Spatiotemporal Features and Deep Learning Methods for Video Classification

Rukiye Savran Kızıltepe

A thesis submitted for the degree of Doctor of Philosophy

School of Computer Science and Electronic Engineering

University of Essex

January 2022

Abstract

Classification of human actions from real-world video data is one of the most important topics in computer vision and it has been an interesting and challenging research topic in recent decades. It is commonly used in many applications such as video retrieval, video surveillance, human-computer interaction, robotics, and health care. Therefore, robust, fast, and accurate action recognition systems are highly demanded.

Deep learning techniques developed for action recognition from the image domain can be extended to the video domain. Nonetheless, deep learning solutions for two-dimensional image data cannot be directly applicable for the video domain because of the larger scale and temporal nature of the video. Specifically, each frame involves spatial information, while the sequence of frames carries temporal information. Therefore, this study focused on both spatial and temporal features, aiming to improve the accuracy of human action recognition from videos by making use of spatiotemporal information.

In this thesis, several deep learning architectures were proposed to model both spatial and temporal components. Firstly, a novel deep neural network was developed for video classification by combining Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). Secondly, an action template-based keyframe extraction method was proposed and temporal clues between action regions were used to extract more informative keyframes. Thirdly, a novel decision-level fusion rule was proposed to better combine spatial and temporal aspects of videos in two-stream networks. Finally, an extensive investigation was conducted to find out how to combine various information from feature and decision fusion to improve the video classification performance in multi-stream neural networks. Extensive experiments were conducted using the proposed methods and the results highlighted that using both spatial and temporal information is required in video classification architectures and employing temporal information effectively in multi-stream deep neural networks is crucial to improve video classification accuracy.

Acknowledgements

First, I would like to express my gratitude to my thesis supervisor Prof. John Q Gan for his continuous advice, support, patience, and encouragement. Completing my PhD thesis would have been impossible without his help and support.

I would like to thank my supervisory panel members, Dr. Alba García Seco de Herrera and Dr. Haider Raza, for their advice and helpful suggestions during my PhD journey. I would like to extend my thanks to my PhD examiners, Dr. Lily Meng and Dr. Amit Kumar Singh, for their review.

I would like to acknowledge the Turkish Ministry of National Education for providing me with the scholarship for pursuing my PhD studies. I could not have imagined having a PhD degree in the UK without this scholarship.

I would like to thank my friends, Beyza Uçar and Edward Longford, for their continuous support, encouragement, and delicious brownies. My warmest thanks also go to all my friends, especially Nihan Arı, Şeyma Karameşe, Ezgi Sarıbaz, Ayşe Cihan, Gamze Nur Eskici, as they made walking on this way easier altogether.

My special thanks go to my loved husband Fırat Kızıltepe who always supported and encouraged me to study. He moved to the UK leaving his job for me, making a difficult and brave decision. I am forever grateful for his sacrifice.

My last thankful words are kept for my parents, my sister, my brother, and my baby. I really appreciate the endless support of my family. I would not be here today without their dedication and effort to my education. My sister, Şeyda Savran Çelik, always helped me with her professional mentor skills and her little daughter, Fatma Rana, supported me with her pure and endless love. Finally, I would like to thank my grandmother and grandfather for their sincere prayers.

Contents

Abstract **i**

Acknowledgements **iii**

Contents **v**

List of Figures **ix**

List of Tables **xiv**

Nomenclature **xvi**

1 Introduction **1**

1.1 Motivation 2

1.2 Problem Statement 4

1.3 Research Objectives 4

1.4 Thesis Overview 5

2 Literature Review **8**

2.1 Introduction 8

2.2 Video Preprocessing 10

2.2.1 Shot Detection 10

2.2.2 Keyframe Extraction 11

2.3 Video Representation and Feature Extraction 12

2.3.1 Hand-crafted Feature Extraction 12

2.3.2 Deep Learning Based Feature Extraction 16
2.4 Video Classification 16
2.4.1 Image-Based Video Classification 17
2.4.2 End-to-End Video Classification 18
2.5 Deep Learning Architectures and Learning Algorithms 25
2.5.1 Neural Networks 25
2.5.2 Recurrent Neural Networks 28
2.5.3 Convolutional Neural Networks 30
2.6 Information Fusion 33
2.6.1 Image Level 33
2.6.2 Feature Level 36
2.6.3 Decision Level 37
2.7 Datasets 38
2.7.1 KTH 38
2.7.2 Weizmann 38
2.7.3 UCF Sports 39
2.7.4 HMDB-51 39
2.7.5 UCF-101 39
2.7.6 YouTube 41
2.7.7 Choice of Datasets 41
2.8 Open Problems and Research Questions 42

3 Combining Very Deep Convolutional Neural Networks and Recurrent Neural Networks for Video Classification 45
3.1 Introduction 46
3.2 Related Work 47
3.3 Methods 49
3.3.1 Preprocessing 49
3.3.2 Feature Extraction 49
3.3.3 Different VGG-16 Based Architectures for Video Classification 50

3.4 Experiments 53
3.4.1 Dataset 53
3.4.2 Experiment Design 54
3.4.3 Experiment Setup 54
3.4.4 Evaluation 55
3.5 Results and Discussion 55
3.6 Conclusion 57

4 Criteria and Methods for Keyframe Selection 59
4.1 Introduction 59
4.2 Related Work 61
4.3 The Proposed Method 63
4.3.1 The Proposed Keyframe Extraction Approach 64
4.3.2 Deep Neural Network Architectures Based on VGG-16 for Video Classification 68
4.4 Experiments 70
4.4.1 Experiment Design 71
4.4.2 Datasets and Evaluation Method 71
4.5 Results and Discussion 72
4.6 Conclusion 77

5 Extracting Motion Information and Decision Fusion for Video Classification 78
5.1 Introduction 78
5.2 Related Work 80
5.3 Methods 82
5.3.1 Two-Stream Convolutional Neural Networks 83
5.3.2 Decision Fusion 83
5.3.3 Dataset 85
5.3.4 Preprocessing and Feature Extraction 85
5.3.5 Performance Evaluation 85

5.4 Results and Discussion . . . 87
5.5 Conclusion . . . 89

6 Combination of Feature and Decision Fusion for Video Classification 90

6.1 Introduction . . . 91
6.2 Related Work . . . 91
6.3 Methods . . . 93
6.3.1 Network Architectures . . . 93
6.3.2 Decision Fusion . . . 94
6.3.3 Feature Fusion . . . 94
6.3.4 Datasets and Evaluation . . . 96
6.4 Results and Discussion . . . 97
6.5 Conclusion . . . 102

7 Conclusions 103

7.1 Contributions . . . 103
7.2 Limitations and Future Work . . . 105

Bibliography 107

Appendices 133

A List of Publications 133

B Post Hoc Comparisons 134

List of Figures

2.1 The overview of literature review chapter 9

2.2 Hierarchical structure of a video 10

2.3 Single-stream network architectures. Convolutional, normalisation, and pooling layers are indicated by pink, green, and blue boxes, respectively [80]. 19

2.4 Two-stream network architecture that processes spatial features and temporal features separately [81]. 19

2.5 Effect of learning rate to cost function [120] 26

2.6 Werbos' backpropagation algorithm: error back propagation [119] 27

2.7 A typical neural network consisting of three types of layers: input, hidden, and output layers 30

2.8 An example architecture of CNN consisting of different types of layers: input, convolutional, pooling, fully connected and output layers. 31

2.9 Three levels of information fusion for video classification 34

2.10 The KTH dataset action examples 38

2.11 The categories of the UCF Sports dataset 39

2.12 The categories of the UCF dataset [101] 40

3.1 The architectures of networks used in the experiment VGG-16-VOTE(a) and VGG-16-VOTE(b) are designed as baseline methods using the pre-trained VGG-16 by [72]. 51

3.2 The distribution of categories in training and test splits 53

3.3 (a) Training samples on the left; (b) Testing samples on the right from the UCF-101 Human Action Dataset. Samples are from three different categories: Apply Eye Makeup, Archery, and Basketball. 54

4.1 Defining the possible locations of an action template. Red, green, and blue boxes represent the borders of three possible templates. 66

4.2 An example of action template identification 67

4.3 The architectures of the networks used in VGG-16-LSTM and VGG-16-ConvLSTM experiments 70

5.1 The architecture of CNNs used in the experiment 83

5.2 The number of clips and the distribution of clip duration in the UCF-101 [101] 86

6.1 The main architecture of the networks used in the experiments 93

6.2 Two-stream CNN architecture used in the experiments with RGB frames and optical flows. Two-stream architectures with one stream removed were also considered in the experiments, as shown in Table 6.1. 94

6.3 Single-level decision fusion architecture used in the experiments with RGB frames, optical flows, and HOG features. 95

6.4 Multi-level decision fusion architecture used in the experiments with RGB frames, optical flows, and HOG features. Architectures with other combinations were also considered in the experiments, as shown in Table 6.1. 95

6.5 Single-level feature fusion architecture used in the experiments with RGB frames, optical flows, and HOG frames. Architectures with two input channels were also considered in the experiments, as shown in Table 6.2 96

6.6 Multi-stream network architecture used in the experiments combining multi-level feature and decision fusion. Architectures with other combinations were also considered in the experiments, as shown in Table 6.2 96

List of Tables

2.1 State-of-the-art methods on the UCF-101 dataset 24

3.1 Average accuracy scores achieved by different architectures 55

3.2 Independent-Samples T-Test results 56

4.1 Average accuracy scores achieved by different architectures on the UCF-101 dataset with 20 categories [5] 73

4.2 Average accuracy scores achieved by LSTM and ConvLSTM network architectures based on the UCF-101 (20 categories) using two keyframe extraction methods where (1) and (2) indicate one frame per second and the proposed method, respectively. 73

4.3 Average accuracy scores achieved by LSTM and ConvLSTM network architectures on datasets KTH and UCF-101 (101 categories) using three keyframe extraction methods where (1), (2), and (3) indicate: one frame per second method, the proposed action template-based method and optical flow-based keyframe extraction method, respectively. 74

4.4 One-way ANOVA of performance achieved by different network architectures where df, SS, MS, and F refer to degrees of freedom, sum of squares, mean sum of squares, and F score, respectively. The values are significant at the 0.05 level. 74

4.5 Post hoc comparisons using Tukey's HSD on the KTH Dataset. The values are significant at the 0.05 level. 75

4.6 Post hoc comparisons using Tukey's HSD on the UCF-101 dataset. The values are significant at the 0.05 level. 76

5.1 The descriptive statistics of top-1 and top-5 accuracy scores of decision fusion approaches: N, $\bar{x}$, σ, and $\sigma_{\bar{x}}$ denotes the number of samples, mean, standard deviation, and standard error, respectively. 87
5.2 One-way between-groups ANOVA of performance by fusion functions where df, SS, MS, and F refer to degrees of freedom, sum of squares, mean sum of squares, and F score, respectively. 88
5.3 Post hoc comparisons using Tukey's HSD. The values are significant at the 0.05 level. 88

6.1 Decision fusion based architectures used in the experiments 95
6.2 Feature fusion based architectures used in the experiments 96
6.3 Training, validation, and testing accuracy scores achieved by single-stream networks on the KTH dataset 98
6.4 Classification accuracy achieved by multi-stream networks based on decision and feature fusion 99
6.5 One-way between-groups ANOVA of performance achieved by different fusion architectures on the KTH and UCF-101 datasets where df, SS, MS and F refer to degrees of freedom, sum of squares, mean sum of squares, and F score, respectively. 99
6.6 One-way between-groups ANOVA of performance achieved by different fusion architectures on the KTH dataset 101
6.7 One-way between-groups ANOVA of performance achieved by different fusion architectures on the UCF-101 dataset 101

B.1 Post hoc comparisons using Tukey's HSD - *RGB* on the KTH Dataset 135
B.2 Post hoc comparisons using Tukey's HSD - *HOG* on the KTH Dataset 135
B.3 Post hoc comparisons using Tukey's HSD - *OF* on the KTH Dataset 136
B.4 Post hoc comparisons using Tukey's HSD - *D-RGB-HOG* on the KTH Dataset . 136
B.5 Post hoc comparisons using Tukey's HSD - *D-RGB-OF* on the KTH Dataset . . 137
B.6 Post hoc comparisons using Tukey's HSD - *D-HOG-OF* on the KTH Dataset . 137

B.7 Post hoc comparisons using Tukey's HSD - *D-RGB-HOG-OF* on the KTH Dataset 138
B.8 Post hoc comparisons using Tukey's HSD - *D1* on the KTH Dataset 138
B.9 Post hoc comparisons using Tukey's HSD - *D2* on the KTH Dataset 139
B.10 Post hoc comparisons using Tukey's HSD - *D3* on the KTH Dataset 139
B.11 Post hoc comparisons using Tukey's HSD - *F-RGB-HOG* on the KTH Dataset . 140
B.12 Post hoc comparisons using Tukey's HSD - *F-RGB-OF* on the KTH Dataset . . 140
B.13 Post hoc comparisons using Tukey's HSD - *F-HOG-OF* on the KTH Dataset . 141
B.14 Post hoc comparisons using Tukey's HSD - *F-RGB-HOG-OF* on the KTH Dataset 141
B.15 Post hoc comparisons using Tukey's HSD - *FD1* on the KTH Dataset 142
B.16 Post hoc comparisons using Tukey's HSD - *FD2* on the KTH Dataset 142
B.17 Post hoc comparisons using Tukey's HSD - *FD3* on the KTH Dataset 143
B.18 Post hoc comparisons using Tukey's HSD - *RGB* on the UCF-101 Dataset . . . 143
B.19 Post hoc comparisons using Tukey's HSD - *HOG* on the UCF-101 Dataset . . . 144
B.20 Post hoc comparisons using Tukey's HSD - *OF* on the UCF-101 Dataset 144
B.21 Post hoc comparisons using Tukey's HSD - *D-RGB-HOG* on the UCF-101 Dataset 145
B.22 Post hoc comparisons using Tukey's HSD - *D-RGB-OF* on the UCF-101 Dataset 145
B.23 Post hoc comparisons using Tukey's HSD - *D-HOG-OF* on the UCF-101 Dataset 146
B.24 Post hoc comparisons using Tukey's HSD - *D-RGB-HOG-OF* on the UCF-101 Dataset 146
B.25 Post hoc comparisons using Tukey's HSD - *D1* on the UCF-101 Dataset 147
B.26 Post hoc comparisons using Tukey's HSD - *D2* on the UCF-101 Dataset 147
B.27 Post hoc comparisons using Tukey's HSD - *D3* on the UCF-101 Dataset 148
B.28 Post hoc comparisons using Tukey's HSD - *F-RGB-HOG* on the UCF-101 Dataset 148
B.29 Post hoc comparisons using Tukey's HSD - *F-RGB-OF* on the UCF-101 Dataset 149
B.30 Post hoc comparisons using Tukey's HSD - *F-HOG-OF* on the UCF-101 Dataset 149
B.31 Post hoc comparisons using Tukey's HSD - *F-RGB-HOG-OF* on the UCF-101 Dataset 150
B.32 Post hoc comparisons using Tukey's HSD - *FD1* on the UCF-101 Dataset . . . 150
B.33 Post hoc comparisons using Tukey's HSD - *FD2* on the UCF-101 Dataset . . . 151

B.34 Post hoc comparisons using Tukey's HSD - *FD3* on the UCF-101 Dataset . . . 151

Nomenclature

Acronyms / Abbreviations

ANOVA	Analysis of Variance
BLSTM	Bidirectional Long Short-Term Memory
BOW	Bag of Words
CNN	Convolutional Neural Network
ConvLSTM	Convolutional Long Short-Term Memory
HOF	Histogram of Flow
HOG	Histogram of Oriented Gradient
HPF	High Pass Filtering
HSD	Honest Significant Difference
IDT	Improved Dense Trajectories
IHS	Intensity-Hue-Saturation
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
LDA	Linear Discriminant Analysis
LGD	Local and Global Diffusion
LSTM	Long Short-Term Memory
MBH	Motion Boundary Histograms
MSE	Mean Squared Error
PCA	Principal Component Analysis

ReLU	Rectified Linear Units
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
SIFT	Scale Invariant Feature Transform
STIP	Spatiotemporal Interest Points
VLAD	Vector of Locally Aggregated Descriptors

Chapter 1

Introduction

Between 2012 and 2021, worldwide internet usage has grown from 30% of the global population to 59.5%[1]. Moreover, the expansion of easily available and powerful hardware, combined with low-cost video editing software has allowed a greater range of people to contribute to video content generation. These advancements have triggered significant interest in generation and consumption of video contents online. For example, YouTube has one billion users, who produce on average 500 hours of content per minute and consume more than one hundred million hours of media content every single day[2].

However, this massive increase in the amount of video uploaded to the Internet has produced serious challenges in video indexing, archiving, and retrieval systems. The primary focus of the visual media research community is understanding human actions in videos on the social networking platforms, which require the automatic prediction of video semantic content to address these challenges. Due to humans' limited ability to analyse these actions in real time, a demand has emerged for intelligent systems capable of evaluating and recognising activities. Video classification techniques have addressed the human action recognition problem using video data and demonstrated the need for simple, fast, and robust systems. However, the classification of such complicated video data remains a challenging task as the accuracy of recent action recognition approaches are significantly below human performance on a per video basis.

[1]https://www.statista.com/forecasts/1146844/internet-users-in-the-world

[2]https://www.statista.com/statistics/259477/hours-of-video-uploaded-to-youtube-every-minute

This chapter presents the problem of action recognition from videos. The main applications of action recognition are summarised in Section 1.1, and Section 1.2 introduces the problem of action recognition. Section 1.3 lists the research objectives and this chapter is concluded with the thesis overview in Section 1.4.

1.1 Motivation

Video classification systems have many potential applications. Action recognition is essential for wide range of applications, such as video retrieval, video surveillance, human-computer interaction, entertainment industry, robotics, and health care.

Video retrieval is the process of providing search options across video content archives, where these options are generated as a result of an analysis of the content to obtain indexable data. The exponential increase in the number of published videos result in a flood of unstructured video content. Therefore, video retrieval has become one of the most popular and rapidly growing research areas in the multimedia technology. With massive amounts of video data being posted to the Internet every second and the growing popularity of watching videos and movies online, video understanding and action recognition algorithms have become critical for retrieving relevant content.

Video surveillance is the process of employing video cameras to monitor individuals and objects. More and more people interact with surveillance equipment every day. Surveillance cameras are utilised to analyse actions almost everywhere in our daily life such as terminals, airports, stations, shopping centres, banks, offices, schools, public places, and roads. Video surveillance footage is frequently analysed manually by a human to examine for example theft, violent acts, harassment, and used in crime prevention. However, human operators are unable to maintain those tasks with the massive number of cameras and long video streams (e.g. there are currently over 691000[3] security cameras in London and more than 5 million[4] in Great Britain). Recent video surveillance systems are aimed at automatically tracking the individuals or a crowd and recognising their activities, e.g. detecting suspicious and abnormal activities and reporting to the authorities for immediate action. Therefore, there is a growing need for action recognition systems in cities and towns to reduce the human effort and alert security teams.

Human-computer interaction is a field of computer science that explores how people and computers interact. Nowadays, human-computer interaction with a personal computer is not restricted to keyboard and mouse interaction occurring through a variety of sensory channels, including facial, body, speech, and gesture expressions. Video cameras are also employed in human-computer interaction and the entertainment sector, as they enable users to interact with machines in a more natural manner. Natural engagement with the system is becoming crucial in many human-computer interaction applications. As a result, the human-computer interaction and entertainment industries require systems that can identify and recognise actions and gestures in a video stream automatically.

Health monitoring and preventative care applications enable people to be detected and tracked inside their own environment. Human action recognition is becoming more significant and often utilised in medicine, particularly for the aim of monitoring the daily life activities of elderly individuals. Action recognition systems can detect patients' abnormal activities to identify and intervene in possible physical or mental problems at an early stage. For example, one of the most critical abnormal activities is falling which causes serious injury to elderly people. In the literature, different action recognition systems have been proposed to monitor elderly people, especially for fall detection [1, 2, 3, 4]. Usually, these systems capture the continuous movement of elderly people, automatically recognise their activities and detect any abnormality as it happens, such as falling down, having a stroke or having respiration issues. Therefore, action recognition applications are important for health care institutions.

[3]https://www.comparitech.com/vpn-privacy/the-worlds-most-surveilled-cities
[4]https://www.calipsa.io/blog/cctv-statistics-in-the-uk-your-questions-answered

1.2 Problem Statement

In this thesis, the problem of human action recognition from videos has been addressed. The main goal is to automatically learn human activities from training videos and to recognise them in previously unseen, unique, and real-world videos.

Many human activities have been explored in this study, including single-person activities like walking, running, and boxing, to more complicated interactions like haircut, salsa spin, knitting, horse race and other forms of human actions.

Deep learning approaches have been used successfully in action recognition problems. The majority of the current methods treat videos by allocating a label for all video frames. Nonetheless, considering all frames equally in the video will weaken the performance as some frames have more distinctive information than the remaining frames which are irrelevant. Furthermore, the motion information in video is important to be taken into account during classification as it differs an action from others. In the existing approaches, visual features of videos are mainly used to extend image-based deep learning approaches to video domain. If we make use of motion information to extract more informative features for each action category, deep networks can predict actions more accurately. Moreover, multi-stream neural networks combine various features and decisions to make better prediction for video classification problems. However, how to combine various information effectively from different level of fusion sources is still unclear.

This thesis aims to address the mentioned action recognition problems by introducing new methods for keyframe extraction, feature extraction, information fusion and classification to improve the accuracy of action recognition from videos.

1.3 Research Objectives

The objectives of this thesis are listed as follows:

- To review the state-of-the art approaches in video scene understanding and understand the possible limitations of the existing techniques and figure out the gaps for research contributions.

- To develop a novel classification architecture for human action recognition using transfer learning technique and compare the effectiveness of local or global features.
- To develop a novel keyframe extraction method considering the similarity between action regions in consecutive frames.
- To extract and make use of spatial and temporal features for video classification.
- To improve video classification accuracy by developing new decision fusion methods in order to make use of temporal information in two-stream neural networks.
- To investigate the effect of feature and decision fusion in multi-stream neural networks for video classification.

1.4 Thesis Overview

This thesis is organised into seven chapters, beginning with this introduction chapter. The remaining chapters of the thesis are as follows:

- **Chapter 2 -** *Literature Review:*

  This chapter gives a brief summary of the literature on video scene understanding. Video preprocessing and feature extraction methods are firstly presented. This chapter provides a review of video classification methods and present an introduction to deep learning architectures and learning algorithms. Following that, the most commonly used information fusion approaches and datasets for video classification are reviewed. Finally, current problems and challenges are highlighted.
- **Chapter 3 -** *Combining Very Deep Convolutional Neural Networks and Recurrent Neural Networks for Video Classification:*

  This chapter presents an investigation of how temporal information between frame sequences can be used to improve the performance of video classification using RNNs. Seven video classification network architectures using transfer learning are compared in this chapter. The architectures use either global or local features extracted by VGG-16, a very deep CNN pre-trained for image classification. Each network architecture for video classification was tested many times with different data splits, with the best architecture being determined using an Independent-Samples T-test. The experimental results indicate that the network architecture utilising pre-trained CNN and Convolutional Long Short-Term Memory (ConvLSTM) for temporal information extraction can achieve the highest accuracy in video classification. This chapter is a modified version of *"Combining Very Deep Convolutional Neural Networks and Recurrent Neural Networks for Video Classification"* [5] published in *Advances in Computational Intelligence*, which is a part of the *Lecture Notes in Computer Science* book series (volume 11507) and presented at the *15th International Work-Conference on Artificial Neural Networks (IWANN)* in 2019.

- **Chapter 4 -** *Criteria and Methods for Keyframe Selection:*
  This chapter introduces a novel keyframe extraction method to improve the performance of the video classification architectures. This method identifies the informative region of each frame and selects keyframes based on their similarity. In this chapter, extensive experiments using ConvLSTM-based video classifiers on the KTH and UCF-101 datasets have been conducted. The experimental results are analysed using one-way Analysis of Variance (ANOVA), which demonstrates the effectiveness of the proposed keyframe extraction method in terms of significantly improving video classification accuracy. This chapter is a modified version of *"A Novel Keyframe Extraction Method for Video Classification Using Deep Neural Networks"* [6] published in *Neural Computing and Applications* as a journal paper in 2021.
- **Chapter 5 -** *Extracting Motion Information and Decision Fusion for Video Classification:*
  This chapter describes a novel decision-level fusion technique for two-stream neural networks. Asymmetrical multiplication is proposed for decision fusion, with the aim of better combining spatial and temporal information for video classification. The experimental results are evaluated using one-way ANOVA. They indicate that the proposed asymmetrical multiplication method for decision fusion outperforms both the Mycin and average methods significantly. This chapter is a modified version of *"Simple Effective Methods for Decision-Level Fusion in Two-Stream Convolutional Neural Networks for Video Classification"* [7] published in *Intelligent Data Engineering and Automated*

Learning, which is a part of the *Lecture Notes in Computer Science* book series (volume 12489) and presented at the *21st International Conference on Intelligent Data Engineering and Automated Learning (IDEAL)* in 2020.

- **Chapter 6 -** *Combination of Feature and Decision Fusion for Video Classification:*
  This chapter investigates the effect of feature and decision-level fusion on the final classification performance of multi-stream neural networks used for video classification. A novel multi-level fusion approach is developed by combining feature and decision fusion to make better prediction using spatial and temporal features. The experimental results demonstrate that while it is required to combine spatial and temporal features, employing temporal features as dominant representations in conjunction with two distinct spatial features outperforms alternative combinations. Feature fusion involving RGB, HOG, and optical flow features outperforms other architectures, and the proposed multi-level decision fusion approach achieves comparable results to feature fusion with the advantage of lower memory requirement and computational cost.
- **Chapter 7 -** *Conclusion:*
  This chapter concludes the thesis. In addition to the summary of contributions of this thesis, the limitations, and possible extensions of the proposed methods for future research are presented in this chapter.

Chapter 2

Literature Review

2.1 Introduction

In the past decades, there has been arousing interest in web multimedia systems that can generate and distribute massive amount of multimedia data. Data mining, particularly multimedia mining, is mainly applied to extract knowledge from the multimedia data [8, 9]. The main difficulty is to deal with the unstructured complex multimedia data structure such as text, audio, image, and video [10].

Video has become one of the most popular contents in recent years. Thus, video processing using data mining techniques provides a significant potential for video content-based applications in various areas such as entertainment, education, medicine, communication, and security. The number of videos available has increased sharply both on television and the Internet. For example, the total hours of video uploaded to YouTube every minutes has increased by around ten times with 500 hours per minute between 2012 and 2021[1]. Therefore, this tremendous demand has been a remarkable issue and brought serious problems along for video indexing, archiving, retrieval in aforementioned application areas [11, 12].

In order to record people doing things, videos are used in social media, which are also used to automatically predict the semantic meaning of what is in the video. Human action recognition in

[1]https://www.statista.com/statistics/259477/hours-of-video-uploaded-to-youtube-every-minute

the real world is a highly demanded skill, which is why so many classification approaches have focused on dealing with the human factor. Current action recognition algorithms fail to match human performance, and accurately classifying complicated video contents is difficult.

In order to successfully solve problems with action recognition, deep learning techniques have been developed. Most existing approaches label each video frame, but equal consideration of all frames leads to an inefficient method as it confuses things with frames having more obvious content and frames which do not have enough meaning. It is also essential to keep in mind that video provides significant information for classification since the action distinction is the crucial information in that context. The research in image-based deep learning has utilised visual aspects of videos, but it has had difficulty applying the results to video content. Deep networks could generate better predictions about future actions if they rely on motion information to pull out relevant characteristics from each category.

In this chapter, the solutions towards scene understanding from video analysis are explored. A critical analysis is conducted for the approaches utilised solving this problem. The structure of this chapter is demonstrated in Fig.2.1. Firstly, video preprocessing and feature extraction methods are introduced. Then, main video classification methods are reviewed. Deep learning architectures and learning algorithms are presented. Afterwards, common information fusion techniques and datasets used for video classification are summarised. Finally, the existing problems and challenges are identified.

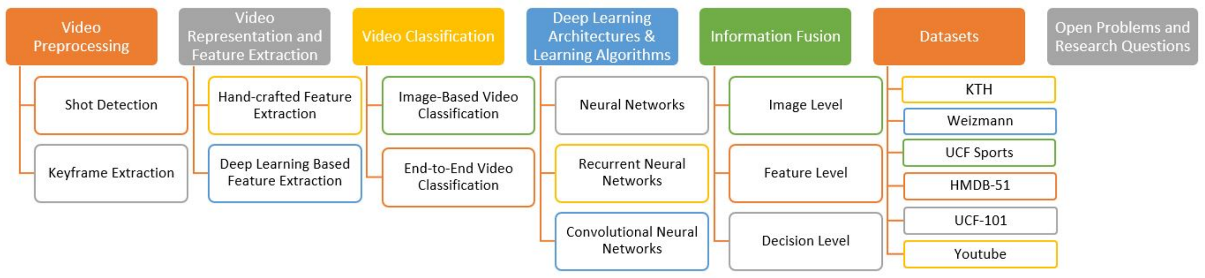
Video Preprocessing
Shot Detection
Keyframe Extraction
Video Representation and Feature Extraction
Hand-crafted Feature Extraction
Deep Learning Based Feature Extraction
Video Classification
Image-Based Video Classification
End-to-End Video Classification
Deep Learning Architectures & Learning Algorithms
Neural Networks
Recurrent Neural Networks
Convolutional Neural Networks
Information Fusion
Image Level
Feature Level
Decision Level
Datasets
KTH
Weizmann
UCF Sports
HMDB-51
UCF-101
Youtube
Open Problems and Research Questions

Figure 2.1: The overview of literature review chapter

2.2 Video Preprocessing

Preprocessing is a data mining technique that converts raw data to a format that machines can understand. Real-world data is inherently incomplete and cannot be processed by a machine learning model. This would result in certain errors, and thus preprocessing data is required prior to feature extraction and classification. In this section, main preprocessing steps for video classification are summarised.

2.2.1 Shot Detection

An important step in the automation of content-based video indexing and retrieval is the recognition of video shot boundaries. Frames, shots, and scenes can be imagined as a hierarchy of video as depicted in Fig. 2.2. Shot is a continuous streaming of motion in a series of video frames. Shot can be defined as a continuous video sequence with only graphic elements as its contents. The process of detecting shot boundaries is the detection of the transitions between two consecutive shots [13].

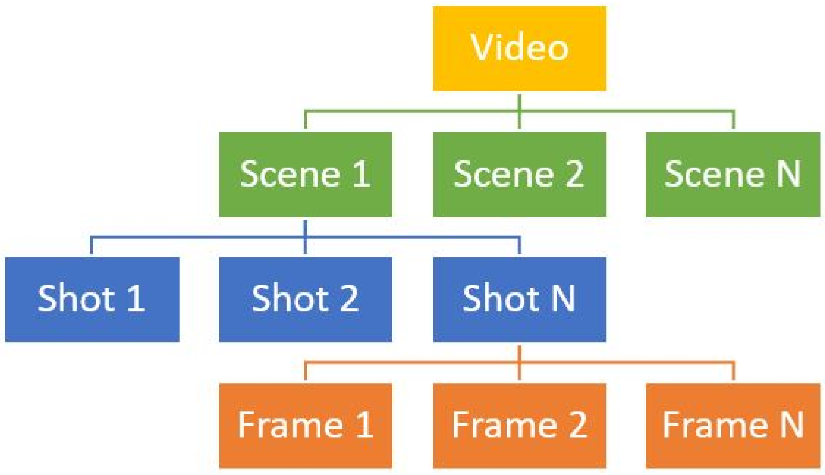

Figure 2.2: Hierarchical structure of a video

Numerous algorithms have been developed to identify video shot boundaries, the different types of shot, and shot transition. There are mainly two types of techniques: content-based and time-based. Temporal techniques include measuring the variations in video frames by time

[14, 15, 16], while content-based techniques focus on the content of each frame to find the shot boundary [17, 18]. Moreover, a few studies have been published that compare the various shot detection algorithms [19, 20, 21].

2.2.2 Keyframe Extraction

Keyframe extraction starts with decomposition of the input video into temporal components such as shots or scenes. There have been many ways to extract keyframes from all frames. The basic approach is taking out the middle frame of each shot as a keyframe. One of the most common ways is to compare the colour histograms of consecutive frames. To select keyframes along the video, two subsequent frames' histograms are calculated, and the difference between them is compared with a threshold value identified by user. If the difference is greater than the threshold value, the frame is selected as a keyframe. Another way is to use the image entropy. After computing the entropy for each frame, the difference between two consecutive frames is compared with a threshold value to select keyframes. Moreover, clustering methods are used to extract keyframes. The idea behind it is that frames are grouped based on their low-level features by using a clustering method like k-means. Then, the most similar frames with the groups' centres are selected. Furthermore, sequential search on video is employed. The idea behind it is starting with a root, a random candidate of keyframe, and analysing the next frames respectively until obtaining a frame with significantly different low-level features which is the root of the next keyframe.

Wolf proposed a novel algorithm for selecting keyframes within shots from video [22]. This algorithm employs optical flow computations to detect local minima of action in a single shot. They measure the action in a shot by utilising optical flow analysis, and keyframes are selected at the local minima of the action [22]. Yan et al. introduced a two-stream CNN to detect keyframes in human action videos. This network learns how the location of keyframes can be predicted based on the features extracted by Linear Discriminant Analysis (LDA). The trained CNN can predict the importance of keyframes [23]. In addition, there have been several attempts to solve this problem by using clustering [24], pooling [25], and deep framework [26].

2.3 Video Representation and Feature Extraction

Recognising and classifying actions from video data has been extensively studied in the fields of computer vision and image processing. One of the primary concerns of this task is the development of an effective video representation. Generally, this problem can be expressed in classification as a representation of a frameset via feature extraction techniques.

Feature extraction and classification are two different steps of video classification frameworks. Feature extraction is applied in the early stage of classification to obtain input representation. Then those representations are fed into a classifier to reach the final classification.

Feature extraction is one of the main processes in machine learning applications, which is based on a collection of techniques used for obtaining information regarding input. In the concept of video classification, features can be extracted from either independent frames or subsequent frames. Feature extraction methods can be categorised into two approaches: hand-crafted feature representation approach and deep learning-based feature representation approach.

2.3.1 Hand-crafted Feature Extraction

Scene understanding based on video and still image processing is one of the research topics in computer vision. Video analysis deals with diverse approaches utilised to observe the changes in the present scene of a particular video. Nowadays, video analysis is one of the momentous areas of computer science. There have been several attempts to review the literature on video scene understanding. The majority of the methods exploited in the reviewed works put into practice hand-crafted features with classic machine learning pipelines for human action recognition [27, 28, 29, 30, 31].

Earlier versions of the video classification problem used a combination of traditional hand-crafted features and state-of-the-art machine learning methods to classify the videos. In the hand-crafted approaches, the features are extracted using certain feature extractors, and those features are used as inputs to the classifiers. However, in the deep learning-based representation approaches, both feature extraction and classification are conducted within the same framework and different parts of the neural networks are used to complete both tasks.

Commonly used hand-crafted feature extraction methods include Histogram of Oriented Gradients (HOG) [32], Histogram of Flow (HOF) [33], Scale Invariant Feature Transform (SIFT) [34], Speeded Up Robust Features (SURF) [35, 36], Spatiotemporal Interest Points (STIPs) [37], Motion Boundary Histograms (MBH) [38], and Fisher Vectors [39].

The HOG descriptor describes an object's structure and shape by extracting the gradient and orientation of the edges. It keeps track of the occurrences of gradient orientation in certain regions of an image or frame. These orientations are determined using 'localised' segments. This means that the entire image is divided into smaller sections, and the gradients and orientation of each region are determined. Finally, the HOG would create a distinct histogram for each of these sections. Histograms are constructed utilising the gradients and orientations of the pixel values. The HOG descriptor was firstly introduced by Dalal and Triggs [32]. Klaser et al. [40] expanded static image histogram of gradient features to the space-time dimension and presented 3D-HOG features to describe activities.

The HOF and MBH descriptors are based on optical flow representations and describe a motion on an image or frame. Gibson [41] pioneered the notion of optical flow. Laptev et al. [33] proposed a method to aggregate optical flow responses and Dalal et al. [38] proposed MBH by calculating changes of optical flow. Wang et al. proposed a descriptor based on MBH which is a video representation using dense trajectories and motion boundary descriptors [42]. A comprehensive experiment using different descriptors was conducted on nine datasets. The MBH descriptor outperformed other powerful descriptors and their approach outperformed the existing state-of-the-art results on all datasets [43].

The SIFT descriptor [34] detects, describes, and matches local features in images. SIFT extracts and utilises a significantly larger number of features, reducing the impact of mistakes produced by these local changes in the average error of all feature matching errors. It is capable of robust object identification even in the presence of clutter and partial occlusion, as the SIFT feature descriptor is invariant to orientation and illumination, uniform scaling, changes, and partially invariant to affine distortion [34]. The SIFT descriptor has been extended to 2+1-dimensional spatiotemporal data in the context of human activity recognition in video sequences [44, 45]. The two-dimensional SIFT algorithm's generation of local position-dependent histograms is extended to three dimensions in order to express SIFT characteristics in a spatiotemporal domain. ST-SIFT detector extends the SIFT detector spatially and temporally [46]. It is effective in representing events in video because spatiotemporal interest points identified by the ST-SIFT detector reflect both spatial and temporal texture information. The evaluations strongly suggest that SIFT-based descriptors that are region-based are the most robust and distinctive, and thus the best candidates for feature matching. SURF descriptor was later demonstrated to be comparable to SIFT in terms of performance while being significantly faster [47]. According to other studies, SIFT outperforms SURF when speed is not a factor [48, 49].

