

VIDEO CLASSIFICATION BY USING NEURAL NETWORK

Submitted in partial fulfillment of the
requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering

By

FEBIN THOMAS (Reg. No. 37110214)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
OF SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600 119**

APRIL- 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **FEBIN THOMAS** (Reg. No. 37110214) who carried out the project entitled as **VIDEO CLASSIFICATION BY USING NEURAL NETWORK** under my supervision from September 2020 to April 2021.

Internal Guide

Dr. M. SARAIVANAN, M.E., Ph.D.,

Head of the Department

Dr. S. VIGNESHWARI, M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **FEBIN THOMAS** (Reg. No. 37110214) AND **E.ROHAN REDDY** (Reg. No. 37110210) hereby declare that the Project Report entitled as “**VIDEO CLASSIFICATION BY USING NEURAL NETWORKS**” project done by me under the guidance of **Dr. M SARVANAN, M.E., Ph.D.**, and at Satyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

SIGNATURE OF THE CANDIDATE

PLACE:

ACKNOWLEDGEMENT

I am pleased to acknowledge our sincere thanks to Board of Management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. SASIKALA, M.E., Ph.D., Dean, School of Computing** and also **Dr. S. VIGNESHWARI, M.E., Ph.D., and Dr. L. LAKSHMANAN, ME., Ph.D., Heads of the Department, Department of Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. M. SARVANAN, M.E., Ph.D.,** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Video classification has been extensively researched in computer vision due to its wide spread use in many important applications such as human action recognition and dynamic scene classification. It is highly desired to have an end-to-end learning framework that can establish effective video representations while simultaneously conducting efficient video classification. The convolution 3-D (C3-D) and VGG (vision and graphics group) are first deployed to extract temporal and spatial features from the input videos cooperatively, which establishes comprehensive and informative representations of videos. VGG and C3-D are chosen due to their strong capability of extracting complementary spatial and temporal features for comprehensive video representations. The introduced RLE layer is further deployed to encode the initial outputs of classifiers, which is followed by a weighting layer jointly learned in the end-to-end framework to combine classification results. In this proposed system, we propose the NEURAL network method for action recognition in video. The input video is converted into number of frames. The converted frames will be stored in the database. Then the ROI (Region of interest) segmentation algorithm is used in order to detect the particular part of the frame. Finally, the NEURAL network classifier is used to detect the given input video is abnormal or not.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	v
	LIST OF FIGURES	ix
	LIST OF ABBREVIATION	x
1	INTRODUCTION	1
	1.1 OUTLINE OF THE PROJECT	1
	1.2 IMAGE PROCESSING	3
	1.3 TYPES OF DIGITAL IMAGE	4
	1.4 COLOR MODEL	4
	1.4.1 RBG COLOR MODEL	4
	1.4.2 CONVERTING COLOR TO GRAYSCALE	5
	1.5 DIGITAL IMAGE PROCESSING TECHNIQUES	5
	1.5.1 IMAGE ENHANCEMENT	6
	1.5.2 IMAGE SEGMENTATION	6
2	LITERATURE SURVEY	7
	2.1 DYNAMIC SCENE CLASSIFICATION	7
	2.2 IMAGE NET CLASSIFICATION WITH DEEP CONVOLUTIONAL NEURAL NETWORKS	7
	2.3 TWO-STREAM CONVOLUTION NETWORKS FOR ACTION RECOGNITION IN VIDEOS	8
	2.4 SUPERVISED LOCAL DESCRIPTOR LEARNING FOR HUMAN ACTION RECOGNITION	9
	2.5 RECOGNIZING ACTIONS BY SHAPE-MOTION PROTOTYPE TREES	10
	2.6 OPTIMIZING KERNEL MACHINES USING DEEP LEARNING	10
	2.7 DYNAMIC SCENE CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORKS	11

2.8 HYBRID DEEP LEARNING FOR FACE VERIFICATION	12
--	----

CHAPTER NO.	TITLE	PAGE NO.
	2.9 TEMPORAL SEGMENT NETWORKS	13
	2.10 JOINT BINARY CLASSIFIER LEARNING FOR ECOC-BASED MULTI-CLASS CLASSIFICATION	14
3	METHODOLOGY	15
	3.1 EXISTING SYSTEM	15
	3.2 PROPOSED SYSTEM	15
	3.3 MODULE DESCRIPTION	15
	3.4 REGION OF INTEREST (ROI) SEGMENTATION	17
	3.5 NEURAL NETWORKS	20
	3.5.1 HISTORICAL BACKGROUND	22
	3.5.2 THE BASICS OF NEURAL NETWORKS	22
	3.5.3 NEURAL NETWORKS VS CONVENTIONS	23
	3.5.4 NETWORK LAYERS	24
	3.6 SYSTEM REQUIREMENTS	25
	3.6.1 HARDWARE REQUIREMENTS	25
	3.6.2 SOFTWARE REQUIREMENTS	25
	3.7 MATLAB	25
	3.8 INTRODUCTION TO MATLAB R2018a	26
	3.8.1 THE MATLAB SYSTEM	26
	3.8.2 COMPUTER VISION TOOLBOX	28
	3.8.3 IMAGE ACQUISITION TOOLBOX	29
	3.9 SYSTEM ARCHITECTURE	30
	3.9.1 ARCHITECTURE	30
	3.9.2 SEQUENCE DIAGRAM	31
	3.9.3 ACTIVITY DIAGRAM	32
	3.9.4 USE CASE DIAGRAM	33
	3.9.5 ER DIAGRAM	33

4	RESULTS AND DISCUSSIONS	34
.	4.1 RESULTS	34
	4.2 SCREENSHOTS	34
	4.2.1 FRAME CONVERSION IMAGE	34
CHAPTER	TITLE	PAGE
NO.		NO.
	4.2.2 ROI	35
	4.2.3 SEGMENTATION	35
	4.2.4 NEURAL NETWORK	36
	4.3 DISCUSSION	36
5	CONCLUSIONS AND FUTURE ENHANCEMENTS	37
	5.1 CONCLUSION	37
	5.2 FUTURE ENHANCEMENTS	37
	REFERENCES	38
	APPENDIX	40
	SOURCE CODE	40
	SCREENSHOTS	42
	PUBLICATION WITH PLAGIARISM REPORT	43

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
3.1	SYSTEM ARCHITECTURE	30
3.2	SEQUENCE DIAGRAM	30
3.3	ACTIVITY DIAGRAM	31
3.4	USE CASE DIAGRAM	31
3.5	CONVERTING IMAGE	32
3.6	GETTING ROI FROM INPUT VIDEO	32
3.7	NEURAL NETWORK FROM SEGMENTED IMAGE	33
3.8	FINAL RESULT FROM INPUT VIDEO	33
4.1	FRAME CONVERSION IMAGE	34
4.2	ROI	35
4.3	SEGMENTATION	35
4.4	NEURAL NETWORK	36

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
NNs	Neural networks
CNNs	Convolution Neural networks
LSTM	Long short-term memory
RBM	Restricted Boltzmann machine
RGB	Red Green and Blue
TSN	Temporal segment network
ECOC	Error correcting output coding
ROI	Region of interest
VGG	Vision and graphics group

CHAPTER 1

INTRODUCTION

1.1 OUTLINE OF THE PROJECT

Video classification has been extensively researched in computer vision due to its wide spread use in many important applications such as human action recognition and dynamic scene classification and so on. However, video sequences present great variability due to huge scale changes, viewpoint variation, and camera motion, which pose great challenges for both video representations and classification. With the great success of deep learning, Neural networks (NNs) and their 3-D variants have been studied in the video domain for a large variety of classification tasks. It is highly desired to have an end-to-end learning framework that can establish effective video representations while simultaneously conducting efficient video classification. Video representations have played a crucially important role in video analysis, which directly affect the ultimate performance of video classification. Both the spatial and temporal information should be captured and encoded for comprehensive and informative representation of video sequences. The two-stream CNNs that extract spatial and temporal features have shown better performance than the one-stream counterpart. Deep feature level fusion schemes have great capability of video representations for improving the performance of video classification. For instances, Karpathy et al. proposed various fusion schemes to process the features; Sun et al. used restricted Boltzmann machine to perform the inference from complementary high-level features extracted from two ConvNet groups with a two-level average pooling hierarchy; Ng et al. designed an end-to-end framework using the long short-term memory (LSTM) to process the features extracted from frames by NNs. Those feature-level fusion methods concatenate the features simply for the following processing, which do not investigate the ensemble essence from the classification point of view. When it comes to video classification tasks, one has to consider not only the appearance information in the spatial domain but also the complex temporal evolution of the scenes. Due to the fusion

of spatial and temporal features, it would unavoidably induce high-dimensional video representations, which raise great challenge for efficient and accurate video classification. Ensemble learning has shown great capability of handling high-dimensional features to conduct accurate and efficient classification tasks. By working with deep learning models, ensemble learning can largely improve the performance for instance, both vision and graphics group (VGG) and Alex Net use different initializations to retrain the networks and average the predictions. Dropout trains multiple CNNs by randomly throwing away certain activations, and then combines these CNNs into one network. Such methods incorporate the ensemble theory into deep learning by training many subnetworks. To leverage the strength of ensemble learning, we propose incorporating random projections-based ensemble learning into the deep learning framework, which enables effectively handling high-dimensional video representations for efficient and more accurate video classification. Error-correcting output coding (ECOC) is a successful strategy to deal with multiclass classification problems by decomposing the original problem into a series of binary classification subproblems. Considering the classification results from the ensemble of classifiers are dense with huge redundancy, we propose a rectified linear encoding (RLE) layer to further encode the initial outputs of classifiers inspired by the idea of ECOC. By doing so, the computation afterwards could be reduced a lot. In this paper, the RLE layer turns out to be an activation layer similar to the rectified linear units (ReLUs) It effectively fuses the initial classification results on the label level and feeds them as the input into the fully connected layer to produce the final results. The end-to-end learning framework for video classification is illustrated in Fig. 1. The convolutional 3-D (C3-D) and VGG are first deployed to extract temporal and spatial features from the input videos cooperatively, which establishes comprehensive and informative representations of videos. VGG and C3-D are chosen due to their strong capability of extracting complementary spatial and temporal features for comprehensive video representations. The resultant high-dimensional representations are further reduced by random projections into a set of lower dimensional subspaces, on

which an ensemble of efficient classifiers is trained with these lower dimensional features.

