature similar to on-line stochastic approximation algorithms. This feature is particularly useful for large datasets. The dictionary \mathbf{D}_t after incorporating the t^{th} example, is calculated with respect to the previous dictionary \mathbf{D}_{t-1} as :

$$\arg \min_{\mathbf{D} \in C} \frac{1}{t} \sum_{i=1}^t \frac{1}{2} \|\mathbf{Y} - \mathbf{D}_{t-1} \mathbf{X}_{t-1}\|_2^2 + \lambda \|\mathbf{x}_i\|_1, \quad (3.4)$$

where C determines the action classes to be trained for.

3.1.3 Sparsity based classification

Suppose we have N classes, C_1, C_2, \dots, C_N consisting of K_1, K_2, \dots, K_N number of training features, respectively. The features belonging to the same class C_i lie approximately close to each other in a low-dimensional subspace [132]. Let \mathbf{b} be a input feature belonging to the p^{th} class, then it is represented as a linear combination of the training samples belonging to class p :

$$\mathbf{b} = \mathbf{D}_p \mathbf{x}_p, \quad (3.5)$$

where \mathbf{D}_p is a $m \times K_p$ dictionary whose columns are the training samples in the p^{th} class and \mathbf{x}_p is a sparse vector for the same class.

In the classification process, the sparse vector \mathbf{x}_j is found for the test feature \mathbf{b}_j using

the dictionaries of training samples $\mathbf{D} = [\mathbf{D}_1, \dots, \mathbf{D}_N]$ by solving the following optimization problem:

$$\arg \min_{\mathbf{x}} \frac{1}{2} \|\mathbf{b}_j - \mathbf{D}\mathbf{x}_j\|_2^2 \quad (3.6)$$

subject to

$$\|\mathbf{x}_j\|_1 \leq T \quad (3.7)$$

and

$$\hat{i} = \arg \max_i \|\delta_i(\mathbf{x}_j)\|_1, i = 1, \dots, N \quad (3.8)$$

where δ_i is a characteristic function that selects the coefficients for class C_i , T represents the sparsity threshold. A test clip \mathbf{b}_j is assigned to class C_i if the absolute sum of sparsity coefficients associated with the i^{th} dictionary is maximum among other classes. This criteria was chosen instead of counting the number of non-zero coefficients as it was found to be better at classification. The reason for using sparsity as classification is that while forming a dictionary for a class, we admit the sparsest representation of features belonging to that class. So, if a test feature belongs to a certain class, it should ideally admit the sparsest representation with respect to that class dictionary and no other.

3.2 Experimental results

In this section, we present the performance of dictionary learning based sparse representation and evaluate the same for different types of feature descriptors. Further, the optimal dictionary size is determined with respect to classification accuracy for each dataset considered for evaluation. This is followed by a thorough analysis of the dictionary learning classification performance on each action class. Finally, a comparison of the proposed sparse representation is presented with state-of-the-art approaches on the benchmark datasets.

3.2.1 Datasets

For evaluation of dictionary based representation, we consider two large action datasets - HMDB51 and UCF50.

HMDB51

The HMDB51 dataset is a very large human action dataset containing 51 action categories, with at least 101 clips for each category. The dataset includes a total of 6,766 video

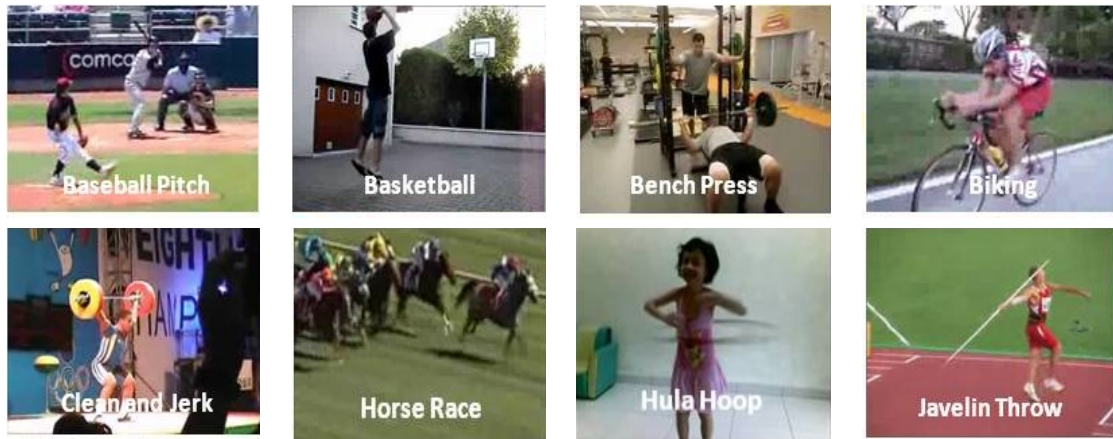


Figure 3.3: Sample actions from HMDB51 dataset

clips (approximately 18 hours) extracted from movies, the Prelinger archive, YouTube and Google videos. Such a variety of sources which have contributed to this database make it very realistic and challenging. Three distinct training and testing splits have been selected from the dataset as provided in [58], with 70 training and 30 testing clips for each category. Some of the sample actions are shown in Figure 3.3.

UCF50

The UCF50 dataset was introduced in [92], consists of 50 sport action categories and all the videos (approximately 17 hours) denoting the actions were collected from YouTube. The dataset consists of more than 100 video clips for each category and gives plenty of variety in terms of camera motion, object appearance and pose, object scale, viewpoint, cluttered background, illumination conditions, etc. The official train/test splits are available at [109] and were used in this work to maintain comparability with the previous literature on these datasets.

3.2.2 Evaluation on different feature descriptors

Different feature descriptors like three dimensional scale invariant feature transform (3D-SIFT) [98], action-gons [129], and others are considered for establishing the robustness of sparse representation. Table 3.1 presents a comparison of classification performance among various features used for learning dictionaries on the HMDB51 dataset. As reported in Table 3.1, the best classification performance of 22.08% was obtained for 3D-SIFT features with a dictionary of size 80. Other features previously used for building dictionaries include IDT features[88] and action-gons[129]. All these representations are based on

spatio-temporal interest points but yield lower performance than action bank. Low-level features capture the entire motion profile in a small spatio-temporal window which makes it highly dense. This means that dictionaries constructed from such features lose significant information during the process of sparsification. However, for sparse features like action-bank that are sparse, the process of construction of dictionaries is not sparse. This explains the effectiveness of dictionaries in representing such features. In the rest of the experiments, we consider only action-bank features for demonstrating the performance of dictionary learning.

Table 3.1: Performance comparison of sparsity-based dictionaries using different features on the HMDB51 action dataset

Feature	Accuracy (%)
3D-SIFT	22.08
Action-gons [129]	58
Improved dense trajectory [88]	59.7
Action Bank	99.87

3.2.3 Classification Performance vs. Dictionary Size

The primary objective of dictionary learning is reconstruction. However, over-fitted dictionaries with perfect reconstruction are not desirable as variability in test examples cannot be handled effectively leading to more mis-classification. Table 3.2 gives the variation of recognition accuracy in terms of dictionary size for HMDB51 and UCF50 datasets. For HMDB51, the maximum performance is noted for dictionary size of 100 with sparsity (lambda value in SPAMS toolbox) set at 2, after which the performance degrades with increase in the dictionary size. In case of UCF50, best classification accuracy is obtained for dictionary size of 120 with sparsity set at 8 after which it degrades sharply. The reason for this decline in performance is that action bank features can be compressed with great effect till the point where all the discriminating characteristics remain. Beyond that point, increasing dictionary size leads to loss of information. This behavior is consistent across datasets and smaller dictionary sizes can produce a fair idea on the average overall classification performance. The only parameter to be tuned is sparsity. It also must be noted that optimal dictionary size is based on the objective at hand and the number of examples available for each class. In our case, the optimal dictionary size is reached where the reconstruction error is relatively low while maintaining high discrimination.

Table 3.2: Effect of dictionary size on performance (in %)

Dictionary Size	HMDB51	UCF50
60	92.33	51.6
80	98.11	60
100	99.87	63.9
120	99.51	72.46
140	98.23	69.6
160	97.56	69.6

3.2.4 Visualization of dictionaries

Dictionaries constructed for sample classes of HMDB51 and UCF50 are presented in Figures 3.4 & 3.5, respectively. The variability in actions of HMDB51 in terms of body movement, posture, and overall appearance is adequately captured in the dictionaries. It is clearly evident that the dictionaries formed for the classes of HMDB51 are indeed distinct from one another. This illustrates that features belonging to different classes do not share a sparse neighbourhood. These distinct dictionaries contribute to better classification performance of dictionaries on the HMDB51 dataset. On the other hand, the dictionaries constructed for few of the classes of UCF50 bear strong similarities. The dictionaries corresponding to classes such as “javelin throw”, “jumping jack”, “kayaking”, “playing guitar”, “nunchunks”, “pole vault”, “pull ups” and “volleyball spiking” are quite similar making it hard to discriminate these classes with sparsity-inducing dictionaries which contributes to lower classification performance on the UCF50 dataset as can be seen in Table 3.4.

In Figure 3.6, the confusion matrix of the UCF50 dataset is presented. *Pole vault* is mis-classified as *kayaking* and *biking* is mis-classified as *juggling balls*. Similarly, *walking with dog* is confused to be *tennis swing*. These confusions are due to the fact that their representative dictionaries are almost identical as shown in Figure 3.5. Also, the confusion matrix for HMDB51 dataset for the best performing dictionary of size 100 is presented in Figure 3.7.

3.2.5 Comparison with state-of-the-art

A summary of the classification performance of previous approaches in literature applied on HMDB51 is presented in Table 3.3. It can be observed that single frame based features like HOG/HOF[59], C2[59], motion interchange patterns [54] demonstrate high mis-classification as they do not consider temporal context while describing action. On the other hand, trajectory features [134], [119], [129] which consider multiple frames to provide temporal description of the motion perform better than single frame based features.

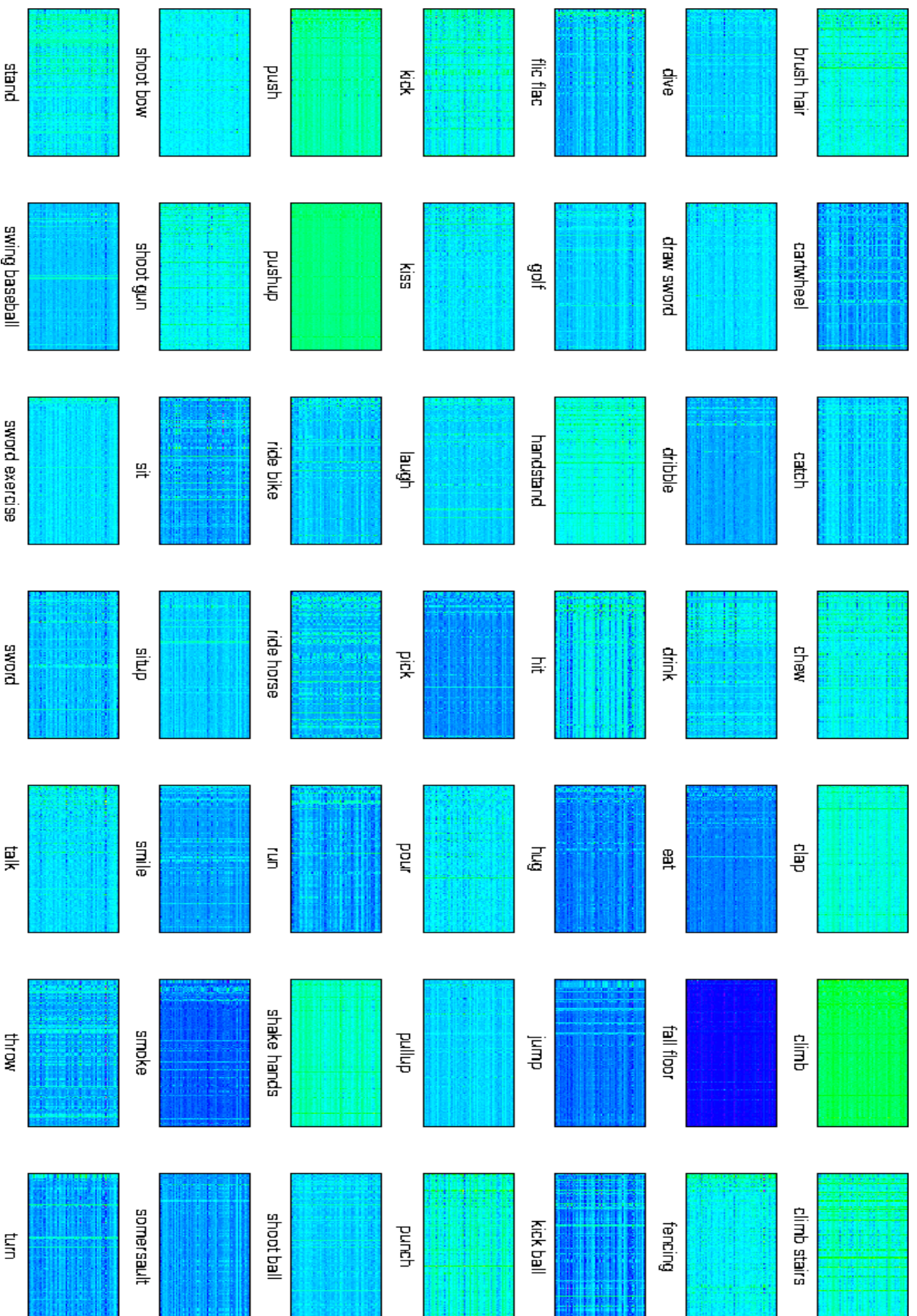


Figure 3.4: Visualization of dictionaries for selected classes in HMDB51. Best viewed in color

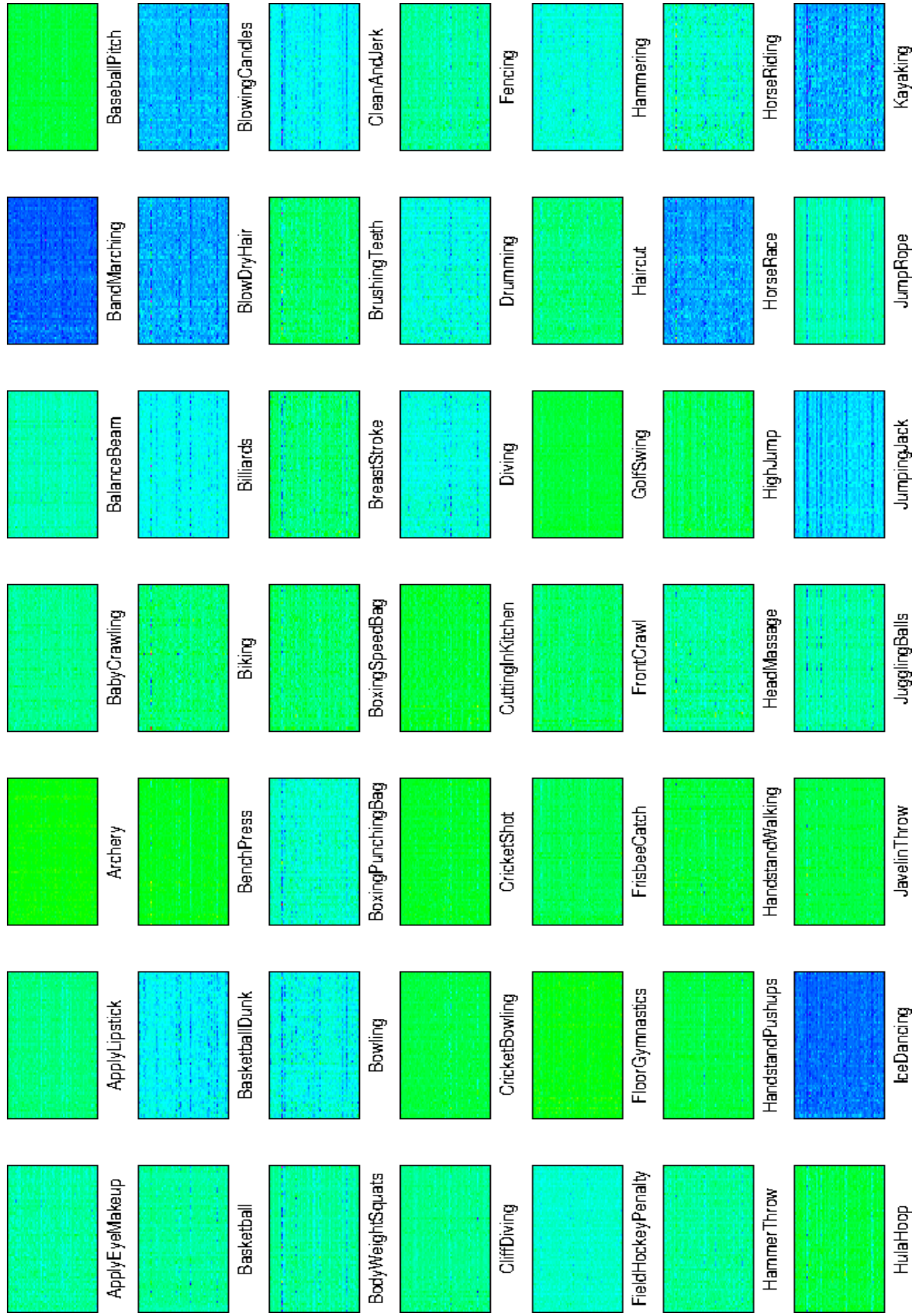


Figure 3.5: Visualization of dictionaries for selected classes in UCF50. Best viewed in color

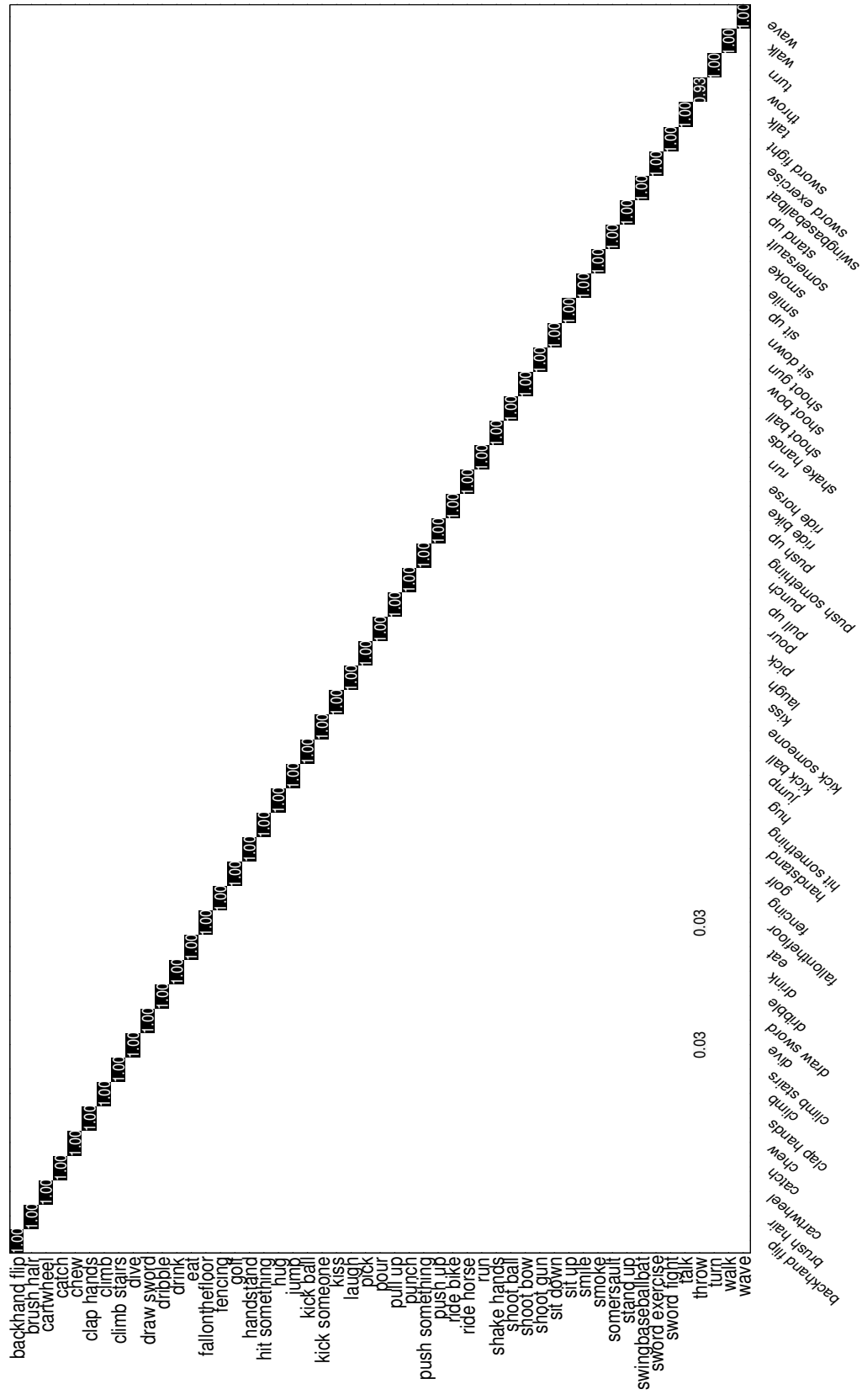


Figure 3.7: Confusion Matrix for HMDB51 dataset for dictionary of size 100. Classification performance : 99.87%

Table 3.3: Comparison of classification performance on the HMDB51 action dataset

Method	Feature	Accuracy (%)
Single-frame based feature		
Kuehne et al. [59]	HOG/HOF	20.20
Kuehne et al. [59]	C2	23.18
Klipper-Gross et al. [54]	Motion Interchange Patterns	29.17
Multiple-frame based feature		
Solmaz et al. [108]	Frequency based 3D spatio-temporal features	29.20
Jiang et al. [47]	Trajectory on motion reference points	40.70
Srivastava et al. [110]	RNN with LSTM	44.1
Wang et al. [119]	Dense trajectory	44.75
Wu et al. [134]	Dense trajectory-aligned	49.46
Liu et al. [75]	Multiple features	49.95
Lan et al. [64]	Local handcrafted features	52.4
Park et al. [86]	Multiple CNNs	54.9
Wang et al. [119]	IDT	57.20
Wang et al. [129]	Action-gons + Sparse Dictionaries	58
Sun et al. [111]	Factorized Spatio-Temporal CNNs	59.1
Simonyan et al. [103]	Two stream CNNs	59.4
Wang et al. [125]	Temporal Pyramid Pooling based CNN	59.7
Peng et al. [88]	IDT + Sparse Dictionaries	59.7
Lan et al. [61]	Space-time Extended Descriptor	62.1
Lan et al. [62]	Long short term motion	63.7
Sadanand et al. [96]	Action bank	26.90
Proposed approach	Action bank + Sparse Dictionaries	99.87

Action bank is also one such representation which uses a spatio-temporal volume across multiple frames but performs slightly better than single frame based features. However, representing action bank features in terms of sparsity-inducing dictionaries improves the performance significantly as shown in Table 3.3. It can be noticed that a similar dictionary transformation of improved dense trajectory features [88] betters the performance only slightly (57.2 to 59.7%). This shows the suitability of action bank features for sparse dictionary based representation. Further, it is also evident from Table 3.3 that the proposed method demonstrates significantly higher classification accuracy than CNN and CNN based

RNN networks presented in [111], [125], [86], [110] and [103].