The SURF method was first published by Bay et al. after the SIFT descriptor [35]. The SURF descriptor has been used to detect and recognise items, persons, and faces, as well as to rebuild three-dimensional scenes, track objects, and extract points of interest. SURF detects interest points by computing an integer approximation of the Hessian blob detector's determinant, which may be computed in three integer operations using a pre-computed integral image. Its feature descriptor is the sum of the Haar wavelet responses in the region of the point of interest [35].

The concept of spatial interest points has been extended in video recognition to include spatiotemporal STIP [37]. Guo [50] presented the ST- SIFT detector in order to address the issue of noise intrusion in videos with complex scenes. Regarding sparse spatiotemporal features, Dollar et al. [51] presented cuboid features derived from the cuboids of spatiotemporally windowed data surrounding a feature point. STIP detectors have a number of limitations, including low temporal efficiency, limited robustness to camera movement, light change, perspective occlusion, and background clutter.

In computer vision, a representation technique known as Bag of Words (BOW) has been used [52]. The core principle of this approach is to depict image data as a normalised histogram referred to as code words. During the learning process, visual words can be constructed by clustering similar patches of an image that can be described by a common feature descriptor. As a result, certain techniques will produce similar histograms for similar images. These can be incorporated into the classification process. Numerous studies on action recognition have used BOW-based methods, including [51, 53, 54, 55, 56].

Another commonly used technique for representing features is the Fisher vector descriptor, which can be regarded as a global descriptor. This technique optimises the calibration of a generative model in order to more accurately model the distribution of extracted local features. The descriptor is constructed using the gradient of a sample's likelihood with respect to the distribution's parameters. It is estimated using a training set and scaled by the Fisher information matrix's inverse square root. Perronnin and Dance [39] proposed first Fisher vector descriptor for image classification. Fisher Kernel framework was introduced using advanced feature encoding in images by Sanchez et al. [57]. Fisher vector, Vector of Locally Aggregated Descriptors (VLAD) have advantages over BOW in local descriptor encoding methods [57]. Fisher vector-based techniques have been employed in many research for video classification such as [58, 59, 60, 61].

The traditional feature extraction methods requires both domain information and human effort to extract human-engineered local features. However, deep learning methods do not require human effort and domain knowledge. Before the prevalence of deep learning-based methods, the focus had been on traditional feature engineering techniques to improve video classification implementations. After the major breakthrough in the video representation by successfully training classifiers, the focus was changed into deep feature extraction. Even after deep feature representation, pre-trained networks were employed as feature extractors.

2.3.2 Deep Learning Based Feature Extraction

Traditional feature extraction methods are usually time-consuming and requires feature engineering and domain knowledge. However, the recent trend to extract robust feature representations is applying deep learning to raw video data. Multiple levels of feature representations can be learned to understand various types of data, including speech, image, and text. These methods are capable of processing raw image and video data automatically for the purpose of feature extraction, description, and classification. These methods for action recognition and representation frequently make use of trainable filters and multi-layer models.

Section 2.5 describes several significant deep learning models that have been used to recognise human actions. However, training a deep learning model from scratch with limited data is extremely difficult. As a result, models are frequently restricted to appearance representation.

CNN is one of the popular deep learning architectures to extract deep features. CNNs involve two main parts: they begin with a sequence of convolution and pooling layers, and they conclude the classification with a densely connected classifier. Pre-trained networks have been used to extract new features from different instances using their first part. In the case of convolution networks, the convolutional part of the pre-trained network is included in the place of feature extractor to obtain interesting features from new samples. Then, these features are fed into a classifier on top of the output layer. Therefore, representations learned by convolutional part of previously trained model on large datasets can be used without a need of domain knowledge. The related works are found in the following references: [62, 63, 64, 65]. More related works are reviewed and presented in the following section.

2.4 Video Classification

Video classification has become an important research area and gained a significant success in recent years. There are several video classification approaches ranging from text-based [66, 67], audio-based [68, 69] to visual-based. In this thesis, the visual-based approach is followed due to the rich information in visual context. Video classification algorithms can be categorised as follows: image-based and end-to-end video classification.

2.4.1 Image-Based Video Classification

The success of CNNs in image understanding is transferred into video classification. The common idea is to handle with a video as an accumulation of consecutive frames. For image analysis, a variety of CNN architectures have been proposed and investigated. The feature representations for those architectures can be extracted by using a feed-forward pass up to a specific fully-connected layer with pre-trained deep models on ImageNet [70] such as AlexNet [71], VGGNet [72], GoogleNet [73], ResNet [74], and DenseNet [75]. Then, these features are averaged to represent video as input of a video classifier like Support Vector Machine [76, 77].

Deeper networks provide a more accurate approximation of the target and provide more accurate feature representations with stronger discriminatory capacities [78]. Thus, the trend was deepening the network for better image understanding. While deeper networks have better discriminatory power, they require more data for training and more parameters to tune. Finding a large, professionally labelled dataset remains a significant barrier for the research community, limiting the creation of increasingly sophisticated neural networks.

Zha et al. [79] utilised an image-trained CNN to extract features from different layers of deep models. They showed that CNN features can gain satisfactory recognition performance in video classification by using the motion information added via late fusion on the UCF-101. However, the spatiotemporal features learnt could not capture movement information properly in the single-stream networks. In a CNN architecture, the main issue is the way of combining appearance and motion information together. Therefore, most of the researchers have focused on using CNN to gather spatiotemporal information with end-to-end video classification rather than image-based methods.

2.4.2 End-to-End Video Classification

In recent years, as more powerful deep learning architectures have been developed, the trend for video classification tasks has shifted away from hand-crafted approaches to fully automated deep learning approaches. The capability of CNN to extract features from raw data has been a key factor in the development of the end-to-end algorithms.

Two seminal works published in 2014 established deep learning as the foundation for video classification. Karpathy et al. [80] and Simonyan and Zisserman [81] popularised single-stream and two-stream networks in action recognition. Single-stream networks handle spatiotemporal features together and essentially no temporal features in contrast to multi-stream networks. Multi-stream networks process spatial and temporal information separately and the extracted features from each stream are fused to reach final classification decision.

Karpathy et al. [80] investigated how temporal data can be fused using a single-stream 2D CNN and they evaluated the proposed architectures in Fig. 2.3. The architectures described failed to significantly improve performance when a single frame is used. The disadvantage of these models is that they do not adequately capture motion aspects. They claimed that local motion information might not be very essential especially for dynamic dataset. Castro et al. [82]; however, highlighted the importance of motion information in video classification problems by showing that the use of visual information is inadequate in motion-based categories.

The advantage of the single-stream technique is that it enables us to leverage transfer learning from models trained on large-scale image datasets. Additionally, there is no need to preprocess the images for optical flow in this architecture because the RGB image data is used directly. As a result, single-stream networks are an attractive candidate for real-time processing.

Simonyan and Zisserman [81] proposed the two-stream CNN that involves spatial and temporal networks which are merged by late fusion as shown in Fig. 2.4. A single video frame is fed into a 2D CNN, while preprocessed multi-frame optical flow is fed into a different 2D CNN.

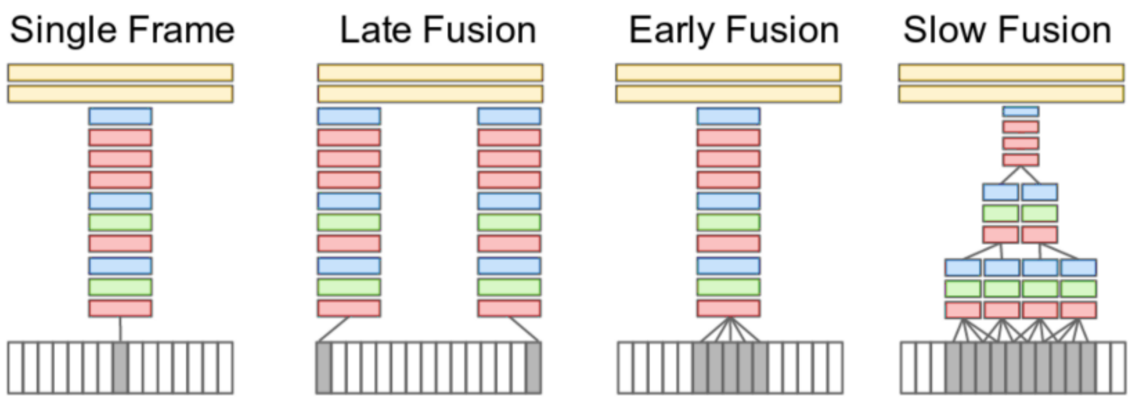

Figure 2.3: Single-stream network architectures. Convolutional, normalisation, and pooling layers are indicated by pink, green, and blue boxes, respectively [80].

Each stream generates a prediction, and their fusion determines the final class score. They also contributed to train the temporal stream on multi-frame dense optical flow to recognise action from movements while the spatial stream recognises actions on raw video frames. Their two-stream model achieved better results on the UCF-101 and HMDB-51 than the models consisting either spatial or temporal stream.

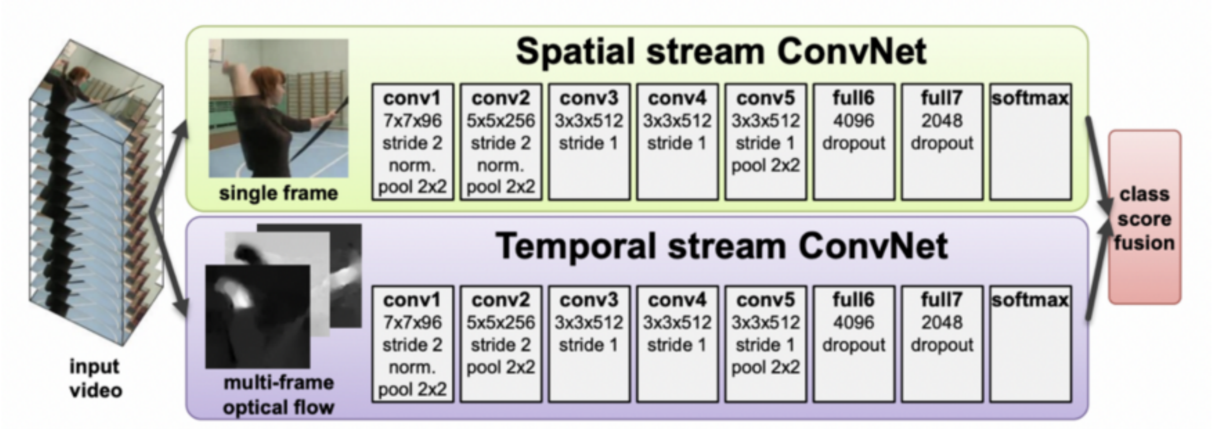

Figure 2.4: Two-stream network architecture that processes spatial features and temporal features separately [81].

This development opened the door for additional study into deep learning applications to video classification. This study demonstrated that CNNs are capable of effectively capturing some motion features and combining them with spatial features to generate accurate predictions for action classes. Following this work, the two-stream architecture has been extended in many works [83, 84, 85, 86, 87, 88].

Donahue et al. popularised the Long Short-Term Memory (LSTM) architecture for video with an architecture known as LRCN [84]. The LRCN is an extension of the encoder-decoder architecture, but it is used for video representations. The LRCN network's strength is that it can handle sequences of varying durations. Additionally, it can be applied to other video-related activities such as image captioning and video description. The LRCN's limitation is that it is unable to outperform the state-of-the-art at the time, but it did make significant gains over single frame designs.

Ji et al. demonstrated a 3D CNN model by extending a special 2D CNN for human action recognition from movies [89]. They used the third dimension of CNN to extract motion information between subsequent frames; therefore, this model has extracted multiple channels having information from the input frames and merged information from whole channels as an ultimate video feature representation.

Tran et al. proposed deep 3D CNNs as the new state-of-the-art in 2015 [54]. They demonstrated that the most successful method for learning spatiotemporal features is the 3D CNN using a 3x3x3 kernel. The 3D CNN network is initially learning spatial appearance, followed by salient motion in the clip's consecutive frames. They also showed that 3D CNNs are more convenient than 2D CNNs to learn spatiotemporal features.

Yao et al. [90] described a novel 3D CNN-RNN encoder-decoder which can capture both local spatial and temporal information. With an attention mechanism within this framework, they capture global context as well.

In 2016, the trend of video classification was back to two-stream networks. Feichtenhofer et al. [88] addressed the issue of fusing spatial and temporal data across streams and generating multi-level loss that could handle long-term temporal relationships. The authors proposed fusing the two streams at two points in the proposed architecture. This network outperformed state-of-the-art improved dense trajectories (IDT) and 3D CNN techniques in terms of capturing motion and spatial data in various subnetworks. While this method retains the limitations of the original two-stream network, it performs better as a result of an upgraded architecture that better fits real-world data.

In 2017, Zhu et al. [91] improved two-stream networks by inventing MotionNet, a hidden stream that learns optical flow [8]. The researchers were able to avoid manually computing optical flow using this end-to-end approach. This means that techniques based on two streams can now be real-time, and that errors resulting from wrong predictions can also be propagated into MotionNet for the purpose of optimising optical flow features. The researchers found that while the two-stream CNNs having hidden stream perform similarly to non-hidden techniques, they can now process up to 10 times more frames per second, enabling the two-stream method to be used in real time.

Carreira and Zisserman compared two-stream architectures consisting of a combination of 3D CNN models. Filters and pooling kernels from 2D CNNs are extended into 3D, giving them an additional temporal dimension. This enables researchers to transfer effective 2D classification frameworks to 3D. In their paper, two 3D networks are employed for two streams where spatial stream includes stacked frames in time dimension rather than single frames. Pre-trained two-stream network on ImageNet and Kinetics outperformed the other architectures. They showed that the use of pre-trained 2D CNN helps to repeat the pre-trained weights in the next dimension [92].

Significant breakthroughs in deep residual learning gained attention and have resulted in the development of novel architectures such as 3D ResNet [93] and pseudo-residual 3D CNN [94].

In 2018, He et al. [95] propounded a novel spatial-temporal network (StNet) which conducts not only local but also global spatial-temporal modelling within movies. 2D convolution on images, whose number of channels is equal to 3 times the number of stacked frames, is applied in order to figure out local spatial-temporal associations. As for global spatial-temporal association, they have implemented 26 temporal convolution on the local spatial-temporal feature maps. The novelty particularly lies in employing temporal Xception block which utilizes a discrete "channel-wise and temporal-wise convolution over the feature sequence of video". It was reported that StNet outperformed several state-of-the-art networks on the Kinetics dataset [95].

Choutas et al. [96] proposed a novel deep architecture by considering appearance and motion features jointly (PoTion). They encoded human joint movement and aggregated the resulting heatmaps temporally. PoTion outperformed other state-of-the-art pose representations in their experimental evaluation. Additionally, it complements conventional appearance and motion streams. They achieved state-of-the-art performance on the JHMDB, HMDB, and UCF101 datasets by combining PoTion and the current two-stream I3D method [92].

In 2019, Tran et al. [97] proposed the use of channel separated convolutional networks (CSNs) for action recognition. The researchers built on the success of group convolution and depth-wise convolution in the Xception and MobileNet models, respectively. By not being completely connected, group convolutions introduce regularisation and require fewer calculations. Convolutions on a depth-wise basis are an extreme instance of group convolutions in which the number of input and output channels equals the number of groups. In their 3D convolutions, conventional CNNs represent both channel interactions and local interactions (both spatial and spatiotemporal). The researchers reduced the number of parameters in the network greatly and incorporated a robust form of regularisation. The channel-separated blocks enable the network to learn both spatial and spatiotemporal features at their own distinct layers.

Oiu et al. [98] proposed a novel two-stream framework for speeding the learning of spatial-temporal representations via Local and Global Diffusion (LGD). Their network architecture simultaneously learns local and global features via the usage of LGD blocks, where each block updates both features by simulating the diffusions between these two representations. The

LGD-3D two-stream network outperforms current architectures, demonstrating that splitting 3D CNN into 2D spatial and 1D temporal CNN results in improved performance.

In 2020, Duan et al. [99] demonstrated Omnisource, a novel architecture for using online data to train video classification models for improved recognition performance. For web-supervised learning, this new methodology overcomes the limitations of data formats such as images, short videos, and long untrimmed movies. OmniSource is more data-efficient in training, as demonstrated by experiments, by leveraging data from different sources and formats. They emphasized the importance of large-scale pre-training utilising extremely massive neural network models.

Gowda et al. [100] concentrated on the SMART frame selection problem in video classification in order to increase classification accuracy. When all frames from a movie are retrieved and feature extraction is performed, the model and computations become very time demanding. As a result, smart frame selection is conducted, which selects a constant number of frames from each video and passes them forward to the LSTM model. They demonstrated that selecting high-quality frames improves action detection ability even in the domain of trimmed videos. The SMART method is now the state-of-the-art architecture in the literature, having been evaluated on a variety of benchmark datasets (UCF101, HMDB51, FCVID, and ActivityNet).

UCF-101 is a benchmark action recognition dataset that was published in 2012 [101] by researchers at the University of Central Florida; additional information about this dataset is provided in Section 2.6. Due to the uncontrolled environment in the captured videos, this dataset presents a challenge, but it is widely used by researchers working on video classification problems. As a result, it is commonly used to compare the majority of the existing literature. The existing literature utilising the UCF-101 is compared in Table 2.1, where the methods are ordered ascendingly by performance. In addition, more research papers can be found in the following review papers [31, 102, 103, 104].

Table 2.1: State-of-the-art methods on the UCF-101 dataset

Method	Accuracy
2D CNN [80]	65.4%
ImageNet + SVM [54]	68.8%
Spatial stream network [81]	72.6%
IDT BoW + SVM [54]	76.2%
LRCN [84]	82.9%
Temporal stream network [81]	83.7%
LSTM composite model [86]	84.3%
3D CNN + SVM [54]	85.2%
IDT Fisher vector [105]	87.9%
Two-stream network [81]	88.0%
LSTM [85]	88.6%
3D CNN + IDT + SVM [54]	90.4%
Transformation CNN [106]	92.4%
Multi-stream multi-class [107]	92.6%
Key volume mining [26]	92.7%
Convolutional two-stream [88]	93.5%
Temporal segment network [108]	94.2%
Flow I3D [92]	96.7%
S3D-G [109]	96.8%
D3D [110]	97.0%
Hidden two-stream [91]	97.1%
Multi-stream I3D [111]	97.2%
BubleNet [112]	97.62%
Two-stream I3D [92]	98.0%
LGD-3D Two-stream [98]	98.2%
PerfNet [113]	98.6%
OmniSource [99]	98.6%
SMART [100]	98.64%

2.5 Deep Learning Architectures and Learning Algorithms

Deep learning is a technique that makes use of deep neural networks to perform complex computations on big data. It is a subset of machine learning that is structured and operated similarly to the human brain. Deep learning has exploded in popularity among scientists, and its algorithms are frequently employed by sectors that deal with complicated problems. Each deep learning method uses different kind of algorithms. This section explains the deep learning algorithms and the key artificial neural networks that have been employed in this thesis.

2.5.1 Neural Networks

The first mathematical model for neural networks was developed, being inspired by the visual perceptron structures of animals [114], by McCulloh and Pitts in 1943 [115]. Frank Rosenblatt developed a "hypothetical nervous system called a perceptron which is a probabilistic model for information storage and organization in the brain" in 1958 [116].

In 1959, Hubel and Wisel found two types of cells in the cat's visual cortex: simple cells and complex cells [114]. Both cells fire in response to exact attributes of visual sensory inputs; however, complex cells manifest more spatial invariance than simple cells. Their discovery gave an inspiration to deep neural network architectures. The first multi-layer networks were trained by group method of data handling which is originated in 1965 by Ivakhnenko and Lapa [117, 118].

In 1975, Werbos' backpropagation algorithm solved the exclusive-or problem, which was a key initiator for the prevalent interest in neural networks [119]. Training in backpropagation networks is conducted in two steps: modelling of training data in the input layer and minimisation of total error using gradient descent.

Gradient descent is a typical optimisation method to minimise a given function. In the concept of neural network, it is applied to minimise a cost function such as sum of squared error, mean squared error and cross-entropy so that weights of network are updated iteratively after each epoch. The term used to control how much the weights of the network are adjusted is learning rate. Sum of squared error is presented in Equation 2.1 where J is the cost of weight, d is the target output and y is the actual output. The importance and tendency of the weight update is calculated taking a step in the reverse direction of the cost gradient as in Equation 2.2 where w is weight, a is learning rate, and J is error with respect to the weight. Weights are updated after each epoch via the update rule in Equation 2.3 where w is the updated weight.

$$J(w) = \frac{1}{2}\sum_i (d_i - y_i)^2 \tag{2.1}$$

$$\Delta w_j = -a\frac{\partial J}{\partial w_j} \tag{2.2}$$

$$w := w + \Delta w \tag{2.3}$$

The learning rate, the most important hyper-parameter, affects the gradient descent depending on its value. If it is very small, gradient descent might be slow whereas if it is very large, gradient descent might overshoot the minimum and fail to converge to local minima. The effectiveness of learning rates on cost function is depicted in Fig. 2.5.

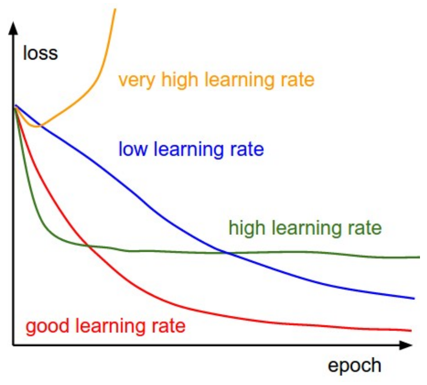

Figure 2.5: Effect of learning rate to cost function [120]

The backpropagation algorithm makes training multi-layer networks efficient and feasible because it allocates the error back up through the layers, by changing the weights at each node as shown in Fig. 2.6. Moreover, backpropagation algorithm was efficiently employed to minimise cost functions by adapting weights [121].

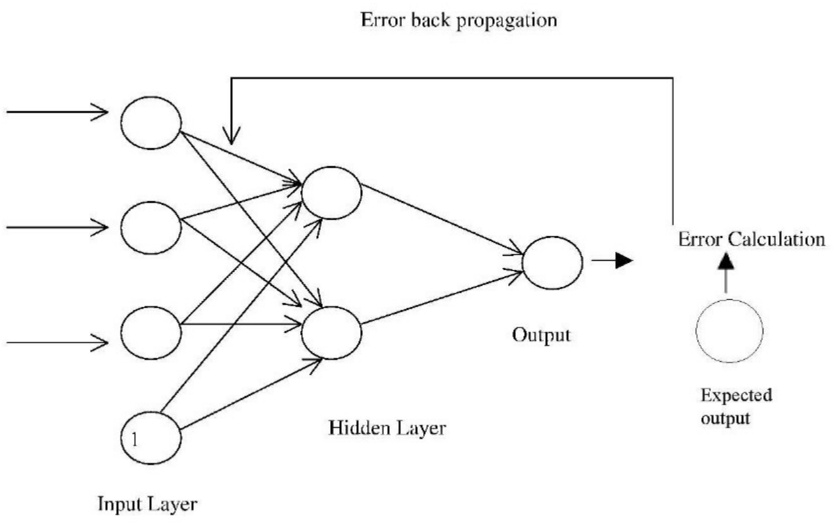

Figure 2.6: Werbos' backpropagation algorithm: error back propagation [119]

In 1979, Fukushima proposed an artificial neural network named "neocognitron" which uses local connections between neurons [122], based on which CNN was introduced, "where the receptive field of a convolutional unit with given weight vector is shifted step by step across a 2-dimensional array of input values, such as the pixels of an image" [122].

In the mid-1980s, Rumelhart and McCelland introduced a term 'connectionism' to demonstrate neural processes [123]. The connectionism elucidates mental phenomenon by utilizing artificial neural networks in cognitive science and indicates parallel distributed processing. The essential model used in the neural network is connectionism although there have been various neural network models. For example, a mental state can be demonstrated as an (N)-dimensional vector of activation values on neural units in the concept of neural network. Memory is allocated by changing the weights of the connections between these neural units. The connection weights are generally represented as a NxN matrix.

In the 1990s, neural networks have been employed in several application ranging from speech recognition [124] to driving a car [125]. In 1993, support vector machines have been extended with the kernel trick and achieved satisfactory results in non-linear classification problems [126].

Afterwards, support vector machines and other simple methods outperformed neural networks [127] in machine learning problems and many researchers abandoned neural networks due to the slow speed of backpropagation with multi-layers and insufficient local optima in gradient.

Neural networks learn from observational data, solving problems on their own. Except for a few specialised issues, researchers did not know how to train neural networks to outperform more traditional approaches until 2006. In 2006, Hinton et al. [128] developed deep neural networks named Deep Belief Networks. This launched the third wave of neural networks, which popularised the terminology deep learning.

Deep learning algorithms have been improved, and currently deep neural networks and deep learning produce exceptional performance on a wide variety of challenging issues in computer vision, speech recognition, and natural language processing. They are currently being used extensively by companies such as Google, Microsoft, and Facebook.

2.5.2 Recurrent Neural Networks

A RNN is a kind of artificial neural network that performs on sequential or time series data. This type of neural networks are frequently used to solve ordinal or temporal challenges like natural language processing [129], handwriting recognition [130], speech recognition [131], and image captioning [132], video classification. RNNs learn from training input and they are distinguished by their memory, which allows them to modify the current input and output based on information from previous inputs [133]. While typical deep neural networks presume that their inputs and outputs are independent, RNNs' outputs are dependent on the sequence's prior elements.

In 1982, the first working example of RNNs was developed by Hopfield called Hopefield network [134] as a means of storing information and performing data storage and retrieval functions.

In 1986, Rumelhart et al. published a manuscript in which backpropagation was reintroduced [135].

They demonstrated empirically that this learning algorithm is capable of producing useful internal representations and so is applicable to general neural network learning applications.

RNNs compute gradients using a backpropagation through time algorithm [136], which is slightly different from regular backpropagation because it is optimised for sequence data. It operates on the same principles as traditional backpropagation, in which the model trains itself by calculating errors. These computations enable us to accurately alter and fit the model's parameters. Backpropagation through time algorithm is distinguished from the standard technique in that it adds errors at each time step, whereas feedforward networks do not require summation because they do not share parameters between layers.

Basic formula of RNN is presented in Equation 2.4 at a time step t where $x^{(t)}$ and $z^{(t)}$ represents input and output vectors through a series of hidden states $h^{(t)}$:

$$\begin{aligned} h^{(t)} &= \sigma(W_x x^{(t)} + W_h h^{(t-1)} + b_h) \\ z^{(t)} &= softmax(W_z h^{(t)} + b_z) \\ \sigma(x) &= \frac{1}{1+e^{-x}} \end{aligned} \tag{2.4}$$

where W_x, W_h, and W_z are weight matrices and b_h and b_z are biases. Softmax refers to the softmax function while σ describes sigmoid function.

RNN is incapable of modelling long-range dependencies and cannot retain information about previous inputs over a long time [137]. Moreover, two well-known issues exist in training RNNs: the vanishing gradient problem refers to the exponential shrinking of the magnitude of gradients as they are propagated back through time; and the exploding gradient problem refers to the explosion of long-term components caused by the gradient's norm increasing significantly during training sequences with long-term dependencies. To address these challenges, researchers developed models of LSTM [138]. In this thesis, detailed experiments conducted using LSTM and more details are presented in the research chapters.

2.5.3 Convolutional Neural Networks

CNNs are similar to neural networks mentioned in the previous section. A CNN is comprised of input, output, and hidden layers just as typical neural networks. Furthermore, the hidden layers are comprised of multiple layers such as convolutional layers, pooling layers, fully connected layers, and normalisation layers. The visualisation of regular neural network is presented in Fig. 2.7. CNN organises its neurons in three dimensions: width, height, and depth. Each layer of a CNN employs the three dimensional input volumes to three dimensional output volume of neuron activations. For instance given an image input, height and width are the same with the input image's width and height while the depth is the number of color channels such as 3 for red, green, blue color channels.

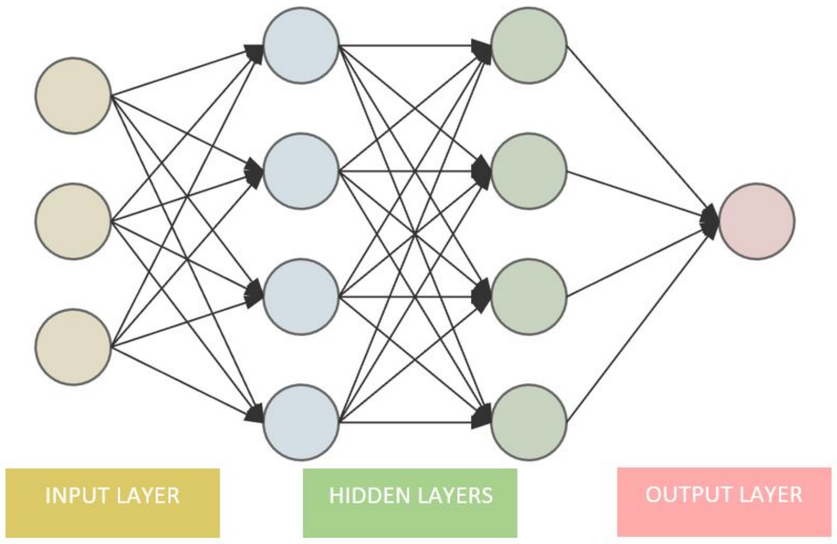

Figure 2.7: A typical neural network consisting of three types of layers: input, hidden, and output layers

A CNN is composed of multiple layers, each of which converts one volume of activations to another using a differentiable function. The first layer of CNN is input layer which takes the raw pixel values of input image or video frame. The second layer is convolutional layer which computes the output of neurons connected to local district in the input. Each neuron employs a dot product between its weights and a related region where it is connected to the input volume. The dimension of the output computed by this convolutional layer will be the number of filters for this layer. Then, activation layer applies an element-wise activation function such as sigmoid and Rectified Linear Unit (ReLU) to the output of convolutional layer. The results will have the same dimensions as the activation does not shrink or expand the current dimension. Afterwards, pooling layer is a subsampling operation by shrinking the spatial dimension of representation depending on filter, stride, and padding sizes. There can be several connected blocks consisting of convolutional layer and pooling layer. Finally, fully connection layers calculate the class scores from each input to all classes. An example is demonstrated in Fig. 2.8.

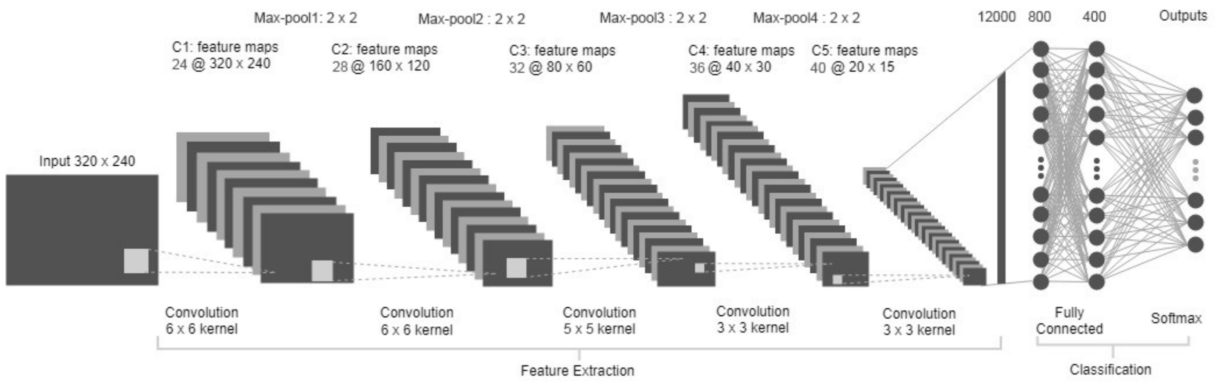

Figure 2.8: An example architecture of CNN consisting of different types of layers: input, convolutional, pooling, fully connected and output layers.

As for the related works with CNNs, Le Cun et al. [139] firstly demonstrated the application of backpropagation networks in image recognition problems. Inspired by Fukushima's computational model [122], they developed a CNN network called LeNet-5. Indeed, they used feature maps to store local features rather than the use of feature detector. Without preprocessing and feature engineering they proposed this network which is fed with normalized images, not with feature vectors. For this purpose, a neuron having a local receptive field examines the input image, and feature map layer stores its states in given locations. The capability of backpropagation networks can be emphasised to tackle low-level information. LeNet-5 is trained to utilise the backpropagation algorithm with 5 layers. As a result, LeNet-5 achieved state-of-the-art performance on the handwritten digit recognition datasets [139]. Nevertheless, LeNet-5 did not perform successfully on the more complex problems due to the fact that it was designed with limited training data and computational power till the recent breakthrough in computer vision.

Hinton et al. made a breakthrough by introducing deep belief networks that were trained by using greedy layer-wise pre-training in 2006 [128]. These networks can be used with labelled, unlabelled, and semi-labelled data. A deep belief network can enhance accuracy and prevent overfitting when they are used with unlabelled data. This net can learn feature representations by itself from the input data, which also decreases the effort of feature extraction required to extract new and relevant features. Therefore, Hinton et al. proposed the use of deep belief networks as feature detectors and classifier with labelled data [128]. After this research, many researchers have developed more techniques to deal with the difficulties within CNN training phase.

In 2012, Krizhevsky, Sutskever and Hinton proposed a deep CNN, known as AlexNet, to classify 1.2 million high-resolution images in the ImageNet Large Scale Visual Recognition Challenge 2010 (ILSVRC2010) into thousand categories [71, 140]. The model trained with deep CNNs with 60 million parameters using an efficient GPU implementation succeeded by achieving by the lowest error rate (17.0%) and outperformed the previous state-of-art methods on the ILSVRC2010 dataset in 2012. The main components introduced in the AlexNet are ReLUs and dropout. ReLUs make the training phase faster by replacing tanh units whilst dropout is proven as a highly effective way to prevent overfitting.

Since 2012, there has been considerable rise in the number of computer vision achievements based on CNNs. Due to the fact that training CNNs is relatively easier and more successful than traditional methods, implementation and understanding CNNs might be more attractive to developers for computer vision such as object classification, object localization and face recognition. Moreover, CNNs can automatically learn feature representations, which eliminates the need for feature engineering.

After AlexNet was implemented in 2012 with the widening of the ILSVRC2012 dataset, the same model further reduced the error rate (15.3%) [71]. In contrast to LeNet-5, AlexNet outperformed the previous classification models using more training data.

Simonyan and Zisserman investigated the depth of CNN and its effects on both accuracy and error rate [72]. They proposed two VGGNet versions by using larger depth than AlexNet (8 layers): VGGNet-16 having 16 layers and VGGNet-19 having 19 layers. Moreover, it has been proven the ability of extracting more details from raw image inputs with very small convolution filters (3×3). Thus, VGGNet made such a significant improvement that the previous configurations can be achieved by pushing the depth to 16–19 weight layers. More networks can be found by following references: GoogleNet [73], ResNet [74], ResNet v2 [141].

Schmidhuber reviewed historical developments of deep learning in neural networks from 1940 until 2013. In that study, many works were summarised under the following subtopics; deep supervised learning, unsupervised learning, reinforcement learning and evolutionary computation [142]. Furthermore, the extended review on deep learning can be found in LeCun et al. [143].

2.6 Information Fusion

Information fusion is the combining process of information from the identical object or scene to attain more complicated, predictable, and accurate information. In the concept of scene understanding, information fusion is the way in which more relevant information from two or more inputs are merged into a single one. The ultimate output gives more information than any of the input sources. Information fusion is applied at multiple levels such as image, feature and decision as shown in Fig. 2.9. In this section, those fusion levels are explained and the related work from literature are presented.

2.6.1 Image Level

Image-level (low-level) fusion is achieved when different pixels from multiple sources are combined. The purpose of image-level fusion is to create a single image which presents a more informative description of the related scene than any of the images. These images to be associated can be taken from multiple same types of sensors or different types of sensors. Therefore, this

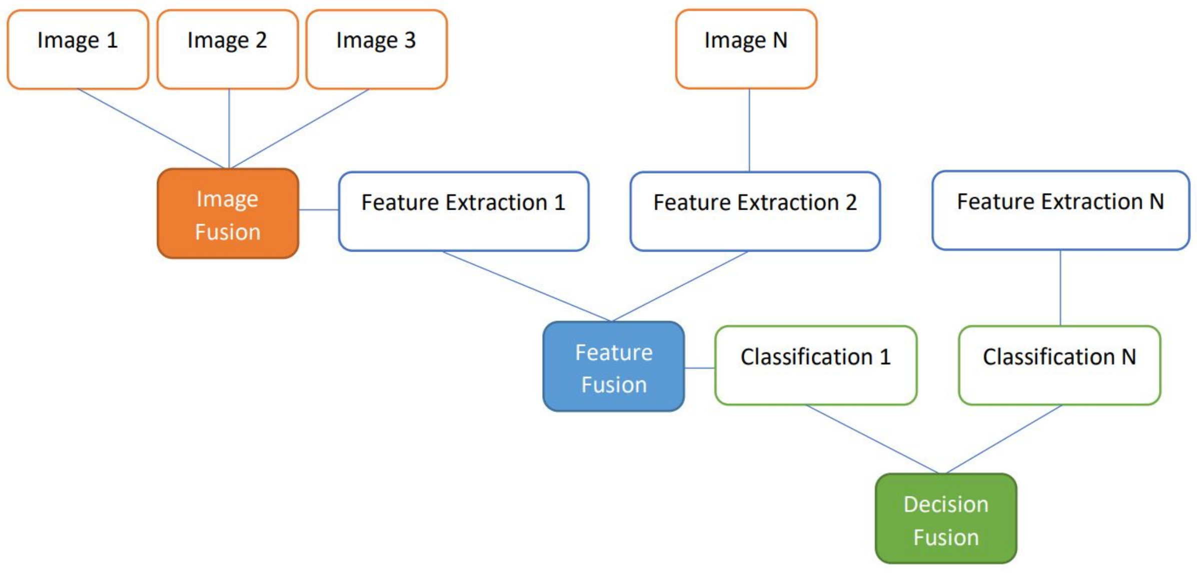

Figure 2.9: Three levels of information fusion for video classification

fusion type is also called as pixel-level multisensor fusion [144]. In computer vision area, image fusion has become a commonly used method for image and video analysis.