1.2 IMAGE PROCESSING

Image Processing is a method to enhance raw images received from cameras/sensors placed on satellites, space probes and aircrafts or pictures taken in normal day-to-day life for various applications. Various techniques have been developed in Image Processing during the last four to five decades. Most of the techniques are developed for enhancing images obtained from unmanned spacecrafts, space probes and military reconnaissance flights. Image Processing systems are becoming popular due to easy availability of powerful personnel computers, large size memory devices, graphics software's etc. Image processing is a physical process used to convert an image signal into a physical image. The image signal can be either digital or analog. The actual output itself can be an actual physical image or the characteristics of an image.

Image processing is photography. In this process, an image is captured or scans using a camera to create a digital or analog image. In order to produce a physical picture, the image is processed using the appropriate technology based on the input source type. In digital photography, the image is stored as a computer file. This file is translated using photographic software to generate an actual image. The colors, shading, and nuances are all captured at the time the photograph is taken the software translates this information into an image. When creating images using analog photography, the image is burned into a film using a chemical reaction triggered by controlled exposure to light. The image is processed in a darkroom, using special chemicals to create the actual image. This process is decreasing in popularity due to the opening of digital photography, which requires less effort and special training to product images. The field of digital imaging has created a whole range of new applications and tools that were

previously impossible. Face recognition software, medical image processing and remote sensing are all possible due to the development of digital image processing. Specialized computer programs are used to enhance and correct images.

1.3 TYPES OF DIGITAL IMAGE

For photographic purposes, there are two important types of digital images: color and grayscale. Color images are made up of colored pixels while grayscale images are made of pixels in different shades of gray.

- **Grayscale Images:** A grayscale image is made up of pixels, each of which holds a single number corresponding to the gray level of the image at a particular location. These gray levels span the full range from black to white in a series of very fine steps, normally 256 different grays. Assuming 256 gray levels, each black and white pixel can be stored in a single byte (8 bits) of memory.
- **Color Images:** A color image is made up of pixels, each of which holds three numbers corresponding to the red, green and blue levels of the image at a particular location. Assuming 256 levels, each color pixel can be stored in three bytes (24 bits) of memory. Note that for images of the same size, a black & white version will use three times less memory than a color version.
- **Binary Images:** Binary images use only a single bit to represent each pixel. Since a bit can only exist in two states- ON or OFF; every pixel in a binary image must be one of two colors, usually black or white. This inability to represent intermediate shades of gray is what limits their usefulness in dealing with photographic images.

1.4 Color Model:

1.4.1 *RGB Color Model*

A representation of additive color mixing, projection of primary color lights on a screen shows secondary colors where two overlap; the combination of all three of red, green, and blue in appropriate intensities makes white.

The RGB color model is an additive color model in which red, green, and blue light is added together in various ways to reproduce a broad array of colors. The name of the model comes from the initials of the three additive primary colors, red, green, and blue. The main purpose of the RGB color model is for the sensing, representation, and display of images in electronic systems, such as televisions and computers, though it has also been used in conventional photography.

1.4.2 *Converting Color To Grayscale*

Conversion of a color image to grayscale is not unique; different weighting of the color channels effectively represents the effect of shooting black-and-white film with different-colored photographic filters on the cameras.

To convert any color to a grayscale representation of its luminance, first one must obtain the values of its red, green, and blue (RGB) primaries in linear intensity encoding, by gamma expansion. Then, add together 30% of the red value, 59% of the green value, and 11% of the blue value (these weights depend on the exact choice of the RGB primaries, but are typical). Regardless of the scale employed (0.0 to 1.0, 0 to 255, 0% to 100%, etc.) the resultant number is the desired linear luminance value; it typically needs to be gamma compressed to get back to a conventional grayscale representation.

To convert a gray intensity value to RGB, simply set all the three primary color components red, green and blue to the gray value, correcting to a different gamma if necessary.

1.5 DIGITAL IMAGE PROCESSING TECHNIQUES

The various Image Processing techniques are:

- Image enhancement
- Image segmentation

1.5.1 *Image Enhancement*

Sometimes images obtained from satellites and conventional and digital cameras lack in contrast and brightness because of the limitations of imaging sub systems and illumination conditions while capturing image. Images may have different types of noise. In image enhancement, the goal is to accentuate certain image features for subsequent analysis or for image display. Examples include contrast and edge enhancement, pseudo-coloring, noise filtering, sharpening, and magnifying. Image enhancement is useful in feature extraction, image analysis and an image display. The enhancement process itself does not increase the inherent information content in the data. It simply emphasizes certain specified image characteristics. Enhancement algorithms are generally interactive and application-dependent.

1.5.2 *Image Segmentation*

Image segmentation is the process that subdivides an image into its constituent parts or objects. The level to which this subdivision is carried out

depends on the problem being solved, i.e., the segmentation should stop when the objects of interest in an application have been isolated e.g., in autonomous air-to-ground target acquisition, suppose our interest lies in identifying vehicles on a road, the first step is to segment the road from the image and then to segment the contents of the road down to potential vehicles.

CHAPTER 2

LITERATURE SURVEY

2.1 Project Title: Dynamic Scene Classification: Learning Motion Descriptors with Slow Features Analysis

Author Name: Christian Theriault, Nicolas Thome, Matthieu Cord

Year Of Publish: 2013

Abstract

In this paper, we address the challenging problem of categorizing video sequences composed of dynamic natural scenes. Contrarily to previous methods that rely on handcrafted descriptors, we propose here to represent videos using unsupervised learning of motion features. Our method encompasses three main contributions: 1)

Based on the Slow Feature Analysis principle, we introduce a learned local motion descriptor which represents the principal and more stable motion components of training videos. 2) We integrate our local motion feature into a global coding/pooling architecture in order to provide an effective signature for each video sequence. 3) We report state of the art classification performances on two challenging natural scenes data sets. In particular, an outstanding improvement of 11% in classification score is reached on a data set introduced in 2012.

2.2 Project Title: Image Net Classification with Deep Convolutional Neural Networks

Author Name: Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton

Year Of Publish: 2013

Abstract

We trained a large, deep convolutional Neural network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The Neural network which has 60 million parameters and 650,000 neurons, consists of five convolution layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way soft max. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of the convolution operation. To reduce over fitting in the fully-connected layers we employed a recently-developed regularization method called “dropout” that proved to be very effective. We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.

2.3 Project Title: Two-Stream Convolution Networks for Action Recognition in Videos

Author Name: Karen Simonyan, Andrew Zisserman

Year Of Publish: 2015

Abstract

We investigate architectures of discriminatively trained deep Convolutional Networks (ConvNets) for action recognition in video. The challenge is to capture the complementary information on appearance from still frames and motion between frames. We also aim to generalize the best performing hand-crafted features within a data-driven learning framework. Our contribution is three-fold. First, we propose a two-stream ConvNet architecture which incorporates spatial and temporal networks. Second, we demonstrate that a ConvNet trained on multi-frame dense optical flow is able to achieve very good performance in spite of limited training data. Finally, we show that multitask learning, applied to two different action classification datasets, can be used to increase the amount of training data and improve the performance on both. Our architecture is trained and evaluated on the standard video actions benchmarks of UCF-101 and HMDB-51, where it is competitive with the state of the art. It also exceeds by a large margin previous attempts to use deep nets for video classification.

2.4 Project Title: Supervised Local Descriptor Learning for Human Action Recognition

Author Name: Xiantong Zhen, Feng Zheng, Ling Shao, Xianbin Cao, Dan Xu

Year Of Publish: 2017

Abstract

Local features have been widely used in computer vision tasks, e.g., human action recognition, but it tends to be an extremely challenging task to deal with large-scale local features of high dimensionality with redundant information. In this paper, we propose a novel fully supervised local descriptor learning algorithm called discriminative embedding method based on the image to- class distance

(I2CDDE) to learn compact but highly discriminative local feature descriptors for more accurate and efficient action recognition. By leveraging the advantages of the I2C distance, the proposed I2CDDE incorporates class labels to enable fully supervised learning of local feature descriptors, which achieves highly discriminative but compact local descriptors. The objective of our I2CDDE is to minimize the I2C distances from samples to their corresponding classes while maximizing the I2C distances to the other classes in the low-dimensional space. To further improve the performance, we propose incorporating a manifold regularization based on the graph Laplacian into the objective function, which can enhance the smoothness of the embedding by extracting the local intrinsic geometrical structure. The proposed I2CDDE for the first time achieves fully supervised learning of local feature descriptors. It significantly improves the performance of I2C-based methods by increasing the discriminative ability of local features while greatly reducing the computational burden by dimensionality reduction to handle large-scale data. We apply the proposed I2CDDE algorithm to human action recognition on four widely used benchmark datasets. The results have shown that I2CDDE can significantly improve I2C-based classifiers and achieves state-of-the-art performance.

2.5 Project Title: Recognizing Actions by Shape-Motion Prototype Trees

Author Name: Zhe Lin, Zhuolin Jiang, Larry S. Davis

Year Of Publish: 2010

Abstract

A prototype-based approach is introduced for action recognition. The approach represents an action as a sequence of prototypes for efficient and flexible action matching in long video sequences. During training, first, an action prototype tree is learned in a joint shape and motion space via hierarchical k-means clustering;

then a lookup table of prototype-to-prototype distances is generated. During testing, based on a joint likelihood model of the actor location and action prototype, the actor is tracked while a frame-to-prototype correspondence is established by maximizing the joint likelihood, which is efficiently performed by searching the learned prototype tree; then actions are recognized using dynamic prototype sequence matching. Distance matrices used for sequence matching are rapidly obtained by look-up table indexing, which is an order of magnitude faster than brute-force computation of frame-to-frame distances. Our approach enables robust action matching in very challenging situations (such as moving cameras, dynamic backgrounds) and allows automatic alignment of action sequences. Experimental results demonstrate that our approach achieves recognition rates of 91.07% on a large gesture dataset (with dynamic backgrounds), 100% on the Weizmann action dataset and 95.77% on the KTH action dataset.