In Table 3.4, we present the performance of the proposed method on the UCF50 dataset. It can be seen the dictionaries constructed from action bank features perform reasonably well as compared to state-of-the-art but not as well as action bank features. This shows that original features are more discriminative than the sparsity-inducing dictionaries. Further, it also illustrates that applying sparsity constraints while constructing dictionaries may not always lead to better discriminative representation.

Table 3.4: Classification performance on the UCF50 dataset

Method	Accuracy (%)
Kliper-Gross et al. [54]	72.60
Solmaz et al. [108]	73.70
Reddy and Shah [92]	76.90
Todorovic et al. [112]	81.03
Sadanand et al. [96] (Action bank)	76.40
Proposed approach	72.46

3.3 Summary

The main goal of this chapter was to explore the sparsity of feature descriptors to develop a discriminative representation for actions. We showed that using dictionary learning, such a sparse representation could be obtained which could easily distinguish actions in datasets with large number of classes. When applied to different existing feature descriptors, it was found that action-bank features that describe the entire video clip using a single descriptor are the most suitable candidates for building dictionaries. By exploring the inherent sparsity of action-bank features, dictionary based sparse representation was shown to correctly classify almost all test examples in the HMDB51 dataset. Finally, we showed comparable or better performance than the existing state-of-the-art approaches on benchmark datasets.

Chapter 4

Recognition of actions with rapid body deformations using sparse representation

In the last chapter, dictionary learning was used for the sparse representation of actions in order to achieve better discriminability. This was shown to be effective as actions comprise of human movements which have limited degrees of freedom and hence span a sparse subspace [26]. Similarly, structural sparsity has been used to represent different objects, including humans, for continuous tracking [141, 32, 79]. Especially, human body representation is the backbone for many approaches like [46] which rely on human body representation for extraction of local features. However, when the human body is highly contorted and the videos are low-resolution, body representation using pose estimation is difficult. In this chapter, we leverage sparsity to represent and identify the human body under rapid and heavy deformation.

Athletes perform complex movements during actions like *platform diving*, *somersaulting* etc. which cause their bodies to contort in poses which are difficult to recognize. These movements also make continuous tracking a challenge during such sporting events. Further, as athletes can appear at any position in the frame in any pose, the issue of tracking is more complicated than pedestrian tracking [78]. Previous attempts at human tracking involve algorithms that are built by considering a certain shape estimate of the human body [85]. Such trackers require a clear foreground shape model of the object being tracked which is hard to obtain in the case of high deformations and volatile background motion. An alternative suggested in literature is the use of part-based models to track the different parts of the body which are then used for human tracking [25, 140]. While using such methods has shown to improve the accuracy of tracking, the requirements include 1) a large

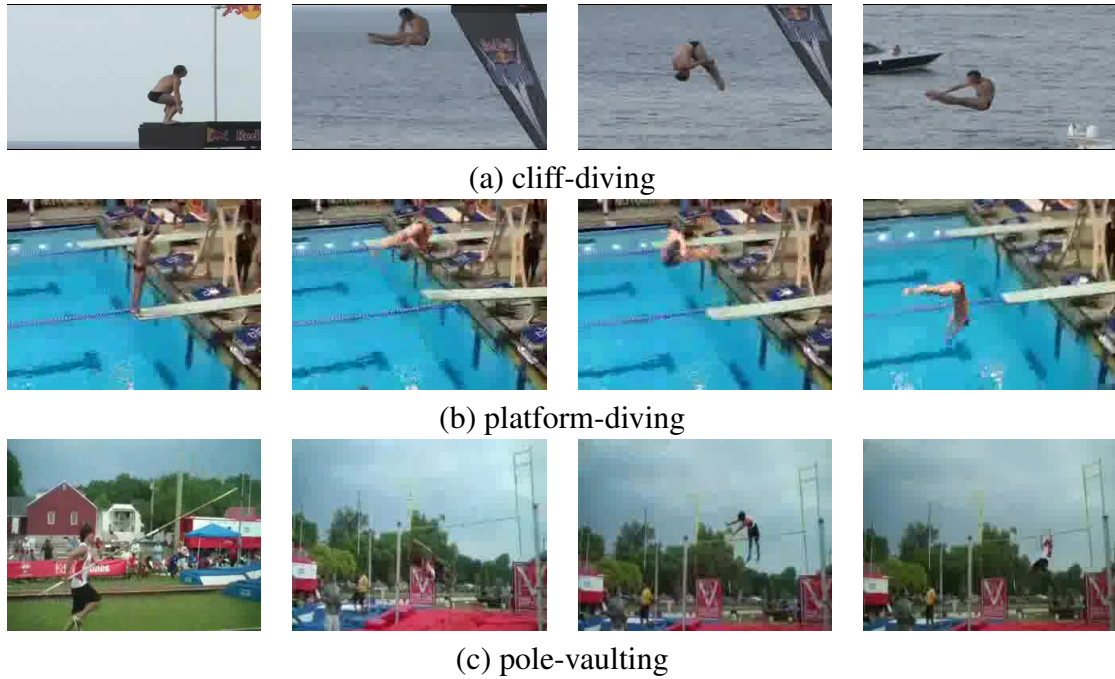


Figure 4.1: Scenarios depicting highly articulated motion leading to complex poses. Best viewed in color.

number of examples with manual annotation, and 2) dependency estimation between the various parts in the form of conditional random fields (CRF) or graph structures. In the case of highly deformed poses, as presented in Figure 4.1, dependency estimation between parts is seldom reliable as most of the body parts are either occluded [21] or do not follow the normal dependency structure [137, 136]. Also, it can be observed from Figure 4.1 that the entire body of the athlete is so tiny that locating and annotating parts automatically is quite challenging. Further, a massive change in body posture can be seen in consecutive images (each frame from left to right is 5 frames apart) for each action presented in Figure 4.1.

From these works, we hypothesize that as all human poses (highly deformed or otherwise) are structurally sparse. This motivates us to create a sparse representation from all the poses of the human body in the form of a dictionary which can be used later for on-line tracking. Notably, this dictionary is also shown to recognize the body of an athlete from the unstructured background in most of the scenarios. Figure 4.2 gives the description of the proposed hybrid framework. For training the dictionary, candidate samples are generated offline from a proposal generator based on the ground-truth available for training images. A convolutional neural network (CNN) modelled on the multi-domain convolutional neural network (MDNet) [83] (discussed in detail in Section 4.2) using pre-trained weights is then used for extraction of features which is used for training a dictionary with a sparsity

constraint. During testing, the first frame is supplied with the ground truth which initializes the tracker. The CNN tracker which is adapted for human target detection is used along with the dictionary verification for final target detection in each frame.

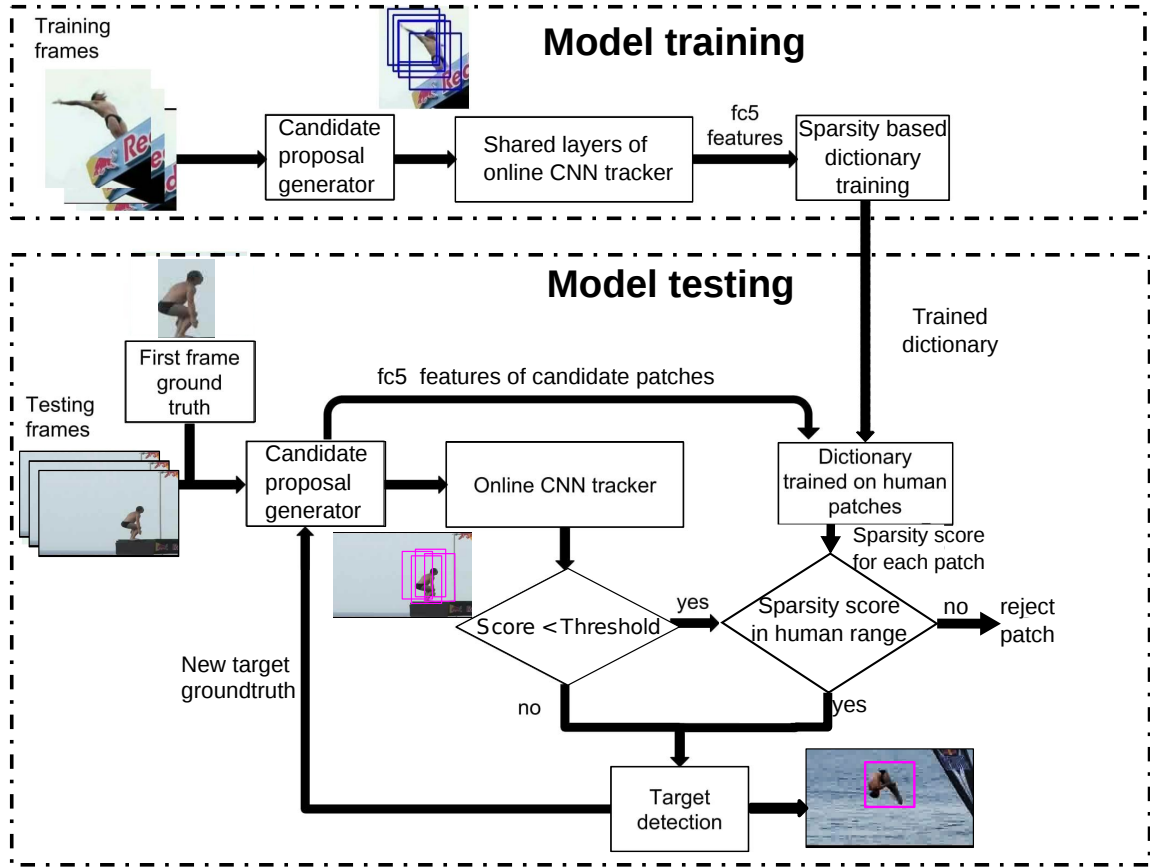


Figure 4.2: Dictionary-CNN hybrid framework for human tracking

The rest of the chapter is organized as follows. In Section 4.1, a brief review of human body representation is presented and the proposed framework based on sparse representation is discussed in Section 4.2. This is followed by experimental results and comparison with state-of-the-art approaches in Section 4.3 and the summary is presented in Section 4.4.

4.1 Review of human body representation

The representation of human body is critical in deduce the action from videos. Especially, recognizing the human body under rapid deformations is key to understanding the action being performed. This is why several diverse approaches have been tried out at both feature extraction and feature association levels for faithful representation and spatial localization. At the feature level, representations ranging from low-level hand-coded features such as

edge and contour detection methods to high-level domain specific features like joint locations, optical flows, depth maps, textures, colour blobs, multi-voxel data, and multiple image silhouettes, have been extensively exploited for building pose estimation based localization frameworks.

At the feature association level, most of the pose estimation approaches can be grouped into three broad categories, mapping functions [52], search strategies [101] and part-based models [99]. Most approaches that come under the first two categories use search strategies and mapping functions in an attempt to estimate all human body articulation parameters simultaneously. As the articulated motion becomes complicated, its description requires high dimensional representation which incurs the curse of dimensionality. Part-based models address this issue by following a bottom-up approach wherein a candidate's loosely connected body parts are detected at first, and all possible configurations of detected parts are inferred at a later stage leading to the low-dimensional representation of complex articulated motion.

A slightly different methodology is followed in [74], where human pose localization is formulated as a joint estimation problem. The modules of foreground segmentation and pose estimation used are a part of a cyclic feedback system. Segmentation supplies a mask for the foreground to a sequential Bayesian filter for pose localization. The localization module, in turn, provides a map of foreground response for the segmentation block. Optimal foreground estimation and pose localization is achieved when the recursive interactions between the two modules reach a steady state causing no significant corrections in either. In [139], a two-stage tree-based optimization is used to build a part-based model where body parts are defined as either associated or abstract. Abstract body parts are used in this model which include not only the loosely connected parts but also the possible constraints between symmetric parts of a given human pose. An association tree built for this model imposes spatio-temporal constraints between body parts in two adjacent frames by generating optimal tracklets from the abstract body parts. Although these human pose localization models perform well to some extent, complex articulated motion poses challenges like the inability to assign tracklets during extremely deformed states of the human body.

Delving further into feature representation, sparsity has been extensively used for transforming features into sparse representations that enhance recognition performance. The inherent structural sparsity associated with human body motion and highly discriminatory property of sparse features hint at the potential of sparse representation in building a robust human pose localization framework. In [106], the structural sparsity of articulated motion of the human body is explored for pose localization by using a probabilistic observation model like the particle filter. However, using the particle filter in a sparsity framework

causes real-time performance issues. To address this challenge, an interest point based representation i.e. a Harris corner detector is applied on all candidate proposal image patches, and a sparsity-based dictionary is trained on the resultant interest point feature descriptors [115]. Though this model handles partial occlusions and scale variances, the limited effectiveness of hand-crafted features limits the performance of sparse representations on highly deformed targets.

Apart from human-specific representations, some approaches have been proposed to track any moving object which can be adapted to the context at hand. Such a framework is MUlti-Store Tracker (MUSTer) [40] where a visual tracker based on short-term and long-term memory model is developed with low-level hand-coded features. An extension can be seen as the multi-domain convolutional neural networks (MDNet) [83], where the update mechanism has been retained but in combination with CNN features. The representation power of CNN features allows MDNet to have better tracking performance than MUSTer. Its architecture consists of two kinds of CNN layers, shared layers (3 convolutional and 2 fully connected) and domain specific-layers (binary classifier), to extract relevant features hierarchically for tracking. The convolutional layer weights are derived from the VGG-16 network [105]. The reason for not using more number of convolutional layers for feature extraction is that spatial information gets diluted as the network depth increases [39]. The next three layers of the network are fully connected layers. Among these fully connected layers, the first two i.e. $fc4$ and $fc5$, are shared layers whereas $fc6$ is a binary classifier. During the offline training phase of the CNN tracker, the weight update takes place only in the $fc4$ and $fc5$ layers which are common to entire domain space i.e. all training video sequences. The $fc6$ layer is made to vary from one domain to the other. This is shown to be immune to several issues like the unconstrained motion of the target, illumination, and scale variances.

4.2 Dictionary-CNN hybrid approach for human body representation

The proposed framework integrates dictionary based representation into an on-line CNN architecture. To motivate the use of the dictionary, we first explain the process of representation by the CNN architecture. The various layers involved in this architecture are depicted in Figure 4.3.

Given a new test frame sequence, and the ground truth of the first frame of the sequence, a predefined overlap threshold is used to separate candidate proposals into positive and

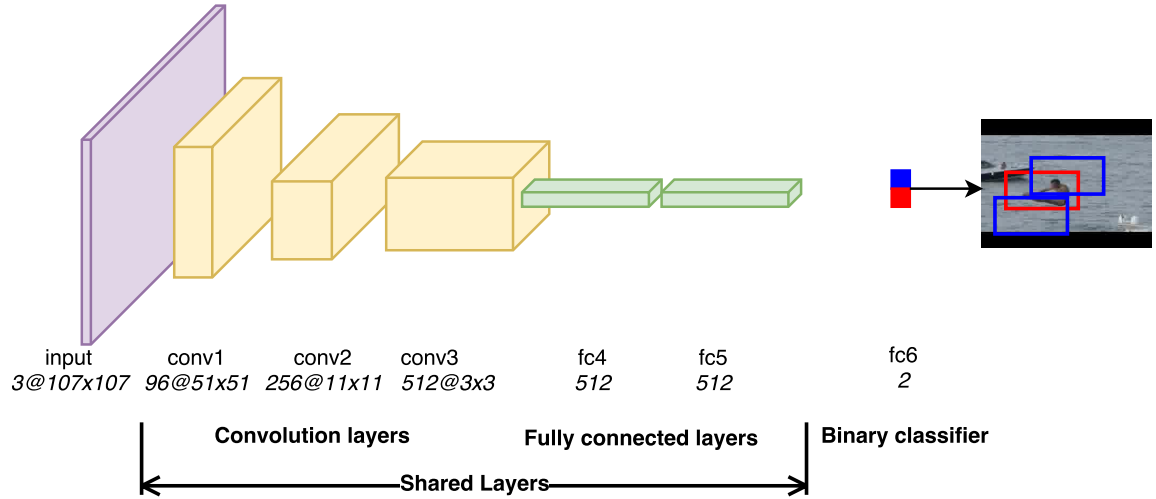


Figure 4.3: CNN architecture adapted from [83]. The input size of 107×107 is designed to obtain 3×3 feature maps in *conv3*: $107 = 75$ (receptive field) + 2×16 (stride). The convolution layers obtained from VGG-16 [105] are not updated. The fully connected layers are learned originally from training videos and shared across videos. The binary classifier is trained from the first frame of the testing video. The fully connected layers and binary classifier are updated when target score drops below the threshold. The red and blue bounding boxes denote positive and negative samples, respectively.

negative samples. The samples are obtained using Gaussian sampling around the provided ground truth. The two classes of samples are then balanced by the number of examples and used to fine-tune *fc4*, *fc5* and *fc6* layer weights. From the second frame on-wards, fine-tuning is performed at regular intervals (5 frames) or if the mean value of the binary classification score for the target at the *fc6* layer drops below a predefined threshold (generally 0.5), whichever happens first. The drop in score indicates the potential failure of the network in estimating the target. Whenever this failure takes place, positive candidate samples that are stored for a short period (5 frames) are used to fine-tune fully connected layers of the tracker. This is called as a short-term update. Apart from the short-term updates, long-term updates are performed at regular frame intervals (100 frames) by using the positive samples collected for a long time-period.

The performance of this on-line CNN architecture on a typical low-resolution video of cliff-diving is shown in Figure 4.4. It is observed that the performance degrades as the target’s shape deforms rapidly and CNN output score becomes highly negative. Invoking either long-term or short-term updates does not help in this case as the deformed target is not contained in either the long-term or short-term memory and hence cannot generate suitable positive candidate proposals. To resolve this issue, we use sparsity to verify the process of candidate proposal selection.

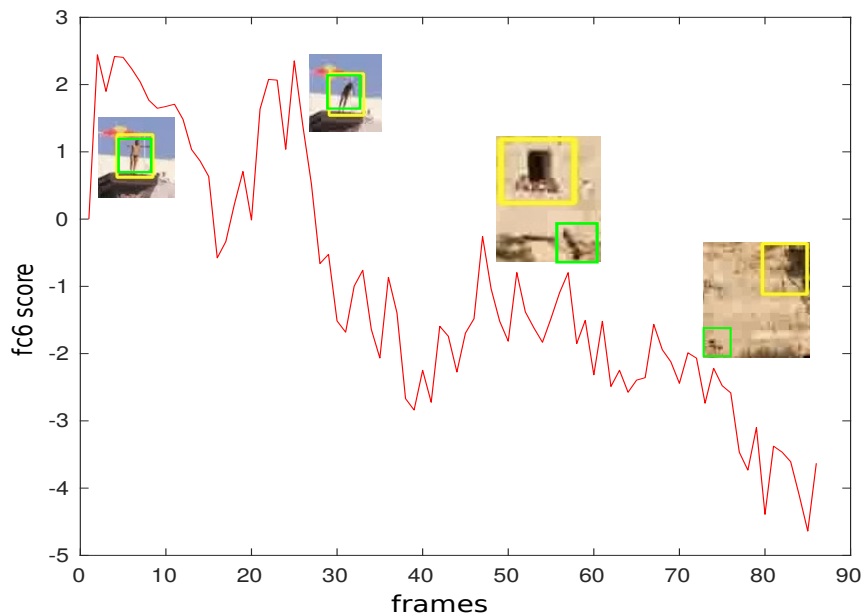


Figure 4.4: $fc6$ scores of MDNet [83] with the tracker output (in yellow) for few of the frames. Green represents the ground truth. Best viewed in color.

4.2.1 Dictionary construction

The dictionary trained in this work is based on human body image patches in different poses. During training, the mean and standard deviation of sparsity scores of features obtained from human patches are noted. In testing, a patch is classified as human if the sparsity score (of its extracted feature) lies within one standard deviation of the training mean for human patches. The number of dictionary atoms k is determined empirically. As we want to minimize the use of manual annotation, a candidate proposal generator is adopted on available ground truths of humans available [116] for HMDB51 [58] and UCF50 [92] to generate a large number of positive samples. The training proposals are generated only from the videos in the official training split available for these two datasets. To demonstrate the performance of sparsity, we choose two features, a) histogram of gradients (HoG) that has been popular for human identification and b) CNN features extracted from the $fc5$ layer of the on-line CNN tracker. Both these features are extracted offline on the positive samples generated from the videos mentioned above.

The sparsity scores for human (in red) and non-human patches (in blue) obtained from the dictionaries learned from HOG, and CNN features ($fc5$ layer features) of human patches are shown in Figure 4.5. The patches whose scores are presented are unseen by the dictionary as they are obtained from the testing videos according to the official training split

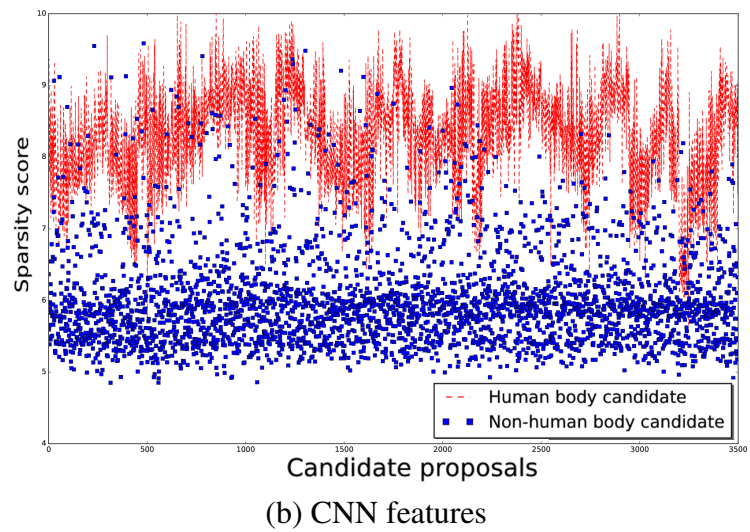
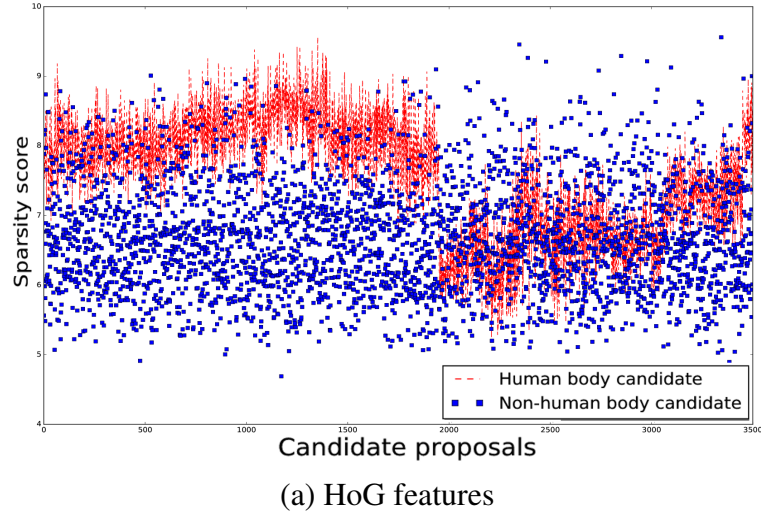


Figure 4.5: Sparsity scores of human patches (in red) and non-human patches (in blue) obtained from dictionaries trained using (a) HoG and (b) CNN features for the 3 classes : cliff-diving, platform-diving, and pole-vaulting. Best viewed in color.

available for UCF50 and HMDB51 datasets. The scores for human and non-human patches are almost identical in case of HoG features which are obtained from a dictionary trained only on human patches. On the other hand, a clear distinction can be seen in the case of CNN features which demonstrates that learning the dictionaries on CNN features is more effective for human representation than HoG features. Further, it should also be noted that during testing, the dictionary trained on CNN features can recognize all poses. This not only demonstrates that the human body is indeed structurally sparse in complex poses but also the same dictionary atoms can be used to represent any human pose, whether normal or highly contorted.