Methods for image fusion can be generally divided into two categories: spatial domain fusion and transform domain fusion [145]. Spatial domain fusion approach employs local spatial features by handling the pixel value of an image such as gradient, spatial frequency and local standard deviation. Some well-known examples of spatial domain fusion are averaging, Brovey method, Principal Component Analysis (PCA), Intensity-hue-saturation (IHS), high pass filtering (HPF) [146]. On the contrary, transform domain fusion firstly computes the Cosine transform and Fourier transform of the pixel value of the input image and conduct fusion on the transformed image. Discrete Wavelet Transform , Stationary Wavelet Transform are commonly used techniques for transform domain fusion [147].

Machine learning techniques have been widely employed in recent years to complete various types of image fusion tasks, and have gained significant success in the image fusion area. To fuse multi-focus images, Yang and Li [148] used the sparse representation technique, in which image patches were represented by an overcomplete dictionary and associated sparse coefficients, and the input images were associated by fusing the sparse coefficients of each pair or set of image pieces. Afterwards, further improvements in algorithmic performance and the ability to fuse various types of images were found with a series of sparse representation-based algorithms [149, 150].

Recent advances in deep learning techniques, particularly CNNs, have accelerated the progress of image fusion [151]. CNN has firstly introduced into image fusion area by Liu et al. and achieved the state-of-the-art multi-focus image fusion performance [152]. Two CNNs have been employed to combine spatiotemporal satellite images and extract salient image features on remote sensing data [153]. In this study, Song et al. proposed one CNN to learn a nonlinear mapping between MODIS and low spatial resolution Landsat images and super resolution between low spatial resolution Landsat and original Landsat images [153]. They also proposed a fusion model including high-pass modulation and a weighting technique to recreate the fused image from the extracted features and the proposed method achieved better fusion performance on both MODIS and Landsat datasets. Zhang et al. proposed a CNN-based image fusion framework as IFCNN [154]. They used two convolutional layers to capture important image features of various input images. An appropriate fusion rule is then used to combine the convolutional features of the various input images, depending on the type of input images (elementwise-max, elementwise-min, or elementwise-mean). Two convolutional layers reconstruct the merged features to create the final fusion image. IFCNN achieved better prediction results when compared to the state-of-the-art image fusion techniques over four types of image datasets including multi-focus image datasets, infrared and visual image dataset, multi-modal medical image dataset, and multi-exposure image dataset [154].

A detailed survey and review of the state-of-the-art image fusion studies which employ deep learning are also presented by Zhang et al. [155]. Deep learning-based approaches can achieve better performance than traditional fusion methods with the robust feature extraction and fitting capabilities of neural networks [155].

2.6.2 Feature Level

Feature-level (middle-level) fusion combines features extracted from different input images. Numerous feature extraction techniques are used in computer vision applications, most notably video classification. However, it is impossible for a single feature category to include sufficient information on all the dynamics involved in actions. The feature-level fusion process extracts correlated features from the various modalities and thus identifies a prominent set of features capable of improving recognition accuracy. Therefore, feature fusion techniques have been commonly applied in such applications. Although it is frequently accomplished through the use of basic operations such as concatenation, product, and summation [156, 157, 158], this may not be the optimal method.

Because eliciting a feature set often needs dimensionality reduction techniques, feature-level fusion presupposes the availability of a substantial amount of training data. Recent research on feature fusion has concentrated on deep neural networks due to the rapid improvement of deep learning and the availability of big datasets in image and video domain.

Karpathy et al. [80] looked towards fusing time information along the video and focused on early, late, and slow fusion. First, pixel-level temporal information was accessible by extending first layer time filters over the video (early fusion). Second, two single-frame networks' outputs are joined later in the first completely connected layer (late fusion). Furthermore, deep layers can collect more global information in both spatial and temporal dimensions (slow fusion). They tested their models on a variety of datasets. On all datasets, the slow fusion model outperformed alternative implementations. Thus, they demonstrated the network's ability to learn appealing motion properties. They discovered that slow fusion outperforms early and late fusion models and that features can be transferred to other classification problems via transfer learning [80].

To enhance recognition accuracy, Qin et al. [159] introduced a new feature fusion approach based on a combination of classical descriptors and 3D CNN. This model extracts numerous spatial-temporal features using a variety of classical descriptors and then fuses them with learned features from a 3D CNN to create a specific fusion feature for classification. Wang et al. [160] proposed Loss Switching Fusion Network to combine spatiotemporal descriptors to improve the video classification performance.

Tianyu et al. [161] classified movies using frame-level features; their solution utilises video classification architecture to generate video-level features from a collection of frames and then temporally fuse these features to learn long-term spatiotemporal information for the movie genre classification problem.

Hu et al. [162] proposed 3D CNNs with multi-layer-pooling selection fusion for video classification, where the integrated deep global feature is combined with information from shallow layer of local feature extraction networks, and then cascaded and used for classification. The suggested 3D CNNs with multi-layer pooling selection fusion performed better in classification.

Recently, attention mechanism [163] has been utilised for various studies in computer vision, especially in visual data analysis such as image and video classification [164, 165, 166]. It has been used recently for feature fusion and achieved comparative results to the existing state-of-the-art methods [167, 168].

2.6.3 Decision Level

Decision-level (high-level) fusion combines multi decisions made by different classifiers. It is a type of high-level information fusion. In comparison to low-level and middle-level fusion, high-level fusion is more precise, supports real-time, and also overcomes the limitations of single sensor imaging.

Decision-level fusion algorithms includes voting, Bayesian inference, evidence theory, fuzzy integral, and other operational methods [169]. Related works for decision fusion are presented in Chapter 5.

2.7 Datasets

The dataset being utilised is one of the most crucial parts of any deep learning project. In this section, publicly known human activity recognition datasets are described. Since recently created datasets have been shown to be commonly used for testing, the primary focus is on these datasets.

2.7.1 KTH

This dataset was introduced by the Royal Institute of Technology in 2004 [170]. The KTH dataset consists of six different human actions performed by 25 actors in four different outdoor conditions. The action categories in the dataset are walking, jogging, running, boxing, hand waving and hand clapping, and an example frame from each category is presented in Fig. 2.10. The highest accuracy recorded on this dataset is 98.2% [171].

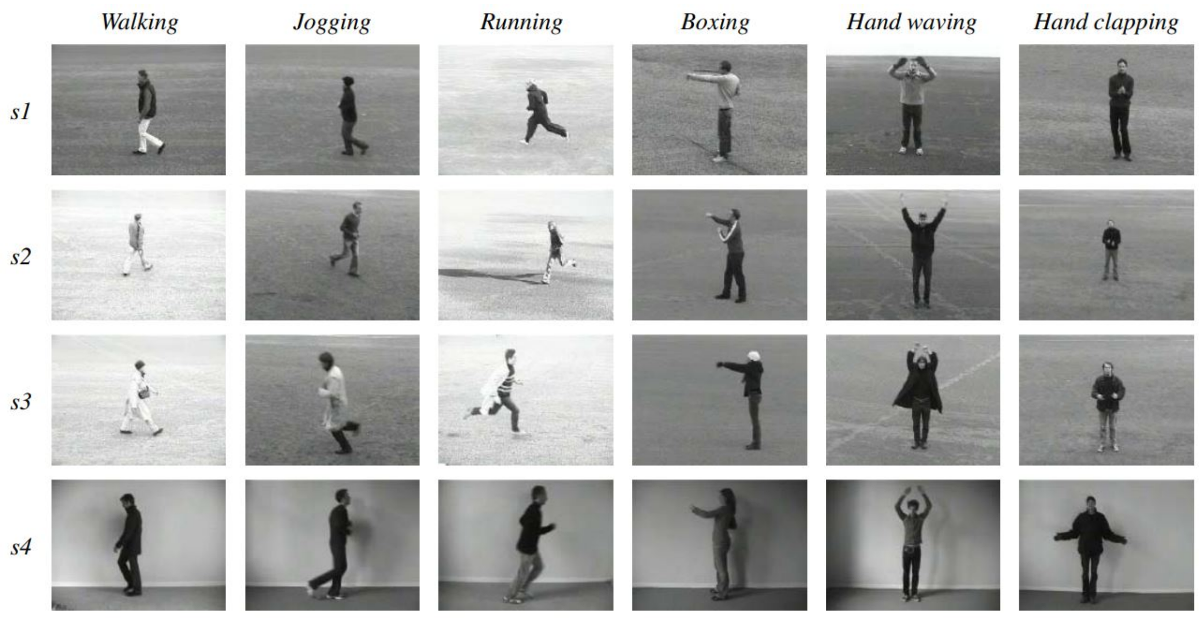

Figure 2.10: The KTH dataset action examples

2.7.2 Weizmann

This dataset was collected and released by the Weizmann Institute of Science in 2005 [172]. The dataset includes 90 video sequences where nine different actors perform basic actions such as running, walking, skipping, jumping, waving hands, and bending. There were many researches using this dataset and a few approaches achieved 100% accuracy on this dataset [173, 174, 175, 176].

2.7.3 UCF Sports

This dataset was introduced by the Centre for Research in Computer Vision at University of Central Florida in 2008 [177, 178]. The UCF Sports dataset contains 150 videos from ten action categories as listed in Fig. 2.11. The state-of-the art performance on this dataset was achieved by [171] with 95% accuracy.

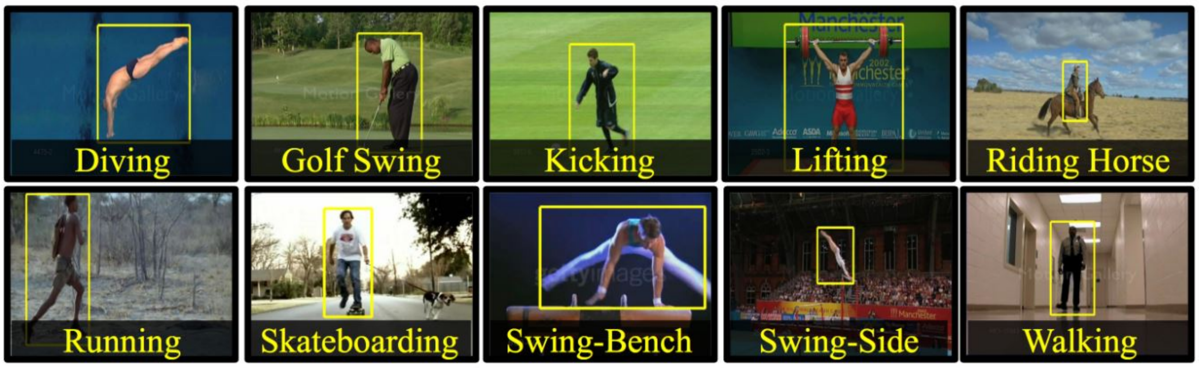

Figure 2.11: The categories of the UCF Sports dataset

2.7.4 HMDB-51

The HMDB-51 dataset was introduced by Serre Lab at Brown University in 2011 [179], which was compiled from a variety of sources, primarily films, but also from public databases such as the Prelinger archive, YouTube, and Google videos. The dataset contains 6849 clips grouped in 51 activity categories, each of which contains at least 101 clips. The highest classification accuracy announced on this dataset is 87.56% by Wang and Koniusz [180] using object and saliency descriptors with three levels of weighted mean pooling.

2.7.5 UCF-101

The UCF-101 action recognition dataset [101] was generated in 2012 at the University of Central Florida's Center for Research in Computer Vision. It is one of the most popular benchmarking dataset. The collection contains 13320 clips representing 101 distinct classes with a frame size of 240 to 360 pixels. The UCF-101 contains clips with a fixed frame rate of 25 frames per second. The categories of the dataset are illustrated in Fig. 2.12 and more details about the dataset are presented in research chapters (Chapter 3, 4, 5, and 6). The state-of-the art performances are achieved with 98.64% by Gowda et al. [100] and 98.6% by Duan et al. [99] and Li et al. [113].

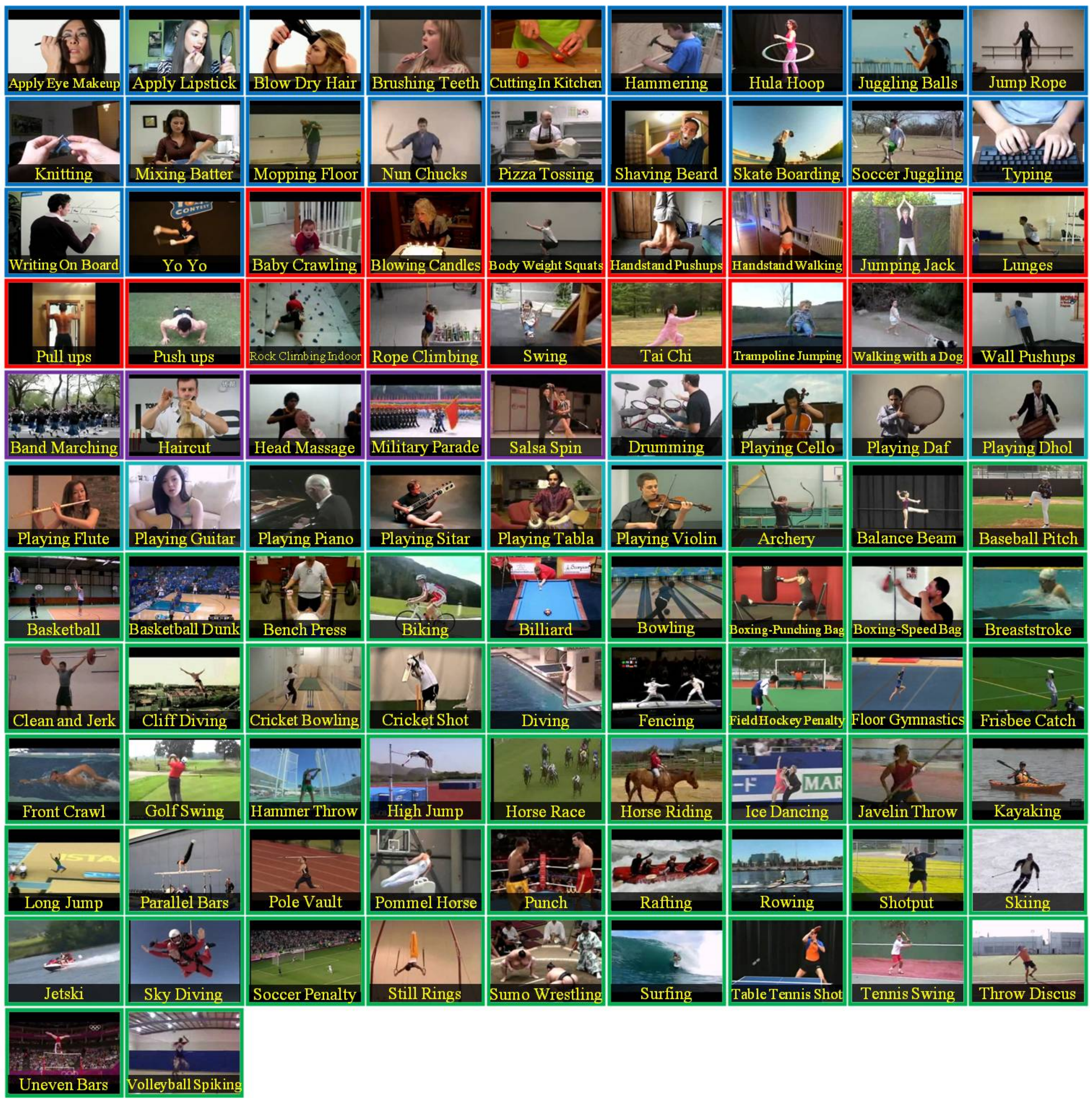

Figure 2.12: The categories of the UCF dataset [101]

2.7.6 YouTube

The YouTube action dataset was introduced by Liu et al. [181] in 2009. It covers 11 activity categories, including basketball shooting, biking/cycling, diving, golf swinging, horseback riding, soccer juggling, swinging, tennis swinging, trampoline leaping, volleyball spiking, and dog walking. Due to the wide range of camera movements, object appearance and posture, object scale, viewpoint, cluttered background, and illumination conditions, this dataset presents significant challenges. Each category's videos are divided into 25 groups, each of which contains at least four action segments. The video clips in a collection may share certain common characteristics, such as the same performer, a similar background, or a similar perspective. Khan et al [182] has recently achieved 100% accuracy on this dataset using fusion of hand-crafted features with CNN.

2.7.7 Choice of Datasets

In this thesis, the KTH and UCF-101 datasets were utilised, which are commonly used for video classification studies and facilitate benchmarking algorithms. The KTH dataset was used for initial experiments as it is considered a small-sized dataset with six action categories. The experiments were repeated on the UCF-101 dataset which gives the largest diversity in terms of actions with 101categories.

The major advantage of using these datasets is that validating results on two datasets contributed to obtaining more convincing results. Another advantage is that both datasets are publicly available and contain various human actions. However, the main disadvantages of these datasets are the KTH has a limited number of categories with few videos and the UCF-101 is a challenging dataset due to the variety of variables involved such as background changes, camera motion, different lighting, and pose. When considering that deep neural networks require a large volume of training data and various datasets to achieve better prediction performance, the KTH and UCF-101 datasets were chosen as they provided adequate video data to conduct the experiments in this thesis.

2.8 Open Problems and Research Questions

Previous studies on human action recognition through video classification have identified a number of significant problems. Fundamentally, developing an accurate and reliable action recognition system is difficult due to the variety of variables involved in real-world video data, such as background changes, camera motion, different lighting, and pose [183, 184].

Deep neural networks require large amount of data to obtain promising results [185]. In the context of video classification, the volume of data causes the main difficulty. The volume of data increases the model complexity and reduce the performance due to the existing redundant information within the movies [23].

The trend is using larger networks to get better classification performance. However, the number of parameters in such networks is getting larger. Thus, the optimisation of network architectures is a challenging issue. Moreover, when training a network for a specific domain, it may not perform well in another domain [186].

Most of the existing video classification techniques are extended from image-based classification techniques [80]. Video data naturally involves time information and using this effectively can improve the video classification performance [187]. In the literature, there are a lot of works conducted to get benefit from time information, but this is still a challenging problem [72, 81, 188, 189]. Most of the recent works use the 3D CNN to get benefit from video temporal information [54, 162, 190, 191], however, it has a high computational cost and excessive memory usage.

Video data contains numerous frames and most current deep learning systems treat videos by labelling each frame. Nonetheless, treating all video frames identically affects efficiency since some frames have more distinct information than others [23]. Finding keyframes improves classification performance and this is needed to be applicable for the existing systems [192]. Existing keyframe extraction methods consider the similarity between consecutive frames to find frame differences, however, major changes between consecutive frames due to uncertain factors such as camera movements, lighting changes, etc. can highly affect the performance of those methods.

Video classification architectures in the literature focus on both appearance and temporal representations to get better classification decision and apply different level of fusion methods. However, combining spatial and temporal features extracted from video input is still challenging problem because of the modality in the video domain [81, 88, 193, 194]. The disadvantage of these architecture is that it cannot be trained end-to-end since optical flow must be computed separately and both streams must be trained independently. While the spatial stream may be trained on large image datasets, the temporal stream requires training on a video dataset. As a result, transfer learning is inapplicable to this design entirely. Additionally, the preprocessing necessary to compute optical flow makes real-time capabilities of this technique challenging.

Training a network requires huge amount of data; therefore, the machine should be equipped with sufficient power. Nowadays, the deep learning solutions for real-world problems require larger memories and multi-core high performing GPUs [195]. The cost of these units and the energy consumption are still problems.

The primary goal of this research is to develop a video classification system that is capable of capturing and recognising human actions in real-world videos through the use of novel techniques involving deep learning, computer vision, and various levels of information fusion. More precisely, it seeks solutions to the following research questions:

1. How to effectively extend deep neural networks for image classification to video classification? *(Chapter 3: Combining Very Deep Convolutional Neural Networks and Recurrent Neural Networks for Video Classification)*
2. How to select discriminative frames for video classification? *(Chapter 4: Criteria and Methods for Keyframe Selection)*
3. How to extract and make use of motion information for video classification? *(Chapter 5:*

Extracting Motion Information and Decision Fusion for Video Classification)

4. How to combine spatial and temporal features along with different decision for video classification? *(Chapter 6: Combination of Feature and Decision Fusion for Video Classification)*

Chapter 3

Combining Very Deep Convolutional Neural Networks and Recurrent Neural Networks for Video Classification

CNNs have been demonstrated to be able to produce the best performance in image classification problems. RNNs have been utilised to make use of temporal information for time series classification. The main goal of this study is to examine how temporal information between frame sequences can be used to improve the performance of video classification using RNNs. Using transfer learning, this work presents a comparative study of seven video classification network architectures, which utilise either global or local features extracted by VGG-16, a very deep CNN pre-trained for image classification. Hold-out validation has been used to optimise the ratio of dropout and the number of units in the fully-connected layers in the proposed architectures. Each network architecture for video classification has been executed a number of times using different data splits, with the best architecture identified using the Independent-Samples T-test. Experimental results show that the network architecture using local features extracted by the pre-trained CNN and ConvLSTM for making use of temporal information can achieve the best accuracy in video classification.

3.1 Introduction

In recent years, the number of published videos has exponentially increased due to wider access to the Internet and more accessible hardware. Analysing the content of videos is essential for both users and suppliers. Users want to know what type of movies they watch and to reach the videos which are related to their interests. Similarly, suppliers are required to manage advertisements and content to suggest interest-based videos to users. For these purposes, video classification has become an important research area in recent years.

However, the classification of high-dimensional data is a challenging task. Action recognition from videos is a highly active research area with state-of-the-art systems still being far from human performance.

CNN has been demonstrated as a powerful architecture for image classification achieving state-of-the-art results in recent years. The performance of CNN in action recognition from videos is not as satisfied as those achieved in image-based tasks such as detection [196], pattern recognition [197], segmentation [198] and classification [71, 72]. While deep networks have been shown to be remarkably effective in the mentioned image processing tasks, it is still unclear how to properly extend these methods to the video domain.

Both the extension of CNNs to video classification and the use of RNNs to understand video content have achieved relatively outstanding results [84, 199]. However, video classification remains a challenging problem because of the temporal information which can be used to achieve better performance. Thus, motivated by these results, in this study seven different network architectures are proposed to classify human actions from videos using RNNs on top of either local or global features extracted by the CNN architecture called VGG-16 that was pre-trained on the ImageNet dataset. Experiments are conducted to explore appropriate approaches to using temporal information by RNNs and find out whether local or global features extracted by the pre-trained CNN can achieve better video classification performance.

The rest of the chapter is organised as follows: Related work is reviewed in Section 3.2 and the proposed methods are described in Section 3.3. Experiments are explained in Section 3.4, and results are reported in Section 3.5, followed by conclusion in Section 3.6.

3.2 Related Work

There are several approaches to video classification, ranging from text-based [68], audio-based [200] to visual-based. In this study, the visual-based approach is followed due to the rich information in a visual context.

Traditional feature extraction methods are usually time-consuming and require feature engineering and domain knowledge. The recent trend to extract robust feature representations is applying deep learning to the raw image or video data. CNNs involve two main parts: a sequence of convolution and pooling layers for feature extraction and densely connected layers for classification. Pre-trained networks have been used to extract new features from different instances using their first part. In the case of convolutional networks, the convolutional part of the pre-trained network is included in the place of the feature extractor to obtain interesting features from new samples. Therefore, representations learned by the convolutional part of the previously trained model on large datasets can be used without domain knowledge [71, 72].

The great success in learning robust feature representations with CNN from raw images has encouraged the use of deep features for video classification. The common idea is to handle a video as an accumulation of consecutive frames. The feature representations could be extracted using a feed-forward pass up to a specific fully-connected layer with pre-trained deep models on the ImageNet [70] such as AlexNet [71], VGGNet [72], GoogleNet [73], and ResNet [74]. Then, these features can be fused as representations at the video level as the input of a classifier for video classification.

In a CNN architecture for video classification, the main issue is the way of combining appearance and motion information. The existing architectures can be categorised into two approaches:

single-stream and two-stream networks.

Single-stream networks handle spatiotemporal features together in contrast to two-stream networks. Karpathy et al. conducted an empirical evaluation of CNNs considering the methods to fuse local temporal information from frame sequences [80]. Their network, which benefits from integrating spatiotemporal information, made significant improvements in classification accuracy in contrast to traditional feature-based baseline methods (55.3% to 63.9%). They showed the capability of CNNs in learning features from video data. They suggested that local motion information might not be very essential especially for the dynamic datasets. However, Castro et al. [82] highlighted the importance of motion information in video classification challenge by showing that the use of visual information only is inadequate in motion-based action recognition.

Zha et al. utilised an image-trained CNN to extract features from different layers of deep models [79]. They showed that CNN features can gain satisfactory recognition performance in video classification by using the motion information added via late fusion on the UCF-101 dataset. However, the spatiotemporal features learnt could not capture movement information properly in single-stream networks.

Two-stream networks tackle with spatial and temporal information in separate networks. Simonyan and Zisserman proposed a two-stream CNN which involves spatial and temporal streams merged by fusion [81]. They used the temporal stream on multi-frame dense optical flow to recognise actions from movements while the spatial stream to recognise actions on raw video frames. Their two-stream model achieved better results on the UCF-101 and HMDB-51 datasets than the models consisting of either spatial or temporal stream only [81]. Following this work, the two-stream architecture has been extended in other studies.

Tran et al. proposed a deep 3D CNN as a feature extractor by using a sequence of RGB frames. The deconvolutional layer is used to interpret model decisions in their study. They also showed that 3D CNNs are more convenient than 2D CNNs to learn spatiotemporal features [54].

Yao et al. proposed a novel 3D CNN-RNN encoder-decoder which can capture both local spatial and temporal information. With an attention mechanism within this framework, they capture global context as well [90].

Feichtenhofer et al. introduced a novel two-stream network fusion architecture. Two streams are fused after the convolution layer, and the output is fused for spatiotemporal loss evaluation [88].

Carreira and Zisserman compared two-stream architectures consisting of a combination of 3D models. Two 3D networks are employed for two streams. The spatial stream includes frames stacked in time dimension rather than single frames. The pre-trained two-stream network on ImageNet and Kinetics datasets outperformed the existing models [92].

3.3 Methods

In this section, the proposed architectures of the networks are described for action recognition in detail. The methods for preprocessing and feature extraction are also described.

3.3.1 Preprocessing

To capture the temporal information between the consecutive frames, the combination of multiple techniques were used. The primary preprocessing was removing the very similar consecutive frames by extracting one frame per second. Secondly, the remaining frames were resized using an interpolation technique from the original image frame size of 320x240 to 240x240.

3.3.2 Feature Extraction

The VGG-16 was trained on 1000 images per each of 1000 categories in 2014 ILSVRC using a subset of ImageNet dataset by Simonyan and Zisserman [72]. It consists of 16 convolutional layers with quite small convolution filters (3x3). It has been shown that the learnt representations can be generalised to other datasets; therefore, the pre-trained VGG-16 was used as a feature extractor for video classification. VGG-16 can be used to extract both local features employing only the convolutional base and global features including the fully-connected layers, as well.

3.3.3 Different VGG-16 Based Architectures for Video Classification

Using transfer learning, in this study, seven video classification architectures are proposed as shown in Fig. 3.1 for action recognition using the UCF-101 dataset. Hold-out validation is used to determine the dropout parameter, the number of units in both the fully-connected layers and the LSTM, number of filters and kernel size in the ConvLSTM. The layers in the pre-trained VGG-16 are fixed and used for feature extraction, and the newly added fully-connected layers are trained using the UCF-101 training data for video classification.

As baseline models, VGG-16-VOTE (a) and VGG-16-VOTE (b) are built based on the general approach which treats frames as single images. For each frame of a video, softmax produces a score for each possible video class. VGG-16-VOTE determines the class of the video by voting in terms of the maximum score among all the frames. The features extracted by the pre-trained VGG-16 convolutional base are fed into the fully connected layer (FC-512) in VGG-16-VOTE (a). To explore the effect of global features, VGG-16-VOTE (b) includes the fully-connected layers of VGG-16.

LSTM is one of the most common approaches for sequence modelling, which has been demonstrated as a robust method for representing long-range dependencies in the previous studies [85] and [138]. The main advantage of LSTM is having its memory cell c_t which accumulates the state information. The cell is modified by controlling the input gate $i_t$and forget gate f_t. Once the cell is fed with new input, it accumulates input information provided that the input gate is on. If the forget gate is activated, the previous cell information c_{t-1} could be forgotten. The output gate o_t checks the current cell output c_t to decide whether it is propagated to the final state h_t or not. In this study, the hidden layer function of LSTM is used as follows [201],

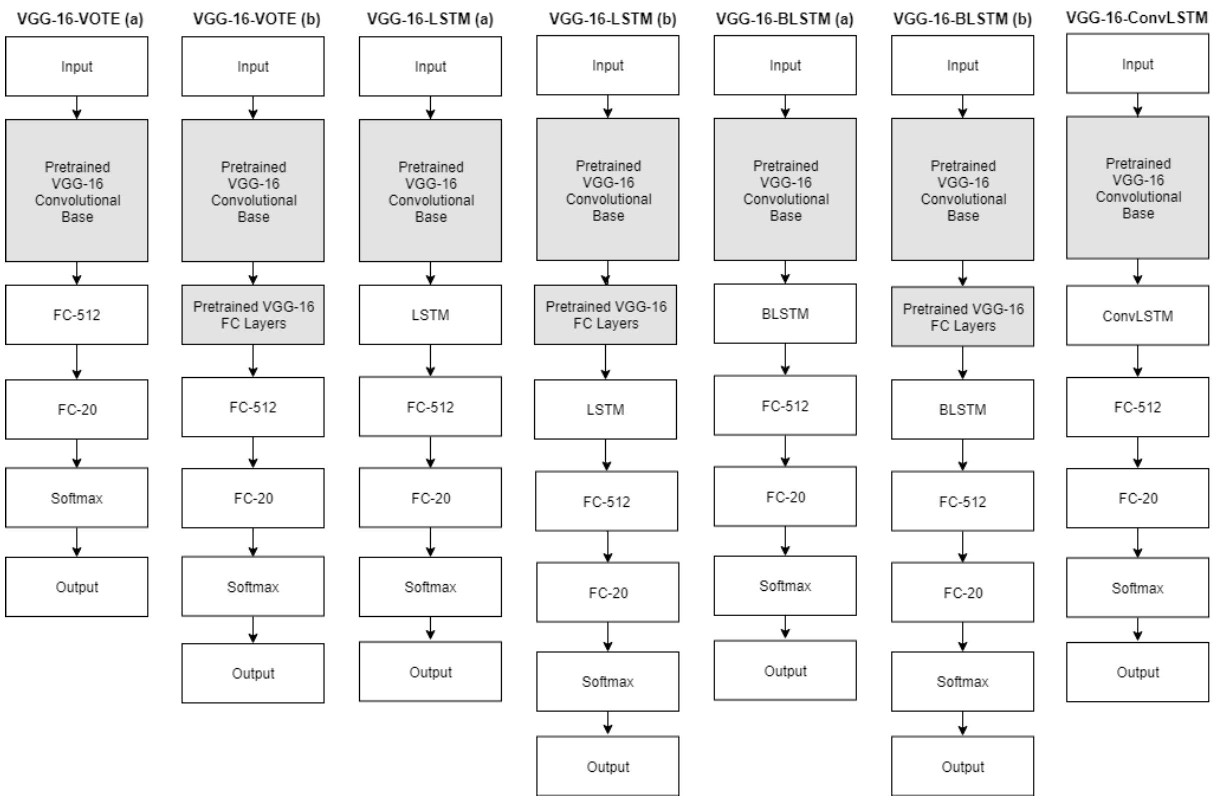

Figure 3.1: The architectures of networks used in the experiment VGG-16-VOTE(a) and VGG-16-VOTE(b) are designed as baseline methods using the pre-trained VGG-16 by [72].

where '·' denotes the Hadamard product:

$$\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \\
c_t &= f_t \cdot c_{t-1} + i_t \cdot \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co} \cdot c_t + b_o) \\
h_t &= o_t \cdot \tanh(c_t)
\end{aligned} \tag{3.1}$$

In VGG-16-LSTM (a) and VGG-16-LSTM (b), the extracted features from video frames are fed into LSTM to access spatiotemporal information. The features for VGG-16-LSTM (a) are local, extracted by the VGG-16 convolutional base, whilst those for VGG-16-LSTM (b) are global, extracted by the VGG-16 fully-connected layers (See Fig. 3.1). The number of units in the output space was set to 512 and ReLU was used as an activation function.

As the second type of RNN used in this study, Bidirectional Long Short-Term Memory (BLSTM) was implemented over the features captured by the VGG-16. The core idea behind BLSTM is connecting two hidden layers of both directions to the same output in order to access information from both the previous and the next status together. Therefore, BLSTM supplies access to long-range sequences in both input directions in contrast to LSTM. BLSTM was applied on top of either local or global features. VGG-16-BLSTM (a) and VGG-16-BLSTM (b) are developed as shown in Fig. 3.1. The parameters of VGG-16-BLSTM (a) and VGG-16-BLSTM (b) were arranged the same as in VGG-16-LSTM (a) and VGG-16-LSTM (b).

An end-to-end trainable ConvLSTM was proposed "by extending the fully connected LSTM to have convolutional structures in both the input-to-state and state-to-state transitions" for precipitation nowcasting [202]. It was shown that the ConvLSTM extracted better spatiotemporal correlations than the fully-connected LSTM for precipitation nowcasting [202]. The formulation of ConvLSTM is applied described as follows [202], where '$\circledast$' and '$\cdot$' denote the convolution operator and Hadamard product, respectively:

$$\begin{aligned}
i_t &= \sigma(W_{xi} \circledast \mathscr{X}_t + W_{hi} \circledast \mathscr{H}_{t-1} + W_{ci} \cdot C_{t-1} + b_i) \\
f_t &= \sigma(W_{xf} \circledast \mathscr{X}_t + W_{hf} \circledast \mathscr{H}_{t-1} + W_{cf} \cdot C_{t-1} + b_f) \\
C_t &= f_t \cdot C_{t-1} + i_t \cdot \tanh(W_{xc} \circledast \mathscr{X}_t + W_{hc} \circledast \mathscr{H}_{t-1} + b_c) \\
o_t &= \sigma(W_{xo} \circledast \mathscr{X}_t + W_{ho} \circledast \mathscr{H}_{t-1} + W_{co} \cdot C_t + b_o) \\
\mathscr{H}_t &= o_t \cdot \tanh(C_t)
\end{aligned} \tag{3.2}$$

Inspired by the mentioned study, the ConvLSTM is used to construct a new architecture for video classification, VGG-16-ConvLSTM as shown in Fig. 3.1, by taking the advantage of its capacity in capturing spatiotemporal information throughout time series. One ConvLSTM layer was added on top of the spatial feature maps extracted by VGG-16 and used the hidden states for video classification. The ConvLSTM layer has 64 hidden states and 7x7 kernels, and the stride of convolution is set to 2 in this experiment.

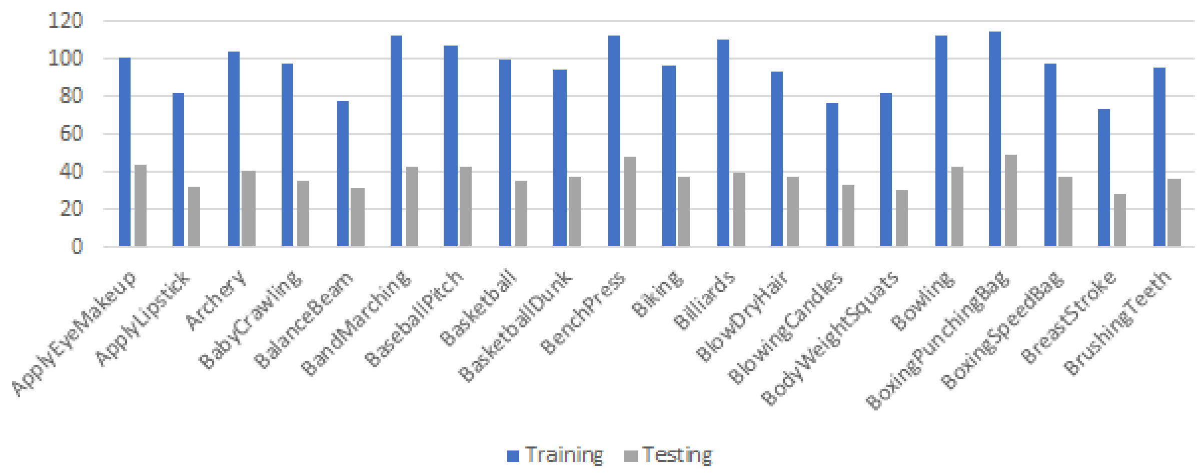

Figure 3.2: The distribution of categories in training and test splits

3.4 Experiments

In this section, the dataset, experiment design and experiment setup and evaluation are described.