2.6 Project Title: Optimizing Kernel Machines using Deep Learning

Author Name: Huan Song, Jayaraman J. Thiagarajan, Prasanna Sattigeri, Andreas Spanias

Year Of Publish: 2017

Abstract

Building highly non-linear and non-parametric models is central to several state-of-the-art machine learning systems. Kernel methods form an important class of techniques that induce a reproducing kernel Hilbert space (RKHS) for inferring non-linear models through the construction of similarity functions from data. These methods are particularly preferred in cases where the training data sizes are limited and when prior knowledge of the data similarities is available.

Despite their usefulness, they are limited by the computational complexity and their inability to support end-to-end learning with a task specific objective. On the other hand, deep Neural networks have become the de facto solution for end-to-end inference in several learning paradigms. In this article, we explore the idea of using deep architectures to perform kernel machine optimization, for both computational efficiency and end-to-end inference. To this end, we develop the DKMO (Deep Kernel Machine Optimization) framework, that creates an ensemble of dense embeddings using Nyström kernel approximations and utilizes deep learning to generate task-specific representations through the fusion of the embeddings. Intuitively, the filters of the network are trained to fuse information from an ensemble of linear subspaces in the RKHS. Furthermore, we introduce the kernel dropout regularization to enable improved training convergence. Finally, we extend this framework to the multiple kernel case, by coupling a global fusion layer with pre-trained deep kernel machines for each of the constituent kernels. Using case studies with limited training data, and lack of explicit feature sources, we demonstrate the effectiveness of our framework over conventional model inferencing techniques.

2.7 Project Title: Dynamic Scene Classification Using Convolutional Neural Networks

Author Name: Aalok Gangopadhyay, Shivam Mani Tripathi, Ishan Jindal and Shanmuganathan Raman

Year of Publish: 2016

Abstract

The task of classifying videos of natural dynamic scenes into appropriate classes has gained a lot of attention in recent years. The problem especially becomes challenging when the camera used to capture the video is dynamic. In this paper, we analyse the performance of statistical aggregation (SA) techniques on various

pre-trained convolution Neural network (CNN) models to address this problem. The proposed approach works by extracting CNN activation features for a number of frames in a video and then uses an aggregation scheme in order to obtain a robust feature descriptor for the video. We show through results that the proposed approach performs better than the-state-of-the art works for the Maryland and YUPenn dataset. The final descriptor obtained is powerful enough to distinguish among dynamic scenes and is even capable of addressing the scenario where the camera motion is dominant and the scene dynamics are complex. Further, this paper shows an extensive study on the performance of various aggregation methods and their combinations. We compare the proposed approach with other dynamic scene classification algorithms on two publicly available datasets Maryland and UPenn to demonstrate the superior performance of the proposed approach.

2.8 Project Title: Hybrid Deep Learning for Face Verification

Author Name: Yi Sun Xiaogang Wang, Xiaoou Tang

Year Of Publish: 2013

Abstract

This paper proposes a hybrid convolutional network (ConvNet)-Restricted Boltzmann Machine (RBM) model for face verification in wild conditions. A key contribution of this work is to directly learn relational visual features, which indicate identity similarities, from raw pixels of face pairs with a hybrid deep network. The deep ConvNets in our model mimic the primary visual cortex to jointly extract local relational visual features from two face images compared with the learned filter pairs. These relational features are further processed through multiple layers to extract high-level and global features.

Multiple groups of ConvNets are constructed in order to achieve robustness and characterize face similarities from different aspects. The top-layer RBM performs inference from complementary high-level features extracted from different

ConvNet groups with a two-level average pooling hierarchy. The entire hybrid deep network is jointly fine-tuned to optimize for the task of face verification. Our model achieves competitive face verification performance on the LFW dataset.

2.9 Title Name: Temporal Segment Networks: Towards Good Practices for Deep Action Recognition

Author Name: Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin,

Xiaoou Tang, and Luc Van Gool

Year Of Publish: 2016

Abstract

Deep convolutional networks have achieved great success for visual recognition in still images. However, for action recognition in videos, the advantage over traditional methods is not so evident. This paper aims to discover the principles to design effective ConvNet architectures for action recognition in videos and learn these models given limited training samples. Our first contribution is temporal segment network (TSN), a novel framework for video-based action recognition. Which is based on the idea of long-range temporal structure modeling? It combines a sparse temporal sampling strategy and video-level supervision to enable efficient and effective learning using the whole action video. The other contribution is our study on a series of good practices in learning ConvNets on video data with the help of temporal segment network. Our approach obtains the state-the-of-art performance on the datasets of HMDB51 (69:4%) and UCF101 (94:2%). We also visualize the learned ConvNet models, which qualitatively demonstrate the effectiveness of temporal segment network and the proposed good practices.

2.10 Title Name: Joint Binary Classifier Learning for ECOC-Based Multi-Class Classification

Author Name: Mingxia Liu, Daoqiang Zhang, Songcan Chen, and Hui Xue

Year Of Publish: 2016

Abstract

Error-correcting output coding (ECOC) is one of the most widely used strategies for dealing with multi-class problems by decomposing the original multi-class problem into a series of binary sub-problems. In traditional ECOC-based methods, binary classifiers corresponding to those sub problems are usually trained separately without considering the relationships among these classifiers. However, as these classifiers are established on the same training data, there may be some inherent relationships among them. Exploiting such relationships can potentially improve the generalization performances of individual classifiers, and, thus, boost ECOC learning algorithms. In this paper, we explore to mine and utilize such relationship through a joint classifier learning method, by integrating the training of binary classifiers and the learning of the relationship among them into a unified objective function. We also develop an efficient alternating optimization algorithm to solve the objective function. To evaluate the proposed method, we perform a series of experiments on eleven datasets from the UCI machine learning repository as well as two datasets from real-world image recognition tasks. The experimental results demonstrate the efficacy of the proposed method, compared with state-of-the-art methods for ECOC-based multi-class classification.

CHAPTER 3

METHDOLOGY

3.1 EXISTING SYSTEM

The convolution 3-D (C3-D) and VGG (vision and graphics group) are first deployed to extract temporal and spatial features from the input videos cooperatively, which establishes comprehensive and informative representations of videos. VGG and C3-D are chosen due to their strong capability of extracting complementary spatial and temporal features for comprehensive video representations. The resultant high-dimensional representations are further reduced by random projections into a set of lower dimensional subspaces, on which an ensemble of efficient classifiers is trained with these lower dimensional features. The introduced RLE layer is further deployed to encode the initial outputs of classifier, which is followed by a weighting layer jointly learned in the end-to-end framework to combine classification results.

3.2 PROPOSED SYSTEM

In this proposed system, we propose the NEURAL network method for action recognition in video. The input video is converted into number of frames. The converted frames will be stored in the database. The frame is feed into the feature extraction (in order to extract the feature from the video frames). Then the ROI (Region of interest) segmentation algorithm is used in order to detect the particular part of the frame. Finally, the Neural network classifier is used to detect the given input video is abnormal or not.

3.3 MODULE DESCRIPTION

Module I: Video to image conversion

This module consists of Matlab executable commands useful for extracting the image frames from the input database video. Here the input video is converted into frames. The location path will be saved in the code itself.

For example 1 sec video it will be converted into 15 frames per sec. Then the converted frames from the video will be stored in the separate folder. From the frame conversion images the feature extraction is used to extract the image features & parameters helpful for analyzing the image overview

Module II: Segmentation

The ROI (Region of interest) segmentation algorithm is used in order to detect the particular part of the frame. A region of interest (ROI) is a portion of an image that you want to filter or perform the operation. A region of interest (ROI) is a portion of an image that we want to filter or perform some other operation on. We define an ROI by creating a binary mask, which is a binary image that is the same size as the image you want to process with pixels that define the ROI set to 1 and all other pixels set to 0.

Filling is a process that fills a region of interest (ROI) by interpolating the pixel values from the borders of the region. This process can be used to make objects in an image seem to disappear as they are replaced with values that blend in with the background area. To fill an ROI, the `roifill` function method is used. This function is useful for image editing, including removal of extraneous details or artifacts.

To create binary masks to define ROIs. However, any binary image can be used as a mask, provided that the binary image is the same size as the image being filtered. For example, suppose we want to filter the grayscale image `I`, filtering only those pixels whose values are greater than 0.5. When creating a region of interest (ROI) mask, `poly2mask` must determine which pixels are included in the region. This determination can be difficult when pixels on the edge of a region are only partially covered by the border line. Finally, we get the image

was what we need the particular portion frame in color and then remaining background image will be in black.

Module III: NEURAL network

This module consists of NEURAL Network with adaptive learning for analyzing the video frames. Then the maximum weight values are taken from the feature extraction frames by using the deep network.

The most common type of Neural network consists of three groups, or layers, of units: a layer of "input" units is connected to a layer of "hidden" units, which is connected to a layer of "output" units. The activity of the input units represents the raw information that is fed into the network. The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units.

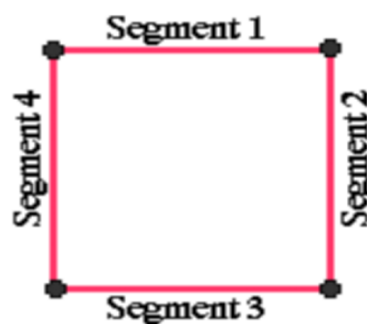
The behavior of the output units depends on the activity of the hidden units and the weights between the hidden and output units. This simple type of network is interesting because the hidden units are free to construct their own representations of the input. The weights between the input and hidden units determine when each hidden unit is active, and so by modifying these weights, a hidden unit can choose what it represents. According to video labels the entire network is trained for action recognition. Finally, the output will check the given input video whether it is abnormal or not.