4.2.2 Dictionary based representation

The trained dictionary is transferred to an on-line representation network for testing new sequences as described in Algorithm 2. For the short-term updates, the *fc5* features of the stored positive samples are sent to the dictionary \mathbf{D} to obtain sparsity scores using l_1 -lasso. Here, l_1 -lasso is used to minimize the sum of squared errors for each example \mathbf{y}_i , i.e.

$$\min \|\mathbf{y}_i - \mathbf{D}\mathbf{x}_i\|_2^2 \quad (4.1)$$

subject to

$$\|\mathbf{x}_i\|_1 \leq s, \quad (4.2)$$

where s is the upper bound on $\|\mathbf{x}_i\|_1$ which is set to the maximum l_1 -norm obtained for negative (non-human) samples on trained dictionary \mathbf{D} so that sparsity scores of all patches (human or non-human) can be considered during testing. Then, only if the sparsity score of a particular proposal agrees with the training threshold (within one standard deviation of mean sparsity score obtained for human patches during training), it is used for fine-tuning. The sparsity threshold ρ ($= 0.2$) and the CNN score threshold γ ($= 0.6$) were determined empirically based on the videos used for training. The output of the CNN tracker is mostly incorrect during any drastic appearance change in the tracked body, and as the short-term updates accumulate, the tracker moves way from the human subject. The dictionary check ensures that such negative proposals are always pruned at the earliest, and the short-term updates always provide reliability in localization. Long-term updates are called when the mean *fc6* scores remain continuously below the threshold for many frame sequences. Such updates can sometimes cure the CNN architecture from going astray during tracking, but they require storing a lot of positive samples which increases update time. Further, a lot of wrong short-term updates can cause the long-term update to betray its objective. Dictionaries save computation time as long-term updates are infrequently invoked, and the quality of short-term updates is improved which aids the long-term updates as well. This is discussed in detail in Section 4.3.3.

4.3 Experimental results

This section explains the dataset used for evaluation, compares the performance of the proposed tracker with the state-of-the-art approaches, and visually demonstrates the result of the proposed approach across various rapid body deformation scenarios.

Algorithm 1: Training of CNN architecture

Input : Frame sequence
Pre-trained weights from first 3 convolution layers of VGG-Net
Ground truths of the targets

- 1 **while** *frame sequence is not empty* **do**
- 2 **if** *current frame is the first frame* **then**
- 3 | Initialize $fc4$, $fc5$, $fc6$ weights randomly
- 4 **end**
- 5 Generate positive and negative candidate proposals around target
- 6 **for all candidate proposals** **do**
- 7 | Perform target versus background binary classification
- 8 **end**
- 9 Fine-tune $fc4$, $fc5$ and $fc6$ weight vectors based on errors generated from positive samples
- 10 Remove current frame from frame sequence
- 11 **end**

4.3.1 Dataset

As the degree of deformation varies in different contexts, we chose *platform-diving* and *cliff-diving* from UCF50 [92] and *dive* from HMDB51. Also, these actions contain somersaults, cartwheels, and other acrobatic movements present in gymnastic routines which means that they cover such actions as well. As gymnastic events are performed indoors with the camera placed closer to the actor than diving, we found that it is easier to track the actor. Additionally, the chosen actions have the following characteristics:

- Rapid deformation in body shape during a short interval of time.
- Low-resolution of the target to be tracked.
- Large change in target appearance from the first frame (the ground truth of the first frame is used for initialization of the tracker).
- Highly occluded body parts while the action is being performed.
- Dense background in most of the cases which may contain other humans.

There are a total of 100, 125 and 150 videos for *platform-diving*, *cliff-diving* and *dive* respectively. All the results subsequently presented in this section are obtained on the testing videos based on the official test splits for HMDB51 and UCF50 datasets.

Algorithm 2: Spatial localization of human body using Dictionary-CNN framework

Input : Frame sequence

Initial location of the target

Pre-trained weights from first 3 convolution layers of VGG-Net

Pre-trained $fc4$ and $fc5$ weight vectors from Algorithm 1

Trained dictionary matrix D

CNN score threshold γ

Sparsity threshold ρ

Output: Estimated location of the target

```
1 Generate positive and negative samples on initial location of the target
2 Perform weight update of  $fc4$ ,  $fc5$  and  $fc6$  using positive and negative samples
3 for frame  $t$  in the frame sequence do
4   Produce candidate region proposals
5   Pick the optimal target location  $o$  based on maximum positive scores at  $fc6$ 
6   if positive score  $\geq \gamma$  then
7     Generate positive and negative samples around  $o$ 
8   end
9   if positive score  $< \gamma$  then
10    for each candidate region proposal  $i$  do
11      Calculate sparsity score  $\alpha_i$  based on  $l_1$ -lasso on  $D$ 
12    end
13    Obtain optimal target location  $o$  based on  $\min_i |\alpha_i - \rho|$ 
14    Generate positive and negative samples around  $o$ 
15    Fine-tune  $fc4, fc5$  and  $fc6$  weight vectors based on errors generated from
      positive samples and  $o$ 
16    continue
17  end
18  if  $t \bmod 100 = 0$  then
19    Fine-tune  $fc4, fc5$  and  $fc6$  weight vectors based on errors generated from
      positive samples and  $o$ 
20  end
21 end
```

4.3.2 Metrics for evaluation

As per the recent literature [71], multiple object tracking accuracy has been used as the most common benchmark. In the case of the dataset considered here, there is only a single object of interest, and hence, we use single object tracking accuracy (OTA) which is computed as follows:

$$OTA = \frac{1}{L} \sum_{l=1}^L TP_l, \quad (4.3)$$

where L is total number of frames in all the videos, and the true positive for frame l is denoted as:

$$TP_l = \begin{cases} 1 & \text{if } O_l > \bar{\tau} \\ 0 & \text{if } O_l < \bar{\tau}, \end{cases} \quad (4.4)$$

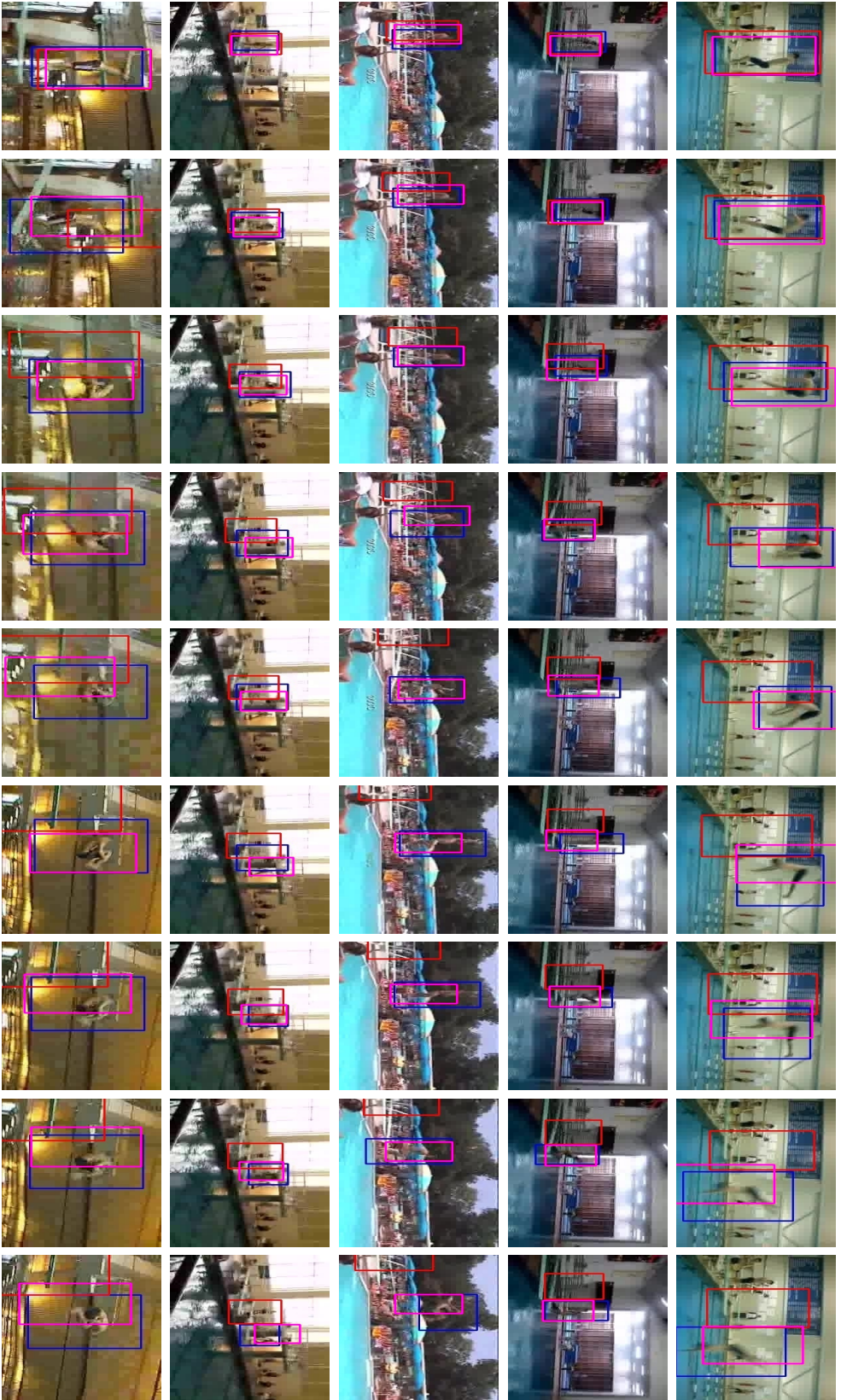
where $\bar{\tau}$ is a pre-defined threshold and $O_l = \frac{|\bar{A}_l \cap A_l|}{|\bar{A}_l \cup A_l|}$, where \bar{A}_l and A_l denote the occupied regions on the image plane for the ground-truth and estimated bounding boxes for frame l , respectively.

To automate the process of recognition and to avoid any manual annotation, we tried obtaining the ground truth in the first frame using the state-of-the-art human detector proposed in [93]. However, due to poor resolution of the target in most of the training videos, we were unable to obtain the bounding box of the target. Hence, the first frame of a video was manually annotated in absence of the ground truth.

4.3.3 Comparative analysis

For comparing the performance of the proposed hybrid framework, a state-of-the-art approach - MDNet [83] is used as the baseline in all the cases demonstrated below. The qualitative results on different instances of both platform-diving and cliff-diving are presented in Figures 4.6 and 4.7, respectively. As it can be observed from these figures, both MDNet and the proposed framework are close to the ground truth in case of small deformations in the human body as shown in Figures 4.6(a) and 4.7(a). However, as the deformations become more complex (Figures 4.6(b) and 4.7(b)), the hybrid tracker clearly gives spatial localization outputs closer to the target than the MDNet tracker [83]. This gap in localization outputs between the two trackers shows the inability of a purely CNN architecture to handle highly deformed human figures. Notably, we have only used the ground truth for evaluation of the trackers and not during the time of tracking in any way.

To demonstrate the effect of dictionary training, we also present the comparison of accuracy between the hybrid dictionary-CNN framework and MDNet for different values of



(a) Small-scale deformations (cols. 1-3)

(b) Large deformations (cols. 4-9)

Figure 4.6: Localization results of Hybrid dictionary-CNN and MDNet [83] for a few platform-diving scenarios. Each row represents a different video sequence. Blue represents ground truth, pink represents hybrid output, and red represents MDNet output. Best viewed in color.



(a) Small-scale deformations (cols. 1-3)

(b) Large deformations (cols. 4-9)

Figure 4.7: Localization results of Hybrid dictionary-CNN and MDNet [83] for a few cliff-diving scenarios. Each row represents a different video sequence. Blue represents ground truth, pink represents hybrid output, and red represents MDNet output. Best viewed in color.

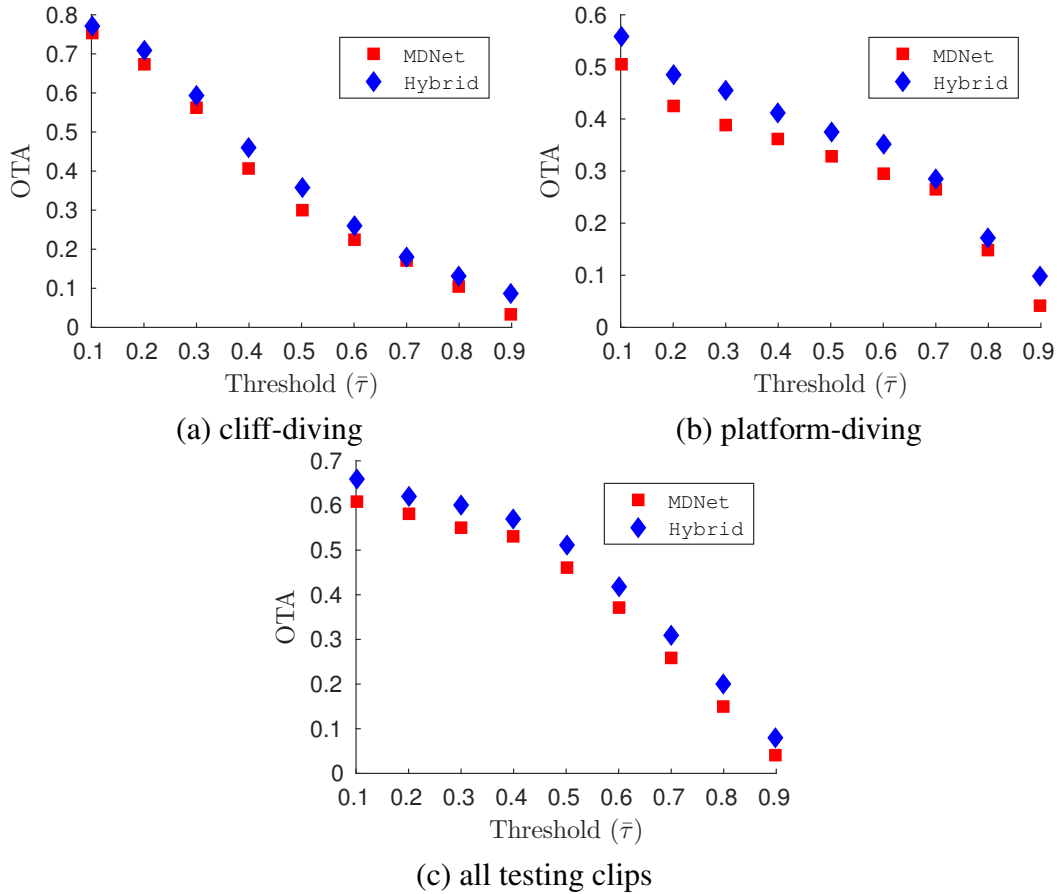


Figure 4.8: Comparison of accuracy (in %) between proposed hybrid framework and MDNet [83]

overlap threshold $\bar{\tau}$ on *cliff-diving*, *platform-diving*, and all clips in Figure 4.8. It can be observed that the hybrid tracker performs consistently better than MDNet for all values of the threshold. The difference in accuracy is a result of the MDNet performing not as well as the hybrid tracker on complex deformation cases. Further, the difference is less pronounced for cliff-diving than for platform-diving as there are fewer humans in the background and the background colour in many cases is in stark to the diver which causes fewer confusions for MDNet. The accuracy presented here encompasses all frames irrespective of the deformation in the target. This is because there is no way to measure the degree of deformation in low-resolution targets whose body-part or pose-based annotations are not available.

As we have seen from the above results, dictionary allows the tracker to work more accurately with less need for short term updates. It is observed that for each testing video, only 3 short-term updates (on an average) were required by the hybrid tracker instead of the 10 short-term updates required by MDNet. This reduces the run-time by 21 seconds (7×3

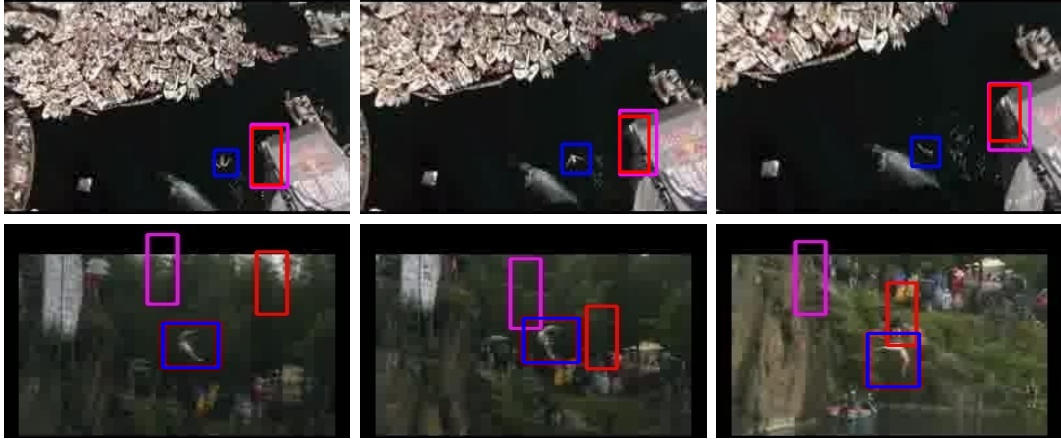


Figure 4.9: Failure cases of human body representation. Each row represents a different scenario. Blue represents ground truth, pink represents the output of the proposed framework, and red represents MDNet output. Best viewed in color.

seconds per update) but after including the time taken for sparsity score calculation, we only save 0.2 seconds per clip over MDNet. However, because of the reduced number of short-term updates, the number of candidate proposals accumulated for long-term update (called once per clip on an average) decreases by 70% which reduces the long-term update time by 3 seconds per clip. Hence, the average savings for a video was observed to be around 3.2 seconds (on a NVIDIA Tesla K20Xm GPU).

The qualitative analysis also produces a few failure cases where neither the hybrid model nor MDNet can track the human target. A few example frames are shown in Figure 4.9 where the target is not identified by either MDNet or the hybrid approach. In these examples, a part of the background structure resembles a human target. This leads to the sparsity output produced by the background structure to be in the sparsity range considered as a human during dictionary training. In low-resolution videos, there may be many possible structures which may look similar to a human body. However, to make the dictionary choose a background structure over a human target, the structure also has to be in close vicinity of the tracked target.

4.4 Summary

In this chapter, we presented a sparse representation based approach to study rapid body deformations in human actions. Particularly, we concentrated on videos captured in low-resolution where it is challenging to estimate the pose of the human body. These scenarios are not handled well by state-of-the-art CNN based localization approaches in the absence of any pose-based or part-based information. We demonstrated that the sparse represen-

tation of the human body can be effectively used to locate the human body even under deformation. Using the sparse representation to guide the CNN based representation was shown to identify low-resolution targets in highly deformed poses. For different types of actions like *cliff diving*, *parallel bars*, *uneven bars*, etc., the proposed approach was able to recognize the human body throughout the action that can further help in the identification of action.

Chapter 5

Unsupervised action recognition using action-vectors

In the last two chapters, we presented supervised dictionary learning based approaches for recognition of actions and also humans performing those actions. However, using supervision requires a large number of annotated examples to train the model in order to achieve a representative model. In this chapter, we present an approach for building an unsupervised action representation for all actions without tracking of actors or action labels. To achieve this, we consider that human actions can be modeled as a sequence of atomic attributes. For example, *boxing* can be interpreted as a combination of attributes like *right-hand punching forward*, *right-hand retracting*, *left-hand punching forward*, and *left-hand retracting*. However, the definition of such attributes is subjective and hence manual annotation of attributes is highly inconsistent [146]. Further, unconstrained videos have inter-actor variability or viewpoint differences causing large deviations within the same attribute and making their explicit extraction difficult. Hence, techniques which require explicit supervision for selection of essential attributes like dictionary learning (see Chapter 3) is not suitable. Further, a sparsity-based representation considers no correlation among the features across actions which is in contradiction to the fact that human actions share correlated attributes like common limb movements. For example, an action like *cricket bowling* contains attributes of another action *running*.

In the aggregation frameworks like Fisher vector [117] and vector of locally aggregated descriptor (VLAD) [73], either the output is a very high dimensional representation, or it is not specifically tuned for each action. So, in the proposed method, we aim to provide a distinct low-dimensional representation for each action. Also, most of the existing works use some supervision in the form of manually annotated bounding boxes for feature extraction[117, 60]. In our proposed method, we extract HOG, HOF, and MBH

features without using bounding boxes or body joint localization. Then we build a universal attribute model (UAM) to estimate the probability density function for implicit attribute extraction of all the actions. Using the UAM removes the need for manual annotation of attributes making it an excellent choice even for fluid actions like *blowing candles* and *playing flute*. Next, a fixed-dimensional super action-vector (SAV) is obtained by concatenating the adapted means of the UAM, given a clip. The super action-vector is intrinsically low-dimensional because an action is composed of only a few attributes. So, to obtain a low-dimensional representation, we decompose the SAV using factor analysis. In the process, we get a low-dimensional representation for each clip which we refer to as an action-vector. Finally, we show that even simple cosine scoring can be used for classifying action-vectors as they are found to be distinctive for each action. This characteristic of the action-vectors eliminates the need for using class labels to build a classifier unlike in most of the existing literature [17, 118, 73]. Figure 5.1 presents a block diagram of the entire process of action-vector extraction.

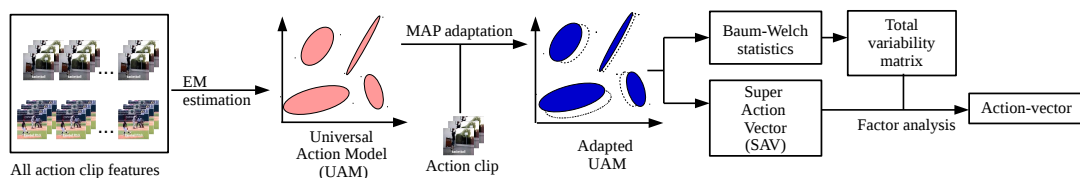


Figure 5.1: Unsupervised attribute modelling and action-vector extraction

The rest of the chapter is organized as follows. In Section 5.1, the process of extraction of action-vectors is presented with universal attribute modelling, super action-vector formation, and action-vector extraction using factor analysis. In Section 5.2, the performance evaluation of action-vectors on UCF101 and HMDB51 datasets is presented and the summary of this chapter is covered in Section 5.3.

5.1 Unsupervised action-vector extraction

We can consider each clip to be a sample function which realizes the random process generating the action. To compare the similarity of two action clips, we need to match the sample functions. Such a match can be based only on the parameters of the *pdf* which describes the random process generating the action. If we assume that the underlying *pdf* can be estimated using a Gaussian mixture model (GMM), then the number of mixtures must be

sufficiently large to accommodate the intra-action variances in unconstrained videos. Unfortunately, a single clip does not have enough data points to estimate the *pdf* of the action. Further, training a GMM for every action has been shown to be challenging especially for actions with few examples [42]. Hence, we propose to train a universal GMM using the clips of all the actions. We call this model the universal attribute model (UAM) which has a large number of mixtures for modelling the attributes of different actions spanning across datasets. However, it was observed that even for a large number of actions spanning multiple datasets, the number of mixtures is not exceedingly large as they share attributes.