3.4.1 Dataset

In this study, the UCF-101 dataset is used to evaluate the neural network architectures utilised to classify human actions from video clips. The dataset contains 13320 clips from 101 non-overlapping classes, with a frame size of 240 to 360 pixels. UCF-101 has defined 3 training-testing splits, which facilitates benchmarking algorithms. It is one of the most popular action recognition datasets and publicly available. Therefore, the UCF-101 dataset is employed in this study. In the experiments, only the first twenty categories of the dataset were used (due to limited time and computing facility), and the first training-testing split was followed to generate train and testing data. Fig. 3.2 presents an overview of the distribution of categories in both training and testing data. In addition, Fig. 3.3 shows some training and testing examples from the UCF-101 dataset.

In this experiment, the hold-out method was used to split training data into the following two groups: 70% of training data for training and 30% of training data for validation. The testing dataset was never used during training and validation, but only used for producing the testing accuracy of each network architecture.

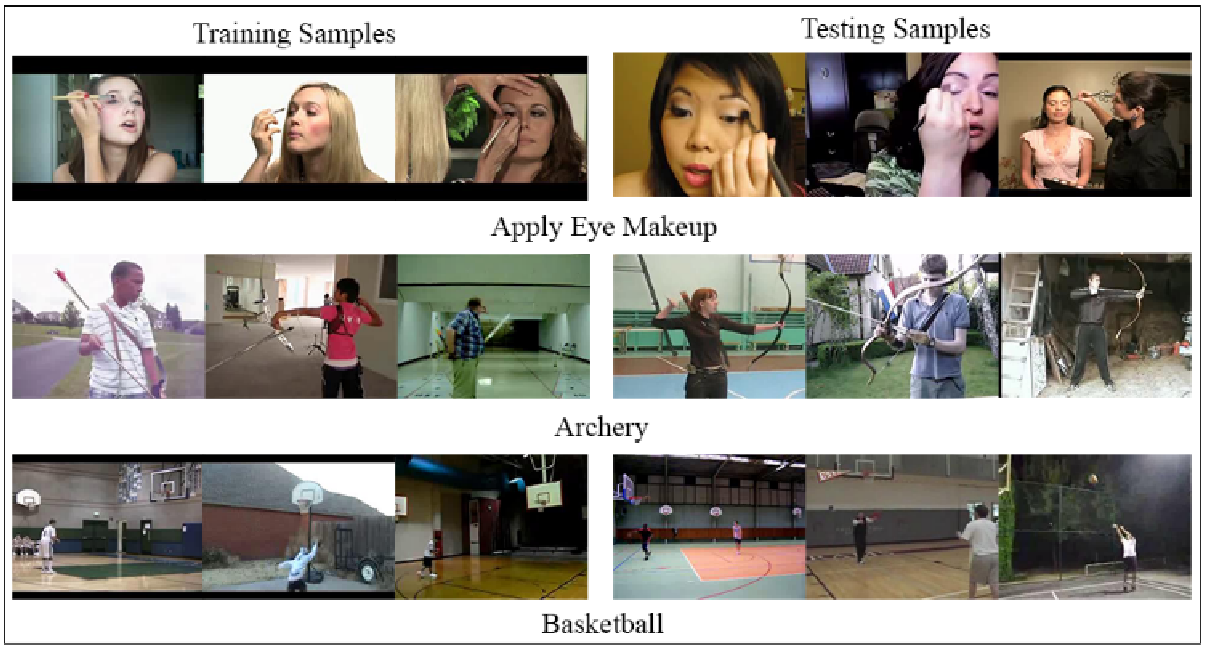

Figure 3.3: (a) Training samples on the left; (b) Testing samples on the right from the UCF-101 Human Action Dataset. Samples are from three different categories: Apply Eye Makeup, Archery, and Basketball.

3.4.2 Experiment Design

During the training process, parameter tuning was applied with the hold-out validation technique. Specifically, if the training is finished with the first possible set of parameters, the next training begins with the second set of possible parameters, and so on. When the validated training is finished, the scores of both training and validation are averaged. The best parameters are identified based on the validation scores. Then, the model with the best parameters is evaluated on the testing data by predicting unseen test videos' classes.

3.4.3 Experiment Setup

The proposed network architectures are implemented by using Tensorflow-gpu v1.12 on an NVIDIA GTX Titan X GPU using the CUDA v9.0 toolkit. The batch size is set to 128, and the cost is minimised by using the Stochastic Gradient Descent (SGD) optimiser. Dropout is used as a regularisation method. The neurons are kept active with the probability $p = 0.5$ after the first fully-connected layer.

Table 3.1: Average accuracy scores achieved by different architectures

Model	Training Accuracy	Validation Accuracy	Testing Accuracy
VGG-16-ConvLSTM	99.24%	97.45%	82.04%
VGG-16-LSTM(a)	98.86%	97.91%	81.27%
VGG-16-BLSTM(a)	97.76%	95.19%	76.20%
VGG-16-VOTE(a)	78.71%	76.55%	73.20%
VGG-16-LSTM(b)	78.41%	79.47%	67.62%
VGG-16-VOTE(b)	73.70%	67.05%	64.04%
VGG-16-BLSTM(b)	89.17%	75.37%	61.18%

3.4.4 Evaluation

In this experiment, classification accuracy is used as a performance metric, which is the percentage of correctly classified video instances. In the experiment, seven different video classification architectures were compared based on their accuracy scores through a statistical significance test. 14 training, validation, and testing accuracy scores were collected for each architecture under the same conditions but different data splits for training, validation and testing. In this context, the dependent variable is testing accuracy while the independent variable is classification architecture. A significance level of 0.05 was considered during the analysis. The distribution of performances was evaluated using the Shapiro-Wilk test, which shows that the testing accuracy scores are normally distributed at the 0.05 level of significance. Moreover, T-test was applied to compare pairs of network architectures in terms of testing accuracy and reported t-scores and p-values in Section 3.5.

3.5 Results and Discussion

In this study, the first twenty categories of the UCF-101 data set have been used to obtain results. Training and testing data were separated using the first part of the train/test splits enabling benchmarking algorithms. The hold-out validation technique was applied to identify the optimal values for the dropout parameter, the number of units in LSTM, the number of filters, strides, and kernel size in ConvLSTM, and the number of units in dense layers.

Table 3.1 demonstrates the average of 14 results for each architecture. It is clear that VGG-16-

Table 3.2: Independent-Samples T-Test results

Architecture	Mean	Std.	*t*	*df*	*p*
VGG-16-ConvLSTM	82.04%	0.01151	2.109	22.473	.046
VGG-16-LSTM(a)	81.27%	0.00757			
VGG-16-ConvLSTM	82.04%	0.01151	7.070	17.076	.000
VGG-16-BLSTM(a)	76.20%	0.02871			
VGG-16-ConvLSTM	82.04%	0.01151	9.196	15.929	.000
VGG-16-VOTE(a)	73.20%	0.03407			
VGG-16-ConvLSTM	82.04%	0.01151	29.357	24.821	.000
VGG-16-LSTM(b)	67.60%	0.01436			
VGG-16-ConvLSTM	82.04%	0.01151	50.232	21.222	.000
VGG-16-VOTE(b)	64.04%	0.00687			
VGG-16-ConvLSTM	82.04%	0.01151	35.477	21.579	.000
VGG-16-BLSTM(b)	61.18%	0.01875			
VGG-16-LSTM(a)	81.27%	0.00757	6.387	14.79	.000
VGG-16-BLSTM(a)	76.20%	0.02871			
VGG-16-LSTM(a)	81.27%	0.00757	8.643	14.28	.000
VGG-16-VOTE(a)	73.20%	0.03407			
VGG-16-LSTM(a)	81.27%	0.00757	31.492	19.704	.000
VGG-16-LSTM(b)	67.60%	0.01436			
VGG-16-LSTM(a)	81.27%	0.00757	63.025	25.761	.000
VGG-16-VOTE(b)	64.04%	0.00687			
VGG-16-LSTM(a)	81.27%	0.00757	37.164	17.127	.000
VGG-16-BLSTM(b)	61.18%	0.01875			

ConvLSTM outperformed the remaining architectures (82.04%). It is followed by VGG-16-LSTM (a) (81.27%). VGG-16-BLSTM (a) was the third best architecture (76.20%). Interestingly, the baseline method with local features, VGG-16-VOTE (a), achieved higher accuracy than the architectures extracting global features.

Regarding the results on the testing dataset, the Independent-Samples T-test was employed for possible pairs of the network architectures to identify whether the difference in their testing accuracy is statistically different. Table 3.2 provides the results of the T-Test, where t, df and p represent t-score, degree of freedom, and probabilistic significance, respectively. There is a significant difference between all architectures at the 0.05 level of significance.

The best model is VGG-16-ConvLSTM, and the second-best model is VGG-16-LSTM (a) without the fully-connected layers of the pre-trained VGG-16. The third-best architecture is VGG-16-BLSTM (a) which performs well using local features captured by the VGG-16. These results show that ConvLSTM plays an important role in improving the video classification performance by making better use of temporal information between frames. They also show that local features extracted by the pre-trained CNN are more effective than its global features for video classification. Because the pre-trained CNN was constructed on a large number of training images, it can overcome overfitting problems to some extent, especially when the local features extracted by the pre-trained CNN are used for video classification, as indicated by the results in Table 3.2.

3.6 Conclusion

This study proposes seven network architectures for video classification by incorporating spatial and temporal features extracted from the VGG-16, LSTM, BLSTM and ConvLSTM. First, the baseline models were optimised, which are VGG-16-VOTE (a) and (b) with local and global features, respectively. Then, ConvLSTM and LSTM were optimised to extract higher-level features from frame sequences, based on the features extracted by the pre-trained CNN (VGG-16). Experimental results show that combining pre-trained CNN with ConvLSTM achieved the highest performance among the seven network architectures for video classification, demonstrating the importance of effective integration of spatiotemporal information in video classification.

Chapter 4

Criteria and Methods for Keyframe Selection

CNNs and RNNs produce a powerful architecture for video classification problems as spatial-temporal information can be processed simultaneously and effectively. Using transfer learning, this study presents a comparative study to investigate how temporal information can be utilised to improve the performance of video classification when CNNs and RNNs are combined in various architectures. To enhance the performance of the identified architecture for an effective combination of CNN and RNN, a novel action template-based keyframe extraction method is proposed by identifying the informative region of each frame and selecting keyframes based on the similarity between those regions. Extensive experiments on the KTH and UCF-101 datasets with ConvLSTM-based video classifiers have been conducted. Experimental results are evaluated using one-way ANOVA, which reveals the effectiveness of the proposed keyframe extraction method in the sense that it can significantly improve video classification accuracy.

4.1 Introduction

Video has become more popular in many applications in recent years due to increased storage capacity, more advanced network architectures, as well as easy access to digital cameras, especially in mobile phones. According to recent statistics, more than 500 hours [1] of video are

[1] https://www.omnicoreagency.com/youtube-statistics

uploaded onto the Internet every minute and the sharp rise in the number of videos is expected to continue in the coming decades due to the increase in demand for video content. Therefore, this increase is a remarkable issue and brings serious challenges for video indexing, archiving, and retrieval systems. The main subject of videos on social networking websites is human actions. Automatic classification of their semantic content is essential for the appropriate use and management of these videos. However, the classification of video content remains a challenging task owing to the complexity of video data.

Action recognition problems have been addressed using deep learning approaches in both image and video domains. CNNs have achieved state-of-the-art results in the recent decade. CNN applications to video-based tasks are not so successful as those in image domains, e.g., classification [71, 72], object detection [196], pattern recognition [197], and segmentation [198]. Therefore, the power of RNNs in sequence learning has been employed to gather temporal information for improving video classification performance. Although combining CNNs and RNNs has achieved satisfactory results [84, 199], the representation of temporal information is still a demanding problem due to complex variations in actions and dynamic background in videos.

The performance of action recognition has been improved remarkably by transfer learning and use of extra training data. Extensive video datasets such as HMDB51 [179], UCF-101 [101], Sports-1M [80], and Youtube-8M [203] have been published and the state-of-the-art results have recently been reported on these benchmarking datasets [92, 98, 99, 191, 204].

The majority of the current video classification methods classify videos by assigning a label to each frame. Nonetheless, considering all frames equally weakens the classification performance as some frames have more distinctive information than others. It is argued that selecting keyframes for better classification performance is essential. Thus, this study proposes a novel keyframe extraction method by identifying an action template to preserve the succinct content, in which the entire video is represented in a set of keyframes. The main novelty of this work is the proposed keyframe extraction algorithm that employs an action template for each video to extract and select the most distinctive frames with both static and dynamic backgrounds, without the need of using complex procedures.

The main contributions of this work are the identification of the best architecture for combining CNNs and RNNs for video classification and the proposal of the action template-based keyframe extraction, which aims to extract more informative frames by calculating the similarity only between action regions, rather than whole frames. The former was partly presented in [5], which has been substantially extended and serves as a baseline to test the newly proposed method. In this study, extensive experiments have shown that the action template-based keyframe extraction method significantly outperformed the frame selection methods used in the experiments for comparison purposes.

The rest of the chapter is organised as follows: related work is reviewed in Section 4.2 and the proposed keyframe extraction method is described in Section 4.3. Experiments conducted are detailed in Section 4.4, while the results are analysed and summarised in Section 4.5. Conclusions are given in Section 4.6.

4.2 Related Work

Keyframe extraction approaches can be generally categorised into six groups: uniform sampling-based, shot boundary-based, shot activity-based, visual content-based, motion analysis-based, and clustering-based. Although uniform sampling-based methods are easy and computationally efficient, these methods may fail to represent the video in two possible scenarios: no enough keyframes for a short semantically important video and too many keyframes with similar content for a long static segment [205].

Early works on keyframe extraction focused on shot boundary-based techniques [19, 206]. Basically, this technique employs the first or middle frame of each shot as the keyframe after shot boundary detection [207]. Video shot boundary detection methods were reviewed by Pal et al. [207]. Although shot boundary-based methods are easy to use and generalise, the extracted keyframe cannot represent the visual content entirely and it is not stable.

Shot activity-based approach is used to select keyframes considering the frame with the least different from other frames in terms of a given similarity measure. Based on this concept, Zhuang et al.[208] proposed a keyframe selection method with the assumption that "every keyframe represents a contiguous interval in a shot". In this work, the limits of intervals and the location of the keyframe within each interval are optimised. Similarly, the Lloyd-Max algorithm is used in the design of a scalar quantizer in [209].

Visual content-based approach has been explored for visual content-based information retrieval and keyframe based video summarization. In this approach, visual features of video clips are extracted to analyse keyframes in movie segments. Zhong and Smoliar proposed an integrated system solution using video content information obtained from a parsing process [210]. Human attention mechanism has been simulated to produce a semantic video summary based on keyframe extraction. Visual attention of each frame is quantified using a descriptor named attention quantifier, which indicates colour conspicuousness and the motion with more attention involved [211]. There have been many attempts to analyse the visual content of video for keyframe extraction for video partitioning and summarization [212, 213].

As for motion analysis-based approach, a novel algorithm was proposed for selecting keyframes within shots from video by employing optical flow computations to detect local minima of action in a single shot [22]. This work measures the motion in a shot by utilising optical flow analysis, where keyframes are selected at the local minima of the action. Mizher et al. has also proposed an action keyframe extraction method based on L1-Norm and accumulated optical flows [214]. A similar approach has been observed for salient region-based keyframe extraction by using optical flow and calculating mutual information entropy [215].

Clustering-based methods have been used to extract keyframes. The idea is that frames are grouped based on their low-level features by using a clustering method like K-means and the most similar frames with the groups' centres are selected [208]. Dynamic Delaunay graph clustering through iterative edge pruning technique has also been used to extract keyframes [24]. Tan et al. demonstrated the KGAF-means method by adopting K-means and the artificial fish swarm algorithms to extract keyframe sequences [216].

The proposed method in this study aims to tackle some important limitations of the aforementioned approaches. Despite extracting keyframes using shot-based approaches is easy to use, early approaches are unable to capture the temporal information. As for clustering-based approaches, they are sensitive to the type of adopted kernel and the number of clusters, and high in time complexity [217]. Furthermore, video is a special kind of media content that includes temporal information and complex background. Another limitation of the mentioned methods is handling entire frame differences rather than a specific region of interest. This work proposes a novel approach based on the similarity between regions of interest in consecutive frames to address these limitations. Different from the previous works, an action template is employed to find the region of interest for each video.

It is noteworthy that some related work on deep neural networks for video classification has been presented in our previous work [5].

4.3 The Proposed Method

Keyframe extraction is a principal preprocessing step in video analysis. The purpose of extracting keyframes is to get more discriminate information from the video in an effective manner. Each video has its own unique characteristics such as saturation, brightness, contrast, camera angle, vibration, blur, location of the action, number of actors, type of action, length, and background. Considering a large number of variables in each video and treating all videos equally bring about a major weakness in keyframe extraction. Thus, it is necessary to recognise the region of action in continuous action video. Considering the variations in the complex video data, it is a challenging task to find the location of the action, based on which this study proposes a new method for keyframe extraction.

4.3.1 The Proposed Keyframe Extraction Approach

In general, the location of action in a movie is related to the point on the screen and camera, to which the reviewer's attention is paid. It is observed that attention is paid to the central area mostly while recording and watching. Therefore, the outside regions of video frames are usually cropped off before identifying the region of interest. Then, the area of action is formulated as a region in the centre of video frames, which produces either the biggest difference or lowest similarity between consecutive frames, leading to a template for the video to track the action area. Calculating only the difference in regions of action between frames throughout the video helps to extract keyframes more accurately and effectively by reducing the influence of possible dynamically changing backgrounds.

The proposed keyframe extraction method consists of four steps: (1) identify an action template; (2) specify the location of an action; (3) calculate action similarities to find distinctive frames; (4) select a preset number of keyframes in chronological order. The four steps of the proposed keyframe extraction method can be summarised as follows:

1) Action template identification:
    - Frame decomposition.
    - Frame cropping.
    - Define three possible regions for action template.
    - Calculate Mean Squared Error (MSE) for each possible region using the first two frames.
    - Choose the region that produces the largest MSE as an action template.
2) Action location specification:
    - Find the region of interest on each frame by matching the action template against overlapped regions in each frame using the correlation coefficient defined in Equation (4.3).
3) Keyframe extraction:

- Calculate the structural similarity measure (S_i) between regions of interest on consecutive frames (f_i, f_{i-1}).
- Compare the similarity score with thresholds $T_1 = [0.65, 0.90]$ and $T_2 = [0.65, 0.95]$ (these threshold values were chosen by analysis of significance in action changes in the experiments):

$$0.65 < S_i < 0.90 \rightarrow \text{add } f_i \text{ to primary list}(p_f)$$

$$0.65 < S_i < 0.95 \rightarrow \text{add } f_i \text{ to alternative list}(a_f)$$

- Repeat the above till end of the video, with N_{p_f} frames extracted into p_f and N_{a_f} frames extracted in to a_f.

4) Keyframe selection:
   - Set the number of keyframes (N_{k_f}).
   - Find keyframe ratio (k):

$$k = \begin{cases} \lfloor \frac{N_{p_f}}{N_{k_f}} \rfloor, & \text{if } N_{p_f} \geq N_{k_f} \\ \lfloor \frac{N_{a_f}}{N_{k_f}} \rfloor, & \text{otherwise} \end{cases}$$

   - Return the indexes of keyframes by choosing a frame from every k frames from keyframe list p_f if $N_{p_f} \geq N_{k_f}$, or from keyframe list a_f otherwise.

In the first step, each frame is cropped by taking an appropriate number of pixels out (depending on frame resolution) from each side of a frame to create a general action area. As depicted in Fig. 4.1, the inner area is then divided into three different candidate templates. It has been observed that background changes between consecutive frames result in small structural similarity (SSIM) and action differences lead to large MSE. The candidate area having the largest MSE between consecutive frames is assigned as an action template. The mean squared error between two regions of frames X and Y is computed as follows:

$$MSE(X,Y) = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} [Y(i,j) - X(i,j)]^2 \tag{4.1}$$

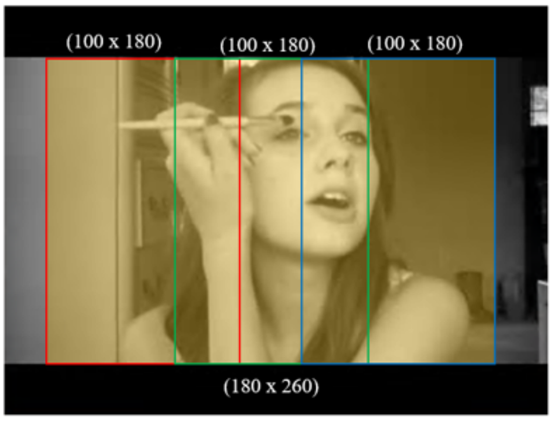

Figure 4.1: Defining the possible locations of an action template. Red, green, and blue boxes represent the borders of three possible templates.

where *m* and *n* are the number of rows and columns in the region of interest, respectively. The structural similarity formulated by Wang et al. [218] is adopted in this study and defined as follows:

$$SSIM(X,Y) = \frac{\left(2\mu_X\mu_Y + C_1\right)\left(2\sigma_{XY} + C_2\right)}{\left(\mu_X^2 + \mu_Y^2 + C_1\right)\left(\sigma_X^2 + \sigma_Y^2 + C_2\right)} \tag{4.2}$$

where μ_X and μ_Y denotes the average of pixel values in X and Y, σ_X and σ_Y are the variance of pixel values in X and Y, and σ_{XY} is the co-variance of pixel values in X and Y. $C_1 = (k_1L)^2$ and $C_2 = (k_2L)^2$ are small constants where L denotes the dynamic range of pixel values. MSE and SSIM are calculated for each frame region and used to select frames well representing the action (see Fig. 4.2).

In the second step, after having determined the action template for each video, the template-based correlation coefficient matching method is used to find at what position the template most closely matches the data in a region of each frame. This operation slides throughout each frame and compares the overlapped patterns of size *w* x *h* to the template, where *w* and *h* are the width and height of the template. Then, the best matches are found as global maximums. Regarding colour channels, template summation in the function is done over all channels and different mean values are used for each channel. The formula for the template-based matching method is:

$$R(x,y) = \sum_{x',y'} \left(T'(x',y') \cdot I'(x+x',y+y')\right) \tag{4.3}$$

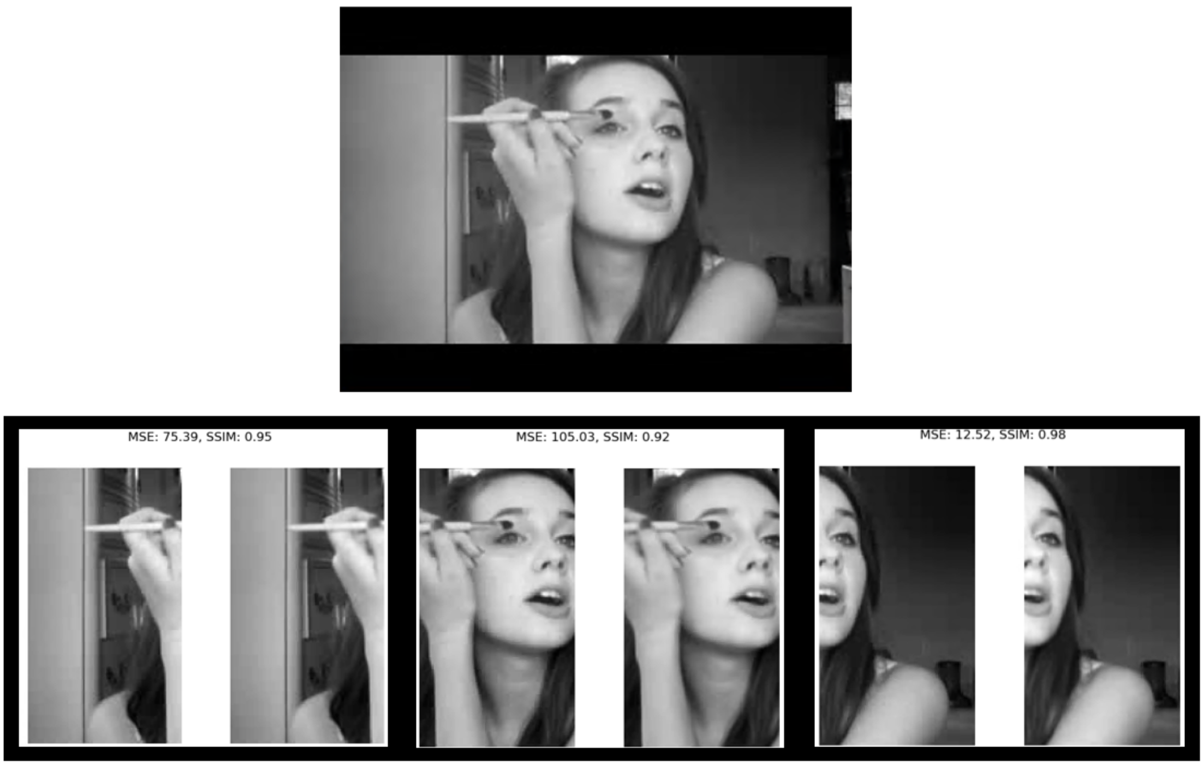

Figure 4.2: An example of action template identification

where $R(x,y)$ is the correlation-coefficient score for a single overlapped position of (x,y) representing the coordinates of each pixel in the frame. $T'(x',y')$ is the average of pixel values of the template T, where (x',y') represents the coordinates of each pixel in the template, given as:

$$T'(x',y') = T(x',y') - \frac{1}{(w \cdot h)} \cdot \sum_{x'',y''} T(x'',y'') \tag{4.4}$$

On the other hand, $I'(x+x',y+y')$ is the average of pixel values of a given frame I in the region overlapped with the template T, given as:

$$I'(x+x',y+y') = I(x+x',y+y') - \frac{1}{(w \cdot h)} \cdot \sum_{x'',y''} I(x+x'',y+y'') \tag{4.5}$$

where $x'' = 0,...,w-1$ and $y'' = 0,...,h-1$ which represent the new coordinates of (x,y) in the template after moving the centre of the template over the frame. $T(x',y')$ is the pixel values for a pixel (x,y) in the template while $I(x+x',y+y')$ is the pixel value for the corresponding pixel position in the frame. After performing the template matching procedure, the region of interest on each frame is localised where the highest matching probability takes place.

In step 3, it is very challenging to distinguish between action changes and background changes in consecutive frames. Through an extensive investigation, it has been discovered that the structural similarity between two regions of interest in two consecutive frames is more sensitive to action changes than background changes. After analysing both background and action changes in these areas, two rules are proposed in this study to specify upper and lower bounds of similarity range. Action changes are found mostly important in the interval $[0.65, 0.90]$, and the lower similarity is mainly due to the dramatic change in dynamic background. Rarely, the difference between regions in consecutive frames is not in this range. However, if there are not enough keyframes extracted using the above interval more frames are extracted by extending the upper bound of the interval up to 0.95.

Finally, keyframes are selected by using a keyframe ratio in chronological order. The pseudocode of the proposed algorithm is demonstrated in Algorithm 1.

4.3.2 Deep Neural Network Architectures Based on VGG-16 for Video Classification

In 2014, Simonyan and Zisserman [72] introduced a VGG-16 network architecture trained on 1,000 image categories using image data for the ILSVRC. VGG-16 consists of 16 convolutional layers with relatively small convolution filters (3x3). The pre-trained neural network VGG-16 was used to generalise the pre-learnt feature representations using transfer learning. In the previous work [5], ConvLSTM and LSTM with local features extracted using VGG-16 outperformed those using global features. Thus, this study uses the ConvLSTM(1) and LSTM(1) architectures used in [5]. Apart from the newly proposed keyframe extraction method, more experiments were conducted using not only 20 but also 101 categories of the UCF-101 dataset and evaluating the proposed methods on the KTH action recognition dataset as well. Moreover, the parameters of the networks were optimised using the hold-out validation method on the validation split of the training dataset. The two video classification architectures to classify 101 categories are shown in Fig. 4.3.

```
Function Keyframe_Extraction($V, N_{k_f}$)
    Input  : The input video stream, $V$
    Input  : The number of keyframes, $N_{k_f}$
    Output : The list of keyframes, $K$
    while input video is available do
        // Action template identification
        Get next video and crop outline areas;
        Frame decomposition $F = F_1, F_2, ..., F_N$;
        Select three candidate action regions $R1_{f1}, R2_{f1}, R3_{f1}$ on the first frame $F_1$ and
          $R1_{f2}, R2_{f2}, R3_{f2}$ the second frame $F_2$;
        Calculate MSE scores between candidate action regions:
          $MS_1 = MSE(R1_{f1}, R1_{f2}), MS_2 = MSE(R2_{f1}, R2_{f2}), MS_3 = MSE(R3_{f1}, R3_{f2})$;
        Choose action template considering the largest MSE score gained from two
          consecutive action regions;
        // Action location specification
        Extract region of interest on each frame by matching the action template against
          overlapped regions $r = r_1, r_2, ..., r_n$
        // Keyframe extraction
        foreach consecutive frame pairs $f_i$ and $f_{i+1}$ do
            Calculate the structural similarity measure $S_i$ between regions of interest ($r_i$ and
              $r_{i+1}$)
            if $0.65 < S_i < 0.90$ then // $T_1 = [0.65, 0.90]$
                Add $F_i$ to the primary keyframe list ($p_f$)
            end
            if $0.65 < S_i < 0.95$ then // $T_2 = [0.65, 0.95]$
                Add $F_i$ to the alternative keyframe list ($a_f$)
            end
            // Keyframe selection. Find keyframe ratio, $k$:
            $$k = \begin{cases} \left\lfloor \frac{N_{p_f}}{N_{k_f}} \right\rfloor, & \text{if } N_{p_f} \geq N_{k_f} \\ \left\lfloor \frac{N_{a_f}}{N_{k_f}} \right\rfloor, & \text{otherwise} \end{cases}$$
            if $N_{p_f} \geq N_{k_f}$ then
                Add every $k$ frames of the primary list $p_f$ to $K$
            else
                Add every $k$ frames of the alternative list $a_f$ to $K$
            end
        end
    end
    return the list of keyframes, K
End
```

Algorithm 1: Action template-based keyframe extraction

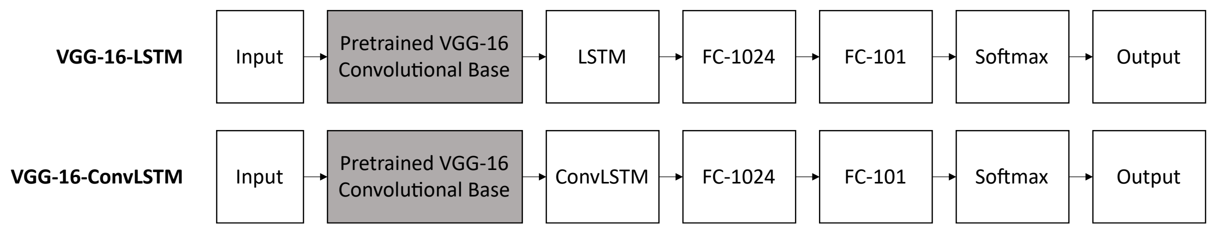

Figure 4.3: The architectures of the networks used in VGG-16-LSTM and VGG-16-ConvLSTM experiments

LSTM is one of the most common approaches for sequence modelling. Previous studies [85, 138, 219] have demonstrated that LSTM is a robust method to represent long-range dependencies. In VGG-16-LSTM, the local features extracted by VGG-16 from video frames are fed into LSTM to access spatiotemporal information. The number of units in the output space was set to 1024 and ReLU was used as an activation function.

An end-to-end trainable ConvLSTM was proposed by extending the fully-connected LSTM to have convolutional structures in both the input-to-state and state-to-state transitions for precipitation nowcasting [202]. The purpose of precipitation nowcasting is to predict future precipitation intensity over a relatively short period of time in a local area and it can be seen as a video prediction problem with a fixed camera with the weather radar [220]. It was shown that ConvLSTM extracts better spatiotemporal correlations than the fully-connected LSTM for precipitation nowcasting [202]. Inspired by this study, ConvLSTM was used to build a new architecture for video classification, which takes the advantage of its capacity in capturing spatiotemporal information throughout time series. One ConvLSTM layer is added on top of the spatial feature maps extracted by VGG-16 and use the hidden states for video classification. This layer contains 64 hidden states, 7x7 kernels, and the stride of convolution is set to 1 to perform the experiments described in Section 4.4.

4.4 Experiments

In this section, the experimental setup, the datasets used and the evaluation method are described.

4.4.1 Experiment Design

During the training process, parameter tuning is carried out with the hold-out validation technique. The best parameters are identified based on the validation scores. After that, the model with the best parameters is evaluated on testing data by predicting unseen test videos' classes.

The proposed network architectures are implemented by using TensorFlow-gpu v1.12 on an NVIDIA TITAN X GPU using the CUDA v9.0 toolkit. The batch size is set to 128, and the cost is minimised by using the SGD optimiser. The number of epochs is determined using early stopping by observing the change in validation loss. Dropout is used as a regularisation method, disabling some neurons within ConvLSTM and fully-connected layers with a probability of 0.5.

4.4.2 Datasets and Evaluation Method

In this study, the UCF-101 and KTH datasets are used to evaluate the neural network architectures with the proposed keyframe extraction method and two more keyframe extraction methods for classifying human actions from video clips. The UCF-101 dataset has defined three training-testing splits, aiming to facilitate benchmarking algorithms. In the previous experiment [5], only the first twenty categories of the dataset were used due to limited time and computing facility, and the first training-testing split was adopted to generate training and testing data. However, the experiments were conducted in this study using all categories with the three training-testing splits of the UCF-101 and KTH datasets. The KTH dataset was used for initial experiments and verification as it is small-sized dataset with six action categories.

In the experiments, confusion matrices are produced for performance analysis and accuracy is used for the comparison of the performances achieved by different architectures. 180 training, validation, and testing accuracy scores have been collected with the three training-testing splits released by the UCF-101 organisation (10 times per split and 30 times per classifier). Similarly, 90 accuracy scores have been collected with the official training, validation, and testing splits of the KTH dataset (15 times per classifier).

The Kolmogorov-Smirnov test is a normality test that compares the observed cumulative distribution with the cumulative distribution that would occur if the data were normally distributed [221], and it has been used for calculating numerical means for assessing normality. As for the test of homogeneity of variances, Levene statistic has been applied to the dependent variable and shows variances of groups are homogeneous based on mean and median. Levene's test is simply a one-way analysis of variance on the absolute values of the differences between each observation and the mean of its group and is appropriate for testing the null hypothesis [222]. Furthermore, an ANOVA test has been conducted to compare the variance differences to figure out whether the results are significant. Afterwards, Tukey's Honest Significant Difference (HSD) test has been run to determine whether the specific groups' means are different. The results are presented in Section 4.5.

4.5 Results and Discussion

The VGG-16-ConvLSTM and VGG-16-LSTM architectures presented in our previous work [5] for video classification are used as baseline methods to evaluate the proposed method in this study. One of the findings of the previous work [5] is that using global features can help achieve better classification performance over local features. It can be highlighted that the fundamental difference between local and global features is the way of representing input frames in terms of the whole frame or frame patches, which provide different information on the input to the video classifier. Seven different classification networks using either local or global features extracted by the pre-trained VGG-16 were compared in the previous work [5]. The extracted features were fed into a newly added fully-connected layer in baseline VGG-16-VOTE (a) and the fully-connected layer of VGG-16 was included in VGG-16-VOTE (b). Similar to the baseline methods, LSTM was employed to access spatiotemporal information over the features in VGG-16-LSTM (a) and VGG-16-LSTM (b). To test the effect of directional connections in LSTM structure for action recognition, VGG-16-BLSTM (a) and VGG-16-BLSTM (b) were implemented by using BLSTM. The VGG-16-ConvLSTM architecture was proposed with convolutional structures in state transitions. Table 4.1 shows the results obtained from the earlier study [5] in which VGG-16-ConvLSTM (82.04%) significantly outperformed the other networks followed by VGG-

16-LSTM (81.27%) with local features at 0.05 significance level ($p = 0.046$). In the previous study, one frame per second was extracted to reduce the number of frames in classification.

Table 4.1: Average accuracy scores achieved by different architectures on the UCF-101 dataset with 20 categories [5]

Model	Training Accuracy	Validation Accuracy	Testing Accuracy
VGG-16-ConvLSTM	99.24%	97.45%	82.04%
VGG-16-LSTM(a)	98.86%	97.91%	81.27%
VGG-16-BLSTM(a)	97.76%	95.19%	76.20%
VGG-16-VOTE(a)	78.71%	76.55%	73.20%
VGG-16-LSTM(b)	78.41%	79.47%	67.62%
VGG-16-VOTE(b)	73.70%	67.05%	64.04%
VGG-16-BLSTM(b)	89.17%	75.37%	61.18%

In this study, experiments with the proposed keyframe extraction method were conducted using the first 20 categories of the UCF-101 dataset in the first place to investigate how the proposed keyframe extraction method can improve the video classification performance of the LSTM and ConvLSTM based network architectures in comparison with the previous work [5].

Table 4.2: Average accuracy scores achieved by LSTM and ConvLSTM network architectures based on the UCF-101 (20 categories) using two keyframe extraction methods where (1) and (2) indicate one frame per second and the proposed method, respectively.