3.4 REGION OF INTEREST (ROI) SEGMENTATION

In computer vision, image segmentation is the process of partitioning a digital image into multiple segments. The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and

easier to analyze. Image segmentation is typically used to locate objects and boundaries in images. More precisely image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics.

The result of image segmentation is a set of segments that collectively cover the entire image, or a set of contours extracted from the image. Each of the pixels in a region are similar with respect to some characteristic or computed property, such as color, intensity, or texture.



For ROIs that enclose an area, each ROI is defined by the (closed path) outline that encloses the region. For example, in the case of a Rectangular ROI, the outline consists of four-line segments joining the four corner points:

To make the computation of ROI statistics feasible, all line segments for an ROI must be straight. So, with Irregular ROIs, what may appear to be a curved outline of the region is actually made up of many short straight-line segments. This can be seen by zooming into a portion of the Irregular ROI outline.

Spline and Elliptical ROIs present a particular issue, since calculation of intensity statistics with a truly curved outline would be problematic and very slow. Calculation of intensity statistics for Spline and Elliptical ROIs is therefore done by representing the outline by a series of straight-line segments. For Elliptical ROIs, if the outline is represented by n straight line segments, then the error between the true area of the ellipse and the approximated area is:

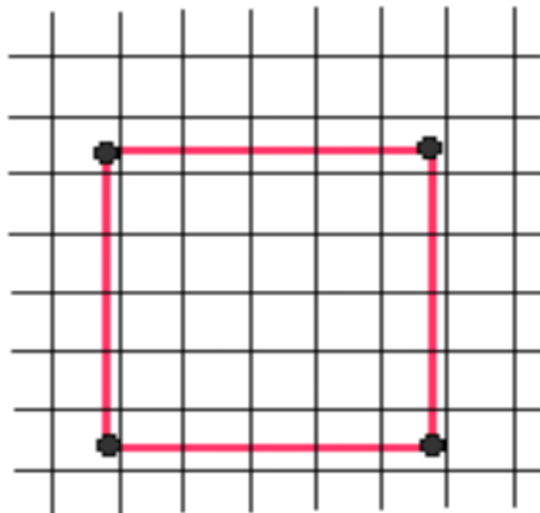
$$(\pi - (n * \sin(\pi/n) * \cos(\pi/n))) \times 100\% / \pi$$

In **Jim** versions up to and including 8.0, n is set to 360 (i.e., for a circular ellipse, each line segment subtends an angle of 1° at the center), which leads to a difference of 0.005% between the true ellipse area and the area of the ellipse represented by straight line segments. Computation of intensity statistics for ROIs involves multiplying pixel intensities by the area of pixels that lie with the outline of the ROI. Given a random distribution of pixel intensities with an Elliptical ROI, we therefore expect an error of the order of 0.005% for intensity statistics.

Note: the area statistic returned for an Elliptical ROI is always the true area of the ellipse ($= \pi \times \text{semi-major-axis} \times \text{semi-minor-axis}$).

For Spline ROIs, the interpolated spline shape is divided into 1000 straight-line segments, and the statistics returned are equivalent to those of a polygonal shape with these segments.

The vertices of an ROI outline may be positioned anywhere with respect to the array of image pixels, so the same Rectangular ROI shown above superimposed on the pixel array may appear like this:



The computation of area and intensity statistics for a generalized ROI composed of straight-line segments goes as follows.

1. Each row of the image is considered in turn.

2. If the row intersects the ROI anywhere, then each pixel in that image row is considered in turn.
3. Each pixel is intersected with the ROI.
4. The total ROI area is the sum of the areas of all the intersected shapes.

For Text, Marker and Line ROIs, the area is zero. Let I_i be the intensity

of pixel i with non-zero intersected area; let a_i be the intersected area of

pixel i ; and let A be the total area of all the intersected shapes (i.e.,

$$\sum(a_i)).$$

5. The mean pixel intensity is

$$\sum(a_i \times I_i) / A.$$

For a Text or Marker ROI, the mean pixel intensity is the intensity of the pixel at the centre of the text or beneath the marker. Let μ be the mean pixel intensity.

6. The standard deviation in pixel intensity is:

$$\sqrt{((\sum(a_i \times I_i \times I_i) - \mu \times \mu \times A) / A)}.$$

The exception to this is for Text and Marker ROIs, where the standard deviation in pixel intensity is always zero.

7. Minimum pixel intensity is the minimum intensity in any intersected shape with an area greater than zero. The exception to this is for Text and Marker ROIs, where the minimum pixel intensity is the intensity of the pixel at the centre of the text or beneath the marker.

8. Maximum pixel intensity is the maximum intensity in any intersected shape with an area greater than zero. The exception to this is for text and Marker ROIs, where the maximum pixel intensity is the intensity of the pixel at the centre of the text or beneath the marker.

The algorithm for Line ROIs is similar to that described above. However, Line ROIs do not enclose any area, and so in the description above substitute "length" for "area".

3.5 NEURAL NETWORK

A Neural Network (NN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain,

process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. NNs, like people, learn by example. An NN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons.

3.5.1 *Historical Background*

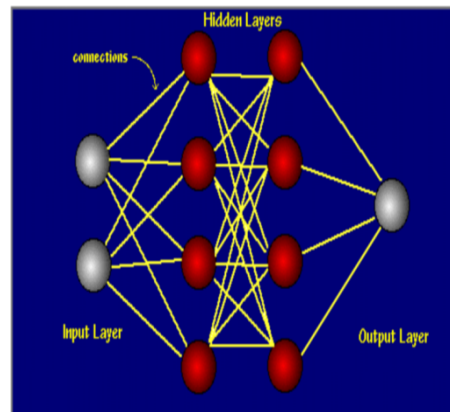
Neural network simulations appear to be a recent development. However, this field was established before the advent of computers, and has survived at least one major setback and several eras.

Many important advances have been boosted by the use of inexpensive computer emulations. Following an initial period of enthusiasm, the field survived a period of frustration and disrepute. During this period when funding and professional support was minimal, important advances were made by relatively few researchers. These pioneers were able to develop convincing technology which surpassed the limitations identified by Minsky and Papert. Minsky and Papert, published a book (in 1969) in which they summed up a general feeling of

frustration (against Neural networks) among researchers, and was thus accepted by most without further analysis. Currently, the Neural network field enjoys a resurgence of interest and a corresponding increase in funding

3.5.2 The Basics of Neural Networks

Neural networks are typically organized in layers. Layers are made up of a number of interconnected 'nodes' which contain an 'activation function'. Patterns are presented to the network via the 'input layer', which communicates to one or more 'hidden layers' where the actual processing is done via a system of weighted 'connections'. The hidden layers then link to an 'output layer' where the answer is output as shown in the graphic below.



Use of Neural networks:

Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained Neural network can be thought of as an "expert" in the category of information it has been given to analyze. This expert can then be used to provide projections given new situations of interest and answer "what if" questions.

Other advantages include:

1. Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience.
2. Self-Organization: An NN can create its own organization or representation of the information it receives during learning time.
3. Real Time Operation: NN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability.
4. Fault Tolerance via Redundant Information Coding: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.

3.5.3 Neural Networks Versus Conventional Computers

Neural networks take a different approach to problem solving than that of conventional computers. Conventional computers use an algorithmic approach i.e. the computer follows a set of instructions in order to solve a problem. Solve the problem. That restricts the problem-solving capability of conventional computers to problems that we already understand and know how to solve. But computers would be so much more unless the specific steps that the computer needs to follow are known the computer cannot be useful if they could do things that we don't exactly know how to do.

Neural networks process information in a similar way the human brain does. The network is composed of a large number of highly interconnected processing elements (neurons) working in parallel to solve a specific problem. Neural networks learn by example. They cannot be programmed to perform a specific task. The examples must be selected carefully otherwise useful time is wasted or even worse the network might be functioning incorrectly. The disadvantage is that

because the network finds out how to solve the problem by itself, its operation can be unpredictable.

On the other hand, conventional computers use a cognitive approach to problem solving; the way the problem is to be solved must be known and stated in small unambiguous instructions. These instructions are then converted to a high level language program and then into machine code that the computer can understand. These machines are totally predictable; if anything goes wrong is due to a software or hardware fault.

Neural networks and conventional algorithmic computers are not in competition but complement each other. There are tasks more suited to an algorithmic approach like arithmetic operations and tasks that are more suited to Neural networks. Even more, a large number of tasks, require systems that use a combination of the two approaches (normally a conventional computer is used to supervise the Neural network) in order to perform at maximum efficiency.

3.5.4 Network Layers

The commonest type of artificial Neural network consists of three groups, or layers, of units: a layer of **"input"** units is connected to a layer of **"hidden"** units, which is connected to a layer of **"output"** units.

The activity of the input units represents the raw information that is fed into the network.

The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units.

The behavior of the output units depends on the activity of the hidden units and the weights between the hidden and output units.

This simple type of network is interesting because the hidden units are free to construct their own representations of the input. The weights between the input

and hidden units determine when each hidden unit is active, and so by modifying these weights, a hidden unit can choose what it represents.

We also distinguish single-layer and multi-layer architectures. The single-layer organization, in which all units are connected to one another, constitutes the most general case and is of more potential computational power than hierarchically structured multi-layer organizations. In multi-layer networks, units are often numbered by layer, instead of following a global numbering.

3.6 SYSTEM REQUIREMENTS

3.6.1 *Hardware Requirements*

Ø Hard Disk	:	40GB and Above
Ø RAM	:	512MB and Above
Ø Processor	:	Pentium III and Above

3.6.2 *Software Requirements*

Ø MATLAB R2018a

Ø Windows Operating System XP and Above

3.7 MATLAB

MATLAB (**matrix laboratory**) is a multi-paradigm numerical computing environment and fourth-generation programming language. A proprietary programming language developed by Math Works, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, Fortran and Python.

Although MATLAB is intended primarily for numerical computing, an optional toolbox uses the Mu PAD symbolic engine, allowing access to symbolic computing abilities. An additional package, Simulink, adds graphical multi-domain simulation and model-based design for dynamic and embedded systems.