5.1.1 Universal attribute model (UAM)

The universal attribute model (UAM) can be represented as follows

$$p(\mathbf{x}_l) = \sum_{c=1}^C w_c \mathcal{N}(\mathbf{x}_l | \boldsymbol{\mu}_c, \boldsymbol{\sigma}_c), \quad (5.1)$$

where the mixture weights w_c satisfy the constraint $\sum_{c=1}^C w_c = 1$ and $\boldsymbol{\mu}_c, \boldsymbol{\sigma}_c$ are the mean and covariance for mixture c of the UAM, respectively. A feature \mathbf{x}_l is part of a clip \mathbf{x} represented as a set of feature vectors $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_L$. This feature can be either a HOF or an MBH descriptor and we train a separate UAM for each during evaluation using standard EM estimation.

We hypothesize that after training the UAM, each Gaussian component in the UAM captures an attribute. This attribute can be specific to a particular action or may be present in multiple actions. In Figure 5.2, we show the UAM posteriors for two actions: *Hulahoops* and *Benchpress*. A posteriors is a representation of posterior probabilities of Gaussian mixtures given the frame (actually the normalized posterior probabilities of all the features extracted from the frame) obtained from the UAM. The posteriors belonging to the clips of action *Hulahoops* (Figure 5.2(a)) show almost identical posterior probability patterns. The black Gaussian mixtures represent the attributes which contribute to the action *Hulahoops*. The slight changes in the patterns for two clips of the same action can be attributed to actor and viewpoint variations. A similar trend follows for the *Benchpress* action in Figure 5.2(b).

As the goal is to find the *pdf* of the action that generates a clip, we need to adapt the UAM parameters using the data in the clip. We perform a maximum *a posteriori* (MAP) adaptation similar to [89, 42] for obtaining the requisite *pdf* which describes the clip.

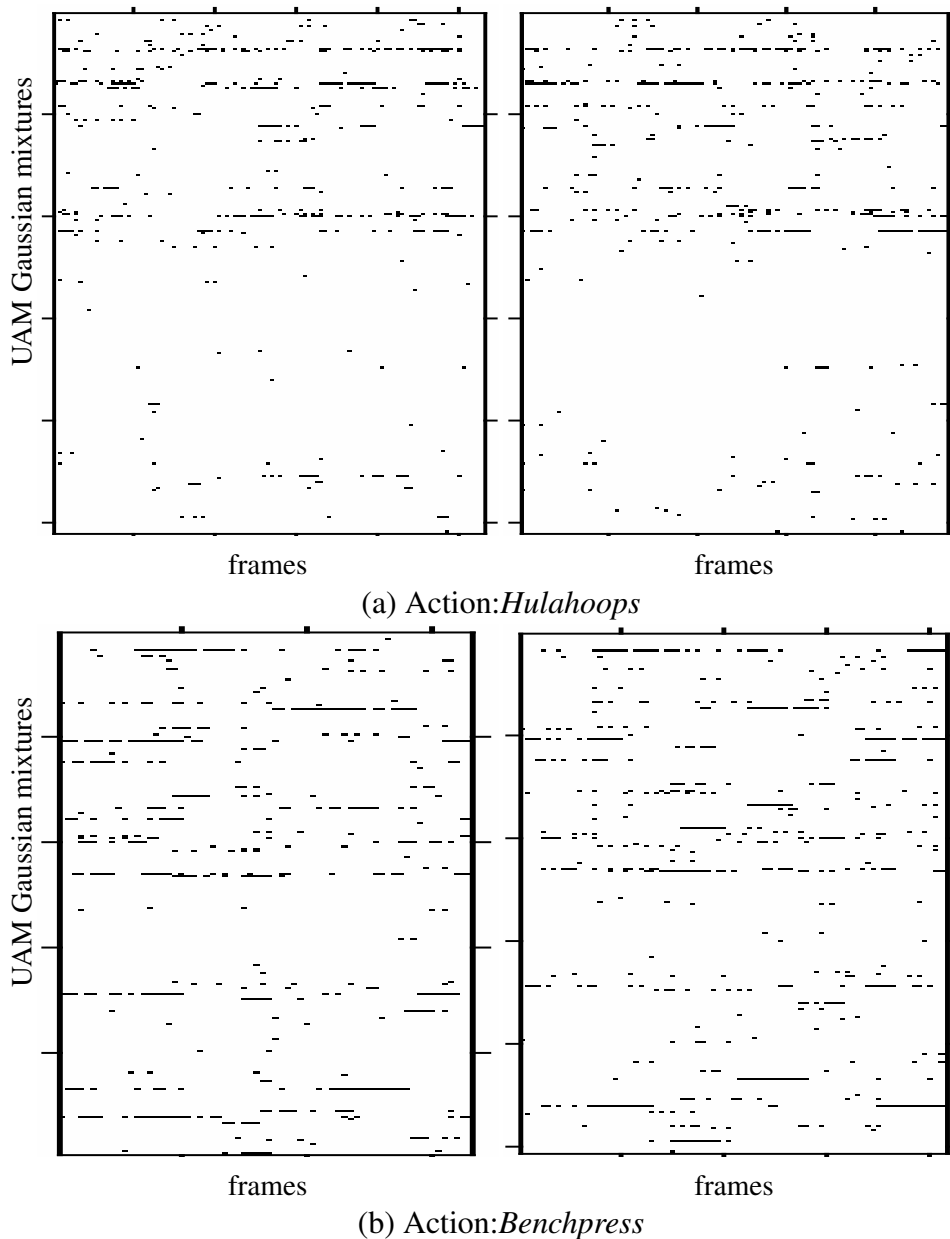


Figure 5.2: UAM posteriograms (using 256 Gaussian mixtures) for two actions of UCF101: (a) *Hulahoops* and (b) *Benchpress*. Although the two clips of *Hulahoops* have variable number of frames, the sequence of Gaussian mixtures having the highest posterior probability (in black) is similar throughout the action. These mixtures represent the attributes which contribute to the action and the slight deviations may be caused by actor or viewpoint variability. Similar behavior can be observed in the clip of *Benchpress*.

5.1.2 Representation of actions as super action-vector (SAV)

The UAM parameters are adapted for every clip to enhance the contribution of the attributes present in it. This adaptation requires updating the parameters of each of the mixture components in the UAM. Given L feature vectors of a clip \mathbf{x} , the probabilistic alignment of these feature vectors into each of the C mixture components of the UAM is calculated as a posterior $p(c|\mathbf{x}_l)$ which is computed as

$$p(c|\mathbf{x}_l) = \frac{w_c p(\mathbf{x}_l|c)}{\sum_{c=1}^C w_c p(\mathbf{x}_l|c)}, \quad (5.2)$$

where \mathbf{x}_l is a $d \times 1$ feature vector and $p(\mathbf{x}_l|c)$ is the likelihood of a feature \mathbf{x}_l arriving from a mixture c . The prior probability $p(c)$ is given by the corresponding mixture weight w_c .

The computed posterior probability $p(c|\mathbf{x}_l)$ is then used to calculate the zeroth and first order Baum-Welch statistics for a clip \mathbf{x} given by

$$n_c(\mathbf{x}) = \sum_{l=1}^L p(c|\mathbf{x}_l), \quad (5.3a)$$

and

$$\mathbf{F}_c(\mathbf{x}) = \frac{1}{n_c(\mathbf{x})} \sum_{l=1}^L p(c|\mathbf{x}_l) \mathbf{x}_l, \quad (5.3b)$$

respectively. The MAP adapted parameters of a clip-specific model can be obtained as a convex combination of the UAM and the clip-specific statistics. For every mixture c of the UAM, the adapted weights and means are calculated as

$$\hat{w}_c = \alpha n_c(\mathbf{x})/L + (1 - \alpha)w_c, \quad (5.4a)$$

$$\hat{\boldsymbol{\mu}}_c = \alpha \mathbf{F}_c(\mathbf{x}) + (1 - \alpha)\boldsymbol{\mu}_c. \quad (5.4b)$$

The covariance is not modified as there is not enough data in one clip to update the entire covariance matrix of the UAM. The adapted means for each mixture are then concatenated to compute a $(Cd \times 1)$ -dimensional SAV for each clip represented as

$$\mathbf{s}(\mathbf{x}) = [\hat{\boldsymbol{\mu}}_1 \hat{\boldsymbol{\mu}}_2 \cdots \hat{\boldsymbol{\mu}}_C]^t. \quad (5.5)$$

Obtaining a fixed-dimensional representation like the super action-vector normalizes the effect of varying length clips but results in a high-dimensional representation. This representation though contains many of the attributes that do not contribute to the clip and hence

are not changed from the original UAM. It follows from Equation 5.2 where the likelihood $p(\mathbf{x}_l|c)$ is close to zero for mixtures in the UAM which are not responsible for modelling the attributes and hence the features belonging to the clip. This translates to posterior probability $p(c|\mathbf{x}_l)$ being close to zero for the same mixtures. Further, the statistics for these mixtures, $n_c(\mathbf{x})$ and $\boldsymbol{\mu}_c(\mathbf{x})$, are also close to zero following Equations 5.3a and 5.3b. Finally, the MAP adapted weights \hat{w}_c and means $\hat{\boldsymbol{\mu}}_c$ for these non-contributing mixtures to be the same as that in the UAM. Since each clip contains only a few of the total UAM mixtures (attributes), only those means are modified. Hence, the SAV is intrinsically low-dimensional, and by using a suitable decomposition, we can extract such a representation which we refer to as an action-vector.

5.1.3 Extraction of compact action-vectors using factor analysis

In order to arrive at a low-dimensional representation, the SAV \mathbf{s} is decomposed as

$$\mathbf{s} = \mathbf{m} + \mathbf{T}\mathbf{w}, \quad (5.6)$$

where \mathbf{m} is assumed to be an actor and viewpoint independent supervector. Such a supervector can be initialized using the UAM supervector as the UAM is trained using large number of actors and viewpoints resulting in a distribution that is marginalized over views and actors. Also, \mathbf{T} is a low-rank rectangular matrix known as the total variability matrix of size $Cd \times r$, and a r -dimensional random vector \mathbf{w} whose prior distribution is assumed to be a standard Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{I})$ [18]. We refer to this random vector as an *action-vector* which is a hidden variable and is defined by its posterior distribution $P(\mathbf{w}|\mathbf{x})$ after observing a clip \mathbf{x} as $P(\mathbf{w}|\mathbf{x}) \propto P(\mathbf{x}|\mathbf{w})\mathcal{N}(\mathbf{0}, \mathbf{I})$

$$\begin{aligned} &\propto \exp\left(\mathbf{w}\mathbf{T}^t\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{s}}(\mathbf{x}) - \frac{1}{2}\mathbf{w}^t\mathbf{T}^t\mathbf{N}(\mathbf{x})\boldsymbol{\Sigma}^{-1}\mathbf{T}\mathbf{w} - \frac{1}{2}\mathbf{w}^t\mathbf{w}\right) \\ &= \exp\left(\mathbf{w}\mathbf{T}^t\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{s}}(\mathbf{x}) - \frac{1}{2}\mathbf{w}^t\mathbf{M}(\mathbf{x})\mathbf{w}\right) \\ &= \exp\left(-\frac{1}{2}(\mathbf{w} - \mathbf{L}(\mathbf{x}))^t\mathbf{M}(\mathbf{x})(\mathbf{w} - \mathbf{L}(\mathbf{x}))\right) \times \text{constant}, \end{aligned} \quad (5.7)$$

where $\boldsymbol{\Sigma}$ is a diagonal covariance matrix of dimension $Cd \times Cd$ and it models the residual variability not captured by the total variability matrix \mathbf{T} . The matrix $\mathbf{L}(\mathbf{x})$ is defined as

$$\mathbf{L}(\mathbf{x}) = \mathbf{M}^{-1}(\mathbf{x})\mathbf{T}^t\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{s}}(\mathbf{x}) \quad (5.8)$$

where $\tilde{\mathbf{s}}(\mathbf{x})$ is the centered supervector which appears because the posterior distribution of \mathbf{w} is conditioned on the Baum-Welch statistics of the clip centered around the means of the UAM. The first order Baum-Welch statistics centered around the UAM mean can be obtained as

$$\tilde{\mathbf{F}}_c(\mathbf{x}) = \sum_{l=1}^L p(c|\mathbf{x}_l)(\mathbf{x}_l - \boldsymbol{\mu}_c). \quad (5.9)$$

We can now express $\tilde{\mathbf{s}}(\mathbf{x})$ as the concatenated first-order statistics given below

$$\tilde{\mathbf{s}}(\mathbf{x}) = [\tilde{\mathbf{F}}_1(\mathbf{x})\tilde{\mathbf{F}}_2(\mathbf{x}) \cdots \tilde{\mathbf{F}}_C(\mathbf{x})]^t. \quad (5.10)$$

Also, the matrix $\mathbf{M}(\mathbf{x})$ is defined as

$$\mathbf{M}(\mathbf{x}) = \mathbf{I} + \mathbf{T}^t \boldsymbol{\Sigma}^{-1} \mathbf{N}(\mathbf{x}) \mathbf{T}, \quad (5.11)$$

where $\mathbf{N}(\mathbf{x})$ is a diagonal matrix of dimension $Cd \times Cd$ whose diagonal blocks are $n_c(\mathbf{x})\mathbf{I}$, for $c = 1, \dots, C$ and \mathbf{I} is the identity matrix of dimension $d \times d$.

From Equation 5.7, the mean and covariance matrix of the posterior distribution are given by

$$E[\mathbf{w}(\mathbf{x})] = \mathbf{M}^{-1}(\mathbf{x}) \mathbf{T}^t \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{s}}(\mathbf{x}) \quad (5.12a)$$

and

$$\text{Cov}(\mathbf{w}(\mathbf{x}), \mathbf{w}(\mathbf{x})) = \mathbf{M}^{-1}(\mathbf{x}), \quad (5.12b)$$

respectively. Using EM algorithm [51], we iteratively estimate the posterior mean and covariance in the E-step and use the same to update \mathbf{T} and $\boldsymbol{\Sigma}$ in the M-step.

In the first E-step of the estimation, \mathbf{m} and $\boldsymbol{\Sigma}$ are initialized with the UAM mean and covariance, respectively. For the total variability matrix \mathbf{T} , a desired rank r is chosen, and the matrix is initialized randomly. Then $E[\mathbf{w}(\mathbf{x})]$ and $\text{Cov}(\mathbf{w}(\mathbf{x}), \mathbf{w}(\mathbf{x}))$ calculated according to Equations 5.12a & 5.12b.

In the M-step, the matrix \mathbf{T} is calculated as the solution of

$$\sum_{\mathbf{x}} \mathbf{N}(\mathbf{x}) \mathbf{T} E[\mathbf{w}(\mathbf{x}) \mathbf{w}^t(\mathbf{x})] = \sum_{\mathbf{x}} \tilde{\mathbf{s}}(\mathbf{x}) E[\mathbf{w}^t(\mathbf{x})], \quad (5.13)$$

which results in a system of r linear equations. The right hand side of Equation 5.13 contains $\tilde{\mathbf{s}}(\mathbf{x})$ which also accounts for the number of features in the clip. As \mathbf{T} is the same for all the clips, the left hand side is also weighed by $\mathbf{N}(\mathbf{x})$ to account for the number of features in the clip.

For each $c = 1, \dots, C$, the residual matrix Σ is estimated mixture by mixture as

$$\Sigma_c = \frac{1}{n_c(\mathbf{x})} \left(\sum_{\mathbf{x}} \tilde{S}_c(\mathbf{x}) - \mathbf{M}_c \right) \quad (5.14)$$

where \mathbf{M}_c denotes the c th diagonal block of the $Cd \times Cd$ matrix $\frac{1}{2} \sum_{\mathbf{x}} \tilde{\mathbf{s}}(\mathbf{x}) E[\mathbf{w}^t(\mathbf{x})] T^t + T E[\mathbf{w}(\mathbf{x})] \tilde{\mathbf{s}}^t(\mathbf{x})$ and $\tilde{S}_c(\mathbf{x})$ is the second-order Baum-Welch statistics of the clip centered on the means of the UAM calculated as

$$\tilde{S}_c(\mathbf{x}) = \text{diag} \left(\sum_{l=1}^L p(c|\mathbf{x}_l) (\mathbf{x}_l - \boldsymbol{\mu}_c) (\mathbf{x}_l - \boldsymbol{\mu}_c)^t \right). \quad (5.15)$$

After the final M-step i.e. estimation of \mathbf{T} and Σ matrices, the action-vector for a given clip can be represented using the mean of its posterior distribution as follows

$$\mathbf{w}(\mathbf{x}) = (I + \mathbf{T}^t \Sigma^{-1} \mathbf{N}(\mathbf{x}) \mathbf{T})^{-1} \mathbf{T}^t \Sigma^{-1} \tilde{\mathbf{s}}(\mathbf{x}). \quad (5.16)$$

This process of obtaining the action-vector is known as *factor analysis* [51]. The \mathbf{T} -matrix contains the eigenvectors of the largest r eigenvalues of the total variability covariance matrix [18]. We hypothesize that these large eigenvalues arrive from the Gaussian mixture(s) which model the attributes in the clip. The original SAV can now be projected onto a r -dimensional action-vector based on T . The 2-D visualization of action-vectors ($r=200$) is presented in Figure 5.3(b) using t -distributed stochastic neighbor embedding (t -SNE) [80]. It follows that most of the actions of UCF101 form easily identifiable clusters that is in contrast to Figure 5.3(a) where highly overlapping MBH features can be seen for the same actions. Hence, the proposed approach can effectively represent the video clip in a fixed dimensional space. The ability to obtain such a lower dimensional embedding confirms our hypothesis that SAVs are intrinsically low-dimension. The visualization of action-vectors for the MBH feature (Figure 5.3(b)) shows that there is a general distinction among actions which is obtained without the use of action labels. This visualization leads us to explore the viability of action-vectors for classification without explicitly training a classifier. The inherent separation means that the factors extracted represent the SAV faithfully. Factor analysis yields such descriptive factors if the data follows a Gaussian distribution. In Figure 5.4, we visualize the SAV of an action class in the HMDB51 dataset which is transformed to 2 dimensions using t -SNE and the histogram of the resulting projections is plotted. It can be observed that indeed SAVs of all action classes considered together follow a Gaussian distribution and this explains the representation power of action-vectors.

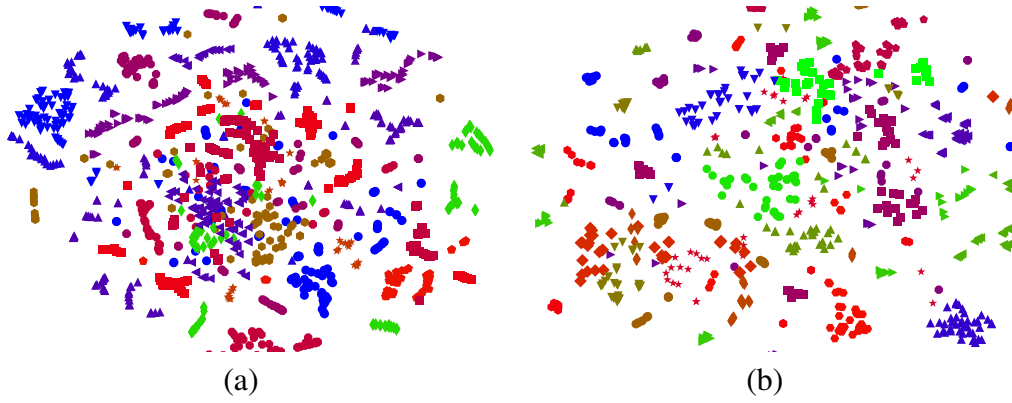


Figure 5.3: t-SNE visualization on selected classes of UCF101 (a) MBH features (b) MBH action-vectors. Best viewed in color.

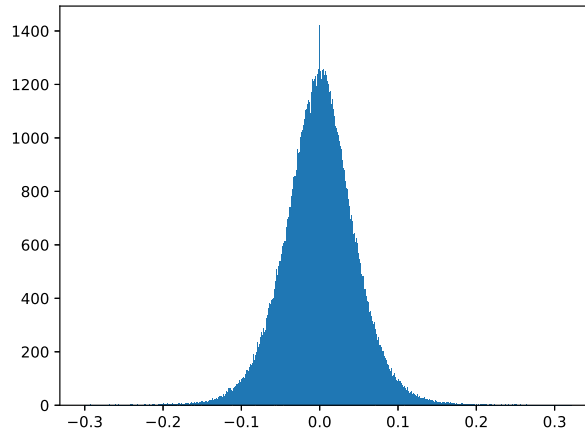


Figure 5.4: Histogram of super action-vector for MBH features of an action class in HMDB51 dataset (reduced to 2 dimensions using t-SNE).

5.1.4 Unsupervised action-vector scoring

We started with the goal of comparing clips based on the underlying *pdf* of their actions and arrived at an action-vector representation. The *pdf* was estimated using a GMM where each mixture is assumed to learn an attribute. Hence, the action-vector extraction method of obtaining useful attributes to represent a clip is equivalent to obtaining the *pdf* for that clip. Now, once we have obtained such a normalized representation (because of $\mathbf{N}(\mathbf{x})$) in Equation 5.16, we can directly compare the action-vectors of any two clips using cosine scoring without the use of labels. For cosine scoring, the distance between a pair of action-vectors is expressed as

$$k(\mathbf{w}_1, \mathbf{w}_2) = \frac{\mathbf{w}_1^t \mathbf{w}_2}{\sqrt{\mathbf{w}_1^t \mathbf{w}_1} \sqrt{\mathbf{w}_2^t \mathbf{w}_2}}. \quad (5.17)$$

During testing, the average cosine similarity of the clip with all the training clips from each class is calculated and the clip is assigned the class with the highest average score.

5.2 Experimental Results

In this section, we present the performance of action-vectors under various configurations of the universal attribute model (UAM) and with different features.

5.2.1 Datasets

A detailed performance analysis of action-vectors is performed on the UCF101 and HMDB51 (details in Section 3.2.1) datasets.

UCF101

UCF101 is one of the largest trimmed action dataset consisting of 13000+ clips of human actions spanning 101 classes [109]. Each video clip contains only the action of interest, and the average duration of each clip is around 7.21 seconds. A three-fold cross validation strategy is followed for evaluation according to the splits are obtained from the UCF101 website.

5.2.2 Experimental Settings

We present action-vector performance with different UAM mixtures ranging from 256 to 2048. It is found empirically that beyond 2048 mixtures, the improvement in classification accuracy is negligible as compared to the enormous increase in the training time for the UAM and factor analysis stages. Further, it was found that action-vector dimension is not critical for classification performance and produces negligible difference. Hence, for all the reported results hereafter, the action-vector dimension is set to be 200 based on empirical validation.

Feature extraction on these datasets in the form of HOG, HOF, and MBH descriptors is done (as part of the iDT framework [118]) without using any human annotations. Though the classification performance of iDT significantly improves when using human bounding boxes [117], annotations are either not available or expensive to acquire for most of the unconstrained videos.

5.2.3 Super action-vector vs. Action-vector

In Table 5.1, the performance of the proposed action-vector is compared with super action-vector (SAV). It can be observed that extracting essential attributes with factor analysis leads to a more discriminative representation that achieves superior classification accuracy. Further, it can be seen that the performance improves slightly for all feature descriptors when the number of UAM mixtures are increased, for the UCF101 dataset. However, the gains become less pronounced as compared to the increase in training time. Hence, it can be concluded that even with a modest number of mixtures (256), reasonably good classification performance can be obtained which shows that this efficacy of the action-vector framework.

Table 5.1: Comparison of super action-vector & action-vector classification performance (%) using cosine scoring for varying UAM mixtures on UCF101 dataset.