Architecture	Training Accuracy	Validation Accuracy	Testing Accuracy
ConvLSTM(2)	98.95%	98.37%	88.15%
LSTM(2)	96.48%	95.53%	83.10%
ConvLSTM(1)	99.24%	97.45%	82.04%
LSTM(1)	98.86%	97.91%	81.27%

Table 4.2 shows the results obtained on the first twenty categories of UCF-101 in which ConvLSTM(1) and LSTM(1) refer to the previous method selecting one frame per second whereas ConvLSTM(2) and LSTM(2) indicate the proposed method. It can be seen that the architectures using keyframes extracted by the proposed method, ConvLSTM(2) and LSTM(2), outperformed the previous method, achieving accuracy scores of 88.15% and 83.10%, respectively. In order to draw more convincing conclusions, further experiments were conducted with the proposed method (2) using all the 101 categories of the UCF-101 dataset and the KTH dataset in comparison with two commonly used keyframe extraction methods: method (1) is a baseline

Table 4.3: Average accuracy scores achieved by LSTM and ConvLSTM network architectures on datasets KTH and UCF-101 (101 categories) using three keyframe extraction methods where (1), (2), and (3) indicate: one frame per second method, the proposed action template-based method and optical flow-based keyframe extraction method, respectively.

Architecture	Dataset	Training Accuracy	Validation Accuracy	Testing Accuracy
ConvLSTM(2)	KTH	87.41%	67.34%	71.13%
LSTM(2)	KTH	65.06%	85.50%	68.66%
ConvLSTM(1)	KTH	84.16%	69.52%	68.27%
ConvLSTM(3)	KTH	82.02%	54.92%	66.17%
LSTM(1)	KTH	58.43%	82.26%	64.58%
LSTM(3)	KTH	76.85%	45.12%	62.83%
ConvLSTM(2)	UCF-101	94.27%	90.02%	67.39%
ConvLSTM(1)	UCF-101	97.19%	92.19%	65.44%
LSTM(2)	UCF-101	93.79%	86.76%	64.27%
LSTM(1)	UCF-101	94.09%	89.64%	63.86%
ConvLSTM(3)	UCF-101	87.71%	82.75%	60.04%
LSTM(3)	UCF-101	92.96%	85.46%	49.62%

method that extracts one frame for each second until the end of the video, and method (3) is a motion-based keyframe extraction method that selects keyframes considering the local minima of action between optical flows [22]. The results are summarised in Table 4.3, in which ConvLSTM(2) achieved the best classification accuracy (71.13%) followed by LSTM(2), ConvLSTM(1), ConvLSTM(3), LSTM(1), and LSTM(3) on the KTH dataset, respectively. Similarly, ConvLSTM(2) outperformed the other methods on the UCF-101 dataset with 67.39% accuracy score. As shown in Table 4.4, there is a statistically significant difference between classifier groups in terms of one-way ANOVA $(F(11.258) = 304.725, p = 0.000)$.

Table 4.4: One-way ANOVA of performance achieved by different network architectures where df, SS, MS, and F refer to degrees of freedom, sum of squares, mean sum of squares, and F score, respectively. The values are significant at the 0.05 level.

	df	SS	MS	F	p-value
Between Groups	11	0.847	0.077	304.725	0.000*
Within Groups	258	0.065	.000		
Total	269	0.913			

The Tukey's HSD post hoc test results on the KTH dataset, as shown in Table 4.5, show that the classification performance achieved by ConvLSTM(2) is statistically significantly higher than LSTM(1), ConvLSTM(1), LSTM(2), LSTM(3), and ConvLSTM(3) ($p < 0.05$) on the KTH

Table 4.5: Post hoc comparisons using Tukey's HSD on the KTH Dataset. The values are significant at the 0.05 level.

Method (I)	(J)	Mean Difference (I-J)	Sd. Error	p-value	95% Confidence Interval Lower Bound	Upper Bound
LSTM(1)	ConvLSTM(1)	-0.016^{*}	.003	.000	$-.023$	$-.008$
	LSTM(2)	-0.004	.003	.612	$-.012$	.003
	ConvLSTM(2)	-0.035^{*}	.003	.000	$-.043$	$-.028$
	LSTM(3)	0.142^{*}	.003	.000	.135	.150
	ConvLSTM(3)	0.038^{*}	.003	.000	.031	.046
ConvLSTM(1)	LSTM(1)	0.016^{*}	.003	.000	.008	.023
	LSTM(2)	0.012^{*}	.003	.000	.004	.019
	ConvLSTM(2)	-0.020^{*}	.003	.000	$-.027$	$-.012$
	LSTM(3)	0.158^{*}	.003	.000	.151	.166
	ConvLSTM(3)	0.054^{*}	.003	.000	.046	.062
LSTM(2)	LSTM(1)	0.004	.003	.612	$-.003$	.012
	ConvLSTM(1)	-0.012^{*}	.003	.000	$-.019$	$-.004$
	ConvLSTM(2)	-0.031^{*}	.003	.000	$-.039$	$-.024$
	LSTM(3)	0.147^{*}	.003	.000	.139	.154
	ConvLSTM(3)	0.042^{*}	.003	.000	.035	.050
ConvLSTM(2)	LSTM(1)	0.035^{*}	.003	.000	.028	.043
	ConvLSTM(1)	0.020^{*}	.003	.000	.012	.027
	LSTM(2)	0.031^{*}	.003	.000	.024	.039
	LSTM(3)	0.178^{*}	.003	.000	.170	.185
	ConvLSTM(3)	0.074^{*}	.003	.000	.066	.081
LSTM(3)	LSTM(1)	-0.142^{*}	.003	.000	$-.150$	$-.135$
	ConvLSTM(1)	-0.158^{*}	.003	.000	$-.166$	$-.151$
	LSTM(2)	-0.147^{*}	.003	.000	$-.154$	$-.139$
	ConvLSTM(2)	-0.178^{*}	.003	.000	$-.185$	$-.170$
	ConvLSTM(3)	-0.104^{*}	.003	.000	$-.112$	$-.097$
ConvLSTM(3)	LSTM(1)	-0.038^{*}	.003	.000	$-.046$	$-.031$
	ConvLSTM(1)	-0.054^{*}	.003	.000	$-.062$	$-.046$
	LSTM(2)	-0.042^{*}	.003	.000	$-.050$	$-.035$
	ConvLSTM(2)	-0.074^{*}	.003	.000	$-.081$	$-.066$
	LSTM(3)	0.104^{*}	.003	.000	.097	.112

dataset. There is a statistically significant difference between all methods except for LSTM(1) and LSTM(2).

The Tukey's HSD post hoc test results on the UCF-101 dataset are presented in Table 4.6, which demonstrate that ConvLSTM(2) significantly outperformed LSTM(1), ConvLSTM(1), LSTM(3), and ConvLSTM(3) ($p < 0.05$).

Table 4.6: Post hoc comparisons using Tukey's HSD on the UCF-101 dataset. The values are significant at the 0.05 level.

Method (I)	(J)	Mean Difference (I-J)	Sd. Error	p-value	95% Confidence Interval Lower Bound	Upper Bound
LSTM(1)	ConvLSTM(1)	−0.037*	.009	.001	−.062	−.012
	LSTM(2)	−0.041*	.009	.000	−.066	−.016
	ConvLSTM(2)	−0.065*	.009	.000	−.091	−.040
	LSTM(3)	0.017	.009	.340	−.008	.043
	ConvLSTM(3)	−0.016	.009	.450	−.041	.009
ConvLSTM(1)	LSTM(1)	0.037*	.009	.001	.012	.062
	LSTM(2)	−0.004	.009	.998	−.029	.021
	ConvLSTM(2)	−0.029*	.009	.017	−.054	−.003
	LSTM(3)	0.054*	.009	.000	.029	.080
	ConvLSTM(3)	0.021	.009	.159	−.004	.046
LSTM(2)	LSTM(1)	0.041*	.009	.000	.016	.066
	ConvLSTM(1)	0.004	.009	.998	−.021	.029
	ConvLSTM(2)	−0.025	.009	.059	−.050	.001
	LSTM(3)	0.058*	.009	.000	.033	.083
	ConvLSTM(3)	0.025	.009	.055	.000	.050
ConvLSTM(2)	LSTM(1)	0.065*	.009	.000	.040	.091
	ConvLSTM(1)	0.029*	.009	.017	.003	.054
	LSTM(2)	0.025	.009	.059	−.001	.050
	LSTM(3)	0.083*	.009	.000	.058	.108
	ConvLSTM(3)	0.050*	.009	.000	.024	.075
LSTM(3)	LSTM(1)	−0.017	.009	.340	−.043	.008
	ConvLSTM(1)	−0.054	.009	.000	−.080	−.029
	LSTM(2)	−0.058*	.009	.000	−.083	−.033
	ConvLSTM(2)	−0.083*	.009	.000	−.108	−.058
	ConvLSTM(3)	−0.033*	.009	.003	−.059	−.008
ConvLSTM(3)	LSTM(1)	0.016	.009	.450	−.009	.041
	ConvLSTM(1)	−0.021	.009	.159	−.046	.004
	LSTM(2)	−0.025	.009	.055	−.050	.000
	ConvLSTM(2)	−0.050*	.009	.000	−.075	−.024
	LSTM(3)	0.033*	.009	.003	.008	.059

The keyframe extraction method proposed in this study uses action templates to identify most important regions related to actions in video frames. The experimental results have demonstrated that this action template-based approach to keyframe extraction can extract frames with distinctive actions to significantly improve the performance of deep convolutional neural networks for action recognition from videos. Keyframe extraction methods have been investigated due to their adaptability to video summarization systems and performance improvement in video classification approaches. Keyframe extraction methods enable using more informative input representation while reducing the number of frames. With the advantage of using fewer but more informative frames, the input dimension is reduced and training time is shortened. Moreover, using selected keyframes can effectively improve the accuracy of video classification.

4.6 Conclusion

In this chapter, a template-based keyframe extraction method is proposed which employs action template-based similarity to extract keyframes for video classification tasks. Combining pre-trained CNN with ConvLSTM has achieved the highest classification accuracy among the other architectures. It can be seen that calculating structural similarity between two relevant regions of consecutive frames effectively prevents dynamic background noise from being treated as actions in keyframe selection. The experimental results and the conducted analysis show that the proposed keyframe extraction method can select informative frames reliably and thus significantly improve the performance of deep neural network architectures for video classification. Finally, when finding the relevant area using the extracted action template the proposed method successfully extracts proper keyframes from human action videos for video classification using deep neural networks. Although the proposed method has outperformed the two commonly used keyframe extraction methods, this study has a few limitations. One of the limitations is that CNN architectures used in the evaluation of the proposed method were not state-of-the-art architecture. This means that it did not produce the best results. The second limitation of the study is that the technical infrastructure of the experiment was weak with one GPU machine only and could not conduct more comprehensive experiments with larger batches. However, the proposed keyframe extraction method has significantly outperformed the commonly used keyframe extraction methods on two different datasets. Therefore, future work could be focused on the application of the proposed algorithm using more powerful architectures for real-world video classification and video summarising problems.

Chapter 5

Extracting Motion Information and Decision Fusion for Video Classification

CNNs have recently been applied for video classification applications where various methods for combining the appearance and motion information from video clips are considered. The most common method for combining the spatial and temporal information for video classification is averaging prediction scores at the softmax layer. Inspired by the Mycin uncertainty system for combining production rules in expert systems, this study proposes using the Mycin formula for decision fusion in two-stream CNNs. Based on the intuition that spatial information is more useful than temporal information for video classification, this study also proposes multiplication and asymmetrical multiplication for decision fusion, aiming to better combine the spatial and temporal information using two-stream CNNs. The experimental results show that (i) both spatial and temporal information is important, but the decision from the spatial stream should be dominating with the decision from the temporal stream as complementary and (ii) the proposed asymmetrical multiplication method for decision fusion significantly outperforms the Mycin method and average method as well.

5.1 Introduction

Over the past few years, research interest in the detection and recognition of human actions from video data streams has increased substantially. While humans have adapted to understanding the behaviour of other humans (at least within their cultural framework), the identification and classification of human activities have remained a significant challenge for computer vision research. This problem is exacerbated when these actions are mirrored within different environmental settings. Furthermore, understanding multi-level human actions within complex video data structures is crucial to meaningfully apply these techniques to a wide range of real-world domains.

There have been several approaches to classify human actions throughout a video. One of the most popular approaches is an extension of single image classification tasks to multiple image frames, followed by combining the predictions from each frame. Despite the fact that there has been recent success in using deep learning for image classification, more research should be focused on deep neural network architecture design and representation learning for video classification.

Real-world video data involves a complex data structure, which generally consists of two basic components, spatial and temporal features. Research has shown that only using spatial features in video classification, by extending the approaches from image-based architectures, has not achieved the same level of interpretative accuracy in multi-image domains compared to single image domains.

CNNs have been widely utilised in video classification due to their powerful structure and robust feature representation capability. However, one of the main issues is the methodology behind combining appearance and motion information in CNN architectures, which is by information fusion in general. The fusion of information can take place at different levels: signal, image, feature, and symbol (decision) level. Signal-level fusion is the combination of multiple signals to produce a new signal, having the same format as the input signals. Fusion at image-level, or pixel-level, combines a set of pixels from each input image. Feature-level fusion refers to combining different features extracted from multiple images. Decision-level fusion provides the fused representations from several inputs at the highest level of abstraction. The input image(s) are normally analysed individually for feature extraction and classification. The results are multiple symbolic representations which are then combined based on decision rules. The highest level of fusion is symbolic fusion as the existence of more complexity at this level.

The main contribution of this study is to have proposed simple but effective methods for decision-level fusion in two-stream CNNs for video classification, which have taken into account the asymmetry of the spatial and temporal information in videos about human actions.

This chapter is organised as follows: Section 5.2 presents a brief overview of the existing fusion approaches in two-stream neural networks for video classification. The proposed method is described in Section 5.3, the results are presented in Section 5.4, and conclusions are drawn in Section 5.5.

5.2 Related Work

Action recognition is attracting widespread interest due to the need to capture context information from an entire video instead of just extracting information from each frame (a process that is an extension of image-based classification). CNN has received increased attention in video classification research over the last decade because of its powerful architecture in image-based tasks [80]. A common extension of image-based approaches is to extend 2D CNNs into time dimension to capture spatiotemporal information with the first layer of CNNs [54, 190]. The progress of video classification architectures has been slower than image classification architectures [71, 73]. The possible challenges in video classification are huge computational cost, difficulty in complex architectures, overfitting, and capturing spatiotemporal information [223].

Karpathy et al. explored several fusion approaches to extract temporal information from consecutive frames using pre-trained 2D CNNs: single frame, late, early, and slow fusion [80]. The single frame model utilises a single-stream network that combines all frames at the final step whereas the late fusion approach utilises two networks with shared parameters a distance of 15 frames apart and then fuses predictions at the last stage. The early fusion involves pixel-level fusion by combining information from the entire video time window. The slow fusion model is a form of a balanced mixture of the late and early fusion models by fusing temporal information throughout the network. The authors highlighted that the spatiotemporal feature representations could not capture motion features properly as the single frame method outperformed the spatiotemporal network architectures [80].

Simmoyan and Zisserman implemented a deep two-stream architecture to learn motion features by modelling these features in stacked optical flow vectors [81]. Spatial and temporal streams tackle with spatial and temporal information in separate networks and the fusion occurs in the last classification layer of the network. Although the temporal information is absent, video-level predictions were gathered after averaging predictions over video clips. Their achievement on the UCF-101 and HMDB-51 over the models consisting of either spatial or temporal stream has led to investigating multiple deep neural networks for video classification.

As opposed to stream-based design, Yue-Hei et al. employed RNN on trained feature maps to explore the use of LSTM cell to capture temporal information over video clips [85]. They also explored the necessity of motion information and highlighted the use of optical flow in action recognition problems, although using optical flow is not always beneficial, especially for the videos taken from the wild. Donahue et al. built an end-to-end training architecture by encoding convolution blocks and a form of LSTM as a decoder [84]. They also showed that the weighted average between spatial and temporal networks returned average scores. A temporal attention mechanism within 3D CNN and RNN encoder-decoder architecture was proposed by Yao et al. to tackle local temporal modelling and learn how to select the most relevant temporal segments given the RNN [224]. Sun et al. [225] proposed factorised spatiotemporal CNNs that factorise the original 3D convolution kernel learning, as a sequential process, which learns 2D spatial kernels in the lower layers by using separate spatial and temporal kernels rather than different networks. They also proposed a score fusion scheme based on the sparsity concentration index which puts more weights on the score vectors of class probability that have a higher degree of sparsity [225].

Feichtenhofer et al. [88] explored how the fusion of spatial and temporal streams affects the performance from spatiotemporal information gathered throughout the networks. They proposed a novel spatiotemporal architecture involving a convolutional fusion layer between spatial and temporal networks with a temporal fusion layer after extensively examining feature-level fusion on feature maps using sum, max, concatenation, convolutional, bilinear, and 3D CNN and 3D Pooling fusion [88]. Moreover, spatial-temporal information fusion with frame attention has been proposed by Ou and Sun [226].

Combining both spatial and temporal features has been a common problem for video classification. Researchers use pixel-level, feature-level and decision-level fusion approaches to take the best advantage of spatiotemporal information [227, 228, 229]. This study investigates how decision-level fusion affects the final prediction performance without using complex architectures. Based on the fact that both spatial and temporal information is useful for video classification and they play an asymmetrical role in recognising human actions from videos and inspired by the Mycin uncertainty system [230] for combining production rules in expert systems, this study aims to propose simple but effective decision fusion methods in two-stream CNNs for video classification.

5.3 Methods

In this section, the architecture of the two-stream neural networks, the decision fusion functions, the dataset, the details of preprocessing and feature extraction, and the performance evaluation approach are presented. The proposed two-stream architecture is implemented in [81] with some modifications. The architecture is one of the most commonly used networks in computer vision applications as it is the first implementation of decision fusion on the top of two-stream neural networks by averaging the scores from both streams. Nonetheless, it is limited in the temporal domain as the input of the spatial stream is a single frame and that of the temporal stream is a stack of fixed frames. Due to its popularity, researchers commonly use averaging for the final decision by taking both spatial and temporal information equally.

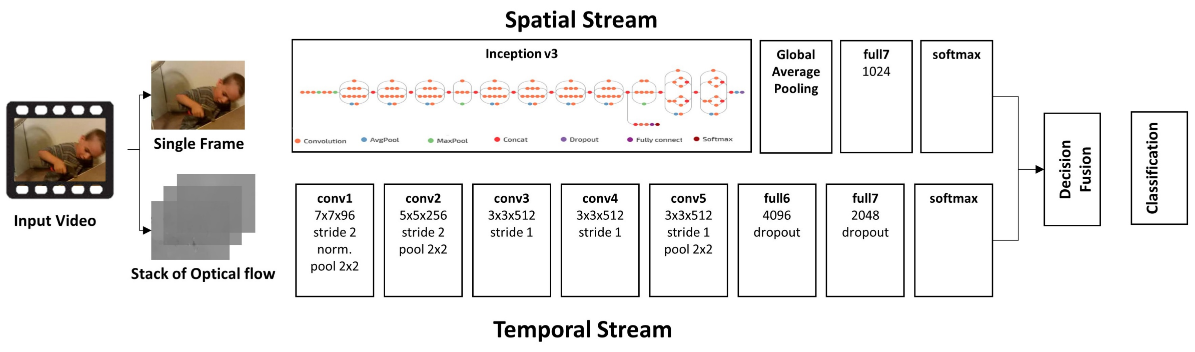

Figure 5.1: The architecture of CNNs used in the experiment

5.3.1 Two-Stream Convolutional Neural Networks

Two-stream neural networks have been implemented by using Tensorflow-gpu v.1.13 on NVIDIA TITAN X GPU using CUDA v9.0 toolkit. For the spatial stream network, pre-trained Inception v3 on ImageNet is employed to extract high-level feature representations from video clips. With all convolutional layers of Inception pre-trained network frozen, its fully connected layers are trained by initialising weights randomly. After that, the top 2 and 3 inception blocks are trained by freezing the rest of the network separately. The temporal stream network is created by mitigating the same structure defined by Simonyan and Zisserman [81] for the two-stream CNN as shown in Fig. 5.1. The model is evaluated on a holdout validation dataset after every epoch and the performance of the model is monitored based on the loss during the training by using the early stopping technique.

5.3.2 Decision Fusion

In this study, the focus is decision fusion in two-stream neural networks to investigate the effect of different operations on the final decision at the highest level of abstraction. Many decision fusion rules have been proposed to combine information gathered from different sources based on the same input in video classification applications. The common technique is averaging or weighted averaging which is generally used to get better representation for the combination of spatial and temporal information extracted by two different input representations.

The softmax activation function is widely employed in the output layer of a deep neural network to present a categorical distribution of a list of labels and to calculate the probabilities of each input element belonging to a category. The softmax function $S : \mathbb{R}^K \to \mathbb{R}^K$ is defined by the formula

$$S_i(x) = \frac{exp(x_i)}{\Sigma_j^K exp(x_j)} \tag{5.1}$$

where $i = 1, ..., K$ and $x = x_1, ..., x_K$ is the output of the final dense layer of the neural network.

The softmax scores from the networks can be fused to give a better representation than could be obtained from any individual stream. Spatial features are mainly used along with temporal features in many video classification applications. These spatial features have taken the same importance as temporal information in the same scope, by using the average function to make the final decision. In this study, it is argued that the temporal information is also important for video classification, but it may not be as important as spatial information for video classification, that is, they are asymmetrical. Inspired by the Mycin uncertainty system for combining production rules in expert systems, this study proposes using the Mycin formula for decision fusion in two-stream CNNs. Based on the intuition that spatial information is more useful than temporal information for video classification, this study also proposes multiplication and asymmetrical multiplication for decision fusion, aiming to better combine the spatial and temporal information for video classification using two-stream CNNs. To test the above ideas, this study proposes to investigate the following four decision fusion functions for the two-stream neural networks for video classification: Average (A), Mycin (B), Multiplication (C), and Asymmetrical Multiplication (D).

$$A_{X,Y} = \frac{X+Y}{2} \tag{5.2}$$

$$B_{X,Y} = X + Y - XY \tag{5.3}$$

$$C_{X,Y} = XY \tag{5.4}$$

$$D_{X,Y} = X(1 - min(X,Y)) \tag{5.5}$$

where X and Y represent appearance and motion softmax scores, respectively. When the softmax scores X and Y are asymmetrical, using min(X,Y) in the decision fusion function would bring about extra information in combining them for decision fusion. This will be investigated further by experiments in Section 5.4.

5.3.3 Dataset

The dataset used in this study is the UCF-101 dataset and it contains 101 categories of human actions divided into five types: 1) Human-Object Interaction, 2) Body-Motion Only, 3) Human-Human Interaction, 4) Playing Musical Instruments, and 5) Sports. This dataset is one of the most common action recognition datasets with 101 categories and over 13000 clips. The number of clips per action category and the distribution of clip duration are demonstrated in Fig. 5.2. Three training and testing splits officially released are adopted to test the neural networks used to classify human actions from video clips.

5.3.4 Preprocessing and Feature Extraction

The first step of preprocessing was extracting and saving video frames from video samples. Then, the frames were resized by cropping the centre of frames from the original rectangular image size 320x240 to a 240x240 square. To feed the temporal stream with the temporal information between the frame sequences, the preprocessed tvl1 optical flow representations were used [88]. The Inception v3 is a CNN which is pre-trained on the ImageNet dataset containing 1,000 image categories. The pre-trained Inception v3 was employed as a spatial feature extractor in the spatial stream network [231].

5.3.5 Performance Evaluation

In this experiment, classification accuracy is used as a performance metric, which is the percentage of correctly classified video instances. In the experiment, four different decision

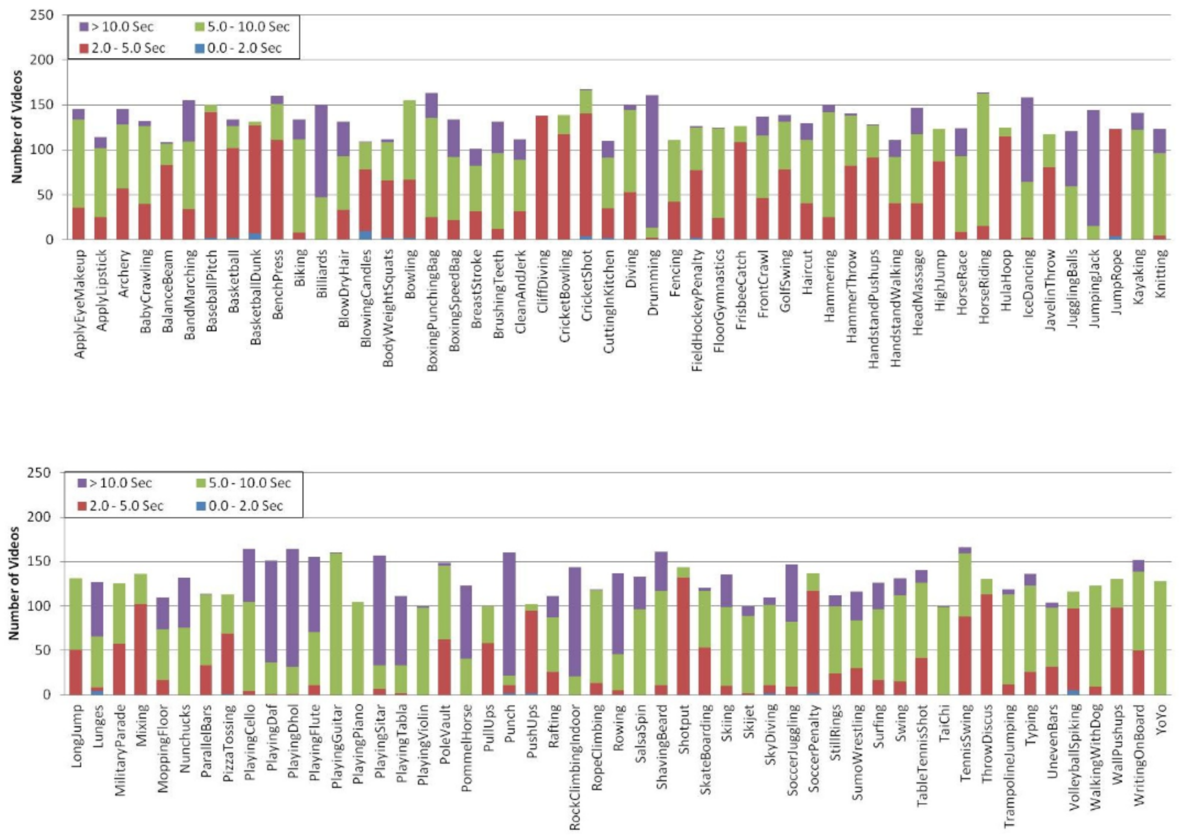

Figure 5.2: The number of clips and the distribution of clip duration in the UCF-101 [101]

fusion functions were compared based on their accuracy scores through a statistical significance test. 48 training, validation, and testing accuracy scores were collected under the same conditions but with either different data splits for training, validation and testing data or different configuration. A significance level of 0.05 was considered during the analysis. The distribution of performance data was evaluated using the Kolmogorov-Smirnov test to check whether the accuracy scores are normally distributed. As for the test of homogeneity of variances, the Levene statistic was applied to the dependent variables. Furthermore, an ANOVA test has been conducted to compare the variance differences to figure out whether the results are significant. Afterwards, the Tukey's HSD test was run to determine whether the specific groups' means are different. The results are presented in Section 5.4.

5.4 Results and Discussion

Four different decision-level fusion functions were compared for the two-stream neural network for video classification. Twelve different runs were conducted for each function over the 3 training, validation and testing splits. In total, 48 sets of top-1 and top-5 accuracy scores were collected, and the results are presented in this section.

Table 5.1 demonstrates the descriptive statistics of dependent variables, top-1 and top-5 accuracy scores, obtained from the decision fusion functions, A, B, C, and D which are defined in Section 5.3. The highest mean accuracy scores were achieved by the proposed asymmetrical multiplication, D, with 59.14% and 83.94% for top-1 and top-5 accuracy scores, respectively, which is followed by the multiplication function, C, at 58.94% (top-1) and 83.75% (top-5). The mean accuracy score achieved by the average is the lowest. It is interesting to see the performance gap between the average and the multiplication functions is over 3%.

Table 5.1: The descriptive statistics of top-1 and top-5 accuracy scores of decision fusion approaches: N, $\bar{x}$, σ, and $\sigma_{\bar{x}}$ denotes the number of samples, mean, standard deviation, and standard error, respectively.

	Group	N	$\bar{x}$	σ	$\sigma_{\bar{x}}$
top-1	A	12	55.56%	.01492	.00431
	B	12	55.54%	.01259	.00363
	C	12	58.94%	.01252	.00361
	D	12	59.14%	.01352	.00390
top-5	A	12	80.87%	.01285	.00371
	B	12	80.67%	.01127	.00325
	C	12	83.75%	.01309	.00378
	D	12	83.94%	.01642	.00474

As shown in Table 5.2, there was a statistically significant difference at the $p < 0.05$ level in top-1 accuracy scores achieved by the four decision-level fusion functions: F (3,44) = 27.137, p = 0.00. The mean top-5 accuracy scores varied significantly as well: F (3,44) = 20.755, p = 0.00.

Additional comparisons using the Tukey's HSD test, as shown in Table 5.3, indicated that

Table 5.2: One-way between-groups ANOVA of performance by fusion functions where df, SS, MS, and F refer to degrees of freedom, sum of squares, mean sum of squares, and F score, respectively.

		SS	df	MS	F	Sig.
top-1	Between Groups	.015	3	.005	27.137	.000
	Within Groups	.008	44	.000		
	Total	.023	47			
top-5	Between Groups	.011	3	.004	20.755	.000
	Within Groups	.008	44	.000		
	Total	.019	47			

there were significant differences in prediction performance between the proposed asymmetrical multiplication and the average function (mean difference = 0.03607, p = 0.00), and between the average and multiplication functions (mean difference = 0.03403, p = 0.00). Although there was a significant difference between the proposed asymmetrical multiplication and Mycin functions (mean difference = 0.03582, p=0.00), results did not support the significant difference between the average and Mycin functions (mean difference = 0.00025, p = 1.00). The experimental results have strengthened the argument that the importance of spatial features is not the same as that of temporal features and the decision fusion making use of this asymmetry works well.

Table 5.3: Post hoc comparisons using Tukey's HSD. The values are significant at the 0.05 level.

Method		Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
(I)	(J)				Lower Bound	Upper Bound
A	B	.00025	.00548	1.000	-.0144	.0149
	C	-.03378*	.00548	.000	-.0484	-.0192
	D	-.03582*	.00548	.000	-.0505	-.0212
B	A	-.00025	.00548	1.000	-.0149	.0144
	C	-.03403*	.00548	.000	-.0487	-.0194
	D	-.03607*	.00548	.000	-.0507	-.0214
C	A	.03378*	.00548	.000	.0192	.0484
	B	.03403*	.00548	.000	.0194	.0487
	D	-.00204	.00548	.982	-.0167	.0126
D	A	.03582*	.00548	.000	.0212	.0505
	B	.03607*	.00548	.000	.0214	.0507
	C	.00204	.00548	.982	-.0126	.0167

* The mean difference is significant at the 0.05 level.

Although this research presents significant results, there are several limitations, which must be addressed in future research. First, the proposed asymmetrical multiplication method was developed and evaluated on only two-stream neural networks for video classification. It can be extended to multi-stream neural networks and other domains. Second, the dataset used in the experiments was the UCF-101. Future research on different datasets might be needed before generalising these findings to other datasets. Finally, the temporal streams in two-stream architectures were trained from scratch, which is much costly than the spatial streams which were trained using transfer learning. Although learning from scratch leads to better prediction results, using neural networks pre-trained on large motion datasets can help to reduce the cost of temporal streams.

5.5 Conclusion

This study has investigated decision fusion methods for combining spatial and temporal information in two-stream neural networks for video classification. The proposed asymmetrical multiplication function for decision fusion outperformed the other three methods compared in this study, significantly better than the famous Mycin method and commonly used averaging method. The findings of this study support the idea that spatial information should be treated as dominant while temporal information must be complementary. This work has some limitations. The most important limitation is that the two-stream neural network used is not state-of-the-art due to the limited computing facility used in the experiment and cannot produce the best results. Future work could be focused on more powerful feature learning in deep neural networks and combining decision fusion with feature fusion for further improving video classification performance.

Chapter 6

Combination of Feature and Decision Fusion for Video Classification

It has been well-known that combining spatial and temporal information effectively is critical in improving the performance of video classification. Information fusion is widely used to combine sensory data, features, or decisions, and make use of the modality between different features and classifiers for getting better results. The purpose of this chapter is to investigate the effect of feature and decision level fusion on the final classification performance of multi-stream neural networks used for video classification. The experimental results indicate that (i) using both spatial and temporal features is necessary, but using temporal features as dominant representations in conjunction with two distinct spatial features outperforms other combinations, (ii) feature fusion involving RGB, HOG, and optical flow features outperforms other fusion methods investigated in this chapter, (iii) the proposed multi-level decision fusion approach achieves results comparable to those from to feature fusion, but has the advantage of lower memory requirement and computational cost, and (iv) the combinations of feature and decision fusion methods achieve performances close to that of decision-level methods and would be improved further using different parameters and weights.

6.1 Introduction

In recent years, autonomous action-based video analysis has been a popular research topic in computer vision and pattern recognition. It has a variety of potential applications in the field of content-based video retrieval.

Existing studies on video classification algorithms are primarily depend on data structures, and algorithm optimisation is still primarily performed on computers with low computational resources. Due to the increasing amount of data and data complexity, relying exclusively on the algorithm and single-computer computing resources is insufficient. The usage of distributed computing platforms to manage algorithms' massive time and space complexity consumption in big data environments has become a serious challenge.

With the development of deep learning and the trend in sharing large-scale labelled datasets, many approaches have attempted to learn the semantic representation of videos using CNNs and RNNs. However, one of the main concerns is how to merge appearance and motion information in deep architectures, which is generally referred to as information fusion.

The main contribution of this work is to have proposed methods for decision and feature fusion at different levels in multi-stream CNNs for video classification. This chapter is organised as follows: Section 6.2 provides an overview of the related work. Section 6.3 describes the methodology and the proposed architectures for feature and decision fusion at different stages, Section 6.4 presents the results, and Section 6.5 draws conclusions.

6.2 Related Work

Recognising and classifying activities from video data has received significant research attention in the disciplines of computer vision. One of the major tasks in action recognition from videos is to create an effective video representation, which is usually based on feature extraction. Another major task is to develop an effective approach for classification. Both feature extraction and classification can be carried out at multiple stages and could be combined in different manners.

CNN's ability to extract features from raw data has been a key to the development of end-to-end video classification techniques. Two important papers published in 2014 set the basis for classification using deep learning. In action recognition, Karpathy et al. [80] and Simonyan and Zisserman [81] popularised single and two-stream networks. In comparison to multi-stream networks, single-stream networks combine spatiotemporal and temporal variables and generally ignore temporal features. Multi-stream networks evaluate spatial and temporal data independently and combine the derived features from each stream to reach a final classification result.

Karpathy et al. [80] investigated how temporal data can be fused using a single-stream 2D CNN. Simonyan and Zisserman [81]] proposed the two-stream CNN that involves spatial and temporal networks which are merged by late fusion. The two-stream architecture has now been extended by other researchers [83, 84, 85, 86, 87, 88].

Combining spatial and temporal features has been a major problem for video classification. Information fusion is commonly employed for video classification architectures at multiple levels, including image, feature, and decision. The feature-level fusion process extracts correlated features from the various modalities and thus identifies a prominent set of features capable of improving recognition accuracy. It is frequently accomplished through the use of basic operations such as concatenation, product and summation [156, 157, 158]. The decision-level fusion combines multi decisions made by different classifiers at the highest level. Related work on decision fusion for video classification is described in Chapter 5.

This study investigates how feature fusion and decision fusion affects the final prediction performance without applying complex architectures. Several single and multi-stream neural networks have been developed and novel combinations of feature and decision level fusion approaches have been proposed for video classification.

6.3 Methods

This section describes the architectures of the deep neural networks, the details of the feature and decision fusion implementations and the evaluation method. ConvLSTM is used to create the proposed networks since it outperformed the other architectures as presented in the previous chapters. Moreover, 17 video classification architectures were implemented using a VGG-16 pre-trained neural network with three different input types: RGB frames, optical flows, and HOG representations.

6.3.1 Network Architectures

In this study, several CNN-based network architectures were implemented containing single-stream and multi-stream CNNs. Single-stream architecture is made of ConvLSTM architecture used in the previous chapter and it is depicted in Fig. 6.1, whose input can be RGB frames (*RGB*), optical flows (*OF*), and HOG features (*HOG*), separately.

In this study, the effect of different information fusion levels on the final decision in multi-stream neural networks for video classification was investigated. Several deep classification designs were proposed for the purpose of combining information from multiple sources such as different inputs or classifiers. Two-stream architectures were constructed using two different input types: RGB frames and optical flows (*RGB-OF*), RGB frames and HOG features (*RGB-HOG*), or HOG features and optical flows (*HOG-OF*). Two-stream RGB-OF architecture is presented in Fig. 6.2. Moreover, a three-stream neural network was implemented to test classification performance by using all three input types (*RGB-HOG-OF*) as shown in Fig. 6.3.