The basic data structure in MATLAB is the array, an ordered set of real or complex elements. This object is naturally suited to the representation of images, real-valued ordered sets of color or intensity data.

MATLAB stores most images as two-dimensional arrays (i.e., matrices), in which each element of the matrix corresponds to a single pixel in the displayed image. (Pixel is derived from picture element and usually denotes a single dot on a computer display.) For example, an image composed of 200 rows and 300 columns of different coloured dots would be stored in MATLAB as a 200-by-300 matrix. Some images, such as RGB, require a three-dimensional array, where the first plane in the third dimension represents the red pixel intensities, the second plane represents the green pixel intensities, and the third plane represents the blue pixel intensities.

This convention makes working with images in MATLAB similar to working with any other type of matrix data, and makes the full power of MATLAB available for image processing applications. For example, you can select a single pixel from an image matrix using normal matrix subscripting.

3.8 INTRODUCTION TO MATLAB R2018a:

3.8.1 The MATLAB System

The MATLAB system consists of these main parts:

1. Desktop Tools and Development Environment

This is the set of tools and facilities that help you use and become more productive with MATLAB functions and files. Many of these tools are graphical user interfaces. It includes the MATLAB desktop and Command Window, a command history, an editor and debugger, a code analyzer and other reports, and browsers for viewing help, the workspace, files, and the search path.

2. Mathematical Function Library

This is a vast collection of computational algorithms ranging from elementary functions, like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix Eigen values, Bessel functions, and fast Fourier transforms.

3. The Language

This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create large and complex application programs.

4. Graphics

MATLAB has extensive facilities for displaying vectors and matrices as graphs, as well as annotating and printing these graphs. It includes high-level functions for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level functions that allow you to fully customize the appearance of graphics as well as to build complete graphical user interfaces on your MATLAB applications.

5. External Interfaces

This is a library that allows you to write C and FORTRAN programs that interact with MATLAB. It includes facilities for calling routines from MATLAB (dynamic linking), for calling MATLAB as a computational engine, and for reading and writing MAT-files.

6. Image Processing Toolbox

The Image Processing Toolbox is a collection of functions that extend the capability of the MATLAB numeric computing environment. The toolbox supports a wide range of image processing operations, including:

- Spatial image transformations
- Morphological operations
- Neighborhood and block operations
- Linear filtering and filter design
- Transforms
- Image analysis and enhancement
- Image registration
- De blurring
- Region of interest operations

Many of the toolbox functions are MATLAB M-files, a series of MATLAB statements that implement specialized image processing algorithms. You can extend the capabilities of the Image Processing Toolbox by writing your own M-files, or by using the toolbox in combination with other toolboxes, such as the Signal Processing Toolbox and the Wavelet Toolbox.

3.8.2 Computer Vision Toolbox

The field is of importance for such various applications as autonomous vehicles, navigating with the help of images, captured by a mounted camera, and high precision measurements using images, taken by calibrated cameras. In this paper, we will present a number of numerical routines, implemented in MATLAB, that are useful in a variety of computer vision applications. The collection of routines will be called the Computer Vision Toolbox. One of the main problems in Computer Vision is to calculate the 3D-structure of the scene and the motion of the camera from measurements in the images taken from different view-points in the scene. This problem is called structure and motion, referring to the fact that both the structure of the scene and the motion of the camera are calculated from image measurements only. A number of different sub problems, arising from different knowledge of the intrinsic properties of the camera appear. Other important problems are to calculate the structure of the 3D-scene given the motion of the camera and to calculate the motion of the camera given the structure of the scene. These problems are somewhat simpler than the general structure and motion problem, but are nevertheless important for navigation and obstacle avoidance. A related problem is to calibrate a camera, i.e. calculate the focal distance, the principal point etc., from images of a known scene. This calibration may be used to make a more precise reconstruction. The problem of estimating structure and motion from image sequences are closely related to the field of closed range photogrammetric. In this field images are used to make precise measurements of three-dimensional objects from a number of images, often taken by calibrated cameras.

The main difference is that in computer vision, the focus is not on accuracy, but instead on reliability and speed of calculation. In all these problems different kind of so called features can be used.

The simplest feature is points that are easily detected in the sequence, such as corners or painted marks. Other types of features are lines and curves, which are easier to detect and track but more difficult to use in structure and motion algorithms. In order to use our computer vision routines these features

must be detected in advance and the correspondence between different images must be established. The main emphasis of our computer vision toolbox is to use these detected features to solve for structure and motion.

3.8.3 Image Acquisition Toolbox

The Image Acquisition Toolbox's is a collection of functions that extend the capability of the MATLAB numeric computing environment. The toolbox supports a wide range of image acquisition operations including

- Acquiring images through many types of image acquisition devices, from professional grade frame grabbers to USB-based Webcams
- Viewing a preview of the live video stream
- Triggering acquisitions (includes external hardware triggers)
- Configuring call back functions that execute when certain events occur
- Bringing the image data into the MATLAB workspace

Many of the toolbox functions are MATLAB M-files. You can extend the capabilities of the Image Acquisition Toolbox by writing your own M-files, or by using the toolbox in combination with other toolboxes, such as the Image Processing Toolbox and the Data Acquisition Toolbox. The toolbox also includes a Simulink interface called the Image Acquisition Block set. This block set extends Simulink with a block that lets you bring live video data into a model.

3.9 System Architecture

3.9.1 Architecture

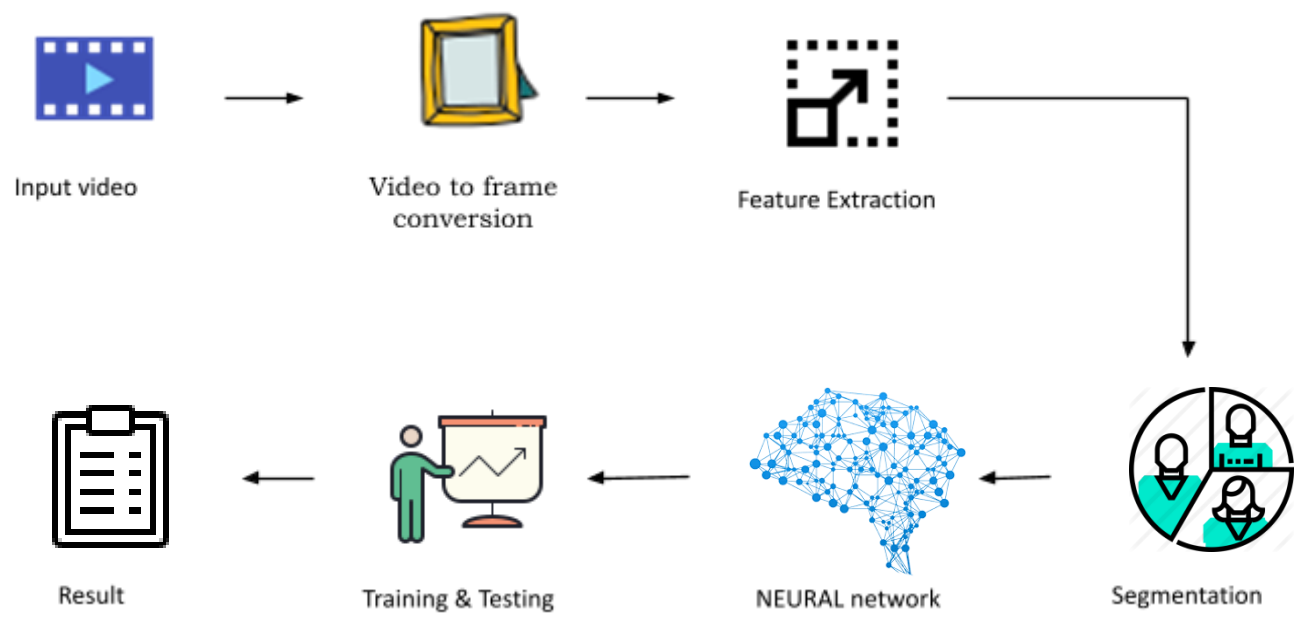
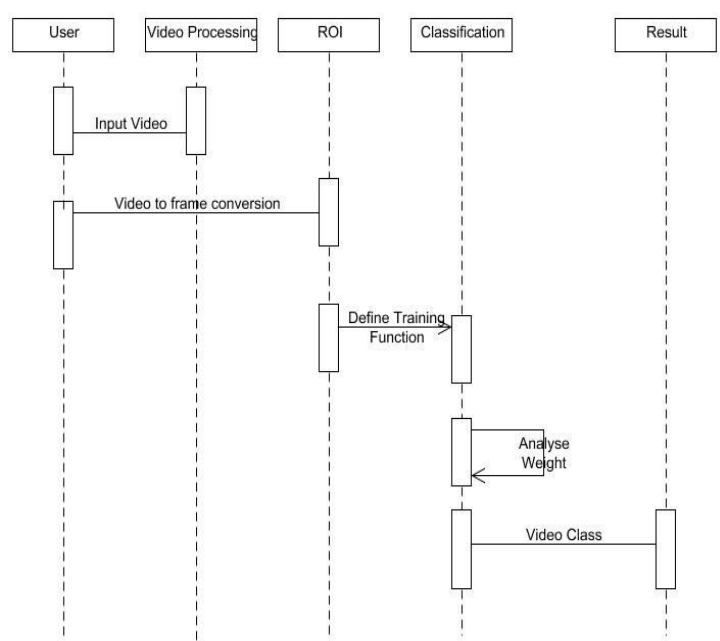