Representation	# of UAM mixtures											
	256			512			1024			2048		
	HOG	HOF	MBH	HOG	HOF	MBH	HOG	HOF	MBH	HOG	HOF	MBH
super action-vector	59.89	61.23	62.45	60.46	62.99	63.56	59.11	62.40	64.28	64.22	64.96	65.12
action-vector	86.47	89.50	90.47	86.47	89.44	90.74	86.64	89.67	90.78	87.17	88.80	90.67

In Table 5.2, we evaluate the performance of action-vectors on the HMDB51 dataset. Clearly, action-vectors perform classification better than super action-vectors over all descriptors and UAM configurations. It is interesting to note that the performance of both SAV and action-vector dips as the number of UAM mixtures increase. We hypothesize that the number of distinct attributes in the actions of HMDB51 is fairly low and as a result, training such a large number of mixtures becomes challenging. However, it was found that decreasing the number of mixtures to 128 causes a massive dip in the performance which indicates that at least 256 mixtures are required to capture the attributes of all the actions. It is important to note that action-vectors with MBH features perform consistently better than both HOG and HOF. This shows that MBH expresses the attributes of an action reasonably better than other features which lead to better discrimination.

Table 5.2: Comparison of super action-vector & action-vector classification performance (%) using cosine scoring for varying UAM mixtures on HMDB51 dataset.

Representation	# of UAM mixtures											
	256			512			1024			2048		
	HOG	HOF	MBH	HOG	HOF	MBH	HOG	HOF	MBH	HOG	HOF	MBH
super action-vector	54.21	55.36	55.34	52.15	54.71	55.44	56.98	52.88	53.21	48.68	53.49	55.12
action-vector	74.11	76.67	76.99	73.24	75.62	76.08	71.88	74.18	74.51	68.44	70.72	72.55

5.2.4 Action-vectors vs. other aggregation methods

Finally, the performance of action-vectors is compared with other aggregation methods like Fisher vector and VLAD in Table 5.3. The concatenated action-vectors used for comparison are formed using the action-vectors three different feature descriptors, HOG, HOF, and MBH. The final action-vector is 600-dimensional and harnesses complimentary information of the descriptors to enhance the classification accuracy over the individual action-vectors. Further, it can be observed that unsupervised cosine scoring of action-vectors performs on par with supervised classification methods using convolutional neural network based descriptors combined with Fisher vectors [113, 123, 17]. This shows that the proposed methods of using factor analysis for extracting a aggregate representation is effective at discrimination even without supervision.

Table 5.3: Comparison with other state-of-the-art aggregation methods

Method	Accuracy (in %)	
	UCF101	HMDB51
C3D features + iDT(fisher) [113]	90.4	-
Traj-pooled deep CNN + iDT(fisher) [123]	91.5	65.9
VLAD ³ + iDT(fisher) [73]	92.2	-
iDT-FV + DNN Hybrid [17]	92.4	70.4
Action-vector concatenated (HOG + HOF + MBH) + cosine	91.3	77.1

5.2.5 Analysis on untrimmed videos

We extend the analysis of action-vectors to temporally untrimmed videos present in the THUMOS14 dataset [48]. The videos in THUMOS14 are significantly longer than UCF101 and the actions of interest have significantly less duration as compared to other background activities. The actions of interest are based on the 101 action classes in UCF101. In Figure 5.5, we show the similarity between action-vectors obtained for the same action in two different datasets i.e. UCF101 and THUMOS14. It can be observed that even in untrimmed videos, the similarity of the actions is noticeable while the separation across different actions is maintained. We hypothesize that the reason behind the similarity is that the total variability matrix \mathbf{T} can faithfully reconstruct only the foreground actions that have been learned through the UAM and not arbitrary background movements. This is important as the average length of the clips in THUMOS14 is around 3 minutes, but the duration of relevant actions is only around 4-5 seconds which makes action-vectors an important tool for relevant action spotting.

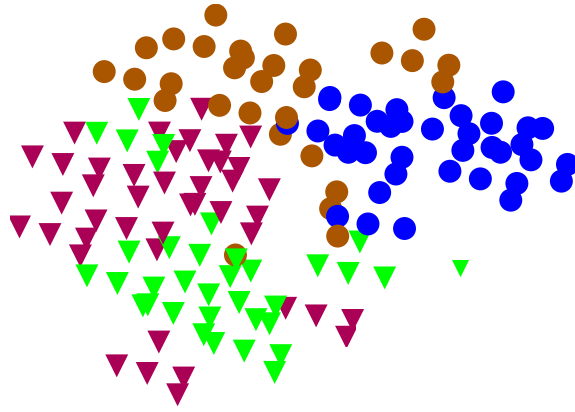


Figure 5.5: t-SNE plot shows similarity in action-vectors of UCF101 and THUMOS14 across two classes - WritingOnBoard (UCF101, \blacktriangledown) (THUMOS14, \blacktriangledown) and WallPushups (UCF101, \bullet) (THUMOS14, \bullet)

In order to understand how action-vectors enhance the mixtures belonging to the attributes of a specific action, an untrimmed clip containing the action *blowing candles* is presented in the form of an entropy plot of UAM posteriors in Figure 5.6. It can be observed that during the action of interest, the entropy value is low which shows that only a few of the UAM mixtures get activated. These mixtures represent the attributes which form a part of the action. Apart from the actions of interest, the video also contains activities which are not in the 101 actions of UCF101 and have not been modeled by the UAM. During such background activities, higher entropy values are observed which shows that an arbitrarily large number of mixture components get activated simultaneously. We expect to see this behavior from any action that is not modeled by the UAM as there will be no particular set of attributes (components) which will be affiliated with the action. It can also be observed that the entropy values change gradually when the clip transitions from the action to the background or vice-versa which is a result of MBH features being extracted over a period of 15 frames.

5.3 Summary

In this chapter, we proposed an compact and discriminative representation of actions called action-vectors. Attributes of all actions were learned in a universal attribute model and for each clip and the essential attributes were extracted from this model. It was shown that this extraction can be performed in an unsupervised manner by using factor analysis to produce low-dimensional action-vectors. Even without supervision, this approach was shown to produce comparable classification accuracy as compared to other aggregation based rep-

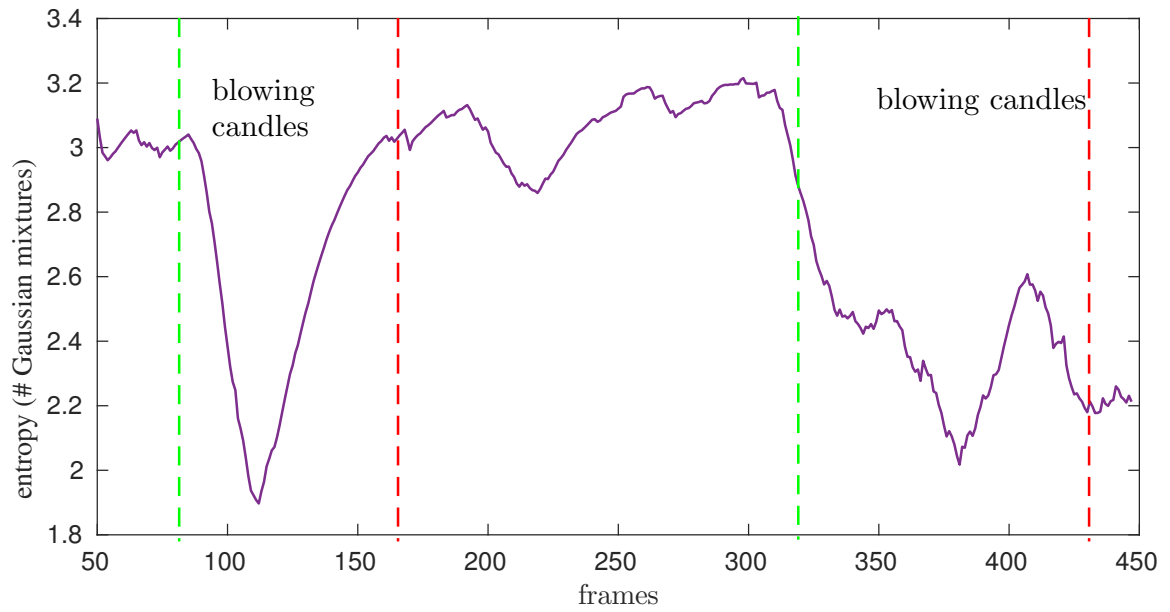


Figure 5.6: Entropy plot for the number of UAM posteriors (using MBH features) for an untrimmed clip of *blowing candles* in THUMOS14.

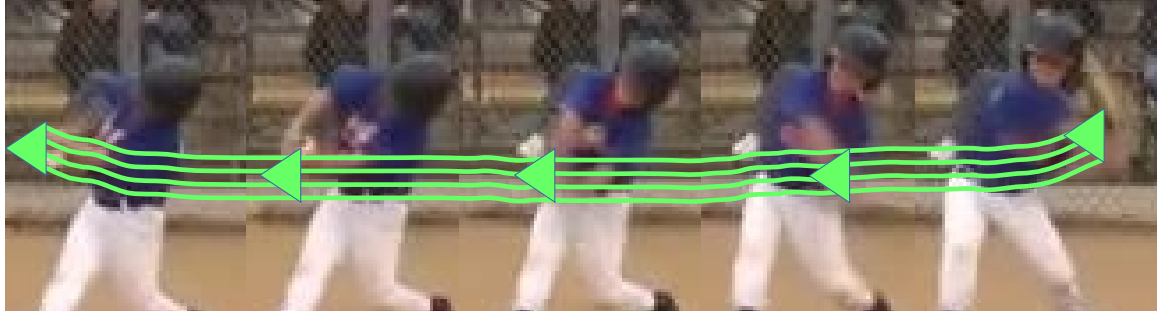
representations like Fisher vectors and VLAD paired with classifiers. Also, we showed that action-vector can spot actions interspersed with large segments of background actions in temporally untrimmed videos.

Chapter 6

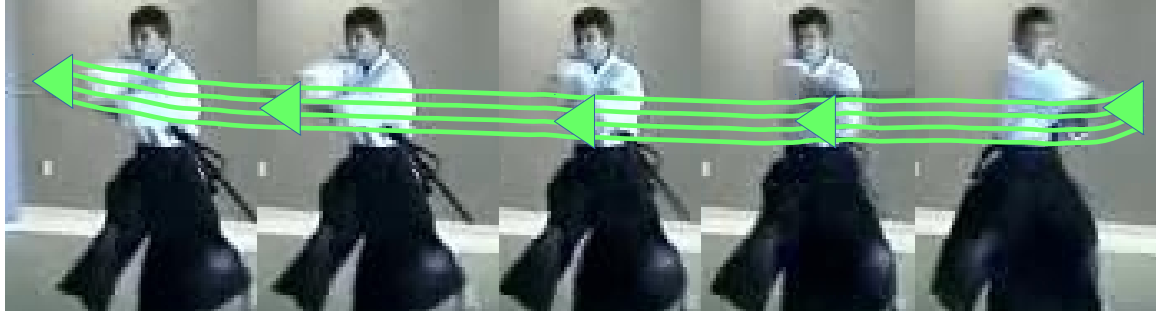
Supervised action recognition using linear and non-linear embedding of action-vectors

In the last chapter, a compact and low-dimensional representation called action-vector was proposed which was formed by extracting the attributes from a universal attribute model (UAM). As the extraction process promotes the choice of relevant attributes for each action, similar actions with analogous attributes result in nearly identical action-vectors. For example, two different actions like *baseball swing* and *sword exercise* appear visually similar based on the attributes i.e. motion trajectories as shown in Figure 6.1. The trajectories show that the limb movements of the actors which resemble closely though the duration and stance of the person performing the action is quite different. So, representation of actions should not only be based on discovering similarity across different instances of the same action but also realizing minute differences across instances of similar actions. Invariably, in such cases, class labels need to be used to discriminate among these actions and this motivates the use of discriminative embedding of attribute based action-vector representation.

Low-dimensional discriminative embedding has been used on action representations like 3D CNN[113] and improved dense trajectory (iDT) [117] to improve classification performance [12, 22]. Though mostly linear embedding methods like linear discriminant analysis (LDA) has been used extensively, non-linear methods like kernel based embedding have also been used for compact representation of human actions [70]. Even graph based embedding of human actions in terms of Kronecker graphs have been explored for explaining the composition of human actions [112]. In this chapter, we discuss linear decomposition techniques like LDA and probabilistic LDA (PLDA) in detail in Section 6.1 and extend the discussion to non-linear decomposition techniques like Siamese networks in



(a) *baseball swing*



(b) *sword exercise*

Figure 6.1: Inter-action similarity shown with motion trajectory. Best viewed in color.

Section 6.2. In Section 6.3, we present experimental results on benchmark action datasets and conclude with a summary in Section 6.4.

6.1 Linear decomposition using LDA and PLDA

Intra-action variabilities are introduced when recording the same action from different camera views which can reveal or hide certain action attributes. In the action-vector space, these variabilities result in action-vectors of the same action to be far apart. Comparing action-vectors using cosine scoring which considers the distance between action-vectors for classification results in misclassification. Hence, we propose to use class information using supervised decomposition techniques like linear discriminant analysis (LDA) to reduce the effect of intra-action variability.

6.1.1 Linear Discriminant Analysis (LDA)

Linear discriminant analysis (LDA), specifically Fisher's LDA assumes that the action-vectors of two actions have means μ_0, μ_1 and covariances Σ_0, Σ_1 . Then, the linear combination of action-vectors yw will have means $y\mu_i$ and variances $y^t\Sigma_i y$ for class $i = 0, 1$, respectively. The separation between these two distributions is the ratio of the between

class variance to the within class variance:

$$\mathbf{S} = \frac{(\mathbf{y}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0))^2}{\mathbf{y}^t(\boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1)\mathbf{y}}. \quad (6.1)$$

The maximum separation is achieved when $\mathbf{y} \propto (\boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1)^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$, i.e., \mathbf{y} is the normal to the discriminating hyperplane.

In case of more than two classes, the aim is to find a subspace which contains all of the class variability. If each of m classes has a mean $\boldsymbol{\mu}_i$ and the same covariance $\boldsymbol{\Sigma}$, then the scatter between class variability is defined as the sample covariance of the class means

$$\boldsymbol{\Sigma}_b = \frac{1}{m} \sum_{i=1}^m (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^t, \quad (6.2)$$

where $\boldsymbol{\mu}$ is the mean of the class means. The class separation in a direction \mathbf{u} in this case will be given by

$$\mathbf{S} = \frac{\mathbf{y}^T \boldsymbol{\Sigma}_b \mathbf{y}}{\mathbf{y}^t \boldsymbol{\Sigma} \mathbf{y}}. \quad (6.3)$$

This means that when \mathbf{y} is an eigenvector of the projection matrix $\mathbf{A} = \boldsymbol{\Sigma}^{-1} \boldsymbol{\Sigma}_b$ and the separation will be equal to the corresponding eigenvalue. If the projection matrix \mathbf{A} is diagonalizable, the variability between features will be contained in the subspace spanned by the eigenvectors corresponding to the $m-1$ largest eigenvalues (as \mathbf{A} is at most of rank $m-1$). Figure 6.2 shows the LDA projected action-vectors onto 100 dimensions (highest available for 101 actions). Clearly, the different actions are better separated into clusters than Figure 5.3(b).

6.1.2 Probabilistic Linear Discriminant Analysis (PLDA)

Even though LDA solves the intra-class variability issue, the number of dimensions available for projection is always limited by the number of classes. In PLDA, there is no dimensionality constraint and it has been shown to produce better results than LDA for tasks like face recognition [90, 11] and object recognition [43]. Hence, we propose to use PLDA based action-vector scoring which is derived with a two-covariance model similar to LDA. The two-covariance model is known as a generative linear-Gaussian model, where latent vectors \mathbf{y} representing actions are assumed to be distributed according to the prior distribution

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y} | \boldsymbol{\mu}, \mathbf{S}_b). \quad (6.4)$$

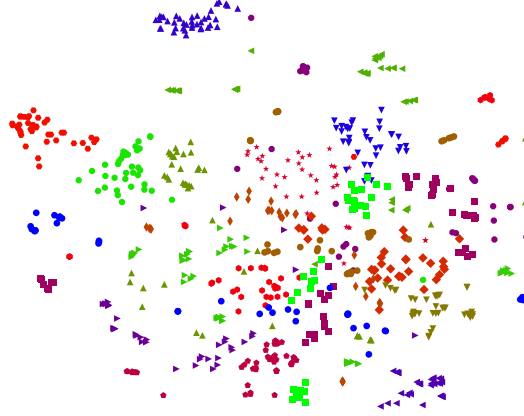


Figure 6.2: t-SNE visualization of LDA projected MBH action-vectors on selected classes of UCF101. Best viewed in color.

For a given action represented by a latent vector \hat{y} , the distribution of action-vector \mathbf{w} is assumed to be

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\hat{y}, \mathbf{S}_w). \quad (6.5)$$

The maximum-likelihood estimates of the model parameters, $\boldsymbol{\mu}$, \mathbf{S}_b , and \mathbf{S}_w , can be obtained using EM algorithm. Now, in case of LDA, the projection of all the vectors using the projection matrix \mathbf{A} would be followed by cosine scoring. In PLDA, the projection matrix \mathbf{A} is not obtained explicitly and scoring is done using for every pair of action-vectors $(\mathbf{w}_1, \mathbf{w}_2)$ using the following two hypotheses:

- *Null hypothesis \mathcal{H}_s* : A pair of action-vectors \mathbf{w}_1 and \mathbf{w}_2 is generated from the distribution $p(\mathbf{w}|\hat{y})$ which involves only a single latent vector \hat{y} representing a particular action.
- *Alternative hypothesis \mathcal{H}_d* : A pair of action-vector \mathbf{w}_1 or \mathbf{w}_2 is generated using two latent vectors representing two different actions that are independently generated from $p(\mathbf{y})$.

The score can now be calculated as a log-likelihood ratio between the two hypotheses \mathcal{H}_s and \mathcal{H}_d as

$$k(\mathbf{w}_1, \mathbf{w}_2) = \log \frac{p(\mathbf{w}_1, \mathbf{w}_2 | \mathcal{H}_s)}{p(\mathbf{w}_1, \mathbf{w}_2 | \mathcal{H}_d)} \quad (6.6)$$

$$= \log \frac{\int p(\mathbf{w}_1 | \mathbf{y}) p(\mathbf{w}_2 | \mathbf{y})}{p(\mathbf{w}_1) p(\mathbf{w}_2)}, \quad (6.7)$$

where in the numerator, we integrate over the distribution of action-vectors to determine the likelihood of producing both action-vectors from the same latent action model. For the hypothesis that action-vectors belong to separate actions, the product of the marginal likelihoods $p(\mathbf{w}_1)$ and $p(\mathbf{w}_2)$ is used.

6.2 Non-linear decomposition using Siamese networks

We have discussed two important linear embedding methods, LDA and PLDA in the previous section. However, linear embedding may not lead to the discovery of higher order non-linear relationships across feature dimensions. To extract such dependencies, non-linear hierarchical structures like deep Siamese networks are useful. Especially, Siamese neural networks have been successfully used for verification tasks like person re-identification [13], object recognition [128], and gesture recognition [5], where there is a need to find correspondence between two items among thousand different items. Further, it shows promise where less number of examples are available as in one-shot recognition of image categories [23, 55]. More recently, in [127], actions were recognized using Siamese networks. Each action was divided into *precondition* (*cause*) and *effect* parts which are fed as input to separate CNN stream. For the *precondition* part, a 4096-D feature vector is obtained for each frame and average pooled to obtain a single 512-D feature representation for the entire part. Then, this representation is transformed for each action separately, and the output is compared to the 512-D output of the *effect* part. This approach requires the division of an action video into *precondition* and *effect* which is not only subjective but requires laborious manual annotation. Instead, we propose to use a complete representation of the entire action i.e., action-vector as input to a Siamese deep neural network which is trained using contrastive loss for discriminating actions. Figure 6.3 presents the training of Siamese networks after extracting action-vectors which modifies Figure 5.1 from the previous chapter.

As Siamese networks are trained to differentiate between inputs rather than classify inputs, the low-dimensional representation of the video is used for differentiating between videos of different classes. The Siamese network consists of two identical sister neural networks with the same weights. The sister networks are given the action-vector represen-

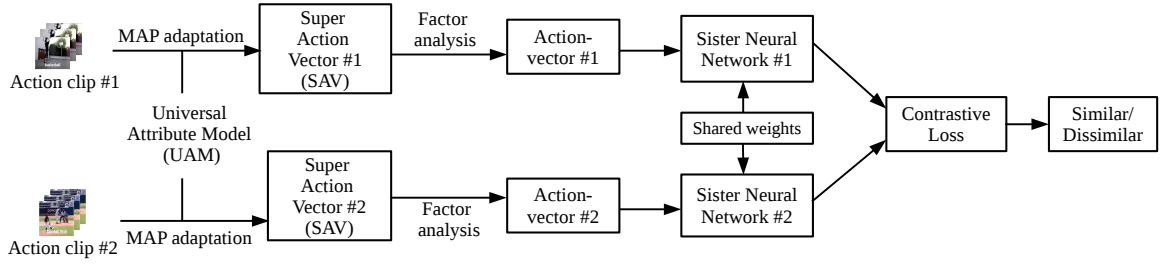


Figure 6.3: Siamese networks for non-linear decomposition of action-vectors

tations of different clips as their input. The last layer of each network then culminates in a contrastive loss function which evaluates the similarity between the two videos.

The Siamese network is optimized using a contrastive loss computed as

$$(1 - y) \frac{1}{2} e_w^2 + (y) \frac{1}{2} \{ \max(0, a - e_w) \}^2, \quad (6.8)$$

where $e_w = \|\mathbf{g}_w(\mathbf{w}_1) - \mathbf{g}_w(\mathbf{w}_2)\|_2$ is the Euclidean distance between the two outputs of the sister networks and is a measure of the semantic similarity between the inputs. The term \mathbf{g}_w is the output of either one of the sister networks and \mathbf{w}_1 and \mathbf{w}_2 are the inputs. The label y is either 1 if the inputs are from the same class and 0 otherwise and the margin a is set to be greater than 0 so that dissimilar pairs beyond this margin do not contribute to the loss. This ensures that the network is optimized for dissimilar pairs that the networks consider as fairly similar. For training the network, the contrastive loss value is calculated using both the inputs, and then back propagated to both the networks.

In Figure 6.4, we present a case where Siamese networks are able to differentiate between two visually similar actions *baseball swing* and *sword exercise* (as shown in Figure 6.1). It can be observed that action-vectors (200 dimensional) for both actions are scattered. On the other hand, the Siamese network 1st hidden layer output (100 dimensions) is clearly separable for these actions.