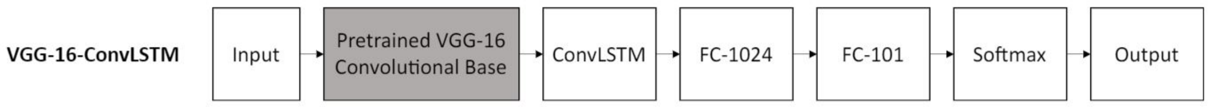

Figure 6.1: The main architecture of the networks used in the experiments

Seventeen different architectures were implemented to compare classification performance using either feature or decision fusion technique. All methods were evaluated on a holdout validation

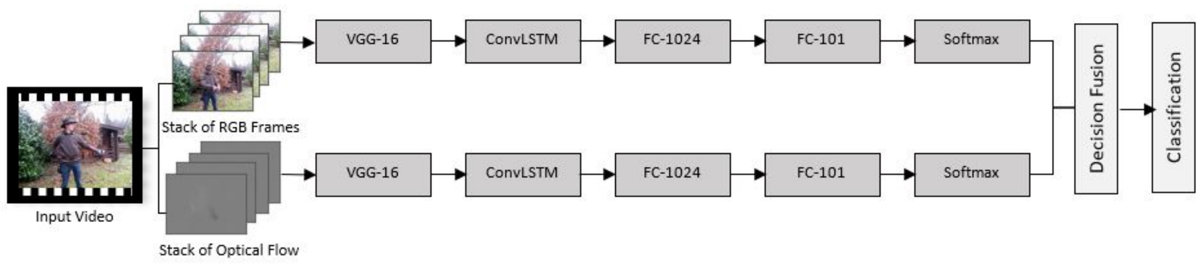

Figure 6.2: Two-stream CNN architecture used in the experiments with RGB frames and optical flows. Two-stream architectures with one stream removed were also considered in the experiments, as shown in Table 6.1.

dataset after every epoch and the performance of the methods were monitored based on the loss during the training by using the early stopping technique.

The Cuda v.9.0 toolkit on NVIDIA TITAN X GPU was employed to optimise the VGG-16 based CNN architectures implemented by using Tensorflow-gpu v.1.13 on the UCF-101 and KTH datasets.

6.3.2 Decision Fusion

Two types of decision fusion-based classification architectures are presented: single-level and multi-level. In the single-level architecture, depicted in Fig. 6.3, different inputs are transmitted into the classifiers in each stream and the final decision is based on the average scores from those streams. However, as illustrated in Fig. 6.4, the final decision is determined progressively based on the scores acquired from each pair of inputs in the multi-level decision fusion architectures.

In this study, the spatial and temporal representations are fused in various architectures, which are listed in Table 6.1, and the ultimate decision is made using the average function.

6.3.3 Feature Fusion

For single-level feature fusion, different inputs are fed into pre-trained VGG-16 neural networks, respectively. The features extracted from each stream are then fused using the concatenation technique. The combined features are used as an input of ConvLSTM as the final classifier, as illustrated in Fig. 6.5. Four different single-level feature fusion architectures with different

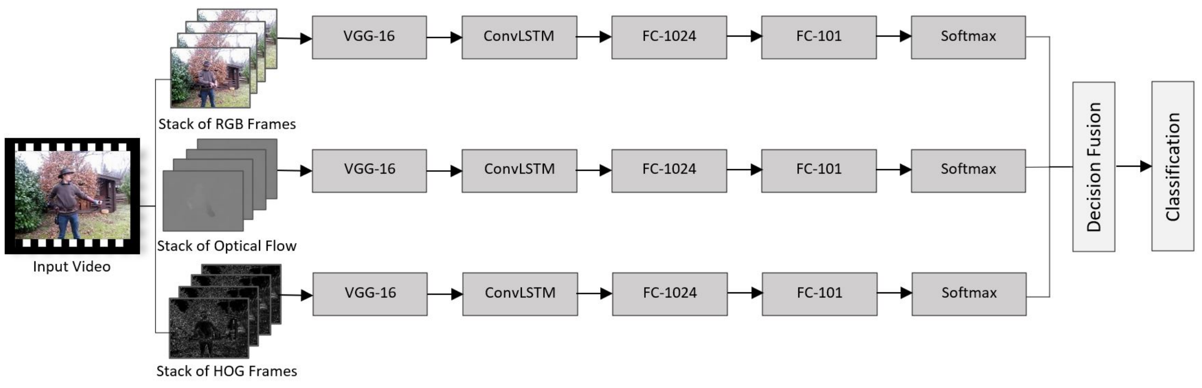

Figure 6.3: Single-level decision fusion architecture used in the experiments with RGB frames, optical flows, and HOG features.

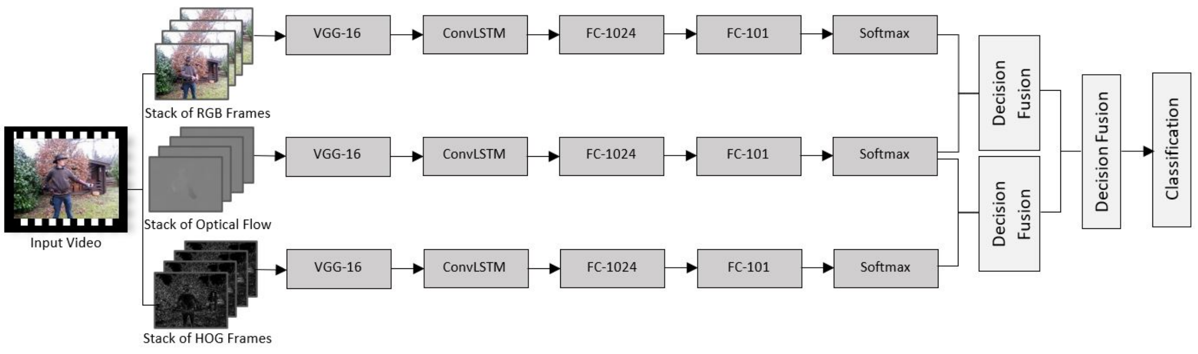

Figure 6.4: Multi-level decision fusion architecture used in the experiments with RGB frames, optical flows, and HOG features. Architectures with other combinations were also considered in the experiments, as shown in Table 6.1.

Table 6.1: Decision fusion based architectures used in the experiments

Method	Fusion Level	Input
D-RGB-OF	single-level	RGB-OF
D-RGB-HOG	single-level	RGB-HOG
D-HOG-OF	single-level	HOG-OF
D-RGB-HOG-OF	single-level	RBG-HOG-OF
D1	multi-level	D-RGB-OF and D-RGB-HOG
D2	multi-level	D-RGB-OF and D-HOG-OF
D3	multi-level	D-RGB-HOG and D-HOG-OF

combinations of input types were implemented in the experiments, as shown in Table 6.2.

The effect of combining feature and decision fusion in multi-stream neural networks for video classification was also investigated in the experiments. Spatial and temporal representations are fused by using feature fusion and then the softmax scores are combined based on the decision fusion rule as depicted in Fig. 6.6. Three different architectures for combining feature fusion and

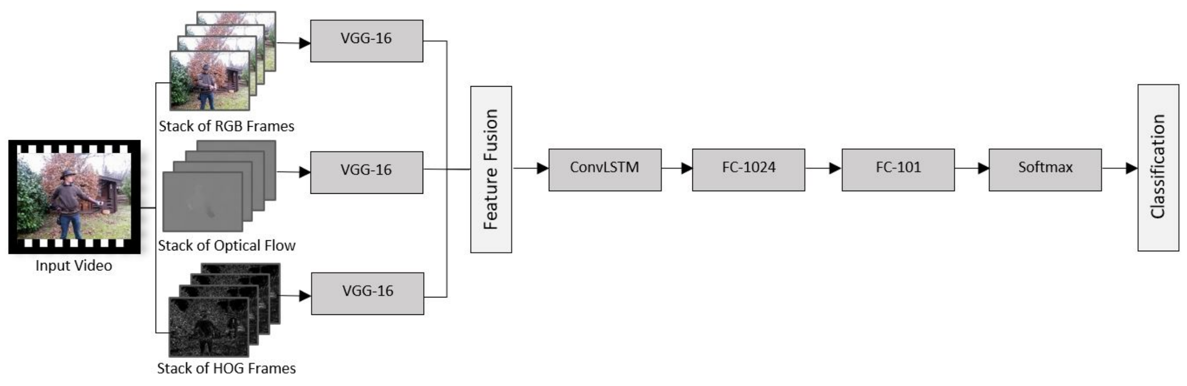

Figure 6.5: Single-level feature fusion architecture used in the experiments with RGB frames, optical flows, and HOG frames. Architectures with two input channels were also considered in the experiments, as shown in Table 6.2

Table 6.2: Feature fusion based architectures used in the experiments

Method	Fusion Level	Input
F-RGB-OF	single-level	RGB-OF
F-RGB-HOG	single-level	RGB-HOG
F-HOG-OF	single-level	HOG-OF
F-RGB-HOG-OF	single-level	RBG-HOG-OF
FD1	multi-level	F-RGB-OF and F-RGB-HOG
FD2	multi-level	F-RGB-OF and F-HOG-OF
FD3	multi-level	F-RGB-HOG and F-HOG-OF

decision fusion were implemented, as shown in Table 6.2.

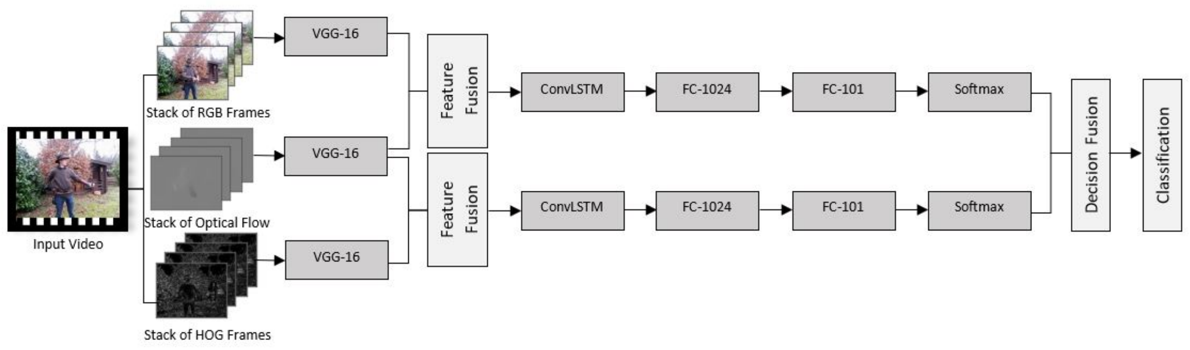

Figure 6.6: Multi-stream network architecture used in the experiments combining multi-level feature and decision fusion. Architectures with other combinations were also considered in the experiments, as shown in Table 6.2

6.3.4 Datasets and Evaluation

The UCF-101 and KTH datasets are utilised in this study to test neural network architectures built for video classification. The UCF-101 dataset contains 13,320 clips with a resolution of

240x360 pixels from 101 different classes while the KTH action recognition dataset contains almost 2,300 video sequences representing six different types of human actions. The datasets are described in detail in Chapter 2.

In the experiments, confusion matrices are produced for performance analysis and accuracy is used for the comparison of the performances achieved by different architectures. 255 training, validation, and testing accuracy scores have been collected with the three training-testing splits released by the UCF-101 organisation (5 times per split and 15 times per classifier). Similarly, 204 accuracy scores have been collected with the official training, validation, and testing splits of the KTH dataset (12 times per classifier).

The Kolmogorov-Smirnov [221] test was used to compare the observed cumulative distribution to the cumulative distribution that would occur if the data were normally distributed. The Levene statistic [222] reveals that variances of groups are homogeneous based on mean and median. ANOVA test was also used to compare variance differences to see if the results were significant. Finally, the Tukey's HSD test was used to see if the means of the groups were different. The results are summarised in the next section.

6.4 Results and Discussion

In this study, seventeen different CNN-based fusion architectures were designed and compared for video classification. Twelve different experiments were undertaken for each architecture on the KTH dataset (total 204 runs), whereas 15 experiments were conducted on the UCF-101 dataset (total 255 runs). After each run, training, validation and testing accuracy scores were collected and the results are presented in this section.

Table 6.3 presents the average accuracy scores achieved by single-stream neural networks, *RGB*, *HOG*, and *OF*, respectively, on the KTH and UCF-101 datasets. Interestingly, the *OF* architecture achieved the greatest test accuracy score of 81.44% and 73.66% on the KTH and UCF-101, respectively. Temporal network, *OF*, significantly outperforms the spatial networks, *HOG* and

RGB, demonstrating the critical role of motion information in video classification. On the KTH dataset, the *HOG* architecture outperformed the *RGB* architecture by nearly four percentage, while its performance remained close to that of the *RGB* architecture on the UCF dataset. The reason for this difference can be spatial differences between both datasets such as the number of colour channels, action complexity, colour intensity, and other appearance related dynamics.

Table 6.3: Training, validation, and testing accuracy scores achieved by single-stream networks on the KTH dataset

Method	Dataset	Training Accuracy	Validation Accuracy	Testing Accuracy
RGB	KTH	87.18%	65.89%	73.07%
HOG	KTH	92.40%	70.62%	77.93%
OF	KTH	98.52%	73.41%	81.44%
RGB	UCF	86.56%	68.77%	70.03%
HOG	UCF	88.33%	68.21%	70.80%
OF	UCF	93.81%	70.64%	73.66%

The performances achieved by the multi-stream neural network architectures are summarised in Table 6.4. In general, single-level feature fusion based architectures outperformed the decision fusion based architectures. In both architectures, the methods fed with optical flows and HOG features, *D-HOG-OF, D2, F-HOG-OF* performed better than those with other feature pairs. However, the highest classification performances were observed when fusing RGB, HOG, and optical flow features together at feature-level with 84.10% on the KTH and 77.98% on the UCF-101. This is followed by *D2* where optical flow features are dominantly fused with both RGB and HOG features. These results confirm that the use of temporal information is essential in video classification.

As shown in Table 6.5, there was a statistically significant difference at the $p < 0.05$ level in test accuracy scores achieved by the different fusion architectures on the KTH, F (16,187) = 85.239; p = 0.001, and the UCF-101, F (16,238) = 209.524; p = 0.001. Additional comparisons were performed using the Tukey's HSD test. As illustrated in Tables B.1-B.17 and B.18-B.34 (See in Appendix B), the results demonstrated that there were significant differences in classification performance across the majority of architectures (* indicates significant differences at the 0.05 level).

Table 6.4: Classification accuracy achieved by multi-stream networks based on decision and feature fusion

Method	Fusion Level	KTH Test Accuracy	UCF Test Accuracy
D-RGB-HOG	single-level	78.24%	71.89%
D-RGB-OF	single-level	81.83%	74.87%
D-HOG-OF	single-level	82.91%	76.32%
D-RGB-HOG-OF	single-level	81.79%	75.89%
D1	multi-level	79.78%	73.39%
D2	multi-level	83.68%	76.31%
D3	multi-level	80.90%	74.23%
F-RGB-HOG	single-level	71.80%	73.34%
F-RGB-OF	single-level	81.25%	74.89%
F-HOG-OF	single-level	81.33%	75.79%
F-RGB-HOG-OF	single-level	84.10%	77.98%
FD1	multi-level	80.75%	73.40%
FD2	multi-level	81.48%	75.16%
FD3	multi-level	82.25%	76.07%

Table 6.5: One-way between-groups ANOVA of performance achieved by different fusion architectures on the KTH and UCF-101 datasets where df, *SS*, *MS* and F refer to degrees of freedom, sum of squares, mean sum of squares, and F score, respectively.

		SS	df	MS	F	Sig.
KTH	Between Groups	.217	16	.014	85.239	.001
	Within Groups	.030	187	.000		
	Total	.247	203			
UCF	Between Groups	.105	16	.007	209.524	.001
	Within Groups	.007	238	.000		
	Total	.113	254			

Tables 6.6 and 6.7 show which architecture pairs have significantly different means of the test accuracy scores on the KTH and UCF-101 datasets. The post hoc test provides homogeneous subset results, with the groups listed in ascending order of mean. The means in each subset are not statistically different from one another. There are 17 groups in the factor, and their dependent variable means were sorted according to their factor level, i.e., group *F-RGB-HOG* and group *RGB* had the lowest mean, while group *F-RGB-HOG-OF* had the highest mean. There are 8 different subsets on the KTH dataset while 10 on the UCF-101 dataset. On both datasets, *F-RGB-HOG-OF* is significantly different from all other groups as it does not appear in a subset together with any of the other groups.

The proposed methods *D2* and *FD3* have achieved results comparable to the *F-RGB-HOG-OF* method and they have the advantage of lower memory requirement and computational cost. The results suggest that higher-dimensional feature vectors are advantageous and this approach may benefit from dimensionality reduction. The accuracy scores achieved by the proposed methods may be further improved by optimising the fusion algorithm such as using different weighting or fusion techniques. However, conducting these experiments requires a significant allocation of resources.

The combinations of combined multi-level feature and decision fusion architectures, *FD1, FD2, and FD3*, have achieved performances close to that achieved by either decision fusion methods or feature fusion methods. There is no significant difference between any of these methods on the KTH dataset whereas there is a significant difference between all combinations on the UCF-101 dataset. Interestingly, the combined methods are not significantly different from either their decision fusion or feature fusion streams. Unfortunately, combining feature fusion with decision fusion did not work well as expected. In the experiments, all features are treated equally and only concatenation and averaging were used to combine features and decisions. All features are used in each combination, thus any of the combinations did not make a significant difference. Based on the experimental results, it is expected that using different parameters and weights for each feature would affect the performance of this proposed method. For example, the rest of the results show that using optical flows in both streams can contribute to reach higher prediction performance and giving higher weight to the optical flow features would improve the performance further. According to confusion matrices, optical flow features helped to reduce the confusion between similar action categories in spatial feature-based approaches. For example, in the KTH dataset, action pairs having similar spatial features (hand waving-hand clapping, running-jogging, and running-walking) had lower confusion between them when optical flow features were employed. Optical flow extraction helped to differentiate those action pairs by making use of temporal changes; therefore, optical flow-based architectures achieved better classification performance than spatial feature-based architectures.

Table 6.6: One-way between-groups ANOVA of performance achieved by different fusion architectures on the KTH dataset

Method	1	2	3	4	5	6	7	8
F-RGB-HOG	71.80%							
RGB	73.07%							
HOG		77.93%						
D-RGB-HOG		78.24%	78.24%					
D1			79.78%	79.78%				
FD1				80.75%	80.75%			
D3				80.90%	80.90%			
F-RGB-OF				81.25%	81.25%	81.25%		
F-HOG-OF				81.33%	81.33%	81.33%		
OF				81.44%	81.44%	81.44%		
FD2				81.48%	81.48%	81.48%		
D-RGB-HOG-OF					81.79%	81.79%		
D-RGB-OF					81.83%	81.83%		
FD3					82.25%	82.25%	82.25%	
D-HOG-OF						82.91%	82.91%	82.91%
D2							83.68%	83.68%
F-RGB-HOG-OF								84.10%
Sig.	0.527	1.000	0.197	0.092	0.233	0.113	0.318	0.638

Table 6.7: One-way between-groups ANOVA of performance achieved by different fusion architectures on the UCF-101 dataset

Method	1	2	3	4	5	6	7	8	9	10
RGB	70.03%									
HOG		70.80%								
D-RGB-HOG			71.89%							
F-RGB-HOG				73.34%						
D1				73.39%						
FD1				73.40%						
OF				73.66%	73.66%					
D3					74.23%	74.23%				
D-RGB-OF						74.87%	74.87%			
F-RGB-OF						74.89%	74.89%			
FD2							75.16%	75.16%		
F-HOG-OF								75.79%	75.79%	
D-RGB-HOG-OF									75.89%	
FD3									76.07%	
D2									76.31%	
D-HOG-OF									76.32%	
F-RGB-HOG-OF										77.98%
Sig.	1.000	1.000	1.000	0.980	0.296	0.115	0.991	0.157	0.449	1.000

In this research, the commonly used averaging technique is used to combine different softmax scores for decision fusion whilst the concatenation technique is used to combine features for feature fusion. There have been many techniques proposed for both decision fusion and feature fusion in video classification, as summarised in Chapter 2 and 5. Other fusion approaches can be used to further increase the performance of the proposed architectures in this study. For example, in two-stream neural networks, the asymmetrical multiplication method proposed in

Chapter 5 for decision fusion significantly outperformed the average method. Although the asymmetrical multiplication was evaluated on only two-stream neural networks, it might improve the performance of decision fusion in multi-stream neural networks as well. Similarly, weighted averaging can enhance the performance of architectures based on feature fusion. In this study, alternative fusion techniques were not included due to the limited time and resources, which is an important topic for future research.

6.5 Conclusion

This chapter proposes feature and decision fusion methods in multi-stream neural networks for video classification. In contrast to single-stream approaches that take either spatial or temporal inputs, the proposed multi-stream fusion approaches take the advantage of aggregating both spatial and temporal features at different levels. When comparing feature-level fusion versus decision-level fusion algorithms, the feature-level fusion algorithm has a better chance of achieving superior results. Fusion at the feature-level is expected to outperform fusion at the decision-level in terms of performance. The experimental results show that the highest accuracy score was achieved by feature fusion containing all features; however, the proposed multi-stream decision fusion method achieved comparable results with the advantage of lower memory requirement and computational cost. Due to the modalities' unique features, the above-mentioned features may not be compatible in many cases. Feature concatenation may also result in a feature vector with an extremely high dimension. Complex classifier design may be required to handle the concatenated data set at the feature-level of operation. Therefore, decision-level fusion approaches are more likely to be practical.

Chapter 7

Conclusions

In this thesis, several novel methods for video classification, specifically human action recognition, were proposed and evaluated. The details of the proposed methods are presented in the previous chapters. To conclude the thesis, this chapter summarises research contributions in Section 7.1 and discuss limitations and the directions for the possible future work in Section 7.2.

7.1 Contributions

The thesis work aims at developing new deep learning methods for video classification. This research has made several contributions to the relevant field and the main contributions of this research can be summarised as follows.

Firstly, a comprehensive review of the state-of-the-art methods was conducted. This study contributes to our understanding of video scene understanding involving preprocessing, feature extraction, classification and information fusion. The key findings and analyses related to video classification are presented in Chapter 2.

Secondly, a novel video classification architecture was proposed involving spatial and temporal features extracted by using the transfer learning technique. In this work, seven different network architectures involving different RNNs were developed by either using local or global features. The experimental results indicated that integrating pre-trained CNN with ConvLSTM achieves the highest performance for video classification among the seven network architectures tested, highlighting the important role of effective spatiotemporal information integration in video classification. This research also provided an optimised network baseline for the remaining experimental works in this thesis.

Thirdly, a novel action template-based keyframe extraction method was developed by using the visual similarity between consecutive frames. This approach considers the similarity between two action regions rather than whole frames. To achieve this, an action template is detected automatically and similarity is taken into account to extract keyframes. The proposed method was compared with two well-known keyframe extraction methods. The results demonstrate that the proposed keyframe extraction method can select informative frames reliably and thus significantly improve the performance of deep neural network architectures for video classification. This study also showed that when finding the relevant area using the extracted action template the proposed method successfully extracts proper keyframes from human action videos for video classification using deep neural networks.

Fourthly, a novel asymmetrical multiplication function was proposed for decision-level fusion in two-stream neural networks. The proposed asymmetrical multiplication function outperformed the other three approaches tested in this study, outperforming both the well-known Mycin method and the widely utilised averaging method. The findings of this research showed that using both spatial and temporal information is essential, but spatial information can be dominant in two-stream neural networks.

Finally, a comprehensive investigation on how to combine feature and decision fusion in multi-stream neural networks was conducted involving seventeen different classification architectures. In this research, multi-level feature and decision fusion methods were also proposed to improve classification performance. Although feature fusion using three different feature types outperformed the other architectures, the proposed combination method has achieved a comparable result with the advantage of lower memory requirement and computational cost. The results also indicated that using temporal information as dominant along with different spatial features helped to make a better prediction in multi-level fusion architectures.

7.2 Limitations and Future Work

Whilst the thesis has successfully demonstrated that the proposed methods outperformed the existing approaches, some limitations need to be acknowledged. The major limitations of this study are due to limited time and hardware resources. Training and optimising deep neural networks require a long time on video datasets. Deep networks have millions of parameters and the classification architectures used in this thesis are optimised based on cross-validation using a limited set of hyper-parameter values (selected by roughly partitioning the hyper-parameter space). Moreover, there are several techniques available to accelerate the training process and make it more efficient, but these techniques require training on GPUs. However, we had restricted access to GPUs during the experiments. It is unfortunate that the study did not include multiple GPUs and larger datasets. Considering the complex dimension of video data, experiments also require larger memory capacities thus different batch sizes could not be used during the research.

Another limitation with the current study is that CNN architectures used in the evaluation of the proposed methods were not state-of-the-art architectures. Although they outperformed the existing methods under the same experimental conditions, they did not produce the best results in the literature. The current state-of-the-art methods on the UCF-101 dataset are achieved by SMART [100], OmniSource [99], and PERT [113] networks with 96% accuracy whereas the highest performance recorded on the KTH dataset is 98.2% [171]. The scope of this study was limited in terms of comparing the proposed methods with the existing approaches rather than achieving state-of-the-art results. Thus, future work could focus on the application of the proposed algorithm to real-world video classification using more powerful architectures.

Multi-stream neural networks used in this study require pre-computing the optical flow representations. The use of optical flows improves the classification performance; however, it also limits the application of these architectures in real-time.

The proposed method for combining feature and decision fusion in multi-stream neural networks is limited with two fusion techniques: concatenation and averaging. Other fusion methods were not included during the experiments due to the limited computing facility. Using other fusion techniques can improve the classification performance, thus future work could be focused on the extension of this research using more powerful fusion approaches.

The proposed methods in this thesis were evaluated on the same datasets. The KTH dataset focuses on the detection of only one action event for a single person in a video and the UCF-101 dataset consists of multiple people in some videos acting only the same action. However, real-world video data can include multiple people performing different actions at the same time. Similar to most of the related work, this research focused on single action recognition from the videos rather than the recognition of multiple actions performed by more than one person at the same time. In the future, the proposed methods can be improved by detecting and tracking multiple people appearing in the scene and eventually recognising their actions. We also intend to test the proposed methods using larger pre-trained neural networks on multiple datasets in the future to determine whether performance improves or remains the same.

The proposed action template-based keyframe extraction method showed high accuracy in action recognition on the KTH and UCF-101 datasets. This method has the potential to be developed and applied to a variety of other video-based tasks in future including emotion recognition, pedestrian detection, vehicle detection, abnormal behaviour detection, video summarization, etc. The proposed keyframe extraction method would provide further development for video-based models and approaches.

Most of the available deep learning-based video classification methods are not appropriate for real-time classification. It is essential that the existing methods must be extended to be applicable in real-time applications. We plan to make the required update on the proposed methods to make them appropriate for real-time video classification.

By addressing the aforementioned limitations and obtaining answers to relevant concerns, the proposed approaches in this thesis will be greatly improved. These concerns should be addressed properly in future studies.

Bibliography

[1] H. Foroughi, A. Naseri, A. Saberi, and H. S. Yazdi, “An eigenspace-based approach for human fall detection using integrated time motion image and neural network,” in *2008 9th International Conference on Signal Processing*. IEEE, 2008, pp. 1499–1503.

[2] T.-H. Tsai and C.-W. Hsu, “Implementation of fall detection system based on 3d skeleton for deep learning technique,” *IEEE Access*, vol. 7, pp. 153 049–153 059, 2019.

[3] A. Chelli and M. Pätzold, “A machine learning approach for fall detection and daily living activity recognition,” *IEEE Access*, vol. 7, pp. 38 670–38 687, 2019.

[4] H. Li, A. Shrestha, H. Heidari, J. Le Kernec, and F. Fioranelli, “Bi-lstm network for multimodal continuous human activity recognition and fall detection,” *IEEE Sensors Journal*, vol. 20, no. 3, pp. 1191–1201, 2019.

[5] R. Savran Kızıltepe, J. Q. Gan, and J. J. Escobar, “Combining very deep convolutional neural networks and recurrent neural networks for video classification,” in *Advances in Computational Intelligence*, I. Rojas, G. Joya, and A. Catala, Eds. Cham: Springer International Publishing, 2019, pp. 811–822.

[6] R. Savran Kızıltepe, J. Q. Gan, and J. J. Escobar, “A novel keyframe extraction method for video classification using deep neural networks,” *Neural Computing and Applications*, pp. 1–12, 2021.

[7] R. Savran Kızıltepe and J. Q. Gan, “Simple effective methods for decision-level fusion in two-stream convolutional neural networks for video classification,” in *Intelligent Data Engineering and Automated Learning – IDEAL 2020*, C. Analide, P. Novais, D. Camacho, and H. Yin, Eds. Cham: Springer International Publishing, 2020, pp. 77–87.

[8] C. A. Bhatt and M. S. Kankanhalli, “Multimedia data mining: state of the art and challenges,” *Multimedia Tools and Applications*, vol. 51, no. 1, pp. 35–76, 2011.

[9] V. Vijayakumar and R. Nedunchezhian, “A study on video data mining,” *International Journal of Multimedia Information Retrieval*, vol. 1, no. 3, pp. 153–172, 2012.

[10] A. Kumar, S. R. Sangwan, and A. Nayyar, “Multimedia social big data: Mining,” in *Multimedia Big Data Computing for IoT Applications*. Springer, 2020, pp. 289–321.

[11] S. Hiriyannaiah, K. Singh, H. Ashwin, G. Siddesh, and K. Srinivasa, “Deep learning and its applications for content-based video retrieval,” in *Hybrid Computational Intelligence*. Elsevier, 2020, pp. 49–68.

[12] M. A. Smith and T. Chen, “9.1 - image and video indexing and retrieval,” in *Handbook of Image and Video Processing*, second edition ed., ser. Communications, Networking and Multimedia, A. BOVIK, Ed. Burlington: Academic Press, 2005, pp. 993–1012.

[13] C.-W. Ngo, “Video shot detection,” in *Encyclopedia of Database Systems*, L. LIU and M. T. ÖZSU, Eds. Boston, MA: Springer US, 2009, pp. 3316–3320.

[14] I. Koprinska and S. Carrato, “Temporal video segmentation: A survey,” *Signal Processing: Image Communication*, vol. 16, no. 5, pp. 477–500, 2001.

[15] S. Manjunath, D. Guru, M. Suraj, and B. Harish, “A non parametric shot boundary detection: an eigen gap based approach,” in *Proceedings of the Fourth Annual ACM Bangalore Conference*, 2011, pp. 1–7.

[16] D. Guru, M. Suhil, and P. Lolika, “A novel approach for shot boundary detection in videos,” in *Multimedia Processing, Communication and Computing Applications*. Springer, 2013, pp. 209–220.

[17] F. Idris and S. Panchanathan, “Review of image and video indexing techniques,” *Journal of Visual Communication and Image Representation*, vol. 8, no. 2, pp. 146–166, 1997.

[18] B. Patel and B. Meshram, “Content based video retrieval systems,” *International Journal of UbiComp (IJU)*, vol. 3, no. 2, 2012.

[19] J. S. Boreczky and L. A. Rowe, "Comparison of video shot boundary detection techniques," *Journal of Electronic Imaging*, vol. 5, no. 2, pp. 122–128, 1996.

[20] R. W. Lienhart, "Comparison of automatic shot boundary detection algorithms," in *Storage and Retrieval for Image and Video Databases VII*, vol. 3656. International Society for Optics and Photonics, 1998, pp. 290–301.

[21] A. F. Smeaton, P. Over, and A. R. Doherty, "Video shot boundary detection: Seven years of trecvid activity," *Computer Vision and Image Understanding*, vol. 114, no. 4, pp. 411–418, 2010.

[22] W. Wolf, "Key frame selection by motion analysis," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, 1996, pp. 1228–1231.

[23] X. Yan, S. Z. Gilani, H. Qin, M. Feng, L. Zhang, and A. S. Mian, "Deep keyframe detection in human action videos," *CoRR*, vol. abs/1804.10021, 2018. [Online]. Available: http://arxiv.org/abs/1804.10021

[24] S. K. Kuanar, R. Panda, and A. S. Chowdhury, "Video key frame extraction through dynamic delaunay clustering with a structural constraint," *Journal of Visual Communication and Image Representation*, vol. 24, no. 7, pp. 1212–1227, 2013.

[25] A. Kar, N. Rai, K. Sikka, and G. Sharma, "Adascan: Adaptive scan pooling in deep convolutional neural networks for human action recognition in videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3376–3385.

[26] W. Zhu, J. Hu, G. Sun, X. Cao, and Y. Qiao, "A key volume mining deep framework for action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1991–1999.

[27] J. K. Aggarwal and Q. Cai, "Human motion analysis: A review," *Computer Vision and Image Understanding*, vol. 73, no. 3, pp. 428–440, 1999.

[28] L. Wang, W. Hu, and T. Tan, "Recent developments in human motion analysis," *Pattern Recognition*, vol. 36, no. 3, pp. 585–601, 2003.

[29] R. Poppe, "A survey on vision-based human action recognition," *Image and Vision Computing*, vol. 28, no. 6, pp. 976–990, 2010.

[30] J. K. Aggarwal and L. Xia, "Human activity recognition from 3d data: A review," *Pattern Recognition Letters*, vol. 48, pp. 70–80, 2014.

[31] M. Vrigkas, C. Nikou, and I. A. Kakadiaris, "A review of human activity recognition methods," *Frontiers in Robotics and AI*, vol. 2, p. 28, 2015.

[32] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 886–893.

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

[34] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2. Ieee, 1999, pp. 1150–1157.

[35] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European Conference on Computer Vision*. Springer, 2006, pp. 404–417.

[36] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[37] I. Laptev, "On space-time interest points," *International Journal of Computer Vision*, vol. 64, no. 2, pp. 107–123, 2005.

[38] N. Dalal, B. Triggs, and C. Schmid, "Human detection using oriented histograms of flow and appearance," in *European Conference on Computer Vision*. Springer, 2006, pp. 428–441.

[39] F. Perronnin and C. Dance, "Fisher kernels on visual vocabularies for image categorization,"

in *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–8.

[40] A. Klaser, M. Marszałek, and C. Schmid, "A spatio-temporal descriptor based on 3d-gradients," in *BMVC 2008-19th British Machine Vision Conference*. British Machine Vision Association, 2008, pp. 275–1.

[41] J. J. Gibson, "The perception of the visual world." 1950.

[42] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Action recognition by dense trajectories," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 3169–3176.

[43] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Dense trajectories and motion boundary descriptors for action recognition," *International Journal of Computer Vision*, vol. 103, no. 1, pp. 60–79, 2013.

[44] P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional sift descriptor and its application to action recognition," in *Proceedings of the 15th ACM International Conference on Multimedia*, 2007, pp. 357–360.

[45] G. T. Flitton, T. P. Breckon, and N. M. Bouallagu, "Object recognition using 3d sift in complex ct volumes," in *Proceedings of the British Machine Vision Conference*, vol. 1. Citeseer, 2010, pp. 1–12.

[46] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[47] J. Bauer, N. Sünderhauf, and P. Protzel, "Comparing several implementations of two recently published feature detectors," *Proceedings of the 6th IFAC Symposium on Intelligent Autonomous Vehicles*, vol. 40, no. 15, pp. 143–148, 2007.

[48] E. Oyallon and J. Rabin, "An analysis of the surf method," *Image Processing On Line*, vol. 5, pp. 176–218, 2015.

[49] T. Lindeberg, “Image matching using generalized scale-space interest points,” *Journal of Mathematical Imaging and Vision*, vol. 52, no. 1, pp. 3–36, 2015.

[50] Y. Guo, “Spatio-temporal sift interest points detection in videos,” *Zhejiang Univ., Hangzhou, China, Tech. Rep*, 2009.

[51] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, “Behavior recognition via sparse spatio-temporal features,” in *2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*. IEEE, 2005, pp. 65–72.

[52] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, “Visual categorization with bags of keypoints,” in *Workshop on Statistical Learning in Computer Vision, ECCV*, vol. 1, no. 1-22. Prague, 2004, pp. 1–2.

[53] F. S. Khan, J. Van De Weijer, A. D. Bagdanov, and M. Felsberg, “Scale coding bag-of-words for action recognition,” in *2014 22nd International Conference on Pattern Recognition*. IEEE, 2014, pp. 1514–1519.

[54] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, “Learning spatiotemporal features with 3d convolutional networks,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4489–4497.

[55] L. Zhang, R. Khusainov, and J. Chiverton, “Practical action recognition with manifold regularized sparse representation,” in *29th British Machine Vision Conference*. British Machine Vision Association, 2018.

[56] M. F. Aslan, A. Durdu, and K. Sabanci, “Human action recognition with bag of visual words using different machine learning methods and hyperparameter optimization,” *Neural Computing and Applications*, vol. 32, no. 12, pp. 8585–8597, 2020.

[57] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, “Image classification with the fisher vector: Theory and practice,” *International Journal of Computer Vision*, vol. 105, no. 3, pp. 222–245, 2013.

[58] O. Kliper-Gross, Y. Gurovich, T. Hassner, and L. Wolf, “Motion interchange patterns for

action recognition in unconstrained videos," in *European Conference on Computer Vision*. Springer, 2012, pp. 256–269.

[59] D. Oneata, J. Verbeek, and C. Schmid, "Action and event recognition with fisher vectors on a compact feature set," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 1817–1824.

[60] X. Peng, C. Zou, Y. Qiao, and Q. Peng, "Action recognition with stacked fisher vectors," in *European Conference on Computer Vision*. Springer, 2014, pp. 581–595.

[61] G. Luo, J. Wei, W. Hu, and S. J. Maybank, "Tangent fisher vector on matrix manifolds for action recognition," *IEEE Transactions on Image Processing*, vol. 29, pp. 3052–3064, 2019.

[62] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *International Conference on Artificial Neural Networks*. Springer, 2011, pp. 52–59.

[63] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," in *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, 2012, pp. 37–49.

[64] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[65] H. Yang, B. Wang, S. Lin, D. Wipf, M. Guo, and B. Guo, "Unsupervised extraction of video highlights via robust recurrent auto-encoders," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4633–4641.