Fig 3.1 System Architecture

3.9.2 Sequence



3.9.3 Activity Diagram

Fig 3.2 Sequence Diagram

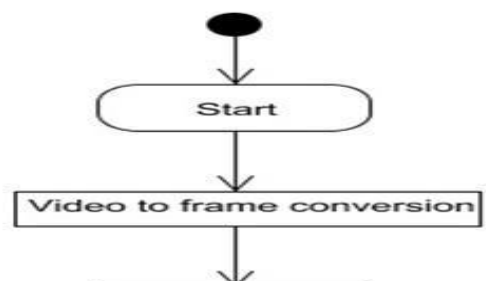


Fig 3.3 Activity Diagram

3.9.4 Use Case Diagram

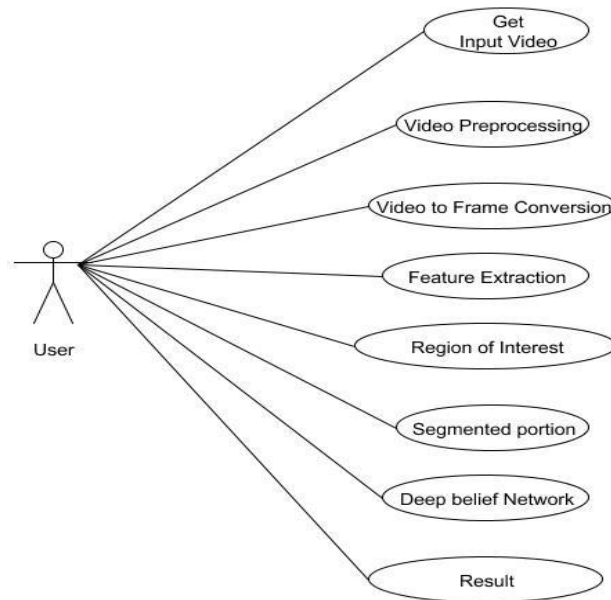


Fig 3.4 Use Case Diagram

3.9.5 ER Diagram

Step 1

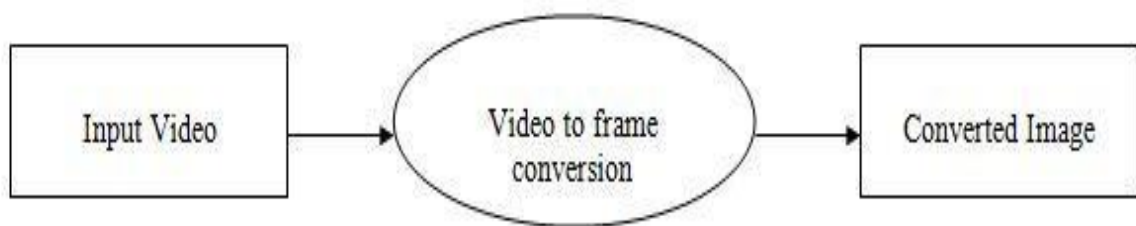


Fig 3.5 Converting Image

Step 2

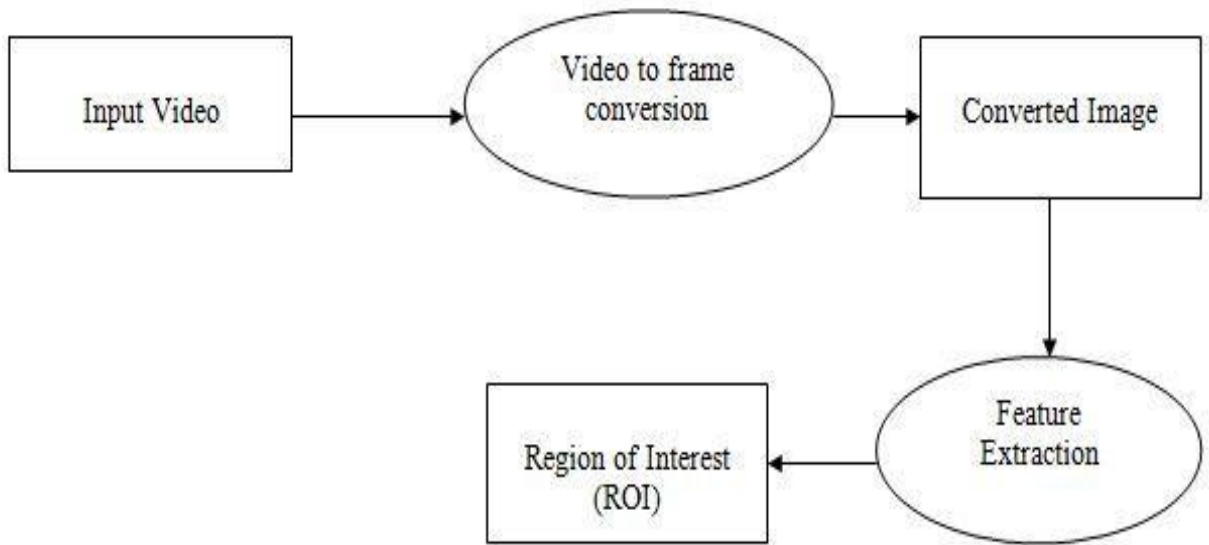


Fig 3.6 Getting ROI from Input Video

Step 3

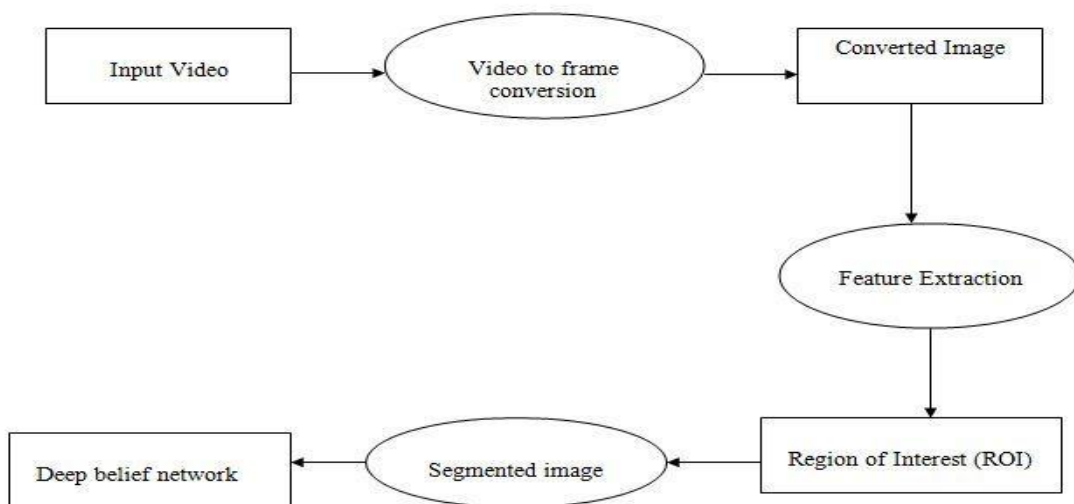


Fig 3.7 NEURAL network from segmented image

Step 4

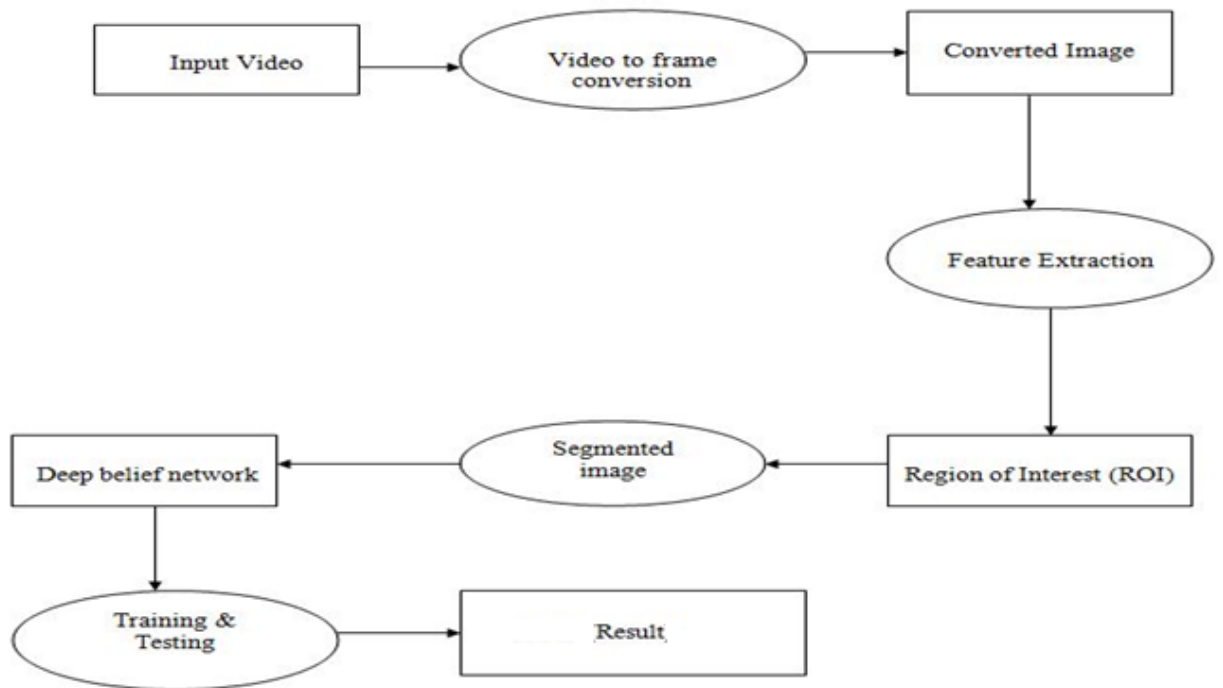


Fig 3.8 Final Result from Input Video

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 RESULTS

We have developed video classification recognition based NEURAL networks. The first step will be a frame collection which can be collected using real time video streaming. As the video gets recorded, action can be predicted. After this all frame collection an image is generated. Then the ROI (Region Of Interest) segmentation algorithm is used in order to detect the particular part of the frame. On the basis of the segmentation, the NEURAL network classifier is used to detect if the given input video is abnormal or not. The recognition of action is accomplished by a NEURAL network classifier. The performance of our method is

evaluated on frames of many actions performed. The experimental results show that the approach performs well for the real time applications.

4.2 SCREENSHOTS

4.2.1 *Frame Conversion Image*

Here the input video is converted into frames. The feature extraction is used to extract the image features & parameters helpful for analyzing the image overview.