6.2.1 Siamese network configuration

To obtain the best possible Siamese network configuration, three different sister neural network are used- NN1: 200 – 100R, NN2: 200 – 100R – 50R, and NN3: 200 – 100R – 100R where R represents ReLU activation. The input action-vector is calculated using a UAM with 256 mixtures trained on the training split of HMDB51. The NN3 configuration achieves the best results among the three overall feature descriptors as shown in Table 6.1 and it is used for reporting the results in subsequent experiments. We use a dropout of 0.1 between the layers except for the output layer, and the learning rate was set to 0.0001 with

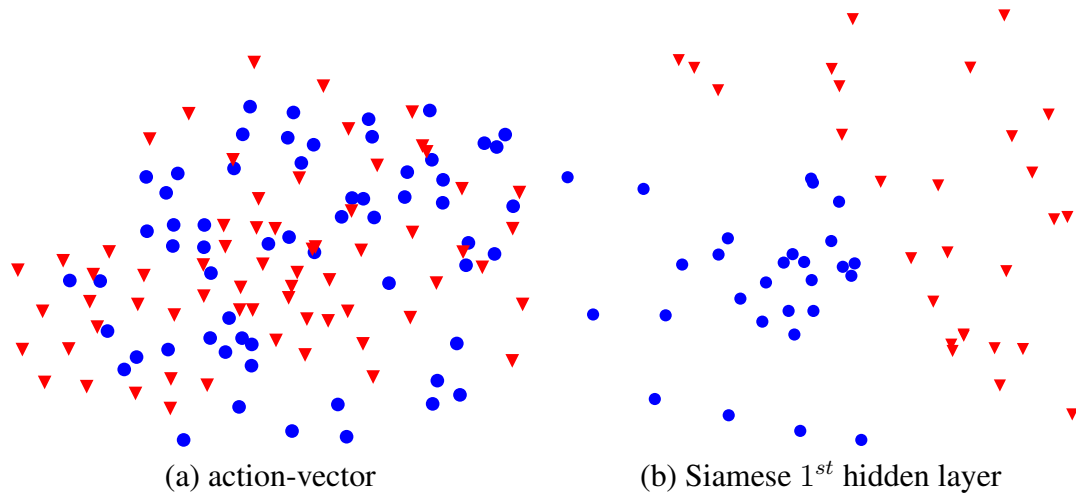


Figure 6.4: t-SNE (stochastic neighbour embedding) plots for similar actions (see Figure 6.1) *baseball swing* (\blacktriangledown) and *sword exercise* (\bullet). Best viewed in color.

RMSprop used as an optimizer. It is found empirically that increasing the number of layers or neurons caused significant deterioration of classification performance. As the number of training instances per action in the HMDB51 dataset is only around 70, it is observed that deeper networks with more number of hidden layers cannot be trained effectively because of over-fitting.

Table 6.1: Performance (in %) of Siamese network with different configurations on HMDB51 dataset

Configuration	HOG	HOF	MBH
NN1	42.1	45.9	49.6
NN2	57.8	59.6	62.4
NN3	72.1	78.5	83.1

6.3 Experimental results

In this section, we present the performance of various non-linear and linear embedding methods. Specifically, intermediate fusion of action-vectors, the various configurations of Siamese networks, and comparison with state-of-the-art action recognition techniques are discussed.

6.3.1 Datasets

The embedding methods discussed in this chapter are evaluated on the following datasets - UCF101, HMDB51, MPII cooking activities, and THUMOS14.

MPII Cooking Activities

The MPII cooking activities dataset [94] contains 5,609 videos of 65 fine-grained cooking activities with low inter-class variability. Each clip is 5 seconds long and recorded at 30 frames per second and all the activities are performed by 12 different subjects.

6.3.2 Experimental settings

The HOG, HOF, and MBH features are extracted using the same spatio-temporal volumes used in improved dense trajectory (iDT)[117]. It is found that changing the feature descriptor and the number of mixtures produces a significant change in the classification performance. However, any change in the action-vector dimension does not result in any significant difference in accuracy, and the value of 200 was chosen as the dimension as it performed marginally better empirically. For LDA, a maximum of 100 dimensions was used for 101 actions in UCF101, 50 dimensions were chosen for 51 actions of HMDB51, and 64 dimensions were chosen for 65 actions in MPII Cooking.

6.3.3 Performance of linear embedding techniques

The two linear embedding techniques - LDA and PLDA are compared on two benchmark datasets, UCF101 and HMDB51. In Table 6.2, the classification performance of both LDA and PLDA is presented with different feature descriptors and varying number of UAM mixtures. It can be observed that the performance of both LDA and PLDA rises steadily with the number of UAM mixtures which shows that more distinctive attributes can be represented using a larger model. Further, it can be seen that PLDA performs better than LDA consistently across descriptors and number of UAM mixtures. This shows that using latent models for each action class and describing action-vectors using the same is more effective than finding the optimal linear discriminant surfaces for projection.

Table 6.2: Classification accuracy (%) for linear embedding techniques on UCF101.

Embedding	# of UAM mixtures											
	256			512			1024			2048		
	HOG	HOF	MBH	HOG	HOF	MBH	HOG	HOF	MBH	HOG	HOF	MBH
LDA	88.17	90.50	79.82	88.2	90.30	91.90	88.40	90.57	92.10	87.24	90.10	92.21
PLDA	88.44	90.90	73.35	88.6	91.07	92.55	88.60	91.24	92.68	88.84	90.74	92.99

In Table 6.3, the analysis of LDA and PLDA over action-vectors is presented on HMDB51 dataset. As seen with UCF101, the performance of PLDA is also superior to LDA across feature descriptors. Similarly, MBH again leads other feature descriptors consistently over

different UAM configurations. This shows that MBH expresses the action attributes better than other features descriptors resulting in better classification. The only trend which is different from UCF101 is that the performance worsens as the number of mixtures are increased. This shows that there are significantly less distinct attributes in HMDB51 that need to be expressed through the UAM. So, as the number of mixtures are increased, there are not enough distinct features which can be used to train these extra mixtures. As these mixtures become overwhelmingly large, they become arbitrarily associated with different actions and cause poor classification performance.

Table 6.3: Classification accuracy (%) for various scoring mechanisms on HMDB51. UAM is trained on HMDB51 training set.

Embedding	# of UAM mixtures											
	256			512			1024			2048		
	HOG	HOF	MBH	HOG	HOF	MBH	HOG	HOF	MBH	HOG	HOF	MBH
LDA	74.55	77.32	78.56	73.98	75.95	77.52	70.54	73.66	74.77	65.10	67.91	67.32
PLDA	76.24	78.17	79.54	74.21	76.67	78.69	71.55	73.53	74.38	63.21	66.14	65.56

6.3.4 Intermediate fusion techniques using PLDA

So far, we have seen the performance of the action-vectors on individual descriptors. In literature, HOG, HOF, and MBH feature descriptors have been shown to extract complementary information and they are combined in various ways to improve action recognition performance [117]. Hence, we also explore different fusion techniques like: (a) intermediate fusion (IF) of action-vectors scored with SVM classifier, (b) intermediate latent dimension fusion (ILDF) using PLDA, and (c) score fusion (SF) using PLDA scores. For the experimental results demonstrated here, the action-vectors considered for each of the individual descriptors are 200-dimensional. In Table 6.4, we present the results using these fusion techniques on action-vectors. To perform intermediate fusion (IF), the action-vectors of different features for the same clip are concatenated and classified using an SVM. In case of IDLF, a PLDA model is trained on the concatenated action-vectors of the training set and then the concatenated action-vectors of the test clips are classified using this model. For score fusion (SF), a convex combination of PLDA scores is used which is optimized for better performance. In all cases, the action-vectors and PLDA classification scores are obtained on 2048 mixture UAMs trained separately for each feature. It can be observed that an action-dependant latent projection technique like PLDA performs better than concatenation of action-vectors (intermediate fusion scored with SVM) or score fusion.

Table 6.4: Comparison of intermediate fusion and SVM (IF + SVM), PLDA-based intermediate latent decomposition fusion (ILDF), and score fusion (SF) on UCF101. The underlying UAM model has 2048 mixtures.

Feature combination	Accuracy (in %)		
	IF + SVM	PLDA-based ILDF	SF
HOG + HOF	90.45	91.23	92.99
HOG + MBH	93.15	93.74	93.92
HOF + MBH	93.56	94.10	93.96
HOG + HOF + MBH	94.21	95.13	93.98

6.3.5 Linear embedding of action-vectors vs. state-of-the-art

Table 6.5 compares the classification performance of the proposed method with the state-of-the-art techniques used for action recognition on the UCF101, HMDB51, and THUMOS14 datasets. The action-vectors for untrimmed videos in THUMOS14 are extracted using the best performing UAM for UCF101 (2048 mixtures for all features) and corresponding \vec{T} -matrix on the UCF101 dataset. This is done in accordance to the THUMOS14 challenge where the UCF101 dataset is used as the training set for the THUMOS14 test clips [48].

Some state-of-the-art techniques listed in Table 6.5 like [147, 63, 114, 104] use long-term features to represent the entire video with a single feature. Other techniques use aggregation models like bag-of-words (BoW) [133], Fisher vector [44, 113, 123, 17] or VLAD [73]. Notably, many of the methods augment CNN features with iDT features to attain the high classification accuracy [114, 123]. Among the methods that use long-term features, temporal linear embedding (TLE) networks [19] perform the best on UCF101. The proposed action-vectors produces comparable performance to TLE on UCF101 and comfortably outperforms it by 10% on the slightly more challenging HMDB51 (according to [58]). We hypothesize that action-vectors capture better long-term temporal dependency than TLE. As the actions HMDB51 are more closer to each other than UCF101 in terms of motion patterns, long-term dependencies are more useful for classification which leads to better performance of action-vectors than TLE on HMDB51.

For the untrimmed THUMOS14 dataset in particular, the state-of-the-art approaches [44, 122, 44] use fixed-sized windows to process untrimmed videos. Fixed-sized windows cannot handle high variations in the duration of actions. To mitigate this issue, fixed-sized windows of different temporal resolutions are extracted from the same clip. The final classification result is based on the aggregation of the classification outputs of each of the windows [123]. Such approaches add to computational overhead as the same video is processed multiple times at different time scales whereas in the proposed approach, we

Table 6.5: Comparison of classification accuracy of proposed approach with existing state-of-the-art methods

Method	Accuracy (in %)		
	UCF101	HMDB51	THUMOS14
Spatio-Temporal CNN [147]	93.0	68.2	-
Multi-skip feature [63]	89.1	63.9	-
Long-term CNNs +iDT [114]	92.7	67.2	-
Two-stream CNN [104]	88.0	59.4	-
Temporal segment networks (TSN) [124]	94.3	69.4	-
HOF + MBH + Event model + BoW [133]	-	49.86	-
Objects + motion [44]	-	-	71.6
C3D features + iDT(fisher) [113]	90.4	-	-
Traj-pooled deep CNN + iDT(fisher) [123]	91.5	65.9	-
VLAD ³ + iDT(fisher) [73]	92.2	-	-
iDT-FV + DNN Hybrid [17]	92.4	70.4	-
iDT+CNN [122]	-	-	62.0
iDT+FV [120]	-	-	63.1
Temporal linear embedding (TLE) [19]	95.6	71.1	-
Action-vector ILDF (HOG + HOF + MBH) + PLDA	95.1	81.1	71.9
Action-vector IF (HOG + HOF + MBH) + SVM	94.3	81.0	70.9

compute a single action-vector for the entire video which outperforms these methods.

6.3.6 Comparison of linear and non-linear embedding techniques

Next, we compare the different linear and non-linear embedding techniques which we have discussed in this chapter. In Table 6.3, such a comparison is presented on the HMDB51 dataset. For non-linear embedding methods, kernel discriminant analysis (kDA) is also introduced for comparison. It can be observed that Siamese networks based embedding of action-vectors performs better than other linear and non-linear embedding techniques. This is also true across the different feature vectors and different UAMs with a varying number of mixtures.

Table 6.6: Performance comparison (%) of various discriminative embedding techniques on HMDB51

Embedding Technique	# UAM mixtures					
	256			512		
	HOG	HOF	MBH	HOG	HOF	MBH
LDA	74.55	77.32	78.56	73.98	75.95	77.52
PLDA	76.24	78.17	79.54	74.21	76.67	78.69
kDA	61.45	65.16	66.12	69.51	70.53	72.75
Siamese (NN3)	72.14	78.53	83.10	76.41	79.35	83.04

Apart from traditional action recognition datasets, we test the performance of action-vectors especially with discriminative embedding on fine grained actions. For evaluation, the MPII cooking activities dataset [95] is used which consists of 65 cooking actions like *cutting*, *peeling*, *cutting slices*, and *cutting dice* that have very low inter-class variability. Even on this challenging dataset, both linear and non-linear embedding of action-vectors performs well as can be seen in Table 6.7. Particularly, Siamese networks and LDA are very close in terms of classification performance and beat the other decomposition techniques.

Table 6.7: Classification accuracy (%) for various discriminative embedding techniques on MPII Cooking

Embedding Technique	# UAM mixtures					
	256			512		
	HOG	HOF	MBH	HOG	HOF	MBH
LDA	74.14	77.48	78.48	76.45	79.48	78.61
PLDA	65.34	67.56	66.85	69.12	67.35	67.63
kDA	63.45	64.16	65.21	64.51	65.53	67.75
Siamese (NN3)	75.48	76.95	78.46	77.81	79.24	80.27

6.3.7 Non-linear embedding of action-vectors vs. state-of-the-art

In Table 6.8, we present the performance of Siamese networks with state-of-the-art techniques on the HMDB51 dataset. Temporal segment networks (TSN) [124], max pooled deep features (CNN +C3D) and iDT features [22], two-stage temporal segment networks [65], and iDT features with deep neural networks (DNN) [17] all provide a single feature representation for a video by utilizing spatial, temporal and spatio-temporal CNNs, and further augment it with iDT features. It can be observed that discriminative embedding of action-vectors using a single feature descriptor MBH which is part of the iDT features is more effective in action recognition than these methods. This shows that factor analysis seems a better choice for discriminative action representation and following it with contrastive loss training aids in classification even further.

In Table 6.9, a comparison with state-of-the-art approaches is presented for MPII cooking activities dataset. Approaches like interaction part mining [144], localized semantic features [145], etc. concentrate on object detection, and its manipulation to recognize the actions. Instead in [12], frame-based CNN features from a video are embedded on Grassmannian manifold. A support vector regressor is learned on the embedded features augmented with iDT features for classification. It can be observed that the proposed Siamese network based embedding scheme performs better than all these embedding and object

interaction based approaches. We hypothesize that Siamese networks are able to better discriminate actions as the comparison is performed on the deep features extracted by the sister networks. As neural networks are universal approximators, they can span the action-vector space better than linear approaches. Further, action-vectors with non-linear discriminative embedding can easily discriminate between action classes with both low and high variability which shows the versatility of the representation.

Table 6.8: Comparison of Siamese network based embedding of action-vectors with state-of-the-art on HMDB51

Method	Accuracy (in %)
3DCNN features + Siamese [127]	63.4
Traj-pooled deep CNN + iDT(fisher) [123]	65.9
Long-term CNNs +iDT [114]	67.2
Temporal segment networks (TSN) [124]	69.4
iDT-FV + DNN Hybrid [17]	70.4
Temporal linear embedding (TLE) [19]	71.1
Max-pool (CNN+C3D) + iDT[22]	73.1
Deep local video feature [65]	75.0
Action-vector (MBH) + Siamese	83.1

Table 6.9: Comparison of Siamese network based embedding of action-vectors with state-of-the-art on MPII Cooking Activities

Method	Accuracy (in %)
CoHOG +iDT [50]	46.6
Localized Semantic Features [145]	70.5
VideoDarwin [27]	72.0
Interaction Part Mining [144]	72.4
GRP-CNN + iDT(Fisher) [12]	75.7
Action-vector (MBH) + Siamese	80.3

6.4 Summary

In this chapter, we explored discriminative embedding of low-dimensional action representations to distinguish between visually similar actions both at coarse and fine level. A comparison of linear and non-linear embedding methods was presented on various benchmark datasets where actions had both low and high inter-class variability. With PLDA based embedding and fusion of action-vectors across different features, state-of-the-art performance

was demonstrated across different datasets, both trimmed and untrimmed. We extended the idea of discriminative embedding with deep non-linear embedding using with Siamese networks and found that even fine grained actions with similar attributes were distinguishable. A comparison with state-of-the-art approaches revealed that such a non-linear embedding comfortably outperforms state-of-the-art approaches.

Chapter 7

Action recognition in surveillance videos

In the last chapter, the efficacy of action-vectors was demonstrated on both trimmed and untrimmed datasets. On temporally untrimmed videos, action-vectors were able to recognize the foreground actions amid a lot of background activities. So, it is a natural fit for abnormal events like snatch thefts which have an extremely low probability of occurrence in long surveillance footage. Manually detecting these rare events or anomalies is challenging in cities as there are hundreds of cameras which need to be monitored. Further, anomalies have localized spatio-temporal signatures where they occur over a small time window in a long sequence or a small spatial region in a wide surveillance area. Another distinguishing feature of actions such as theft is that outside this anomalous spatio-temporal region, no irregularities are present and only regular activities are observed. As anomalous activities like chain and purse snatching are prevalent in many countries, addressing this problem is of immediate concern.

Most of the existing literature on detecting anomalous activities like snatching uses datasets collected in controlled laboratory settings with no crowd or background and with an excellent viewing angle of the activity. Even when conducted in crowded scenes as in [6], the entire pickpocket incident is staged with *a priori* knowledge of how the incident is going to take place which makes analysis a lot easier. So, to analyze real-life thefts, we present a dataset called Snatch 1.0¹ collected from unconstrained surveillance footage. This dataset contains surveillance footage obtained from the traffic police department of the city of Hyderabad in India which includes various instances of snatch thefts (details in Section 7.4.1). It was observed that snatching incidents in surveillance videos can occur in a variety of *scenarios* which are of diverse types and lead to different victim *reactions*. Some of the examples of snatch thefts are shown and described in Figure 7.1.

An ontology of snatch thefts encountered in surveillance videos is presented in Table

¹<http://www.iith.ac.in/vilg/datasets/>




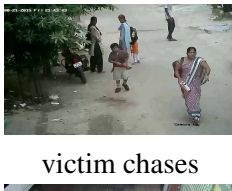
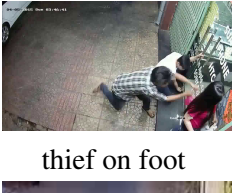

	Scenarios			Reactions
(a)	 thief on motorbike	 pillion rider cannot snatch cleanly	 victim gets dragged	 victim gets dragged
(b)	 thief on foot and inquires	 snatches chain	 runs away	 victim chases
(c)	 thief on motorbike	 pillion rider snatches chain	 rides away	 victim keeps standing
(d)	 thief on motorbike	 pillion rider snatches chain	 rides away	 victim chases
(e)	 thief on foot	 tries to snatch from behind	 victim is dragged	 victim is dragged
(f)	 thief on foot	 snatches chain	 runs away	 victim chases

Figure 7.1: Different snatching scenarios as captured in Snatch 1.0. Best viewed in colour.

Table 7.1: Ontology of snatching scenarios, types and victim reactions

	Snatch Type 1 (Direct grab)	Snatch Type 2 (Inquire and grab)
Scenario 1 (Thief on motorbike)	Reaction 1 (Victim chases)	Reaction 1 (Victim chases)
	Reaction 2 (Victim dragged)	
	Reaction 3 (Victim falls or remains standing)	Reaction 3 (Victim remains standing)
Scenario 2 (Thief on foot)	Reaction 1 (Victim chases)	Reaction 1 (Victim chases)
	Reaction 3 (Victim falls or remains standing)	Reaction 3 (Victim falls or remains standing)

7.1 based on our analysis of the examples depicted in Figure 7.1, . It is evident that snatch thefts are not only complex to model but also the definition of a snatch theft itself is non-trivial. Each interaction between individuals needs to be studied to decide whether or not it is a potential snatch theft or not. These interactions can be considered as part of the larger set of human actions [58]. Hence, we propose a framework for analyzing these interactions. At first, an unsupervised Gaussian mixture model called universal attribute model (UAM) is trained using a variety of human actions containing attributes like *punching* in HMDB51 and UCF101 datasets which is visually similar to *grabbing* in snatch thefts as shown in Figure 7.2. Gaussian mixture models have previously been explored to model attributes of actions [8, 130]. Using factor analysis, the essential attributes useful for describing snatch thefts are extracted and represented in the form of action-vectors. We show that action-vectors perform better than existing state-of-the-art feature descriptors while leveraging a lot of existing video data containing human actions to effectively represent snatch thefts. In addition, we also show that prior modelling based methods like rule-based systems are also not as effective as representation based methods like action-vectors.

The rest of the chapter is as organized as follows. Section 7.1 discusses the related literature for anomaly detection in surveillance videos with a focus on suspicious activities. In Section 7.2, a baseline rule-based system for snatch theft detection is presented for comparison with the action-vector representation. Experimental results on the curated dataset Snatch 1.0 is presented in Section 7.4 and finally the summary is presented in Section 7.5.



(a) *grab*



(b) *punch*

Figure 7.2: Visual similarity in (a) *grabbing* of Snatch 1.0 and (b) *punching* of HMDB51 dataset.

7.1 Related work in surveillance video analysis

A majority of existing literature in the field of anomaly detection is aimed towards detection of generalized abnormal patterns [142, 100] or behaviour in case of individuals or crowds [107]. As the notion of anomaly is subjective, there have been diverse research efforts based on the availability of labelled data for either normal or anomalous event, the frequency of anomalous occurrence, and the nature of the anomaly (general or specific).

As a lot of labelled data for normal behaviour is available, many approaches model normal behaviour extensively and consider events which do not follow these models as anomalous activities. One such approach was proposed for crowded scenarios in [30] by considering three key characteristics: 1) distance between objects, 2) velocity between objects, and 3) area of the objects. The average velocity was monitored and analyzed for any sudden change in speed and direction. A related work [69] used the motion direction in addition to the features as mentioned above, to propose a motion-influence map which localizes both global and local unusual activities. The motion-influence map considers the speed and direction of various objects to determine their relative influence on other objects for detecting unusual motion patterns. In [14], apart from magnitude and direction of motion, entropy information was further added to form a combined histogram of flow orientation, motion, and entropy (HOFME) descriptor. The usage of entropy was to determine the density of motion during normal events. Using the same motion descriptors, sparse representation has also been used in [82] for identifying abnormal activities. Their method proposed that abnormal motion patterns cannot be sparsely represented using a dictionary trained on normal motion patterns.

While all the works mentioned above address generic anomalous activities, there are cases for example when one or more vehicles may slow down while approaching a human

to avoid a collision and later increase their speeds which would then be incorrectly labelled as an anomaly. On a related note, in [41], it was shown that optical flow of motion is consistent during a regular activity or interaction and is disrupted when snatching occurs. However, their work considers examples where there are no other anomalies and even unconditional changes in optical flow which are frequent in vehicular traffic, can be mistaken for a snatching incident.

Apart from estimating better feature descriptors, few works have also incorporated spatial, social, interaction, and sequence contexts for improving anomalous activity detection. In [68], a typical roadside surveillance scene was divided into different zones like traffic lanes, stationary areas, etc. each of which was termed as a scene context and the direction and flow of persons in each of these contexts were measured. Further, to understand normal group behaviour among each scene context, the interaction between any two individuals in the close spatial vicinity was measured for gaze and motion direction information. This was termed as social context and revealed normal behaviour in different scene contexts. In [131], contextual information was used to model the scene in addition to low-level motion information. The contextual information was modelled in the form of prior knowledge as to the type of interactions between actors and the different possible human poses in the scene. Another attempt to model snatching using natural language with the help of clauses was made in [45]. Snatching was defined as a sequence of the thief approaching thief (following or confronting) and the thief snatching the belonging (attack). The attack was detected through primitive actions such as running, walking, and turning. Then a Markov logic network was learned using the rules of activity and probability of actions. These approaches assume that a) the testing scenarios will be same as training scenes regarding pose and actions, b) all the events can be identified, and c) enough labelled data is available to learn the sequences. In real-world scenarios, snatching events rarely happen as compared to regular interactions which makes it difficult to obtain labelled examples. Also, these events occur in crowded scenarios where occlusion makes action detection challenging, and a lot of variations can be encountered in snatching incidents. The dataset presented in this work contains real-world videos with high occlusion and low-resolution making it difficult to employ pose estimation techniques on the detected humans.