[66] J. R. Zhang, Y. Song, and T. Leung, "Improving video classification via youtube video co-watch data," in *Proceedings of the 2011 ACM Workshop on Social and Behavioural Networked Media Access*, 2011, pp. 21–26.

[67] S. Schmiedeke, P. Xu, I. Ferrané, M. Eskevich, C. Kofler, M. A. Larson, Y. Estève, L. Lamel, G. J. Jones, and T. Sikora, "Blip10000: A social video dataset containing spug

content for tagging and retrieval," in *Proceedings of the 4th ACM Multimedia Systems Conference*, 2013, pp. 96–101.

[68] M. C. Darji, D. N. Patel, and Z. H. Shah, "A review on audio features based extraction of songs from movies," *International Journal of Advance Engineering and Research Development*, pp. 2348–4470, 2015.

[69] N. Takahashi, M. Gygli, and L. Van Gool, "Aenet: Learning deep audio features for video analysis," *IEEE Transactions on Multimedia*, vol. 20, no. 3, pp. 513–524, 2017.

[70] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[71] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[72] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[73] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[74] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[75] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4700–4708.

[76] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: an astounding baseline for recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 806–813.

[77] M. Jain, J. van Gemert, C. G. Snoek *et al.*, "University of amsterdam at thumos challenge 2014," *ECCV THUMOS Challenge*, vol. 2014, 2014.

[78] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5455–5516, 2020.

[79] S. Zha, F. Luisier, W. Andrews, N. Srivastava, and R. Salakhutdinov, "Exploiting image-trained CNN architectures for unconstrained video classification," *CoRR*, vol. abs/1503.04144, 2015. [Online]. Available: http://arxiv.org/abs/1503.04144

[80] 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 the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[81] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in Neural Information Processing Systems*, 2014, pp. 568–576.

[82] D. Castro, S. Hickson, P. Sangkloy, B. Mittal, S. Dai, J. Hays, and I. A. Essa, "Let's dance: Learning from online dance videos," *CoRR*, vol. abs/1801.07388, 2018. [Online]. Available: http://arxiv.org/abs/1801.07388

[83] G. Chéron, I. Laptev, and C. Schmid, "P-cnn: Pose-based cnn features for action recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3218–3226.

[84] 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 the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2625–2634.

[85] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in

Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 4694–4702.

[86] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *International Conference on Machine Learning*, 2015, pp. 843–852.

[87] P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "Learning to track for spatio-temporal action localization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3164–3172.

[88] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1933–1941.

[89] 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*, vol. 35, no. 1, pp. 221–231, 2012.

[90] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville, "Describing videos by exploiting temporal structure," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4507–4515.

[91] Y. Zhu, Z. Lan, S. Newsam, and A. Hauptmann, "Hidden two-stream convolutional networks for action recognition," in *Asian Conference on Computer Vision.* Springer, 2018, pp. 363–378.

[92] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4724–4733.

[93] K. Hara, H. Kataoka, and Y. Satoh, "Learning spatio-temporal features with 3d residual networks for action recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3154–3160.

[94] Z. Qiu, T. Yao, and T. Mei, "Learning spatio-temporal representation with pseudo-3d

residual networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5533–5541.

[95] D. He, Z. Zhou, C. Gan, F. Li, X. Liu, Y. Li, L. Wang, and S. Wen, "Stnet: Local and global spatial-temporal modeling for action recognition," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 8401–8408.

[96] V. Choutas, P. Weinzaepfel, J. Revaud, and C. Schmid, "PoTion: Pose MoTion Representation for Action Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Salt Lake City, United States: IEEE, 2018, pp. 7024–7033.

[97] D. Tran, H. Wang, L. Torresani, and M. Feiszli, "Video classification with channel-separated convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 5552–5561.

[98] Z. Qiu, T. Yao, C.-W. Ngo, X. Tian, and T. Mei, "Learning spatio-temporal representation with local and global diffusion," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 056–12 065.

[99] H. Duan, Y. Zhao, Y. Xiong, W. Liu, and D. Lin, "Omni-sourced webly-supervised learning for video recognition," in *European Conference on Computer Vision*. Springer, 2020, pp. 670–688.

[100] S. N. Gowda, M. Rohrbach, and L. Sevilla-Lara, "SMART frame selection for action recognition," *CoRR*, vol. abs/2012.10671, 2020. [Online]. Available: https://arxiv.org/abs/2012.10671

[101] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," *Center for Research in Computer Vision*, vol. 2, 2012.

[102] Z. Wu, T. Yao, Y. Fu, and Y.-G. Jiang, "Deep learning for video classification and captioning," in *Frontiers of Multimedia Research*, 2017, pp. 3–29.

[103] Z. Sun, J. Liu, Q. Ke, H. Rahmani, M. Bennamoun, and G. Wang, "Human action

recognition from various data modalities: A review," *CoRR*, vol. abs/2012.11866, 2020. [Online]. Available: https://arxiv.org/abs/2012.11866

[104] A. Rehman and S. B. Belhaouari, "Deep learning for video classification: A review," *TechRxiv. Preprint. https://doi.org/10.36227/techrxiv.15172920.v1*, 2021.

[105] X. Peng, L. Wang, X. Wang, and Y. Qiao, "Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice," *Computer Vision and Image Understanding*, vol. 150, pp. 109–125, 2016.

[106] X. Wang, A. Farhadi, and A. Gupta, "Actions˜ transformations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2658–2667.

[107] Z. Wu, Y.-G. Jiang, X. Wang, H. Ye, and X. Xue, "Multi-stream multi-class fusion of deep networks for video classification," in *Proceedings of the 24th ACM International Conference on Multimedia*, 2016, pp. 791–800.

[108] 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," in *European Conference on Computer Vision*. Springer, 2016, pp. 20–36.

[109] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy, "Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 305–321.

[110] J. Stroud, D. Ross, C. Sun, J. Deng, and R. Sukthankar, "D3d: Distilled 3d networks for video action recognition," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 625–634.

[111] J. Hong, B. Cho, Y. W. Hong, and H. Byun, "Contextual action cues from camera sensor for multi-stream action recognition," *Sensors*, vol. 19, no. 6, p. 1382, 2019.

[112] B. I. LO, M. V. HC, and W. R. Schwartz, "Bubblenet: A disperse recurrent structure to recognize activities," in *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2020, pp. 2216–2220.

[113] Y. Li, Z. Lu, X. Xiong, and J. Huang, "Perf-net: Pose empowered rgb-flow net," *CoRR*, vol. abs/2009.13087, 2020. [Online]. Available: https://arxiv.org/abs/2009.13087

[114] D. H. Hubel and T. N. Wiesel, "Receptive fields and functional architecture of monkey striate cortex," *The Journal of Physiology*, vol. 195, no. 1, pp. 215–243, 1968.

[115] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[116] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain." *Psychological Review*, vol. 65, no. 6, p. 386, 1958.

[117] A. Ivakhnenko and V. Lapa, "Cybernetic predicting devices. ccm information corporation," *First working Deep Learners with many layers, learning internal representations*, 1965.

[118] A. Ivakhnenko, "Cybernetic systems with combined control," Foreign Technology Division Wright-Patterson Air Force (Ohio), Tech. Rep., 1967.

[119] P. Werbos, "Beyond regression:" new tools for prediction and analysis in the behavioral sciences," *Ph. D. dissertation, Harvard University*, 1974.

[120] F.-F. Li, J. Wu, and R. Gao, "Cs231n: Convolutional neural networks for visual recognition," 2022, retrieved from https://cs231n.github.io/neural-networks-3/.

[121] S. Dreyfus, "The computational solution of optimal control problems with time lag," *IEEE Transactions on Automatic Control*, vol. 18, no. 4, pp. 383–385, 1973.

[122] K. Fukushima and S. Miyake, "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition," in *Competition and cooperation in neural nets*. Springer, 1982, pp. 267–285.

[123] M. S. Thomas and J. L. McClelland, "Connectionist models of cognition," *The Cambridge Handbook of Computational Psychology*, pp. 23–58, 2008.

[124] K. J. Lang, A. H. Waibel, and G. E. Hinton, "A time-delay neural network architecture for isolated word recognition," *Neural Networks*, vol. 3, no. 1, pp. 23–43, 1990.

[125] D. A. Pomerleau, "Knowledge-based training of artificial neural networks for autonomous robot driving," in *Robot Learning*. Springer, 1993, pp. 19–43.

[126] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, 1992, pp. 144–152.

[127] R. Burbidge, M. Trotter, B. Buxton, and S. Holden, "Drug design by machine learning: support vector machines for pharmaceutical data analysis," *Computers & Chemistry*, vol. 26, no. 1, pp. 5–14, 2001.

[128] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[129] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *IEEE Computational Intelligence Magazine*, vol. 13, no. 3, pp. 55–75, 2018.

[130] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 855–868, 2008.

[131] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[132] J. Mao, W. Xu, Y. Yang, J. Wang, and A. L. Yuille, "Deep captioning with multimodal recurrent neural networks (m-rnn)," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6632

[133] A. Graves, "Supervised sequence labelling," in *Studies in Computational Intelligences*. Springer, 2012, pp. 5–13.

[134] J. J. Hopfield, "Neural networks and physical systems with emergent collective

computational abilities," *Proceedings of the National Academy of Sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.

[135] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[136] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[137] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[138] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[139] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[140] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 248–255.

[141] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European Conference on Computer Vision*. Springer, 2016, pp. 630–645.

[142] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[143] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[144] R. C. Luo and M. G. Kay, *Multisensor integration and fusion for intelligent machines and systems*. Intellect Books, 1995.

[145] A. A. Goshtasby and S. G. Nikolov, "Guest editorial: Image fusion: Advances in the state

of the art," *Information Fusion: Special Issue on Image Fusion: Advances in the State of the Art*, vol. 8, pp. 114–118, 2007.

[146] S. K. Shah and D. Shah, "Comparative study of image fusion techniques based on spatial and transform domain," vol. 3, no. 6, 2014, pp. 10 168–10 175.

[147] B. Balachander and D. Dhanasekaran, "Comparative study of image fusion techniques in spatial and transform domain," *Asian Research Publishing Network (ARPN)*, vol. 11, no. 9, pp. 5779–5783, 2016.

[148] B. Yang and S. Li, "Multifocus image fusion and restoration with sparse representation," *IEEE Transactions on Instrumentation and Measurement*, vol. 59, no. 4, pp. 884–892, 2009.

[149] N. Yu, T. Qiu, F. Bi, and A. Wang, "Image features extraction and fusion based on joint sparse representation," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 5, pp. 1074–1082, 2011.

[150] B. Yang and S. Li, "Pixel-level image fusion with simultaneous orthogonal matching pursuit," *Information Fusion*, vol. 13, no. 1, pp. 10–19, 2012.

[151] Y. Liu, X. Chen, Z. Wang, Z. J. Wang, R. K. Ward, and X. Wang, "Deep learning for pixel-level image fusion: Recent advances and future prospects," *Information Fusion*, vol. 42, pp. 158–173, 2018.

[152] Y. Liu, X. Chen, H. Peng, and Z. Wang, "Multi-focus image fusion with a deep convolutional neural network," *Information Fusion*, vol. 36, pp. 191–207, 2017.

[153] H. Song, Q. Liu, G. Wang, R. Hang, and B. Huang, "Spatiotemporal satellite image fusion using deep convolutional neural networks," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 11, no. 3, pp. 821–829, 2018.

[154] Y. Zhang, Y. Liu, P. Sun, H. Yan, X. Zhao, and L. Zhang, "Ifcnn: A general image fusion framework based on convolutional neural network," *Information Fusion*, vol. 54, pp. 99–118, 2020.

[155] H. Zhang, H. Xu, X. Tian, J. Jiang, and J. Ma, "Image fusion meets deep learning: A survey and perspective," *Information Fusion*, 2021.

[156] L. Wang, H. Zhou, S.-C. Low, and C. Leckie, "Action recognition via multi-feature fusion and gaussian process classification," in *2009 Workshop on Applications of Computer Vision (WACV)*. IEEE, 2009, pp. 1–6.

[157] C.-J. Lin, C.-H. Lin, and S.-Y. Jeng, "Using feature fusion and parameter optimization of dual-input convolutional neural network for face gender recognition," *Applied Sciences*, vol. 10, no. 9, p. 3166, 2020.

[158] L. Chen, K. Bo, F. Lee, and Q. Chen, "Advanced feature fusion algorithm based on multiple convolutional neural network for scene recognition," *Computer Modeling in Engineering & Sciences*, vol. 122, no. 2, pp. 505–523, 2020.

[159] Y. Qin, L. Mo, and B. Xie, "Feature fusion for human action recognition based on classical descriptors and 3d convolutional networks," in *2017 Eleventh International Conference on Sensing Technology (ICST)*. IEEE, 2017, pp. 1–5.

[160] L. Wang, D. Q. Huynh, and M. R. Mansour, "Loss switching fusion with similarity search for video classification," in *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019, pp. 974–978.

[161] T. Bi, D. Jarnikov, and J. Lukkien, "Video representation fusion network for multi-label movie genre classification," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 9386–9391.

[162] Z.-p. Hu, R.-x. Zhang, Y. Qiu, M.-y. Zhao, and Z. Sun, "3d convolutional networks with multi-layer-pooling selection fusion for video classification," *Multimedia Tools and Applications*, pp. 1–14, 2021.

[163] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[164] S. Sharma, R. Kiros, and R. Salakhutdinov, "Action recognition using visual attention," in *International Conference on Learning Representations (ICLR) Workshop*, 2016.

[165] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, "Residual attention network for image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3156–3164.

[166] W. Du, Y. Wang, and Y. Qiao, "Recurrent spatial-temporal attention network for action recognition in videos," *IEEE Transactions on Image Processing*, vol. 27, no. 3, pp. 1347–1360, 2017.

[167] S. Pouyanfar, T. Wang, and S.-C. Chen, "Residual attention-based fusion for video classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 478–480.

[168] Y. Dai, F. Gieseke, S. Oehmcke, Y. Wu, and K. Barnard, "Attentional feature fusion," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 3560–3569.

[169] G. Xiao, D. P. Bavirisetti, G. Liu, and X. Zhang, "Decision-level image fusion," in *Image Fusion*. Springer, 2020, pp. 149–170.

[170] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: a local svm approach," in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, vol. 3. IEEE, 2004, pp. 32–36.

[171] S. Sadanand and J. J. Corso, "Action bank: A high-level representation of activity in video," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 1234–1241.

[172] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," in *Tenth IEEE International Conference on Computer Vision (ICCV'05)*, 2005, pp. 1395–1402.

[173] N. Ikizler and P. Duygulu, "Human action recognition using distribution of oriented rectangular patches," in *Workshop on Human Motion*. Springer, 2007, pp. 271–284.

[174] D. Tran and A. Sorokin, "Human activity recognition with metric learning," in *European Conference on Computer Vision*. Springer, 2008, pp. 548–561.

[175] P. Natarajan and R. Nevatia, "Online, real-time tracking and recognition of human actions," in *2008 IEEE Workshop on Motion and Video Computing*. IEEE, 2008, pp. 1–8.

[176] S. A. Rahman, S.-Y. Cho, and M. Leung, "Recognising human actions by analysing negative spaces," *IET Computer Vision*, vol. 6, no. 3, pp. 197–213, 2012.

[177] M. D. Rodriguez, J. Ahmed, and M. Shah, "Action mach a spatio-temporal maximum average correlation height filter for action recognition," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[178] K. Soomro and A. R. Zamir, "Action recognition in realistic sports videos," in *Computer vision in sports*. Springer, 2014, pp. 181–208.

[179] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "Hmdb: a large video database for human motion recognition," in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 2556–2563.

[180] L. Wang and P. Koniusz, "Self-supervising action recognition by statistical moment and subspace descriptors," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021, pp. 4324–4333.

[181] J. Liu, J. Luo, and M. Shah, "Recognizing realistic actions from videos "in the wild"," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 1996–2003.

[182] M. A. Khan, M. Sharif, T. Akram, M. Raza, T. Saba, and A. Rehman, "Hand-crafted and deep convolutional neural network features fusion and selection strategy: an application to intelligent human action recognition," *Applied Soft Computing*, vol. 87, p. 105986, 2020.

[183] M. S. Islam, S. Sultana, U. kumar Roy, and J. Al Mahmud, "A review on video classification with methods, findings, performance, challenges, limitations and future work," *Jurnal Ilmiah Teknik Elektro Komputer dan Informatika (JITEKI)*, vol. 6, no. 2, pp. 47–57, 2020.

[184] M. B. Shaikh and D. Chai, “Rgb-d data-based action recognition: A review,” *Sensors*, vol. 21, no. 12, p. 4246, 2021.

[185] Y. Zhu, X. Li, C. Liu, M. Zolfaghari, Y. Xiong, C. Wu, Z. Zhang, J. Tighe, R. Manmatha, and M. Li, “A comprehensive study of deep video action recognition,” *CoRR*, vol. abs/2012.06567, 2020. [Online]. Available: https://arxiv.org/abs/2012.06567

[186] K. Kawaguchi, L. P. Kaelbling, and Y. Bengio, “Generalization in deep learning,” *Mathematics of Deep Learning*, 2018.

[187] M. Ravanbakhsh, H. Mousavi, M. Rastegari, V. Murino, and L. S. Davis, “Action recognition with image based CNN features,” *CoRR*, vol. abs/1512.03980, 2015. [Online]. Available: http://arxiv.org/abs/1512.03980

[188] A. Ullah, J. Ahmad, K. Muhammad, M. Sajjad, and S. W. Baik, “Action recognition in video sequences using deep bi-directional lstm with cnn features,” *IEEE Access*, vol. 6, pp. 1155–1166, 2017.

[189] A. B. Sargano, P. Angelov, and Z. Habib, “A comprehensive review on handcrafted and learning-based action representation approaches for human activity recognition,” *Applied Sciences*, vol. 7, no. 1, p. 110, 2017.

[190] 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*, vol. 35, no. 1, pp. 221–231, 2012.

[191] M. E. Kalfaoglu, S. Kalkan, and A. A. Alatan, “Late temporal modeling in 3d CNN architectures with BERT for action recognition,” in *Computer Vision - ECCV 2020 Workshops - Glasgow, UK, August 23-28,2020, Proceedings, Part V*, ser. Lecture Notes in Computer Science, A. Bartoli and A. Fusiello, Eds., vol. 12539. Springer, 2020, pp. 731–747.

[192] G. M. E. Elahi and Y.-H. Yang, “Online learnable keyframe extraction in videos and its application with semantic word vector in action recognition,” *Pattern Recognition*, vol. 122, p. 108273, 2022.

[193] S. Cho and H. Foroosh, "Spatio-temporal fusion networks for action recognition," in *Asian Conference on Computer Vision*. Springer, 2018, pp. 347–364.

[194] A. Abdelbaky and S. Aly, "Two-stream spatiotemporal feature fusion for human action recognition," *The Visual Computer*, pp. 1–15, 2020.

[195] V. Sze, Y.-H. Chen, J. Emer, A. Suleiman, and Z. Zhang, "Hardware for machine learning: Challenges and opportunities," in *2017 IEEE Custom Integrated Circuits Conference (CICC)*. IEEE, 2017, pp. 1–8.

[196] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-nn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 91–99.

[197] X. Yin and X. Liu, "Multi-task convolutional neural network for pose-invariant face recognition," *IEEE Transactions on Image Processing*, vol. 27, no. 2, pp. 964–975, 2018.

[198] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587.

[199] N. Ballas, L. Yao, C. Pal, and A. C. Courville, "Delving deeper into convolutional networks for learning video representations," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016. [Online]. Available: http://arxiv.org/abs/1511.06432

[200] N. Takahashi, M. Gygli, and L. Van Gool, "Aenet: Learning deep audio features for video analysis," *IEEE Transactions on Multimedia*, vol. 20, no. 3, pp. 513–524, 2018.

[201] A. Graves, "Generating sequences with recurrent neural networks," *CoRR*, vol. abs/1308.0850, 2013. [Online]. Available: http://arxiv.org/abs/1308.0850

[202] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *Advances in Neural Information Processing Systems*, 2015, pp. 802–810.

[203] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan, "Youtube-8m: A large-scale video classification benchmark," *CoRR*, vol. abs/1609.08675, 2016. [Online]. Available: http://arxiv.org/abs/1609.08675

[204] F. Mao, X. Wu, H. Xue, and R. Zhang, "Hierarchical video frame sequence representation with deep convolutional graph network," in *European Conference on Computer Vision*, 2018, pp. 0–0.

[205] B. T. Truong and S. Venkatesh, "Video abstraction: A systematic review and classification," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 3, no. 1, p. 3, 2007.

[206] A. Nagasaka and Y. Tanaka, "Automatic video indexing and full-video search for object appearances," in *Proceedings of the IFIP TC2/WG 2.6 Second Working Conference on Visual Database Systems II*, 1992, pp. 113–127.

[207] G. Pal, D. Rudrapaul, S. Acharjee, R. Ray, S. Chakraborty, and N. Dey, "Video shot boundary detection: A review," in *Advances in Intelligent Systems and Computing*. Springer International Publishing, 2015, pp. 119–127.

[208] Y. Zhuang, Y. Rui, T. S. Huang, and S. Mehrotra, "Adaptive key frame extraction using unsupervised clustering," in *Proceedings of the IEEE International Conference on Image Processing*, vol. 1, 1998, pp. 866–870.

[209] P. Gresle and T. Huang, "Gisting of video documents: A key frames selection algorithm using relative activity measure," in *The 2nd International Conference on Visual Information Systems*, 1997, pp. 279–286.

[210] H. J. Zhang, J. Wu, D. Zhong, and S. W. Smoliar, "An integrated system for content-based video retrieval and browsing," *Pattern Recognition*, vol. 30, no. 4, pp. 643–658, 1997.

[211] P. Geetha, T. Pandeeswari S., and S. Mohanan, "Visual attention based keyframes extraction and video summarization," in *Computer Science Conference Proceedings in Computer Science & Information Technology (CS&IT)*. Citeseer, 2012, pp. 179–190.

[212] W. Barhoumi and E. Zagrouba, "On-the-fly extraction of key frames for efficient video

summarization," in *Proceedings of the AASRI Conference on Intelligent Systems and Control*, vol. 4. Elsevier, 2013, pp. 78–84.

[213] K. Thakre, A. Rajurkar, and R. Manthalkar, "Video partitioning and secured keyframe extraction of mpeg video," in *Physics Procedia*, vol. 78. Elsevier, 2016, pp. 790–798.

[214] S. N. H. S. Abdullah and K. W. Ng, "Action key frames extraction using l1-norm and accumulative optical flow for compact video shot summarisation," in *Advances in Visual Informatics: 5th International Visual Informatics Conference, IVIC 2017, Bangi, Malaysia, November 28–30, 2017, Proceedings*, vol. 10645. Springer, 2017, p. 364.

[215] G. Bao, D. Li, and Y. Mei, "Key frames extraction based on optical-flow and mutual information entropy," *Journal of Physics: Conference Series*, vol. 1646, no. 1, p. 012112, 2020.

[216] L. Tan, Y. Song, Z. Ma, X. Lv, and X. Dong, "Deep learning video action recognition method based on key frame algorithm," in *Artificial Intelligence and Security*, X. Sun, J. Wang, and E. Bertino, Eds. Cham: Springer International Publishing, 2020, pp. 62–73.

[217] D. Xu and Y. Tian, "A comprehensive survey of clustering algorithms," *Annals of Data Science*, vol. 2, no. 2, pp. 165–193, 2015.

[218] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli *et al.*, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[219] T. Ogawa, Y. Sasaka, K. Maeda, and M. Haseyama, "Favorite video classification based on multimodal bidirectional lstm," *IEEE Access*, vol. 6, pp. 61 401–61 409, 2018.

[220] X. Shi, Z. Gao, L. Lausen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo, "Deep learning for precipitation nowcasting: A benchmark and a new model," in *Advances in Neural Information Processing Systems*, 2017, pp. 5617–5627.

[221] R. J. Freund, D. Mohr, and W. J. Wilson, *Statistical Methods*. Academic Press, 2010.

[222] G. V. Glass, "Testing homogeneity of variances," *American Educational Research Journal*, vol. 3, no. 3, pp. 187–190, 1966.

[223] D. Tran, J. Ray, Z. Shou, S. Chang, and M. Paluri, "Convnet architecture search for spatiotemporal feature learning," *CoRR*, vol. abs/1708.05038, 2017. [Online]. Available: http://arxiv.org/abs/1708.05038

[224] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville, "Describing videos by exploiting temporal structure," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4507–4515.

[225] L. Sun, K. Jia, D.-Y. Yeung, and B. E. Shi, "Human action recognition using factorized spatio-temporal convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4597–4605.

[226] H. Ou and J. Sun, "Spatiotemporal information deep fusion network with frame attention mechanism for video action recognition," *Journal of Electronic Imaging*, vol. 28, no. 2, p. 023009, 2019.

[227] L. Liu and L. Shao, "Learning discriminative representations from rgb-d video data," in *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, 2013, pp. 1493–1500.

[228] M. Li, H. Leung, and H. P. Shum, "Human action recognition via skeletal and depth based feature fusion," in *Proceedings of the 9th International Conference on Motion in Games*, 2016, pp. 123–132.

[229] J. Zhang, W. Li, P. O. Ogunbona, P. Wang, and C. Tang, "Rgb-d-based action recognition datasets: A survey," *Pattern Recognition*, vol. 60, pp. 86–105, 2016.

[230] E. H. Shortliffe and B. G. Buchanan, *Rule-based expert systems: the MYCIN experiments of the Stanford Heuristic Programming Project*. Addison-Wesley Publishing Company, 1985.

[231] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception

architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.

Appendix A

List of Publications

R. Savran Kızıltepe, J. Q. Gan, and J. J. Escobar, “Combining very deep convolutional neural networks and recurrent neural networks for video classification”, in *Advances in Computational Intelligence*, Cham:Springer International Publishing, 2019, pp. 811–822.

R. Savran Kızıltepe and J. Q. Gan, “Simple effective methods for decision-level fusion in two-stream convolutional neural networks for video classification”, in *Intelligent Data Engineering and Automated Learning - IDEAL2020*, Cham: Springer International Publishing, 2020, pp. 77–87.

R. Savran Kızıltepe, J. Q. Gan, and J. J. Escobar, “A novel keyframe extraction method for video classification using deep neural networks”, *Neural Computing and Applications*, 2021.

Appendix B

Post Hoc Comparisons

This section presents the Tukey's HSD test results analysed in Chapter 6 (* indicates significant differences at the 0.05 level in the Mean Difference column of the following tables).

Table B.1: Post hoc comparisons using Tukey's HSD - *RGB* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
HOG	-.0486111109167*	0.0052	0.000	-0.0667	-0.0305
OF	-.0837191357500*	0.0052	0.000	-0.1018	-0.0657
D-RGB-HOG	-.0516975308333*	0.0052	0.000	-0.0698	-0.0336
D-RGB-OF	-.0875771604167*	0.0052	0.000	-0.1056	-0.0695
D-HOG-OF	-.0983796294167*	0.0052	0.000	-0.1164	-0.0803
D-RGB-HOG-OF	-.0871913578333*	0.0052	0.000	-0.1053	-0.0691
D1	-.0671296295000*	0.0052	0.000	-0.0852	-0.0491
D2	-.1060956788333*	0.0052	0.000	-0.1242	-0.0880
D3	-.0783179011667*	0.0052	0.000	-0.0964	-0.0603
F-RGB-HOG	0.0127314817500	0.0052	0.527	-0.0053	0.0308
F-RGB-OF	-.0817901233333*	0.0052	0.000	-0.0999	-0.0637
F-HOG-OF	-.0825617283333*	0.0052	0.000	-0.1006	-0.0645
F-RGB-HOG-OF	-.1103395060833*	0.0052	0.000	-0.1284	-0.0923
FD1	-.0767746911667*	0.0052	0.000	-0.0948	-0.0587
FD2	-.0841049382500*	0.0052	0.000	-0.1022	-0.0660
FD3	-.0918209875000*	0.0052	0.000	-0.1099	-0.0738

Table B.2: Post hoc comparisons using Tukey's HSD - *HOG* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0486111109167*	0.0052	0.000	0.0305	0.0667
OF	-.0351080248333*	0.0052	0.000	-0.0532	-0.0170
D-RGB-HOG	-0.0030864199167	0.0052	1.000	-0.0212	0.0150
D-RGB-OF	-.0389660495000*	0.0052	0.000	-0.0570	-0.0209
D-HOG-OF	-.0497685185000*	0.0052	0.000	-0.0678	-0.0317
D-RGB-HOG-OF	-.0385802469167*	0.0052	0.000	-0.0566	-0.0205
D1	-.0185185185833*	0.0052	0.038	-0.0366	-0.0005
D2	-.0574845679167*	0.0052	0.000	-0.0756	-0.0394
D3	-.0297067902500*	0.0052	0.000	-0.0478	-0.0116
F-RGB-HOG	.0613425926667*	0.0052	0.000	0.0433	0.0794
F-RGB-OF	-.0331790124167*	0.0052	0.000	-0.0512	-0.0151
F-HOG-OF	-.0339506174167*	0.0052	0.000	-0.0520	-0.0159
F-RGB-HOG-OF	-.0617283951667*	0.0052	0.000	-0.0798	-0.0437
FD1	-.0281635802500*	0.0052	0.000	-0.0462	-0.0101
FD2	-.0354938273333*	0.0052	0.000	-0.0536	-0.0174
FD3	-.0432098765833*	0.0052	0.000	-0.0613	-0.0251

Table B.3: Post hoc comparisons using Tukey's HSD - *OF* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0837191357500*	0.0052	0.000	0.0657	0.1018
HOG	.0351080248333*	0.0052	0.000	0.0170	0.0532
D-RGB-HOG	.0320216049167*	0.0052	0.000	0.0140	0.0501
D-RGB-OF	-0.0038580246667	0.0052	1.000	-0.0219	0.0142
D-HOG-OF	-0.0146604936667	0.0052	0.274	-0.0327	0.0034
D-RGB-HOG-OF	-0.0034722220833	0.0052	1.000	-0.0215	0.0146
D1	0.0165895062500	0.0052	0.113	-0.0015	0.0347
D2	-.0223765430833*	0.0052	0.003	-0.0404	-0.0043
D3	0.0054012345833	0.0052	1.000	-0.0127	0.0235
F-RGB-HOG	.0964506175000*	0.0052	0.000	0.0784	0.1145
F-RGB-OF	0.0019290124167	0.0052	1.000	-0.0161	0.0200
F-HOG-OF	0.0011574074167	0.0052	1.000	-0.0169	0.0192
F-RGB-HOG-OF	-.0266203703333*	0.0052	0.000	-0.0447	-0.0086
FD1	0.0069444445833	0.0052	0.995	-0.0111	0.0250
FD2	-0.0003858025000	0.0052	1.000	-0.0185	0.0177
FD3	-0.0081018517500	0.0052	0.978	-0.0262	0.0100

Table B.4: Post hoc comparisons using Tukey's HSD - *D-RGB-HOG* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0516975308333*	0.0052	0.000	0.0336	0.0698
HOG	0.0030864199167	0.0052	1.000	-0.0150	0.0212
OF	-.0320216049167*	0.0052	0.000	-0.0501	-0.0140
D-RGB-OF	-.0358796295833*	0.0052	0.000	-0.0539	-0.0178
D-HOG-OF	-.0466820985833*	0.0052	0.000	-0.0647	-0.0286
D-RGB-HOG-OF	-.0354938270000*	0.0052	0.000	-0.0536	-0.0174
D1	-0.0154320986667	0.0052	0.197	-0.0335	0.0026
D2	-.0543981480000*	0.0052	0.000	-0.0725	-0.0363
D3	-.0266203703333*	0.0052	0.000	-0.0447	-0.0086
F-RGB-HOG	.0644290125833*	0.0052	0.000	0.0464	0.0825
F-RGB-OF	-.0300925925000*	0.0052	0.000	-0.0482	-0.0120
F-HOG-OF	-.0308641975000*	0.0052	0.000	-0.0489	-0.0128
F-RGB-HOG-OF	-.0586419752500*	0.0052	0.000	-0.0767	-0.0406
FD1	-.0250771603333*	0.0052	0.000	-0.0431	-0.0070
FD2	-.0324074074167*	0.0052	0.000	-0.0505	-0.0143
FD3	-.0401234566667*	0.0052	0.000	-0.0582	-0.0221

Table B.5: Post hoc comparisons using Tukey's HSD - *D-RGB-OF* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0875771604167*	0.0052	0.000	0.0695	0.1056
HOG	.0389660495000*	0.0052	0.000	0.0209	0.0570
OF	0.0038580246667	0.0052	1.000	-0.0142	0.0219
D-RGB-HOG	.0358796295833*	0.0052	0.000	0.0178	0.0539
D-HOG-OF	-0.0108024690000	0.0052	0.791	-0.0289	0.0073
D-RGB-HOG-OF	0.0003858025833	0.0052	1.000	-0.0177	0.0185
D1	.0204475309167*	0.0052	0.011	0.0024	0.0385
D2	-.0185185184167*	0.0052	0.038	-0.0366	-0.0005
D3	0.0092592592500	0.0052	0.929	-0.0088	0.0273
F-RGB-HOG	.1003086421667*	0.0052	0.000	0.0822	0.1184
F-RGB-OF	0.0057870370833	0.0052	0.999	-0.0123	0.0239
F-HOG-OF	0.0050154320833	0.0052	1.000	-0.0131	0.0231
F-RGB-HOG-OF	-.0227623456667*	0.0052	0.002	-0.0408	-0.0047
FD1	0.0108024692500	0.0052	0.791	-0.0073	0.0289
FD2	0.0034722221667	0.0052	1.000	-0.0146	0.0215
FD3	-0.0042438270833	0.0052	1.000	-0.0223	0.0138

Table B.6: Post hoc comparisons using Tukey's HSD - *D-HOG-OF* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0983796294167*	0.0052	0.000	0.0803	0.1164
HOG	.0497685185000*	0.0052	0.000	0.0317	0.0678
OF	0.0146604936667	0.0052	0.274	-0.0034	0.0327
D-RGB-HOG	.0466820985833*	0.0052	0.000	0.0286	0.0647
D-RGB-OF	0.0108024690000	0.0052	0.791	-0.0073	0.0289
D-RGB-HOG-OF	0.0111882715833	0.0052	0.744	-0.0069	0.0293
D1	.0312499999167*	0.0052	0.000	0.0132	0.0493
D2	-0.0077160494167	0.0052	0.986	-0.0258	0.0103
D3	.0200617282500*	0.0052	0.014	0.0020	0.0381
F-RGB-HOG	.1111111111667*	0.0052	0.000	0.0930	0.1292
F-RGB-OF	0.0165895060833	0.0052	0.113	-0.0015	0.0347
F-HOG-OF	0.0158179010833	0.0052	0.165	-0.0022	0.0339
F-RGB-HOG-OF	-0.0119598766667	0.0052	0.638	-0.0300	0.0061
FD1	.0216049382500*	0.0052	0.005	0.0035	0.0397
FD2	0.0142746911667	0.0052	0.318	-0.0038	0.0323
FD3	0.0065586419167	0.0052	0.998	-0.0115	0.0246

Table B.7: Post hoc comparisons using Tukey's HSD - *D-RGB-HOG-OF* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0871913578333*	0.0052	0.000	0.0691	0.1053
HOG	.0385802469167*	0.0052	0.000	0.0205	0.0566
OF	0.0034722220833	0.0052	1.000	-0.0146	0.0215
D-RGB-HOG	.0354938270000*	0.0052	0.000	0.0174	0.0536
D-RGB-OF	-0.0003858025833	0.0052	1.000	-0.0185	0.0177
D-HOG-OF	-0.0111882715833	0.0052	0.744	-0.0293	0.0069
D1	.0200617283333*	0.0052	0.014	0.0020	0.0381
D2	-.0189043210000*	0.0052	0.030	-0.0370	-0.0008
D3	0.0088734566667	0.0052	0.950	-0.0092	0.0269
F-RGB-HOG	.0999228395833*	0.0052	0.000	0.0819	0.1180
F-RGB-OF	0.0054012345000	0.0052	1.000	-0.0127	0.0235
F-HOG-OF	0.0046296295000	0.0052	1.000	-0.0134	0.0227
F-RGB-HOG-OF	-.0231481482500*	0.0052	0.001	-0.0412	-0.0051
FD1	0.0104166666667	0.0052	0.834	-0.0076	0.0285
FD2	0.0030864195833	0.0052	1.000	-0.0150	0.0212
FD3	-0.0046296296667	0.0052	1.000	-0.0227	0.0134

Table B.8: Post hoc comparisons using Tukey's HSD - *D1* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0671296295000*	0.0052	0.000	0.0491	0.0852
HOG	.0185185185833*	0.0052	0.038	0.0005	0.0366
OF	-0.0165895062500	0.0052	0.113	-0.0347	0.0015
D-RGB-HOG	0.0154320986667	0.0052	0.197	-0.0026	0.0335
D-RGB-OF	-.0204475309167*	0.0052	0.011	-0.0385	-0.0024
D-HOG-OF	-.0312499999167*	0.0052	0.000	-0.0493	-0.0132
D-RGB-HOG-OF	-.0200617283333*	0.0052	0.014	-0.0381	-0.0020
D2	-.0389660493333*	0.0052	0.000	-0.0570	-0.0209
D3	-0.0111882716667	0.0052	0.744	-0.0293	0.0069
F-RGB-HOG	.0798611112500*	0.0052	0.000	0.0618	0.0979
F-RGB-OF	-0.0146604938333	0.0052	0.274	-0.0327	0.0034
F-HOG-OF	-0.0154320988333	0.0052	0.197	-0.0335	0.0026
F-RGB-HOG-OF	-.0432098765833*	0.0052	0.000	-0.0613	-0.0251
FD1	-0.0096450616667	0.0052	0.903	-0.0277	0.0084
FD2	-0.0169753087500	0.0052	0.092	-0.0350	0.0011
FD3	-.0246913580000*	0.0052	0.000	-0.0428	-0.0066