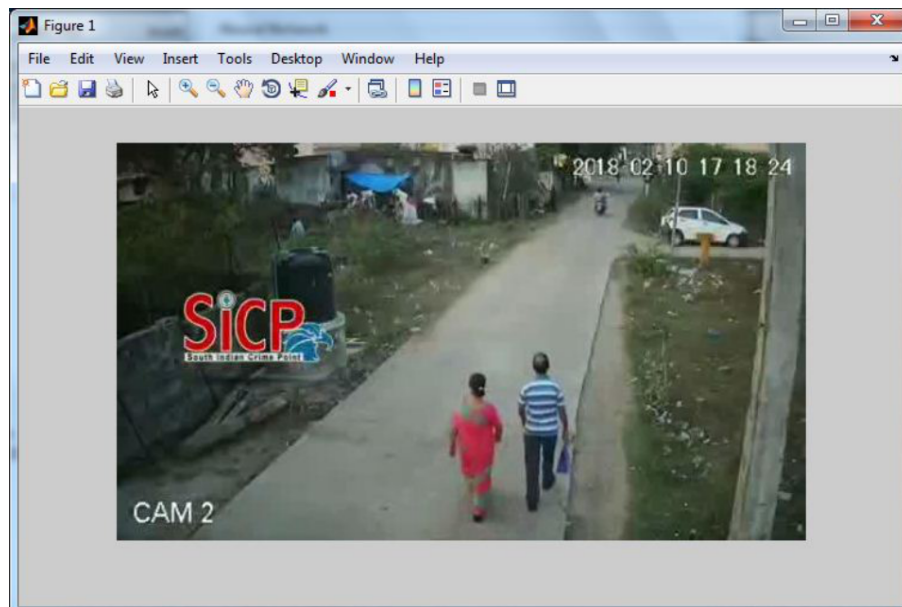


Fig 4.1 Frame Conversion Image

4.2.2 *ROI*

The ROI (Region of Interest) segmentation algorithm is used in order to detect the particular part of the frame.

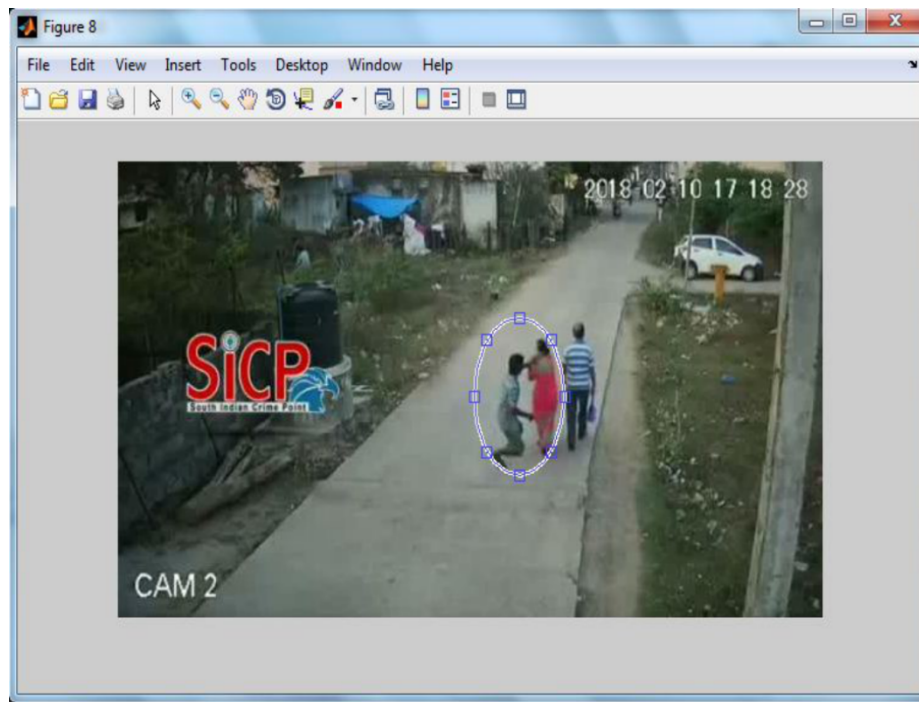


Fig 4.2 ROI

4.2.3 Segmentation

The segmentation algorithm is used in order to detect the particular part of the frame and portion of an image that you want to filter or perform the operation.

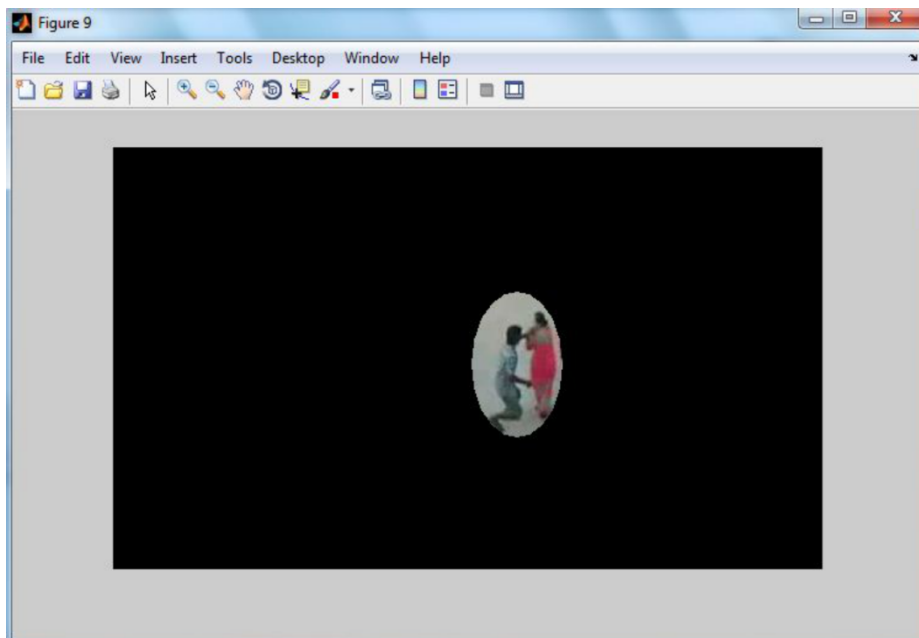


Fig 4.3 Segmentation

4.2.4 NEURAL Network

NEURAL Network is an adaptive learning for analysing the video frames. Then the maximum weight values are taken from the feature extraction frames using the deep network.

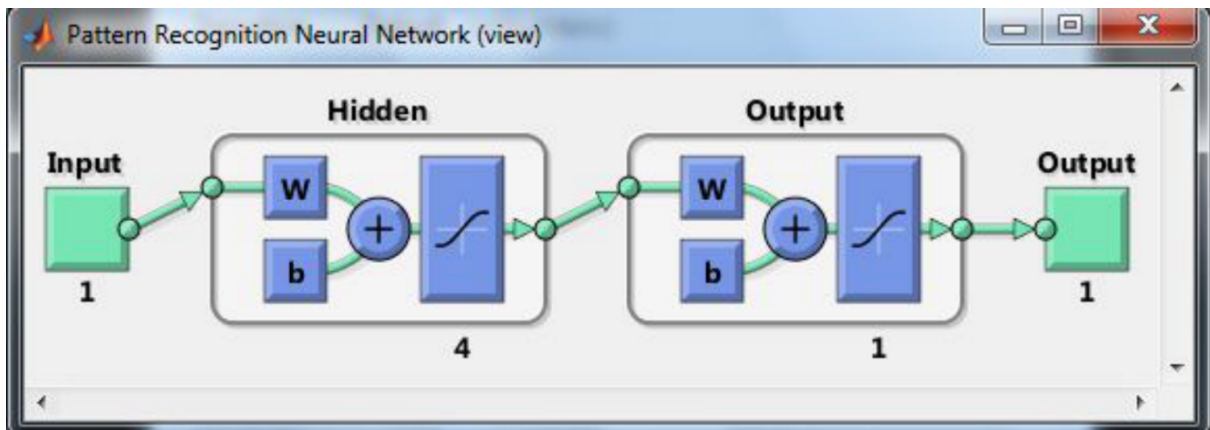


Fig 4.4 NEURAL Network

4.3 DISCUSSION

The performance of the proposed method highly depends on the result of action recognition. However, deep learning algorithms can discriminate the action from the background. Videos can provide in-depth information that can improve the performance of action recognition. Deep learning methods and videos may be used to address the complex background problem and improve the robustness of action recognition. Computers have now become an integral part of our lives and hence their usage should be as trouble free as talking to someone is. But now attempts are being made to make the man machine interaction as natural as possible. Multiple actions can detect and control the appliances from one place.

CHAPTER 5

CONCLUSION AND FUTURE ENHANCEMENTS

5.1 CONCLUSION

In this paper, we propose a Region of Interest (ROI) for video action recognition, which consists of NEURAL network. This architecture is designed according to human visual mechanism in the purpose of obtaining strong representational power for prediction just using RGB data, without optical flow data which needs additional computations. We propose the NEURAL network method for action recognition in video. The input video is converted into number of frames. The converted frames will be stored in the database. Then the ROI (Region of interest) segmentation algorithm is used in order to detect the particular part of the frame. Finally the NEURAL network classifier is used to detect the given input video is abnormal or normal.

5.2 FUTURE ENHANCEMENTS

Future work includes applying the proposed system on different complex datasets such as: sports and real actions ones. These datasets are more complex than the ones used here and the system may need some improvement to achieve acceptable recognition rate. Also the use of a sequence of different actions to segment it then recognize each action is another point of research in the future scope.