Given the works presented above and also based on the categorization provided in [107], a crime like snatching can be characterized as having the following characteristics: 1) snatching activity occurs much more infrequently than regular interactions, 2) many actions which contribute to snatching do not have significantly different characteristics from normal activities like approach, running, etc. Further, the infrequency of snatching events mean that the most of the completely supervised methods where both the training and

testing set should contain large labelled data of abnormal patterns cannot be employed. Similarly, the closeness to normal activities makes the use of completely unsupervised methods unsuitable as in these methods, the normal behaviour is learned, and deviations which are "far-away" are termed anomalous [82]. This motivates the use of representations like action-vectors which can enhance the dissimilarity between snatch thefts and regular actions.

7.2 Baseline: Rule-based snatch theft monitoring

In order to establish the performance of action-vectors which is representation based method, a baseline rule-based system is devised. A rule based method is based on the application of external rules for identifying both normal and anomalous classes and typically include the specification of a set of rules for the feature set and critical thresholds for each of the features being assessed. The system comprises of three stages which are presented below.

7.2.1 Stage I: Detection and tracking of humans in crowded scenarios

Tracking algorithms [35, 83] require the first frame containing the human to be annotated which is not possible in an incoming surveillance stream. Hence, a state-of-the-art detector like YOLO-You Only Look Once [93] is used. We use the detection outputs to initialize a kernelized correlation filter (KCF) based tracker [35]. However, as snatching is a sudden activity and as most of the trackers are designed for detecting smooth motion, KCF cannot cope with the sudden movements. In such a case, YOLO is used for verification of the tracked human and to re-initialize the tracking process. We use a track maintenance system similar to [135] which utilizes both detection results from YOLO and tracking results from KCF. Each *track* is a structure representing a person in the video and consists of 1) *id*: the integer ID of the track, 2) *bbox_list*: list of bounding boxes in every frame where the person is detected, 3) *firstFrame* : first frame where person was detected/tracked, and 4) *lastFrame* : last frame where person was detected/tracked. These tracks are used for interaction identification and processing in the next stage.

7.2.2 Stage II: Victim and thief identification

In this stage, the identity of the potential thief and the victim are uncovered among the thousands of individuals detected. For such identification, the nature of every interaction based on the scenarios mentioned in Table 7.1 needs to be recognized. To identify all the interactions in a surveillance video, every pair of detected humans is checked if they appear

in the same set of frames using the *firstFrame* and *lastFrame* information from their respective *track* structures.

Once some common frames are discovered between a pair of people i and j , these set of frames constitute a meeting. As frame-based detection during the meeting may not always be robust because of either low-resolution or occlusion, every meeting is studied for few frames before and after the meeting for 1) either person i or j moves at a higher speed than the other and/or 2) directions of i and j are same. The first condition checks whether one of the persons darts towards the victim (Snatch Type 2 in Scenario 1, and Snatch Type 1 & 2 in Scenario 2 of Table 7.1). For discovering sudden movement, the average speed before and during the meeting are compared. In the case that either person i or j are detected as slowing down before the meeting and moving suddenly during the meeting then it is highly possible that a snatch theft has taken place and hence, *possibleSnatching* flag is set. The second condition checks whether the victim is being dragged during the meeting (Reaction 3 of Table 7.1). A minimum angle threshold is decided empirically to measure the similarity in the direction of persons i and j . If the direction is found to be same, it can mean that either the victim is chasing or is being dragged. In such a case, *possibleDragging* flag is set to be true which is used later in the identification of snatch thefts.

Finally, if *possibleSnatching* is set as a result of any of the behaviours detected above, a speed comparison of the persons before the meeting is performed to provide the possible victim and thief identity. Here, we assume that thief on a motorbike has higher speed than the victim before the meeting (Scenario 1 of Table 7.1). We additionally use the YOLO tracker [93] to detect motorbike in the vicinity of the meeting to claim thief identity with more confidence.

7.2.3 Stage III: Snatch Theft Verification

After obtaining the identity of the potential thief and victim in the previous stage, the authenticity of the snatch theft is verified in this stage. We exploit the key characteristics of different snatch thefts presented in Table 7.1. If both thief and victim are found to be rushing in the same direction after the meeting with a gap of a few frames, then a snatch theft can be confirmed where the victim is chasing the thief (Reaction 1 of Table 7.1). The average speeds during and after the meeting are compared to check for rapid movement. If either the *possibleDragging* (Reaction 2 of Table 7.1) is set in the previous stage, and the thief is found to be racing after the meeting, then also snatching can be confirmed. Finally, if *possibleSnatching* is set, and the thief rushes after the meeting, snatching is confirmed with victim standing passively or falling (Reaction 3 of Table 7.1). If neither

of these reactions is noted like the victim continues to move in a direction opposite to the thief, then it is mostly a false alarm.

7.3 Unsupervised modeling of snatch theft actions

The process of tracking and subsequent behaviour analysis as mentioned in the baseline system depends largely on critical thresholds for parameters which may sometimes be difficult to obtain for diverse scenarios. Instead, we propose to learn a representation for snatch thefts using its similarities and differences from other human actions. As we have seen that snatch thefts are very diverse, capturing these variations using a Gaussian mixture model (GMM) requires a large number of mixtures. Unfortunately, a single snatch theft video does not have enough data points to estimate the parameters of such a GMM. Hence, we train a universal GMM using the clips of existing human actions datasets like UCF101 and HMDB51 containing 101 and 51 actions, respectively. This GMM is called a universal attribute model (UAM) which implicitly models action attributes. Many of these *attributes* from other actions like punching can be used to describe snatch thefts as shown in Figure 7.2.

The UAM parameters need to be adapted for each action clip like snatch theft to enhance the contribution of the attributes present in it. This adaptation requires updating the parameters of each of the mixture components in the UAM. The MAP adapted parameters of such a clip-specific model can be obtained as a convex combination of the UAM and the clip-specific statistics. The covariance is not modified as there is not enough data in one clip to update the entire covariance matrix of the UAM. The adapted means for each mixture are then concatenated to compute a super action-vector (SAV) for each clip.

Obtaining a fixed-dimensional representation like the SAV, normalizes the effect of varying length clips but results in a high-dimensional representation owing to the large number of mixtures in the UAM. However, the SAV for a clip contains many of the attributes that do not contribute to the action, in this case, snatch thefts. Hence, the SAV is intrinsically low-dimensional, and by using a suitable decomposition, we can extract such a representation which we refer to as an action-vector.

To arrive at a low-dimensional representation, the super action-vector is decomposed as a low-dimensional action-vector whose prior distribution is assumed to be a standard Gaussian. We refer to this random vector as an *action-vector* which is a hidden variable. For a particular clip, the Gaussian mixture(s) which model the attributes in snatch thefts produce high posterior probability. The original super action-vector can then be projected onto a low-dimensional action-vector using only the eigenvectors representing the Gaussian

mixtures contributing to the snatch thefts.

7.4 Experimental results

In the section, we compare action-vector based representation with state-of-the-art descriptors designed for anomaly detection.

7.4.1 Snatch 1.0 : Dataset Description

The videos obtained from the surveillance cameras are low-resolution in most surveillance setups. In cases of wide area surveillance, events of interest can occur further away from the camera because of which person detection becomes more challenging. As snatch theft incidents occur infrequently, we could only obtain a total of 35 chain snatch theft incidents after searching through the archived video footage of over six months from different surveillance cameras placed in the city of Hyderabad, India. The snatch thefts were spread within 4.5 hours of surveillance footage containing 37485 regular interactions. It was observed that snatch thefts are 4-5 seconds in duration, so we divided the entire surveillance footage into 10-second clips which resulted in a total of 816 clips.

7.4.2 Effect of different feature descriptors and UAM mixtures

In this work, action-vectors are formed using two state-of-the-art feature descriptors, namely, HOF and MBH which are obtained as part of the improved dense trajectory set of features [117]. In Table 7.2, the classification performance of these action-vectors is presented for varying number of UAM mixtures. Also, different classifiers like ensemble subspace discriminant analysis (ESDA), k nearest neighbours (k -NN), and support vector machines (SVM) are used for evaluation. For each of the classifiers, 3-fold cross-validation is used. The dimension for the action-vector referred to as r in the previous section is fixed to be 200 as it is found that varying the action-vector dimension does not yield any change in classification performance. It can be observed that action-vector performs consistently across all the settings making it an effective representation. Further, even smaller UAM with less number of mixtures can help in efficient representing snatch thefts leading to proper classification. In Figure 7.3, the t-SNE plot of the action-vectors with HOF and MBH features using 256 UAM mixtures is shown where clear separability can be noticed between the regular interactions and snatch thefts.

Action-vectors are compared against recent state-of-the-art feature descriptors for describing human actions: a) HOFM (histogram of optical flow and magnitude) [14] and b)

Table 7.2: Action-vector classification performance (in %) for Snatch 1.0.

Classifier	# of UAM mixtures					
	HOF			MBH		
	256	512	1024	256	512	1024
ESDA	99.2	99.3	99.3	98.3	99.4	99.4
k -NN	99.5	99.4	99.4	99.7	99.5	99.5
SVM	99.7	99.7	99.8	99.8	99.6	99.4

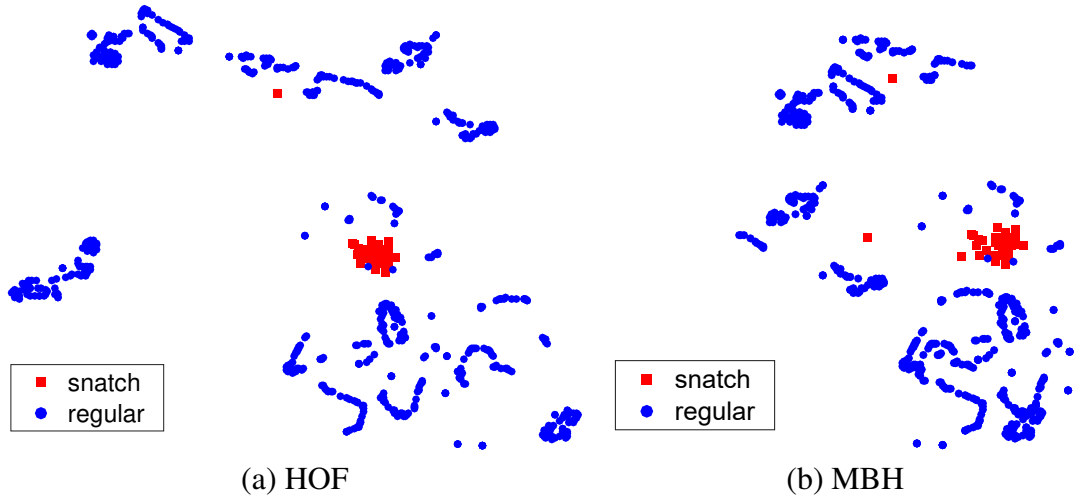


Figure 7.3: Action-vectors for Snatch 1.0 using (a) HOF features and (b) MBH features, using 256 UAM mixtures. Best viewed in color.

3D convolutional neural network (3DCNN) features [113]. For HOFM, we use the parameter settings as per [14] where a regular grid of size $30 \times 30 \times 3$ is created. The first two dimensions correspond to grid cell dimensions (width and height) in space, and the third dimension corresponds to the depth in time. For 3DCNN, a pre-trained network trained on the sports 1M dataset as per [113] was used to obtain the features. Each 3DCNN features summarizes the information in 16 frames into a single descriptor and resulting in 20 feature descriptors for every 10-second clip used in our experiments. The classification outputs of these 20 descriptors are then combined using majority voting to produce the final classification output for the clip.

From Table 7.3, it can be observed that the proposed framework misses only 1 snatch theft as compared 5, 20, and 24 and to the other features. In terms of both accuracy and area under the curve (AUC), action-vector representation outperforms other features as shown in Figure 7.4. Also, when action-vectors calculated using MBH features are compared to MBH features, a significant improvement can be observed. This shows that action-vectors can extract meaningful information from existing descriptors to produce even more discriminative representations.

Table 7.3: Comparison with state-of-the-art feature descriptors using SVM classifier.

Method	Missed Snatches	Accuracy (in %)	AUC (in %)
Rule-based baseline	15	42.9	52.3
HOFM [14]	24	47.6	69.8
MBH [117]	20	59.3	72.4
3DCNN [113]	5	96.6	98.3
action-vector (HOF)	1	99.8	99.8
action-vector (MBH)	1	99.8	99.9

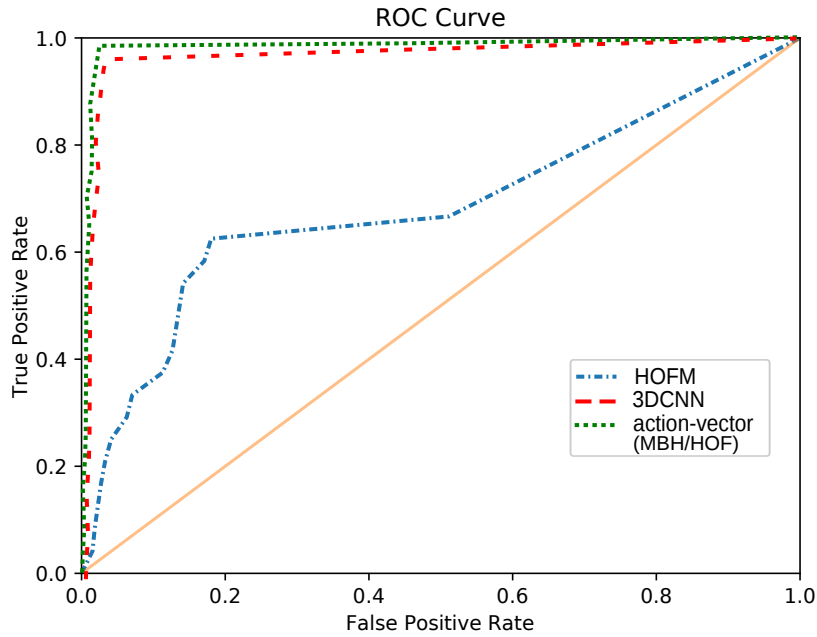


Figure 7.4: ROC for performance comparison of action-vectors with state-of-the-art. Best viewed in colour.

We present some of the detected snatch theft scenarios in Figure 7.5 according to the scenarios explained in Table 7.1. It can be observed that action-vectors recognize a diverse set of snatch thefts which have little similarity to each other. However, a few false positive cases are also encountered where regular interactions are identified as snatch thefts as shown in Figure 7.6. Such cases are difficult to address without identifying and tracking the thief’s limbs.

7.5 Summary

In this chapter, we presented a framework for snatch theft detection in unconstrained videos using action attribute modelling. For evaluation, we introduced a dataset called Snatch 1.0



(a) Scenario 1, Snatch Type 1 and Reaction 1



(b) Scenario 2, Snatch Type 2 and Reaction 3



(c) Scenario 2, Snatch Type 2 and Reaction 1



(d) Scenario 2, Snatch Type 2 and Reaction 3



(e) Scenario 1, Snatch Type 2 and Reaction 1

Figure 7.5: Detection results of snatch theft using the proposed framework. For visualization, a YOLO [93] detector and a kernelised correlation filter [35] based tracker was applied on each of the detected snatch theft clips. Bounding boxes are drawn (in green) with the *id* (in blue) on top of the box for different persons. Best viewed in colour.

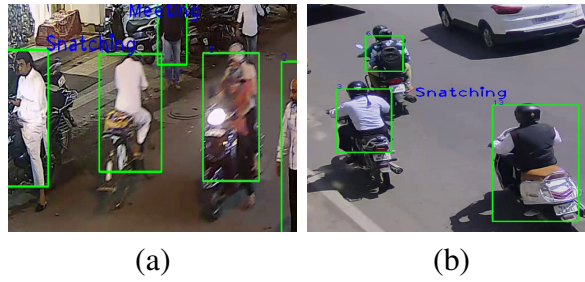


Figure 7.6: False positive cases where normal interactions detected as snatch thefts

that contains snatch thefts in surveillance videos. To obtain the baseline performance on Snatch 1.0, a rule-based system was designed. Our proposed representation for snatch thefts was obtained by leveraging a large GMM trained on action datasets HMDB51 and UCF101. This mitigated the need for labelled snatch theft examples which is difficult to obtain. It was visually shown in the action-vector space, snatch thefts and regular actions can be easily differentiated. Finally, we demonstrated the efficacy of action-vectors over existing state-of-the-art feature descriptors.

Chapter 8

Summary and Conclusions

In this thesis, we presented different approaches for developing robust and discriminative representations for human actions. We showed that the proposed representations were able to address challenges like intra-action variations (due to actors, viewpoints, and duration), inter-action similarity, background motion, and occlusion of actors. Further, we demonstrated that with the non-linear embedding of the proposed representations, actions with fine-grained differences were effectively identified.

In the first approach, we modeled high-level representations of each action class into class-wise dictionaries. We were able to obtain a sparse representation for each action which retained only the discriminative information for that particular action in the dictionary and hence reduces inter-action similarity. A sparsity-based classification method is then proposed to classify the low-rank representation of clips obtained using these dictionaries. However, some of these actions consist of rapid body deformations which impede the extraction of local features necessary for action recognition. Hence, we introduced a sparse representation of convolutional neural network (CNN) features that is robust in locating humans under rapid deformation.

In the next approach, we exploited inter-action similarity to train a universal attribute model (UAM) to learn action attributes (common and distinct) implicitly across all the actions. Using maximum *a posteriori* (MAP) adaptation, a high-dimensional super action-vector (SAV) for each clip was extracted. As this SAV contained redundant attributes of all other actions, we used factor analysis to extract a novel low-dimensional action-vector representation for each clip. Action-vectors were shown to suppress background motion and highlight actions of interest in both trimmed and untrimmed clips. Action-vector were further subjected to linear embedding using linear discriminant analysis (LDA) and probabilistic LDA (PLDA) to address inter-action similarity. Also, Siamese networks were explored for non-linear embedding of action-vectors to mitigate viewpoint variations in

fine-grained action recognition.

Finally, we presented an approach for modeling actions like snatch thefts in surveillance videos. Due to the lack of labeled snatch thefts examples, we leveraged universal attribute models trained on large action datasets to extract the subset of attributes contributing to snatch thefts. The final representation was shown to be effective in distinguishing snatch thefts from regular actions with high accuracy.

8.1 Contributions of the thesis

1. A framework that leverages the sparsity of high-level action features to build discriminative action-specific dictionaries.
2. An approach for recognizing actions with rapid body deformations using sparse representation of convolutional neural network (CNN) features.
3. A novel unsupervised factor analysis based decomposition approach to produce low-dimensional action-vectors. Even without supervision, action-vectors are shown to be highly discriminative for a large number of actions.
4. A method to address inter-action similarity using probabilistic linear discriminant embedding of action-vectors to classify visually similar actions.
5. A method for intermediate latent dimension fusion that combines complimentary information across action-vectors of different feature descriptors. The fused representation obtains state-of-the-art action recognition performance on highly challenging action datasets.
6. An approach for non-linear embedding of action-vectors using deep Siamese networks to discriminate between fine-grained actions with high inter-class similarity.
7. An unsupervised approach for modeling complex and highly infrequent activities like snatch theft detection using action attribute models trained on large action datasets. The final representation obtained is distinct from a variety of regular actions which resulted in excellent classification performance.

8.2 Directions for Further Research

In future, we would like to extend our action-vector based representation for localization of actions in untrimmed videos. Specifically, we would like to design a real-time system that

processes an incoming stream of video data by recursively calculating the action-vectors on short video snippets of the stream. Using a classification module, we should be able to determine the action in that short snippet using action-vectors. We would also like to explore a video recommendation system that uses content-based video indexing to organize large video repositories on the web. The indexing would be based on the action contained in the videos which can then be retrieved by a user-generated query. Such a system would be able to analyze and caption videos automatically which would reduce the burden of manual annotation for the ever-expanding video repositories on the web.

Bibliography

- [1] J. Aggarwal and M. Ryoo. Human activity analysis: A review. *ACM Computing Surveys*, 43(3):16:1–16:43, Apr. 2011.
- [2] M. Aharon, M. Elad, and A. Bruckstein. k -svd: An algorithm for designing over-complete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, Nov 2006.
- [3] W. F. Baum EB, Moody J. Internal representations for associative memory. *Biological Cybernetics*, 59:217–228, 1998.
- [4] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, Mar 1994.
- [5] S. Berlemont, G. Lefebvre, S. Duffner, and C. Garcia. Siamese neural network based similarity metric for inertial gesture classification and rejection. In *Proceedings of IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, volume 1, pages 1–6, May 2015.
- [6] H. Bouma, J. Baan, G. J. Burghouts, P. T. Eendebak, J. R. van Huis, J. Dijk, and J. H. van Rest. Automatic detection of suspicious behavior of pickpockets with track-based features in a shopping mall. In *SPIE Security+ Defence*, pages 92530F–92530F. International Society for Optics and Photonics, 2014.
- [7] S. B. M. T. Brecht M, Schneider M. Whisker movements evoked by stimulation of single pyramidal cells in rat motor cortex. *Nature*, 427:704–710, 2004.
- [8] Z. Cai, L. Wang, X. Peng, and Y. Qiao. Multi-view super vector for action recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 596–603, June 2014.
- [9] A. Castrodad and G. Sapiro. Sparse modeling of human actions from motion imagery. *International Journal of Computer Vision (IJCV)*, 100(1):1–15, 2012.
- [10] R. Chaudhry, A. Ravichandran, G. Hager, and R. Vidal. Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1932–1939. IEEE, 2009.
- [11] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun. Bayesian face revisited: A joint formulation. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 566–579. Springer, 2012.

- [12] A. Chorian, B. Fernando, M. Harandi, and S. Gould. Generalized rank pooling for activity recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1581–1590, July 2017.
- [13] D. Chung, K. Tahboub, and E. J. Delp. A two stream siamese convolutional neural network for person re-identification. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 1992–2000, Oct 2017.
- [14] R. M. Colque, C. Caetano, M. Toledo, and W. R. Schwartz. Histograms of optical flow orientation and magnitude and entropy to detect anomalous events in videos. *IEEE Transactions on Circuits and Systems for Video Technology*, PP(99):1–1, 2016.
- [15] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893. IEEE, 2005.
- [16] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 428–441. Springer, 2006.
- [17] C. R. de Souza, A. Gaidon, E. Vig, and A. M. López. *Sympathy for the Details: Dense Trajectories and Hybrid Classification Architectures for Action Recognition*, pages 697–716. Springer International Publishing, Cham, 2016.
- [18] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet. Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):788–798, May 2011.
- [19] A. Diba, V. Sharma, and L. Van Gool. Deep temporal linear encoding networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2329–2338, 2017.
- [20] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 677–691, June 2015.
- [21] X. Dong, J. Shen, D. Yu, W. Wang, J. Liu, and H. Huang. Occlusion-aware real-time object tracking. *IEEE Transactions on Multimedia*, 19(4):763–771, April 2017.
- [22] I. C. Duta, B. Ionescu, K. Aizawa, and N. Sebe. Spatio-temporal vector of locally max pooled features for action recognition in videos. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3205–3214, July 2017.
- [23] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594–611, April 2006.
- [24] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1933–1941, June 2016.