Table B.9: Post hoc comparisons using Tukey's HSD - *D2* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.1060956788333*	0.0052	0.000	0.0880	0.1242
HOG	.0574845679167*	0.0052	0.000	0.0394	0.0756
OF	.0223765430833*	0.0052	0.003	0.0043	0.0404
D-RGB-HOG	.0543981480000*	0.0052	0.000	0.0363	0.0725
D-RGB-OF	.0185185184167*	0.0052	0.038	0.0005	0.0366
D-HOG-OF	0.0077160494167	0.0052	0.986	-0.0103	0.0258
D-RGB-HOG-OF	.0189043210000*	0.0052	0.030	0.0008	0.0370
D1	.0389660493333*	0.0052	0.000	0.0209	0.0570
D3	.0277777776667*	0.0052	0.000	0.0097	0.0458
F-RGB-HOG	.1188271605833*	0.0052	0.000	0.1008	0.1369
F-RGB-OF	.0243055555000*	0.0052	0.001	0.0062	0.0424
F-HOG-OF	.0235339505000*	0.0052	0.001	0.0055	0.0416
F-RGB-HOG-OF	-0.0042438272500	0.0052	1.000	-0.0223	0.0138
FD1	.0293209876667*	0.0052	0.000	0.0113	0.0474
FD2	.0219907405833*	0.0052	0.004	0.0039	0.0401
FD3	0.0142746913333	0.0052	0.318	-0.0038	0.0323

Table B.10: Post hoc comparisons using Tukey's HSD - *D3* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0783179011667*	0.0052	0.000	0.0603	0.0964
HOG	.0297067902500*	0.0052	0.000	0.0116	0.0478
OF	-0.0054012345833	0.0052	1.000	-0.0235	0.0127
D-RGB-HOG	.0266203703333*	0.0052	0.000	0.0086	0.0447
D-RGB-OF	-0.0092592592500	0.0052	0.929	-0.0273	0.0088
D-HOG-OF	-.0200617282500*	0.0052	0.014	-0.0381	-0.0020
D-RGB-HOG-OF	-0.0088734566667	0.0052	0.950	-0.0269	0.0092
D1	0.0111882716667	0.0052	0.744	-0.0069	0.0293
D2	-.0277777776667*	0.0052	0.000	-0.0458	-0.0097
F-RGB-HOG	.0910493829167*	0.0052	0.000	0.0730	0.1091
F-RGB-OF	-0.0034722221667	0.0052	1.000	-0.0215	0.0146
F-HOG-OF	-0.0042438271667	0.0052	1.000	-0.0223	0.0138
F-RGB-HOG-OF	-.0320216049167*	0.0052	0.000	-0.0501	-0.0140
FD1	0.0015432100000	0.0052	1.000	-0.0165	0.0196
FD2	-0.0057870370833	0.0052	0.999	-0.0239	0.0123
FD3	-0.0135030863333	0.0052	0.417	-0.0316	0.0046

Table B.11: Post hoc comparisons using Tukey's HSD - *F-RGB-HOG* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	-0.0127314817500	0.0052	0.527	-0.0308	0.0053
HOG	-.0613425926667*	0.0052	0.000	-0.0794	-0.0433
OF	-.0964506175000*	0.0052	0.000	-0.1145	-0.0784
D-RGB-HOG	-.0644290125833*	0.0052	0.000	-0.0825	-0.0464
D-RGB-OF	-.1003086421667*	0.0052	0.000	-0.1184	-0.0822
D-HOG-OF	-.1111111111667*	0.0052	0.000	-0.1292	-0.0930
D-RGB-HOG-OF	-.0999228395833*	0.0052	0.000	-0.1180	-0.0819
D1	-.0798611112500*	0.0052	0.000	-0.0979	-0.0618
D2	-.1188271605833*	0.0052	0.000	-0.1369	-0.1008
D3	-.0910493829167*	0.0052	0.000	-0.1091	-0.0730
F-RGB-OF	-.0945216050833*	0.0052	0.000	-0.1126	-0.0765
F-HOG-OF	-.0952932100833*	0.0052	0.000	-0.1134	-0.0772
F-RGB-HOG-OF	-.1230709878333*	0.0052	0.000	-0.1411	-0.1050
FD1	-.0895061729167*	0.0052	0.000	-0.1076	-0.0714
FD2	-.0968364200000*	0.0052	0.000	-0.1149	-0.0788
FD3	-.1045524692500*	0.0052	0.000	-0.1226	-0.0865

Table B.12: Post hoc comparisons using Tukey's HSD - *F-RGB-OF* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0817901233333*	0.0052	0.000	0.0637	0.0999
HOG	.0331790124167*	0.0052	0.000	0.0151	0.0512
OF	-0.0019290124167	0.0052	1.000	-0.0200	0.0161
D-RGB-HOG	.0300925925000*	0.0052	0.000	0.0120	0.0482
D-RGB-OF	-0.0057870370833	0.0052	0.999	-0.0239	0.0123
D-HOG-OF	-0.0165895060833	0.0052	0.113	-0.0347	0.0015
D-RGB-HOG-OF	-0.0054012345000	0.0052	1.000	-0.0235	0.0127
D1	0.0146604938333	0.0052	0.274	-0.0034	0.0327
D2	-.0243055555000*	0.0052	0.001	-0.0424	-0.0062
D3	0.0034722221667	0.0052	1.000	-0.0146	0.0215
F-RGB-HOG	.0945216050833*	0.0052	0.000	0.0765	0.1126
F-HOG-OF	-0.0007716050000	0.0052	1.000	-0.0188	0.0173
F-RGB-HOG-OF	-.0285493827500*	0.0052	0.000	-0.0466	-0.0105
FD1	0.0050154321667	0.0052	1.000	-0.0131	0.0231
FD2	-0.0023148149167	0.0052	1.000	-0.0204	0.0158
FD3	-0.0100308641667	0.0052	0.871	-0.0281	0.0080

Table B.13: Post hoc comparisons using Tukey's HSD - *F-HOG-OF* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0825617283333*	0.0052	0.000	0.0645	0.1006
HOG	.0339506174167*	0.0052	0.000	0.0159	0.0520
OF	-0.0011574074167	0.0052	1.000	-0.0192	0.0169
D-RGB-HOG	.0308641975000*	0.0052	0.000	0.0128	0.0489
D-RGB-OF	-0.0050154320833	0.0052	1.000	-0.0231	0.0131
D-HOG-OF	-0.0158179010833	0.0052	0.165	-0.0339	0.0022
D-RGB-HOG-OF	-0.0046296295000	0.0052	1.000	-0.0227	0.0134
D1	0.0154320988333	0.0052	0.197	-0.0026	0.0335
D2	-.0235339505000*	0.0052	0.001	-0.0416	-0.0055
D3	0.0042438271667	0.0052	1.000	-0.0138	0.0223
F-RGB-HOG	.0952932100833*	0.0052	0.000	0.0772	0.1134
F-RGB-OF	0.0007716050000	0.0052	1.000	-0.0173	0.0188
F-RGB-HOG-OF	-.0277777777500*	0.0052	0.000	-0.0458	-0.0097
FD1	0.0057870371667	0.0052	0.999	-0.0123	0.0239
FD2	-0.0015432099167	0.0052	1.000	-0.0196	0.0165
FD3	-0.0092592591667	0.0052	0.929	-0.0273	0.0088

Table B.14: Post hoc comparisons using Tukey's HSD - *F-RGB-HOG-OF* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.1103395060833*	0.0052	0.000	0.0923	0.1284
HOG	.0617283951667*	0.0052	0.000	0.0437	0.0798
OF	.0266203703333*	0.0052	0.000	0.0086	0.0447
D-RGB-HOG	.0586419752500*	0.0052	0.000	0.0406	0.0767
D-RGB-OF	.0227623456667*	0.0052	0.002	0.0047	0.0408
D-HOG-OF	0.0119598766667	0.0052	0.638	-0.0061	0.0300
D-RGB-HOG-OF	.0231481482500*	0.0052	0.001	0.0051	0.0412
D1	.0432098765833*	0.0052	0.000	0.0251	0.0613
D2	0.0042438272500	0.0052	1.000	-0.0138	0.0223
D3	.0320216049167*	0.0052	0.000	0.0140	0.0501
F-RGB-HOG	.1230709878333*	0.0052	0.000	0.1050	0.1411
F-RGB-OF	.0285493827500*	0.0052	0.000	0.0105	0.0466
F-HOG-OF	.0277777777500*	0.0052	0.000	0.0097	0.0458
FD1	.0335648149167*	0.0052	0.000	0.0155	0.0516
FD2	.0262345678333*	0.0052	0.000	0.0082	0.0443
FD3	.0185185185833*	0.0052	0.038	0.0005	0.0366

Table B.15: Post hoc comparisons using Tukey's HSD - *FD1* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0767746911667*	0.0052	0.000	0.0587	0.0948
HOG	.0281635802500*	0.0052	0.000	0.0101	0.0462
OF	-0.0069444445833	0.0052	0.995	-0.0250	0.0111
D-RGB-HOG	.0250771603333*	0.0052	0.000	0.0070	0.0431
D-RGB-OF	-0.0108024692500	0.0052	0.791	-0.0289	0.0073
D-HOG-OF	-.0216049382500*	0.0052	0.005	-0.0397	-0.0035
D-RGB-HOG-OF	-0.0104166666667	0.0052	0.834	-0.0285	0.0076
D1	0.0096450616667	0.0052	0.903	-0.0084	0.0277
D2	-.0293209876667*	0.0052	0.000	-0.0474	-0.0113
D3	-0.0015432100000	0.0052	1.000	-0.0196	0.0165
F-RGB-HOG	.0895061729167*	0.0052	0.000	0.0714	0.1076
F-RGB-OF	-0.0050154321667	0.0052	1.000	-0.0231	0.0131
F-HOG-OF	-0.0057870371667	0.0052	0.999	-0.0239	0.0123
F-RGB-HOG-OF	-.0335648149167*	0.0052	0.000	-0.0516	-0.0155
FD2	-0.0073302470833	0.0052	0.992	-0.0254	0.0107
FD3	-0.0150462963333	0.0052	0.233	-0.0331	0.0030

Table B.16: Post hoc comparisons using Tukey's HSD - *FD2* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0841049382500*	0.0052	0.000	0.0660	0.1022
HOG	.0354938273333*	0.0052	0.000	0.0174	0.0536
OF	0.0003858025000	0.0052	1.000	-0.0177	0.0185
D-RGB-HOG	.0324074074167*	0.0052	0.000	0.0143	0.0505
D-RGB-OF	-0.0034722221667	0.0052	1.000	-0.0215	0.0146
D-HOG-OF	-0.0142746911667	0.0052	0.318	-0.0323	0.0038
D-RGB-HOG-OF	-0.0030864195833	0.0052	1.000	-0.0212	0.0150
D1	0.0169753087500	0.0052	0.092	-0.0011	0.0350
D2	-.0219907405833*	0.0052	0.004	-0.0401	-0.0039
D3	0.0057870370833	0.0052	0.999	-0.0123	0.0239
F-RGB-HOG	.0968364200000*	0.0052	0.000	0.0788	0.1149
F-RGB-OF	0.0023148149167	0.0052	1.000	-0.0158	0.0204
F-HOG-OF	0.0015432099167	0.0052	1.000	-0.0165	0.0196
F-RGB-HOG-OF	-.0262345678333*	0.0052	0.000	-0.0443	-0.0082
FD1	0.0073302470833	0.0052	0.992	-0.0107	0.0254
FD3	-0.0077160492500	0.0052	0.986	-0.0258	0.0103

Table B.17: Post hoc comparisons using Tukey's HSD - *FD3* on the KTH Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0918209875000*	0.0052	0.000	0.0738	0.1099
HOG	.0432098765833*	0.0052	0.000	0.0251	0.0613
OF	0.0081018517500	0.0052	0.978	-0.0100	0.0262
D-RGB-HOG	.0401234566667*	0.0052	0.000	0.0221	0.0582
D-RGB-OF	0.0042438270833	0.0052	1.000	-0.0138	0.0223
D-HOG-OF	-0.0065586419167	0.0052	0.998	-0.0246	0.0115
D-RGB-HOG-OF	0.0046296296667	0.0052	1.000	-0.0134	0.0227
D1	.0246913580000*	0.0052	0.000	0.0066	0.0428
D2	-0.0142746913333	0.0052	0.318	-0.0323	0.0038
D3	0.0135030863333	0.0052	0.417	-0.0046	0.0316
F-RGB-HOG	.1045524692500*	0.0052	0.000	0.0865	0.1226
F-RGB-OF	0.0100308641667	0.0052	0.871	-0.0080	0.0281
F-HOG-OF	0.0092592591667	0.0052	0.929	-0.0088	0.0273
F-RGB-HOG-OF	-.0185185185833*	0.0052	0.038	-0.0366	-0.0005
FD1	0.0150462963333	0.0052	0.233	-0.0030	0.0331
FD2	0.0077160492500	0.0052	0.986	-0.0103	0.0258

Table B.18: Post hoc comparisons using Tukey's HSD - *RGB* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
HOG	-.0077559010000*	0.0020	0.019	-0.0149	-0.0006
OF	-.0363207498000*	0.0020	0.000	-0.0435	-0.0292
D-RGB-HOG	-.0186848517333*	0.0020	0.000	-0.0258	-0.0115
D-RGB-OF	-.0484082766000*	0.0020	0.000	-0.0556	-0.0413
D-HOG-OF	-.0629146365333*	0.0020	0.000	-0.0701	-0.0558
D-RGB-HOG-OF	-.0586025134667*	0.0020	0.000	-0.0658	-0.0515
D1	-.0336291850667*	0.0020	0.000	-0.0408	-0.0265
D2	-.0628224690667*	0.0020	0.000	-0.0700	-0.0557
D3	-.0420568977333*	0.0020	0.000	-0.0492	-0.0349
F-RGB-HOG	-.0331244980667*	0.0020	0.000	-0.0403	-0.0260
F-RGB-OF	-.0486116803333*	0.0020	0.000	-0.0558	-0.0415
F-HOG-OF	-.0576483417333*	0.0020	0.000	-0.0648	-0.0505
F-RGB-HOG-OF	-.0794914722000*	0.0020	0.000	-0.0866	-0.0723
FD1	-.0337055438000*	0.0020	0.000	-0.0409	-0.0266
FD2	-.0513398346000*	0.0020	0.000	-0.0585	-0.0442
FD3	-.0604168239333*	0.0020	0.000	-0.0676	-0.0533

Table B.19: Post hoc comparisons using Tukey's HSD - *HOG* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0077559010000*	0.0020	0.019	0.0006	0.0149
OF	-.0285648488000*	0.0020	0.000	-0.0357	-0.0214
D-RGB-HOG	-.0109289507333*	0.0020	0.000	-0.0181	-0.0038
D-RGB-OF	-.0406523756000*	0.0020	0.000	-0.0478	-0.0335
D-HOG-OF	-.0551587355333*	0.0020	0.000	-0.0623	-0.0480
D-RGB-HOG-OF	-.0508466124667*	0.0020	0.000	-0.0580	-0.0437
D1	-.0258732840667*	0.0020	0.000	-0.0330	-0.0187
D2	-.0550665680667*	0.0020	0.000	-0.0622	-0.0479
D3	-.0343009967333*	0.0020	0.000	-0.0415	-0.0271
F-RGB-HOG	-.0253685970667*	0.0020	0.000	-0.0325	-0.0182
F-RGB-OF	-.0408557793333*	0.0020	0.000	-0.0480	-0.0337
F-HOG-OF	-.0498924407333*	0.0020	0.000	-0.0570	-0.0427
F-RGB-HOG-OF	-.0717355712000*	0.0020	0.000	-0.0789	-0.0646
FD1	-.0259496428000*	0.0020	0.000	-0.0331	-0.0188
FD2	-.0435839336000*	0.0020	0.000	-0.0507	-0.0364
FD3	-.0526609229333*	0.0020	0.000	-0.0598	-0.0455

Table B.20: Post hoc comparisons using Tukey's HSD - *OF* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0363207498000*	0.0020	0.000	0.0292	0.0435
HOG	.0285648488000*	0.0020	0.000	0.0214	0.0357
D-RGB-HOG	.0176358980667*	0.0020	0.000	0.0105	0.0248
D-RGB-OF	-.0120875268000*	0.0020	0.000	-0.0192	-0.0049
D-HOG-OF	-.0265938867333*	0.0020	0.000	-0.0337	-0.0194
D-RGB-HOG-OF	-.0222817636667*	0.0020	0.000	-0.0294	-0.0151
D1	0.0026915647333	0.0020	0.997	-0.0045	0.0098
D2	-.0265017192667*	0.0020	0.000	-0.0337	-0.0194
D3	-0.0057361479333	0.0020	0.296	-0.0129	0.0014
F-RGB-HOG	0.0031962517333	0.0020	0.980	-0.0040	0.0103
F-RGB-OF	-.0122909305333*	0.0020	0.000	-0.0194	-0.0051
F-HOG-OF	-.0213275919333*	0.0020	0.000	-0.0285	-0.0142
F-RGB-HOG-OF	-.0431707224000*	0.0020	0.000	-0.0503	-0.0360
FD1	0.0026152060000	0.0020	0.998	-0.0045	0.0098
FD2	-.0150190848000*	0.0020	0.000	-0.0222	-0.0079
FD3	-.0240960741333*	0.0020	0.000	-0.0312	-0.0169

Table B.21: Post hoc comparisons using Tukey's HSD - *D-RGB-HOG* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0186848517333*	0.0020	0.000	0.0115	0.0258
HOG	.0109289507333*	0.0020	0.000	0.0038	0.0181
OF	-.0176358980667*	0.0020	0.000	-0.0248	-0.0105
D-RGB-OF	-.0297234248667*	0.0020	0.000	-0.0369	-0.0226
D-HOG-OF	-.0442297848000*	0.0020	0.000	-0.0514	-0.0371
D-RGB-HOG-OF	-.0399176617333*	0.0020	0.000	-0.0471	-0.0328
D1	-.0149443333333*	0.0020	0.000	-0.0221	-0.0078
D2	-.0441376173333*	0.0020	0.000	-0.0513	-0.0370
D3	-.0233720460000*	0.0020	0.000	-0.0305	-0.0162
F-RGB-HOG	-.0144396463333*	0.0020	0.000	-0.0216	-0.0073
F-RGB-OF	-.0299268286000*	0.0020	0.000	-0.0371	-0.0228
F-HOG-OF	-.0389634900000*	0.0020	0.000	-0.0461	-0.0318
F-RGB-HOG-OF	-.0608066204667*	0.0020	0.000	-0.0680	-0.0537
FD1	-.0150206920667*	0.0020	0.000	-0.0222	-0.0079
FD2	-.0326549828667*	0.0020	0.000	-0.0398	-0.0255
FD3	-.0417319722000*	0.0020	0.000	-0.0489	-0.0346

Table B.22: Post hoc comparisons using Tukey's HSD - *D-RGB-OF* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0484082766000*	0.0020	0.000	0.0413	0.0556
HOG	.0406523756000*	0.0020	0.000	0.0335	0.0478
OF	.0120875268000*	0.0020	0.000	0.0049	0.0192
D-RGB-HOG	.0297234248667*	0.0020	0.000	0.0226	0.0369
D-HOG-OF	-.0145063599333*	0.0020	0.000	-0.0217	-0.0074
D-RGB-HOG-OF	-.0101942368667*	0.0020	0.000	-0.0173	-0.0030
D1	.0147790915333*	0.0020	0.000	0.0076	0.0219
D2	-.0144141924667*	0.0020	0.000	-0.0216	-0.0073
D3	0.0063513788667	0.0020	0.149	-0.0008	0.0135
F-RGB-HOG	.0152837785333*	0.0020	0.000	0.0081	0.0224
F-RGB-OF	-0.0002034037333	0.0020	1.000	-0.0074	0.0069
F-HOG-OF	-.0092400651333*	0.0020	0.001	-0.0164	-0.0021
F-RGB-HOG-OF	-.0310831956000*	0.0020	0.000	-0.0382	-0.0239
FD1	.0147027328000*	0.0020	0.000	0.0076	0.0219
FD2	-0.0029315580000	0.0020	0.991	-0.0101	0.0042
FD3	-.0120085473333*	0.0020	0.000	-0.0192	-0.0049

Table B.23: Post hoc comparisons using Tukey's HSD - *D-HOG-OF* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0629146365333*	0.0020	0.000	0.0558	0.0701
HOG	.0551587355333*	0.0020	0.000	0.0480	0.0623
OF	.0265938867333*	0.0020	0.000	0.0194	0.0337
D-RGB-HOG	.0442297848000*	0.0020	0.000	0.0371	0.0514
D-RGB-OF	.0145063599333*	0.0020	0.000	0.0074	0.0217
D-RGB-HOG-OF	0.0043121230667	0.0020	0.785	-0.0028	0.0115
D1	.0292854514667*	0.0020	0.000	0.0221	0.0364
D2	0.0000921674667	0.0020	1.000	-0.0071	0.0072
D3	.0208577388000*	0.0020	0.000	0.0137	0.0280
F-RGB-HOG	.0297901384667*	0.0020	0.000	0.0226	0.0369
F-RGB-OF	.0143029562000*	0.0020	0.000	0.0072	0.0215
F-HOG-OF	0.0052662948000	0.0020	0.449	-0.0019	0.0124
F-RGB-HOG-OF	-.0165768356667*	0.0020	0.000	-0.0237	-0.0094
FD1	.0292090927333*	0.0020	0.000	0.0221	0.0364
FD2	.0115748019333*	0.0020	0.000	0.0044	0.0187
FD3	0.0024978126000	0.0020	0.999	-0.0047	0.0096

Table B.24: Post hoc comparisons using Tukey's HSD - *D-RGB-HOG-OF* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0586025134667*	0.0020	0.000	0.0515	0.0658
HOG	.0508466124667*	0.0020	0.000	0.0437	0.0580
OF	.0222817636667*	0.0020	0.000	0.0151	0.0294
D-RGB-HOG	.0399176617333*	0.0020	0.000	0.0328	0.0471
D-RGB-OF	.0101942368667*	0.0020	0.000	0.0030	0.0173
D-HOG-OF	-0.0043121230667	0.0020	0.785	-0.0115	0.0028
D1	.0249733284000*	0.0020	0.000	0.0178	0.0321
D2	-0.0042199556000	0.0020	0.812	-0.0114	0.0029
D3	.0165456157333*	0.0020	0.000	0.0094	0.0237
F-RGB-HOG	.0254780154000*	0.0020	0.000	0.0183	0.0326
F-RGB-OF	.0099908331333*	0.0020	0.000	0.0028	0.0171
F-HOG-OF	0.0009541717333	0.0020	1.000	-0.0062	0.0081
F-RGB-HOG-OF	-.0208889587333*	0.0020	0.000	-0.0280	-0.0137
FD1	.0248969696667*	0.0020	0.000	0.0177	0.0320
FD2	.0072626788667*	0.0020	0.042	0.0001	0.0144
FD3	-0.0018143104667	0.0020	1.000	-0.0090	0.0053

Table B.25: Post hoc comparisons using Tukey's HSD - *D1* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0336291850667*	0.0020	0.000	0.0265	0.0408
HOG	.0258732840667*	0.0020	0.000	0.0187	0.0330
OF	-0.0026915647333	0.0020	0.997	-0.0098	0.0045
D-RGB-HOG	.0149443333333*	0.0020	0.000	0.0078	0.0221
D-RGB-OF	-.0147790915333*	0.0020	0.000	-0.0219	-0.0076
D-HOG-OF	-.0292854514667*	0.0020	0.000	-0.0364	-0.0221
D-RGB-HOG-OF	-.0249733284000*	0.0020	0.000	-0.0321	-0.0178
D2	-.0291932840000*	0.0020	0.000	-0.0363	-0.0220
D3	-.0084277126667*	0.0020	0.006	-0.0156	-0.0013
F-RGB-HOG	0.0005046870000	0.0020	1.000	-0.0066	0.0077
F-RGB-OF	-.0149824952667*	0.0020	0.000	-0.0221	-0.0078
F-HOG-OF	-.0240191566667*	0.0020	0.000	-0.0312	-0.0169
F-RGB-HOG-OF	-.0458622871333*	0.0020	0.000	-0.0530	-0.0387
FD1	-0.0000763587333	0.0020	1.000	-0.0072	0.0071
FD2	-.0177106495333*	0.0020	0.000	-0.0249	-0.0106
FD3	-.0267876388667*	0.0020	0.000	-0.0339	-0.0196

Table B.26: Post hoc comparisons using Tukey's HSD - *D2* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0628224690667*	0.0020	0.000	0.0557	0.0700
HOG	.0550665680667*	0.0020	0.000	0.0479	0.0622
OF	.0265017192667*	0.0020	0.000	0.0194	0.0337
D-RGB-HOG	.0441376173333*	0.0020	0.000	0.0370	0.0513
D-RGB-OF	.0144141924667*	0.0020	0.000	0.0073	0.0216
D-HOG-OF	-0.0000921674667	0.0020	1.000	-0.0072	0.0071
D-RGB-HOG-OF	0.0042199556000	0.0020	0.812	-0.0029	0.0114
D1	.0291932840000*	0.0020	0.000	0.0220	0.0363
D3	.0207655713333*	0.0020	0.000	0.0136	0.0279
F-RGB-HOG	.0296979710000*	0.0020	0.000	0.0225	0.0368
F-RGB-OF	.0142107887333*	0.0020	0.000	0.0071	0.0214
F-HOG-OF	0.0051741273333	0.0020	0.482	-0.0020	0.0123
F-RGB-HOG-OF	-.0166690031333*	0.0020	0.000	-0.0238	-0.0095
FD1	.0291169252667*	0.0020	0.000	0.0220	0.0363
FD2	.0114826344667*	0.0020	0.000	0.0043	0.0186
FD3	0.0024056451333	0.0020	0.999	-0.0047	0.0096

Table B.27: Post hoc comparisons using Tukey's HSD - *D3* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0420568977333*	0.0020	0.000	0.0349	0.0492
HOG	.0343009967333*	0.0020	0.000	0.0271	0.0415
OF	0.0057361479333	0.0020	0.296	-0.0014	0.0129
D-RGB-HOG	.0233720460000*	0.0020	0.000	0.0162	0.0305
D-RGB-OF	-0.0063513788667	0.0020	0.149	-0.0135	0.0008
D-HOG-OF	-.0208577388000*	0.0020	0.000	-0.0280	-0.0137
D-RGB-HOG-OF	-.0165456157333*	0.0020	0.000	-0.0237	-0.0094
D1	.0084277126667*	0.0020	0.006	0.0013	0.0156
D2	-.0207655713333*	0.0020	0.000	-0.0279	-0.0136
F-RGB-HOG	.0089323996667*	0.0020	0.002	0.0018	0.0161
F-RGB-OF	-0.0065547826000	0.0020	0.115	-0.0137	0.0006
F-HOG-OF	-.0155914440000*	0.0020	0.000	-0.0227	-0.0084
F-RGB-HOG-OF	-.0374345744667*	0.0020	0.000	-0.0446	-0.0303
FD1	.0083513539333*	0.0020	0.007	0.0012	0.0155
FD2	-.0092829368667*	0.0020	0.001	-0.0164	-0.0021
FD3	-.0183599262000*	0.0020	0.000	-0.0255	-0.0112

Table B.28: Post hoc comparisons using Tukey's HSD - *F-RGB-HOG* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0331244980667*	0.0020	0.000	0.0260	0.0403
HOG	.0253685970667*	0.0020	0.000	0.0182	0.0325
OF	-0.0031962517333	0.0020	0.980	-0.0103	0.0040
D-RGB-HOG	.0144396463333*	0.0020	0.000	0.0073	0.0216
D-RGB-OF	-.0152837785333*	0.0020	0.000	-0.0224	-0.0081
D-HOG-OF	-.0297901384667*	0.0020	0.000	-0.0369	-0.0226
D-RGB-HOG-OF	-.0254780154000*	0.0020	0.000	-0.0326	-0.0183
D1	-0.0005046870000	0.0020	1.000	-0.0077	0.0066
D2	-.0296979710000*	0.0020	0.000	-0.0368	-0.0225
D3	-.0089323996667*	0.0020	0.002	-0.0161	-0.0018
F-RGB-OF	-.0154871822667*	0.0020	0.000	-0.0226	-0.0083
F-HOG-OF	-.0245238436667*	0.0020	0.000	-0.0317	-0.0174
F-RGB-HOG-OF	-.0463669741333*	0.0020	0.000	-0.0535	-0.0392
FD1	-0.0005810457333	0.0020	1.000	-0.0077	0.0066
FD2	-.0182153365333*	0.0020	0.000	-0.0254	-0.0111
FD3	-.0272923258667*	0.0020	0.000	-0.0344	-0.0201

Table B.29: Post hoc comparisons using Tukey's HSD - *F-RGB-OF* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0486116803333*	0.0020	0.000	0.0415	0.0558
HOG	.0408557793333*	0.0020	0.000	0.0337	0.0480
OF	.0122909305333*	0.0020	0.000	0.0051	0.0194
D-RGB-HOG	.0299268286000*	0.0020	0.000	0.0228	0.0371
D-RGB-OF	0.0002034037333	0.0020	1.000	-0.0069	0.0074
D-HOG-OF	-.0143029562000*	0.0020	0.000	-0.0215	-0.0072
D-RGB-HOG-OF	-.0099908331333*	0.0020	0.000	-0.0171	-0.0028
D1	.0149824952667*	0.0020	0.000	0.0078	0.0221
D2	-.0142107887333*	0.0020	0.000	-0.0214	-0.0071
D3	0.0065547826000	0.0020	0.115	-0.0006	0.0137
F-RGB-HOG	.0154871822667*	0.0020	0.000	0.0083	0.0226
F-HOG-OF	-.0090366614000*	0.0020	0.002	-0.0162	-0.0019
F-RGB-HOG-OF	-.0308797918667*	0.0020	0.000	-0.0380	-0.0237
FD1	.0149061365333*	0.0020	0.000	0.0078	0.0221
FD2	-0.0027281542667	0.0020	0.996	-0.0099	0.0044
FD3	-.0118051436000*	0.0020	0.000	-0.0190	-0.0047

Table B.30: Post hoc comparisons using Tukey's HSD - *F-HOG-OF* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0576483417333*	0.0020	0.000	0.0505	0.0648
HOG	.0498924407333*	0.0020	0.000	0.0427	0.0570
OF	.0213275919333*	0.0020	0.000	0.0142	0.0285
D-RGB-HOG	.0389634900000*	0.0020	0.000	0.0318	0.0461
D-RGB-OF	.0092400651333*	0.0020	0.001	0.0021	0.0164
D-HOG-OF	-0.0052662948000	0.0020	0.449	-0.0124	0.0019
D-RGB-HOG-OF	-0.0009541717333	0.0020	1.000	-0.0081	0.0062
D1	.0240191566667*	0.0020	0.000	0.0169	0.0312
D2	-0.0051741273333	0.0020	0.482	-0.0123	0.0020
D3	.0155914440000*	0.0020	0.000	0.0084	0.0227
F-RGB-HOG	.0245238436667*	0.0020	0.000	0.0174	0.0317
F-RGB-OF	.0090366614000*	0.0020	0.002	0.0019	0.0162
F-RGB-HOG-OF	-.0218431304667*	0.0020	0.000	-0.0290	-0.0147
FD1	.0239427979333*	0.0020	0.000	0.0168	0.0311
FD2	0.0063085071333	0.0020	0.157	-0.0008	0.0135
FD3	-0.0027684822000	0.0020	0.995	-0.0099	0.0044

Table B.31: Post hoc comparisons using Tukey's HSD - *F-RGB-HOG-OF* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0794914722000*	0.0020	0.000	0.0723	0.0866
HOG	.0717355712000*	0.0020	0.000	0.0646	0.0789
OF	.0431707224000*	0.0020	0.000	0.0360	0.0503
D-RGB-HOG	.0608066204667*	0.0020	0.000	0.0537	0.0680
D-RGB-OF	.0310831956000*	0.0020	0.000	0.0239	0.0382
D-HOG-OF	.0165768356667*	0.0020	0.000	0.0094	0.0237
D-RGB-HOG-OF	.0208889587333*	0.0020	0.000	0.0137	0.0280
D1	.0458622871333*	0.0020	0.000	0.0387	0.0530
D2	.0166690031333*	0.0020	0.000	0.0095	0.0238
D3	.0374345744667*	0.0020	0.000	0.0303	0.0446
F-RGB-HOG	.0463669741333*	0.0020	0.000	0.0392	0.0535
F-RGB-OF	.0308797918667*	0.0020	0.000	0.0237	0.0380
F-HOG-OF	.0218431304667*	0.0020	0.000	0.0147	0.0290
FD1	.0457859284000*	0.0020	0.000	0.0386	0.0529
FD2	.0281516376000*	0.0020	0.000	0.0210	0.0353
FD3	.0190746482667*	0.0020	0.000	0.0119	0.0262

Table B.32: Post hoc comparisons using Tukey's HSD - *FD1* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0337055438000*	0.0020	0.000	0.0266	0.0409
HOG	.0259496428000*	0.0020	0.000	0.0188	0.0331
OF	-0.0026152060000	0.0020	0.998	-0.0098	0.0045
D-RGB-HOG	.0150206920667*	0.0020	0.000	0.0079	0.0222
D-RGB-OF	-.0147027328000*	0.0020	0.000	-0.0219	-0.0076
D-HOG-OF	-.0292090927333*	0.0020	0.000	-0.0364	-0.0221
D-RGB-HOG-OF	-.0248969696667*	0.0020	0.000	-0.0320	-0.0177
D1	0.0000763587333	0.0020	1.000	-0.0071	0.0072
D2	-.0291169252667*	0.0020	0.000	-0.0363	-0.0220
D3	-.0083513539333*	0.0020	0.007	-0.0155	-0.0012
F-RGB-HOG	0.0005810457333	0.0020	1.000	-0.0066	0.0077
F-RGB-OF	-.0149061365333*	0.0020	0.000	-0.0221	-0.0078
F-HOG-OF	-.0239427979333*	0.0020	0.000	-0.0311	-0.0168
F-RGB-HOG-OF	-.0457859284000*	0.0020	0.000	-0.0529	-0.0386
FD2	-.0176342908000*	0.0020	0.000	-0.0248	-0.0105
FD3	-.0267112801333*	0.0020	0.000	-0.0339	-0.0196

Table B.33: Post hoc comparisons using Tukey's HSD - *FD2* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0513398346000*	0.0020	0.000	0.0442	0.0585
HOG	.0435839336000*	0.0020	0.000	0.0364	0.0507
OF	.0150190848000*	0.0020	0.000	0.0079	0.0222
D-RGB-HOG	.0326549828667*	0.0020	0.000	0.0255	0.0398
D-RGB-OF	0.0029315580000	0.0020	0.991	-0.0042	0.0101
D-HOG-OF	-.0115748019333*	0.0020	0.000	-0.0187	-0.0044
D-RGB-HOG-OF	-.0072626788667*	0.0020	0.042	-0.0144	-0.0001
D1	.0177106495333*	0.0020	0.000	0.0106	0.0249
D2	-.0114826344667*	0.0020	0.000	-0.0186	-0.0043
D3	.0092829368667*	0.0020	0.001	0.0021	0.0164
F-RGB-HOG	.0182153365333*	0.0020	0.000	0.0111	0.0254
F-RGB-OF	0.0027281542667	0.0020	0.996	-0.0044	0.0099
F-HOG-OF	-0.0063085071333	0.0020	0.157	-0.0135	0.0008
F-RGB-HOG-OF	-.0281516376000*	0.0020	0.000	-0.0353	-0.0210
FD1	.0176342908000*	0.0020	0.000	0.0105	0.0248
FD3	-.0090769893333*	0.0020	0.002	-0.0162	-0.0019

Table B.34: Post hoc comparisons using Tukey's HSD - *FD3* on the UCF-101 Dataset

(I) Method	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
				Lower Bound	Upper Bound
RGB	.0604168239333*	0.0020	0.000	0.0533	0.0676
HOG	.0526609229333*	0.0020	0.000	0.0455	0.0598
OF	.0240960741333*	0.0020	0.000	0.0169	0.0312
D-RGB-HOG	.0417319722000*	0.0020	0.000	0.0346	0.0489
D-RGB-OF	.0120085473333*	0.0020	0.000	0.0049	0.0192
D-HOG-OF	-0.0024978126000	0.0020	0.999	-0.0096	0.0047
D-RGB-HOG-OF	0.0018143104667	0.0020	1.000	-0.0053	0.0090
D1	.0267876388667*	0.0020	0.000	0.0196	0.0339
D2	-0.0024056451333	0.0020	0.999	-0.0096	0.0047
D3	.0183599262000*	0.0020	0.000	0.0112	0.0255
F-RGB-HOG	.0272923258667*	0.0020	0.000	0.0201	0.0344
F-RGB-OF	.0118051436000*	0.0020	0.000	0.0047	0.0190
F-HOG-OF	0.0027684822000	0.0020	0.995	-0.0044	0.0099
F-RGB-HOG-OF	-.0190746482667*	0.0020	0.000	-0.0262	-0.0119
FD1	.0267112801333*	0.0020	0.000	0.0196	0.0339
FD2	.0090769893333*	0.0020	0.002	0.0019	0.0162