REFERENCES

- [1] Z. Lin, Z. Jiang, and L. S. Davis, “Recognizing Actions By Shape Motion Prototype Trees,” in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 444–451.
- [2] K. Simonyan and A. Zisserman, “Two-Stream Convolutional Networks For Action Recognition In Videos,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 568–576.
- [3] B. Zhang *et al.*, “One-Two-One Networks For Compression Artifacts Reduction In Remote Sensing,” *ISPRS J. Photogramm. Remote Sens.*, to be published, doi: 10.1016/j.isprsjprs.2018.01.003.
- [4] L. Shao, X. Zhen, D. Tao, and X. Li, “Spatio-Temporal Laplacian Pyramid Coding For Action Recognition,” *IEEE Trans. Cybern.*, vol. 44, no. 6, pp. 817–827, Jun. 2014.
- [5] C. Thériault, N. Thome, and M. Cord, “Dynamic Scene Classification: Learning Motion Descriptors With Slow Features Analysis,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2603–2610.
- [6] A. Gangopadhyay, S. M. Tripathi, I. Jindal, and S. Raman, “Dynamic Scene Classification Using Convolutional Neural Networks,” in *Proc. IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, Dec. 2016, pp. 1255–1259.
- [7] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, “Learning Spatiotemporal Features With 3D Convolutional Networks,” in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 4489–4497.

- [8] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional Two-Stream Network Fusion for Video Action Recognition," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 1933–1941.
- [9] L. Wang *et al.*, "Temporal Segment Networks: Towards Good Practices For Deep Action Recognition," in *Proc. Eur. Conf. Comput. Vis.* Berlin, Germany: Springer, 2016, pp. 20–36.
- [10] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond Short Snippets: Deep Networks for Video Classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 4694–4702.
- [11] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-Scale Video Classification With Convolutional Neural Networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1725–1732.
- [12] Y. Sun, X. Wang, and X. Tang, "Hybrid Deep Learning For Face Verification," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1489–1496.
- [13] G. Hinton, "A Practical Guide To Training Restricted Boltzmann Machines," *Momentum*, vol. 9, no. 1, p. 926, 2012.
- [14] H. Song, J. J. Thiagarajan, P. Sattigeri, and A. Spanias, "Optimizing Kernel Machines Using Deep Learning," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published, doi: 10.1109/TNNLS.2018.2804895.

APPENDIX

Source Code

```
clc;
clear all;
close all;

%% Video to frame conversion %%

[filenamefp, pathnamefp]=uigetfile('*.avi*');
fp=strcat(pathnamefp , filenamefp);
readerobj = VideoReader(fp);
vidFrames = read(readerobj);
numFrames = get(readerobj, 'numberOfFrames');

for t = 1 : numFrames
currFrame = read(readerobj, t);
numFramesWritten = 0;
opBaseFileName = sprintf('%3.3d.jpg', t);
opFullFileName = fullfile('D:\Deep ensemble learning\frames', opBaseFileName);
imwrite(currFrame, opFullFileName, 'jpg');
progIndication = sprintf('Wrote frame %4d of %d.', t, numFrames);
disp(progIndication);
numFramesWritten = numFramesWritten + 1;
end

progIndication = sprintf('Wrote %d frames to folder
"%s"',numFramesWritten,'D:\Deep ensemble learning\frames');
disp(progIndication);

I=imread('D:\Deep ensemble learning\frames\024.jpg');
```

```

figure,imshow(I);

%% ROI Selection %%

if('video 1.avi'==filenamefp)
    S1=Wakeboarding_ROI(I);

elseif('video 2.avi'==filenamefp)
    S2=Swimming_ROI(I);

elseif('video 3.avi'==filenamefp)
    S3=Jogging_ROI(I);

elseif('video 4.avi'==filenamefp)
    S4=Skipping_ROI(I);

elseif('video 5.avi'==filenamefp)
    S5=Riding_ROI(I);

elseif('video 6.avi'==filenamefp)
    S4=Class_ROI(I);

elseif('video 8.avi'==filenamefp)
    S4=Fighting_ROI(I);

end

% %% NEURAL network

% hiddenLayerSize = 2;
% net = patternnet(hiddenLayerSize);

% net.inputs{1}.processFcns = {'removeconstantrows','mapminmax'};
% net.outputs{2}.processFcns = {'removeconstantrows','mapminmax'};

% net.divideFcn = 'dividerand'; % Divide data randomly
% net.divideMode = 'sample'; % Divide up every sample
% net.divideParam.trainRatio = 70/100;
% net.divideParam.valRatio = 15/100;
% net.divideParam.testRatio = 15/100;
%
% net.trainFcn = 'traincgb'; % conjugate gradient backPropagation
%
% net.performFcn = 'mse'; % Mean squared error
%
% net.plotFcns = {'plotperform','plottrainstate','ploterrhist', ...
% 'plotregression','plotfit'};

% inputs=[1 2 3 4];

```

```

% targets=[0 1 1 0];
%
% [net,tr] = trainrp(net,inputs,targets);
% outputs = net(inputs);
% ktry=net(targets);
% %
% errors = gsubtract(ktry,outputs);
% targets=ktry;
% performance = perform(net,targets,outputs)
%
% trainTargets = targets .* tr.trainMask{1};
% valTargets = targets .* tr.valMask{1};
% testTargets = targets .* tr.testMask{1};
% trainPerformance = perform(net,trainTargets,outputs);
% valPerformance = perform(net,valTargets,outputs);
% testPerformance = perform(net,testTargets,outputs);

```

Screenshots

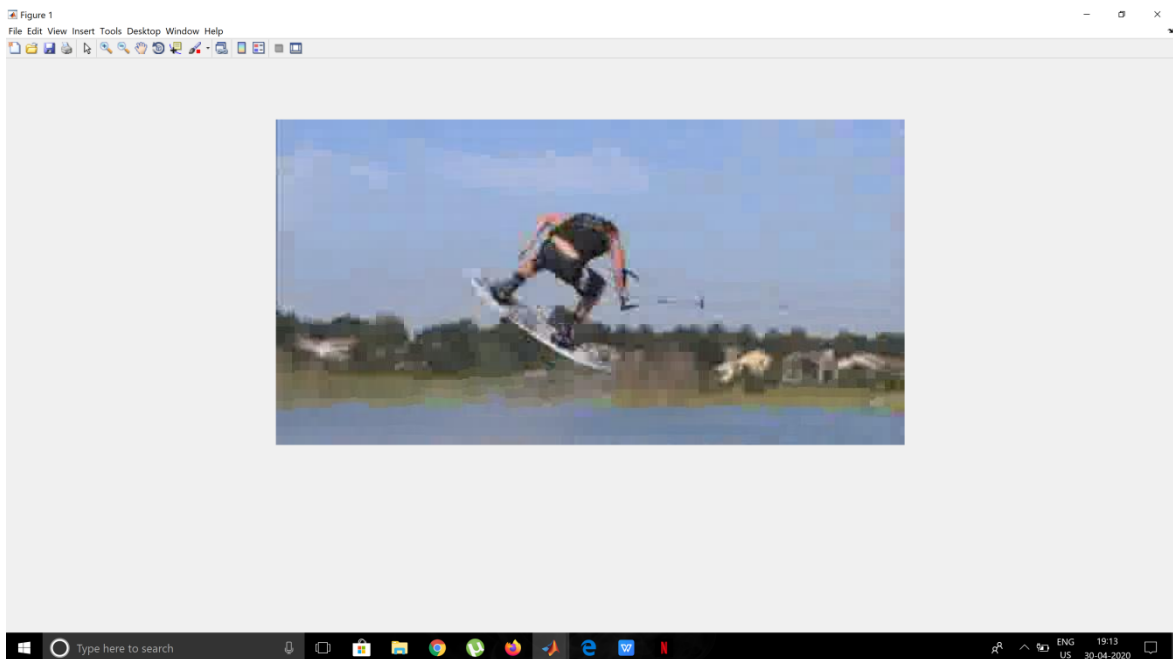


Fig A. Wakeboarding

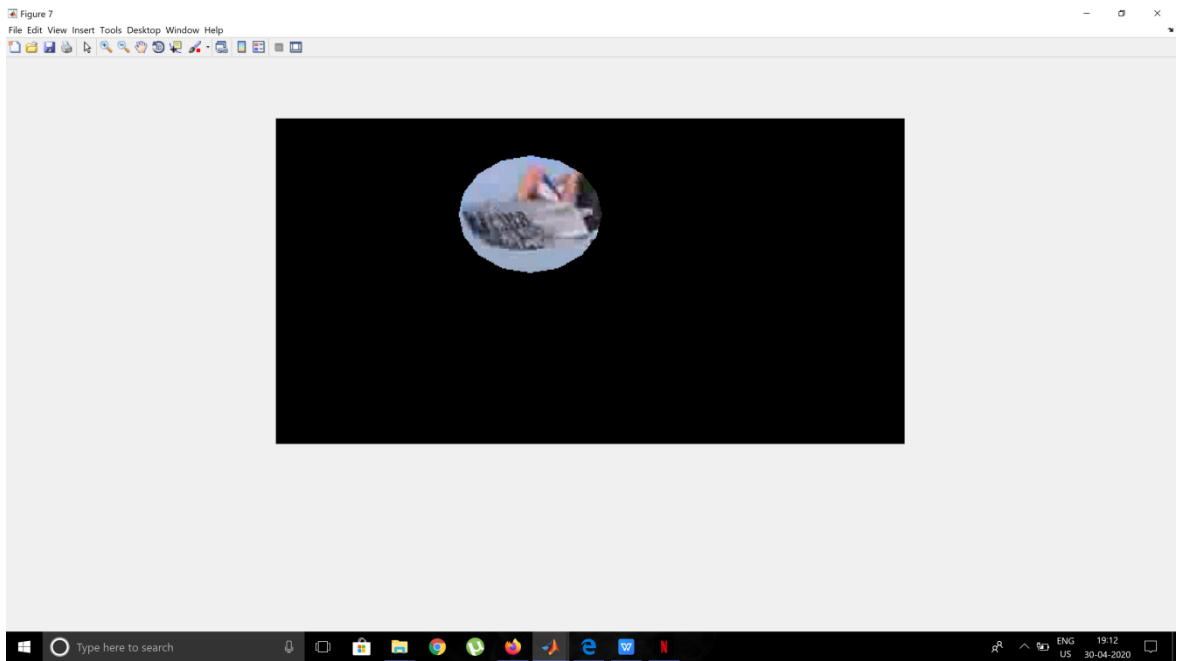


Fig B. Wakeboarding_ROI.m