- [25] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, Sept 2010.
- [26] Y. Feng, M. Ji, J. Xiao, X. Yang, J. J. Zhang, Y. Zhuang, and X. Li. Mining spatial-temporal patterns and structural sparsity for human motion data denoising. *IEEE Transactions on Cybernetics*, 45(12):2693–2706, Dec 2015.
- [27] B. Fernando, S. Gavves, O. Mogrovejo, J. Antonio, A. Ghodrati, and T. Tuytelaars. Modeling video evolution for action recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5378–5387, 2015.
- [28] A. Gaidon, Z. Harchaoui, and C. Schmid. Temporal localization of actions with actoms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2782–2795, Nov 2013.
- [29] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pages 2672–2680, 2014.
- [30] K. Goya, X. Zhang, K. Kitayama, and I. Nagayama. A method for automatic detection of crimes for public security by using motion analysis. In *Proceedings of International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 736–741, Sept 2009.
- [31] T. Guha and R. Ward. Learning sparse representations for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(8):1576–1588, Aug 2012.
- [32] T. Guha and R. K. Ward. Learning sparse representations for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(8):1576–1588, 2012.
- [33] S. Han, R. Fu, S. Wang, and X. Wu. Online adaptive dictionary learning and weighted sparse coding for abnormality detection. In *Proceedings of IEEE International Conference on Image Processing (ICIP)*, pages 151–155, Sept 2013.
- [34] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.
- [35] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015.
- [36] S. Herath, M. Harandi, and F. Porikli. Going deeper into action recognition: A survey. *Image and Vision Computing*, 60:4 – 21, 2017. Regularization Techniques for High-Dimensional Data Analysis.
- [37] S. Herath, M. Harandi, and F. Porikli. Going deeper into action recognition: A survey. *Image and Vision Computing*, 60:4 – 21, 2017. Regularization Techniques for High-Dimensional Data Analysis.
- [38] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

- [39] S. Hong, T. You, S. Kwak, and B. Han. Online tracking by learning discriminative saliency map with convolutional neural network. In *Proceedings of International Conference on Machine Learning (ICML)*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 597–606. JMLR.org, 2015.
- [40] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao. Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 749–758, June 2015.
- [41] N. Ibrahim, M. M. Mustafa, S. S. Mokri, L. Y. Siong, and A. Hussain. Detection of snatch theft based on temporal differences in motion flow field orientation histograms. *International Journal of Advancements in Computing Technology*, 4(12), 2012.
- [42] N. Inoue and K. Shinoda. A fast and accurate video semantic-indexing system using fast map adaptation and gmm supervectors. *IEEE Transactions on Multimedia*, 14(4):1196–1205, Aug 2012.
- [43] S. Ioffe. *Probabilistic Linear Discriminant Analysis*, pages 531–542. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.
- [44] M. Jain, J. C. van Gemert, and C. G. M. Snoek. What do 15,000 object categories tell us about classifying and localizing actions? In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 46–55, June 2015.
- [45] G. Jeong and H. S. Yang. Context-aware activity recognition by markov logic networks of trained weights. In *Proceedings of International Conference on Virtual Systems and Multimedia*, pages 5–12, Oct 2010.
- [46] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):221–231, Jan 2013.
- [47] Y.-G. Jiang, Q. Dai, X. Xue, W. Liu, and C.-W. Ngo. Trajectory-based modeling of human actions with motion reference points. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 425–438. Springer, 2012.
- [48] Y.-G. Jiang, J. Liu, A. Roshan Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes, 2014.
- [49] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1725–1732, 2014.
- [50] H. Kataoka, K. Hashimoto, K. Iwata, Y. Satoh, N. Navab, S. Ilic, and Y. Aoki. Extended co-occurrence hog with dense trajectories for fine-grained activity recognition. In *Proceedings of Asian Conference on Computer Vision (ACCV)*, pages 336–349, Cham, 2015. Springer International Publishing.
- [51] P. Kenny, G. Boulianne, and P. Dumouchel. Eigenvoice modeling with sparse training data. *IEEE Transactions on Speech and Audio Processing*, 13(3):345–354, 2005.

- [52] F. Keskin, Y. E. Kara, and L. Akarun. Hand pose estimation and hand shape classification using multi-layered randomized decision forests. In *Proceedings of European Conference on Computer Vision (ECCV)*, ECCV'12, pages 852–863, Berlin, Heidelberg, 2012. Springer-Verlag.
- [53] A. Klaser, M. Marszalek, and C. Schmid. A spatio-temporal descriptor based on 3d-gradients. In *Proceedings of British Machine Vision Conference (BMVC)*, pages 275:1–10, May. 2008.
- [54] O. Kliper-Gross, Y. Gurovich, T. Hassner, and L. Wolf. Motion interchange patterns for action recognition in unconstrained videos. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 425–438, Oct. 2012.
- [55] G. Koch. *Siamese Neural Networks for One-Shot Image Recognition*. PhD thesis, University of Toronto, 2015.
- [56] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pages 1097–1105, 2012.
- [57] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pages 1097–1105. Curran Associates, Inc., 2012.
- [58] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb: a large video database for human motion recognition. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 2556–2563. IEEE, 2011.
- [59] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: a large video database for human motion recognition. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 2556–2563, Nov. 2011.
- [60] G. I Varol and A. A. Salah. Extreme learning machine for large-scale action recognition. In *Proceedings of European Conference on Computer Vision workshops (ECCV workshops)*, 2014.
- [61] Z. Lan and A. G. Hauptmann. Beyond spatial pyramid matching: Space-time extended descriptor for action recognition. *CoRR*, abs/1510.04565, 2015.
- [62] Z. Lan, X. Li, M. Lin, and A. G. Hauptmann. Long-short term motion feature for action classification and retrieval. *CoRR*, abs/1502.04132, 2015.
- [63] Z. Lan, M. Lin, X. Li, A. G. Hauptmann, and B. Raj. Beyond gaussian pyramid: Multi-skip feature stacking for action recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 204–212, June 2015.
- [64] Z. Lan, S. Yu, M. Lin, B. Raj, and A. G. Hauptmann. Handcrafted local features are convolutional neural networks. *CoRR*, abs/1511.05045, 2015.
- [65] Z. Lan, Y. Zhu, A. G. Hauptmann, and S. Newsam. Deep local video feature for action recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1219–1225. IEEE, 2017.

- [66] I. Laptev. On space-time interest points. *International Journal of Computer Vision (IJCV)*, 64(2):107–123, Sep 2005.
- [67] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, Jun. 2008.
- [68] M. J. Leach, E. Sparks, and N. M. Robertson. Contextual anomaly detection in crowded surveillance scenes. *Pattern Recognition Letters*, 44:71 – 79, 2014.
- [69] D. G. Lee, H. I. Suk, S. K. Park, and S. W. Lee. Motion influence map for unusual human activity detection and localization in crowded scenes. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(10):1612–1623, Oct 2015.
- [70] K. Li, J. Hu, and Y. Fu. Modeling complex temporal composition of actionlets for activity prediction. In *Proceedings of European Conference on Computer Vision (ECCV)*, ECCV’12, pages 286–299, Berlin, Heidelberg, 2012. Springer-Verlag.
- [71] L. Li, T. Nawaz, and J. Ferryman. Pets 2015: Datasets and challenge. In *Proceedings of IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, Aug 2015.
- [72] L.-J. Li, H. Su, L. Fei-Fei, and E. P. Xing. Object bank: A high-level image representation for scene classification & semantic feature sparsification. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pages 1378–1386, 2010.
- [73] Y. Li, W. Li, V. Mahadevan, and N. Vasconcelos. Vlad3: Encoding dynamics of deep features for action recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1951 – 1960, June 2016.
- [74] T. Lim, S. Hong, B. Han, and J. H. Han. Joint segmentation and pose tracking of human in natural videos. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, ICCV ’13, pages 833–840, Washington, DC, USA, 2013. IEEE Computer Society.
- [75] J. Liu, Y. Huang, X. Peng, and L. Wang. Multi-view descriptor mining via codeword net for action recognition. In *Proceedings of IEEE International Conference on Image Processing (ICIP)*, pages 793–797, Sept 2015.
- [76] J. Liu, J. Luo, and M. Shah. Recognizing realistic actions from videos ’in the wild’. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1996–2003, Jun. 2009.
- [77] S. Lu, J. Zhang, Z. Wang, and D. D. Feng. Fast human action classification and {VOI} localization with enhanced sparse coding. *Journal of Visual Communication and Image Representation*, 24(2):127 – 136, 2013. Sparse Representations for Image and Video Analysis.
- [78] W. L. Lu, J. A. Ting, J. J. Little, and K. P. Murphy. Learning to track and identify players from broadcast sports videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(7):1704–1716, July 2013.

- [79] B. Ma, J. Shen, Y. Liu, H. Hu, L. Shao, and X. Li. Visual tracking using strong classifier and structural local sparse descriptors. *IEEE Transactions on Multimedia*, 17(10):1818–1828, Oct 2015.
- [80] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research (JMLR)*, 9(Nov):2579–2605, 2008.
- [81] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 689–696, New York, NY, USA, Jun. 2009. ACM.
- [82] X. Mo, V. Monga, R. Bala, and Z. Fan. Adaptive sparse representations for video anomaly detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 24(4):631–645, April 2014.
- [83] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4293–4302, June 2016.
- [84] J. Y. Ng, M. J. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. *CoRR*, abs/1503.08909, 2015.
- [85] C. Panagiotakis, E. Ramasso, G. Tziritas, M. Rombaut, and D. Pellerin. Shape-motion based athlete tracking for multilevel action recognition. In *Proceedings of International Conference on Articulated Motion and Deformable Objects (AMDO)*, pages 385–394, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.
- [86] E. Park, X. Han, T. L. Berg, and A. C. Berg. Combining multiple sources of knowledge in deep cnns for action recognition. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–8. IEEE, 2016.
- [87] X. Peng, Q. Peng, Y. Qiao, J. Chen, and M. Afzal. A study on unsupervised dictionary learning and feature encoding for action classification. *CoRR*, abs/1309.0309, 2013.
- [88] X. Peng, L. Wang, Y. Qiao, and Q. Peng. A joint evaluation of dictionary learning and feature encoding for action recognition. In *Proceedings of International Conference on Pattern Recognition (ICPR)*, pages 2607–2612, Aug 2014.
- [89] F. Perronnin. Universal and adapted vocabularies for generic visual categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(7):1243–1256, July 2008.
- [90] S. J. Prince and J. H. Elder. Probabilistic linear discriminant analysis for inferences about identity. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 1–8. IEEE, 2007.
- [91] Q. Qiu, Z. Jiang, and R. Chellappa. Sparse dictionary-based representation and recognition of action attributes. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 707–714, Nov 2011.
- [92] K. K. Reddy and M. Shah. Recognizing 50 human action categories of web videos. *Machine Vision and Applications*, 24(5):971–981, 2013.

- [93] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, June 2016.
- [94] M. Rohrbach, S. Amin, M. Andriluka, and B. Schiele. A database for fine grained activity detection of cooking activities. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1194–1201. IEEE, 2012.
- [95] M. Rohrbach, S. Amin, M. Andriluka, and B. Schiele. A database for fine grained activity detection of cooking activities. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1194–1201, June 2012.
- [96] S. Sadanand and J. Corso. Action bank: A high-level representation of activity in video. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1234–1241, Jun. 2012.
- [97] S. Sadanand and J. J. Corso. Action bank: A high-level representation of activity in video. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1234–1241. IEEE, 2012.
- [98] P. Scovanner, S. Ali, and M. Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of ACM International Conference on Multimedia (ACMMM)*, pages 357–360. ACM, 2007.
- [99] C. Shan, T. Tan, and Y. Wei. Real-time hand tracking using a mean shift embedded particle filter. *Pattern Recognition*, 40(7):1958–1970, July 2007.
- [100] M. H. Sharif and C. Djeraba. An entropy approach for abnormal activities detection in video streams. *Pattern Recognition*, 45(7):2543 – 2561, 2012.
- [101] N. Shimada, K. Kimura, and Y. Shirai. Real-time 3-d hand posture estimation based on 2-d appearance retrieval using monocular camera. In *Proceedings of ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems (RATFG-RTS'01)*, RATFG-RTS '01, pages 23–, Washington, DC, USA, 2001. IEEE Computer Society.
- [102] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pages 568–576, 2014.
- [103] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pages 568–576, 2014.
- [104] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pages 568–576. Curran Associates, Inc., 2014.
- [105] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.
- [106] C. Sminchisescu and B. Triggs. Estimating articulated human motion with covariance scaled sampling. *International Journal of Robotics Research*, 22(6):371–391, June 2003. Special issue: Visual Analysis of Human Movement.

- [107] A. A. Sodemann, M. P. Ross, and B. J. Borghetti. A review of anomaly detection in automated surveillance. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):1257–1272, Nov 2012.
- [108] B. Solmaz, S. Assari, and M. Shah. Classifying web videos using a global video descriptor. *Machine Vision and Applications*, 24(7):1473–1485, 2013.
- [109] K. Soomro, A. R. Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *CoRR*, abs/1212.0402, 2012.
- [110] N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using lstms. *CoRR*, abs/1502.04681, 2015.
- [111] L. Sun, K. Jia, D.-Y. Yeung, and B. E. Shi. Human action recognition using factorized spatio-temporal convolutional networks. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 4597–4605, 2015.
- [112] S. Todorovic. Human activities as stochastic kronecker graphs. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 130–143. Springer, 2012.
- [113] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 4489–4497, 2015.
- [114] G. Varol, I. Laptev, and C. Schmid. Long-term Temporal Convolutions for Action Recognition. *arXiv:1604.04494*, 2016.
- [115] R. Venkatesh Babu, P. Parate, and Äniruddha Acharya K. Robust tracking with interest points. *Image and Vision Computing*, 33(C):44–56, Jan. 2015.
- [116] H. Wang, D. Oneata, J. Verbeek, and C. Schmid. Improved trajectories video description.
- [117] H. Wang, D. Oneata, J. Verbeek, and C. Schmid. A robust and efficient video representation for action recognition. *International Journal of Computer Vision (IJCV)*, 119(3):219–238, July 2015.
- [118] H. Wang and C. Schmid. Action recognition with improved trajectories. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 3551–3558, 2013.
- [119] H. Wang and C. Schmid. Action recognition with improved trajectories. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, Sydney, Australia, Oct 2013.
- [120] H. Wang and C. Schmid. Lear-inria submission for the thumos workshop. In *Proceedings of ICCV Workshop on Action Recognition with a Large Number of Classes*, volume 2, page 8, 2013.
- [121] H. Wang, C. Yuan, W. Hu, H. Ling, W. Yang, and C. Sun. Action recognition using nonnegative action component representation and sparse basis selection. *IEEE Transactions on Image Processing*, 23(2):570–581, Feb 2014.
- [122] L. Wang, Y. Qiao, and X. Tang. Action recognition and detection by combining motion and appearance features. *THUMOS14 Action Recognition Challenge*, 1(2):2, 2014.

- [123] L. Wang, Y. Qiao, and X. Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4305–4314, 2015.
- [124] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. *Temporal Segment Networks: Towards Good Practices for Deep Action Recognition*, pages 20–36. Springer International Publishing, Cham, 2016.
- [125] P. Wang, Y. Cao, C. Shen, L. Liu, and H. T. Shen. Temporal pyramid pooling based convolutional neural networks for action recognition. *CoRR*, abs/1503.01224, 2015.
- [126] S. Wang, Z. Ma, Y. Yang, X. Li, C. Pang, and A. G. Hauptmann. Semi-supervised multiple feature analysis for action recognition. *IEEE Transactions on Multimedia*, 16(2):289–298, Feb 2014.
- [127] X. Wang, A. Farhadi, and A. Gupta. Actions~ transformations. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2658–2667, 2016.
- [128] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 2794–2802, 2015.
- [129] Y. Wang, B. Wang, Y. Yu, Q. Dai, and Z. Tu. Action-gons: Action recognition with a discriminative dictionary of structured elements with varying granularity. In *Proceedings of Asian Conference on Computer Vision (ACCV)*, pages 259–274. Springer, 2014.
- [130] Z. Wang, Y. Wang, L. Wang, and Y. Qiao. Codebook enhancement of vlad representation for visual recognition. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1258–1262, March 2016.
- [131] A. Wiliem, V. Madasu, W. Boles, and P. Yarlagadda. A context-based approach for detecting suspicious behaviours. In *Proceedings of Digital Image Computing: Techniques and Applications*, pages 146–153, Dec 2009.
- [132] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, Feb 2009.
- [133] J. Wu and D. Hu. Learning effective event models to recognize a large number of human actions. *IEEE Transactions on Multimedia*, 16(1):147–158, Jan 2014.
- [134] J. Wu and D. Hu. Learning effective event models to recognize a large number of human actions. *IEEE Transactions on Multimedia*, 16(1):147–158, 2014.
- [135] J. Xing, H. Ai, and S. Lao. Multi-object tracking through occlusions by local tracklets filtering and global tracklets association with detection responses. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1200–1207, June 2009.
- [136] R. Yao, S. Xia, Z. Zhang, and Y. Zhang. Real-time correlation filter tracking by efficient dense belief propagation with structure preserving. *IEEE Transactions on Multimedia*, 19(4):772–784, April 2017.

- [137] Y. Yuan, H. Yang, Y. Fang, and W. Lin. Visual object tracking by structure complexity coefficients. *IEEE Transactions on Multimedia*, 17(8):1125–1136, Aug 2015.
- [138] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 818–833. Springer, 2014.
- [139] D. Zhang and M. Shah. Human pose estimation in videos. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, pages 2012–2020, 2015.
- [140] S. Zhang, X. Cheng, H. Guo, L. Zhou, and Z. Wu. Tracking deformable parts via dynamic conditional random fields. In *Proceedings of IEEE International Conference on Image Processing (ICIP)*, pages 476–480, Oct 2014.
- [141] T. Zhang, S. Liu, C. Xu, S. Yan, B. Ghanem, N. Ahuja, and M. H. Yang. Structural sparse tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 150–158, June 2015.
- [142] Y. Zhang, H. Lu, L. Zhang, and X. Ruan. Combining motion and appearance cues for anomaly detection. *Pattern Recognition*, 51:443–452, 2016.
- [143] F. Zhou, F. D. I. Torre, and J. K. Hodgins. Hierarchical aligned cluster analysis for temporal clustering of human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(3):582–596, March 2013.
- [144] Y. Zhou, B. Ni, R. Hong, M. Wang, and Q. Tian. Interaction part mining: A mid-level approach for fine-grained action recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3323–3331, 2015.
- [145] Y. Zhou, B. Ni, S. Yan, P. Moulin, and Q. Tian. Pipelining localized semantic features for fine-grained action recognition. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 481–496, Cham, 2014. Springer International Publishing.
- [146] Z. Zhou, F. Shi, and W. Wu. Learning spatial and temporal extents of human actions for action detection. *IEEE Transactions on Multimedia*, 17(4):512–525, April 2015.
- [147] Y. Zhu and S. Newsam. *Depth2Action: Exploring Embedded Depth for Large-Scale Action Recognition*, pages 668–684. Springer International Publishing, Cham, 2016.
- [148] Y. Zhu, X. Zhao, Y. Fu, and Y. Liu. Sparse coding on local spatial-temporal volumes for human action recognition. In *Proceedings of Asian Conference on Computer Vision (ACCV)*, pages 660–671. Springer, 2010.

Publications

Patents

1. C. Krishna Mohan, Dinesh Singh, C. Vishnu, and **Debaditya Roy**. "*Method and system for detection of accident in traffic surveillance video*", *Indian Complete Patent*, Application no. 201841003604, Jan. 31, 2018
2. **Debaditya Roy**, Dinesh Singh, C. Vishnu, and C. Krishna Mohan. "*Method and system for detection of crime events in surveillance videos*", *Indian Complete Patent*, Application no. 201741041239, Nov. 17, 2017
3. C. Krishna Mohan, Dinesh Singh, C. Vishnu, and **Debaditya Roy**. "*A method and system for real time detection of traffic violation by two-wheeled riders*", *Indian Complete Patent*, Application no. 201741038813, Nov. 1, 2017

Journals

1. **Debaditya Roy** and C. Krishna Mohan. "Snatch Theft Detection in Unconstrained Surveillance Videos Using Action Attribute Modeling", *Pattern Recognition Letters, Elsevier* vol. 108, pp. 56-61, Mar. 2018
2. Dinesh Singh, **Debaditya Roy**, and C. Krishna Mohan. "DiP-SVM : Distribution Preserving Kernel Support Vector Machine for Big Data", in *IEEE Transactions on Big Data*. vol. 3, no. 1, pp. 79-90, Mar. 2017
3. **Debaditya Roy**, M. Srinivas, and C. Krishna Mohan. "Sparsity-inducing Dictionaries for Effective Action Classification", *Pattern Recognition, Elsevier*. vol. 59, pp. 55-62, Nov. 2016
4. **Debaditya Roy**, K. Sri Rama Murty, and C. Krishna Mohan. "Unsupervised Action Recognition using Universal Attribute Modelling", *IEEE Transactions on Multimedia* (under 2nd review), Nov. 2017
5. Nazil Perveen, **Debaditya Roy**, and C. Krishna Mohan. "Unsupervised Spontaneous Expression Recognition using Universal Attribute Model", *IEEE Transactions on Image Processing* (under 2nd review), Jan. 2018

Conferences

1. **Debaditya Roy**, K. Sri Rama Murty and C. Krishna Mohan. “Action Recognition based on discriminative embedding of actions using Siamese networks”, *IEEE Int. Conf. on Image Processing (ICIP2018)* (under review)
2. **Debaditya Roy**, K. Sri Rama Murty and C. Krishna Mohan. “Action-vectors: Un-supervised movement modeling for action recognition”, In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2017)*, pp. 1602-1606, Mar. **2017**
3. M. Srinivas, **Debaditya Roy**, and C. Krishna Mohan. “Discriminative feature extraction from X-Ray images using Deep Convolutional Neural Network”, In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2016)*, pp. 917-921, Mar. **2016**
4. **Debaditya Roy**, K. Sri Rama Murty and C. Krishna Mohan. “Feature selection using deep neural networks”, In *Proc. Int. Joint Conf. on Neural Networks (IJCNN 2015)*, pp.1-6, Jul. **2015**
5. **Debaditya Roy**, M. Srinivas and C. Krishna Mohan. “Sparsifying Dense Features for Action Recognition”, In *Proc. Int. Conf. on Perception and Machine Intelligence (PerMin 2015)*, ACM ICPS, Feb. **2015**
6. M. Srinivas, **Debaditya Roy** and C. Krishna Mohan. “Learning Sparse Dictionaries for Music and Speech Classification”, In *Proc. IEEE 19th Int. Conf. on Digital Signal Processing (DSP 2014)*, pp. 673-675, Aug. **2014**
7. M. Srinivas, **Debaditya Roy** and C. Krishna Mohan. “Music Genre Classification using On-line Dictionary Learning”, In *Proc. 2014 Int. Joint Conf. on Neural Networks (IJCNN 2014)*, pp. 1937-1941, Jul. **2014**

CURRICULUM VITAE

Name: Debaditya Roy

Date of Birth: January 14, 1990

Educational Qualifications:

- August 2013 - present : *Doctor of Philosophy*, Computer Science and Engineering, Indian Institute of Technology Hyderabad
- July 2011 - June 2013 : *Master of Technology (Institute Silver Medal)*, Computer Science and Engineering, National Institute of Technology Rourkela
- August 2007 - June 2011 : *Bachelor of Technology*, Computer Science and Engineering

DOCTORAL COMMITTEE MEMBERS

Chairperson: Dr. M. V. Panduranga Rao

Advisor: Dr. C. Krishna Mohan

Members:

- Dr. Vineeth N. Balasubramanian (Dept. of CSE)
- Dr. N. R. Aravind (Dept. of CSE)
- Dr. Sumohana S. Channappayya (Dept. of EE)