

# **Interaction Recognition in Aerial Videos for Security and Surveillance Applications**



By

Mubashir Shah

(Registration No: 00000364602)

Supervisor: Dr. Tahir Habib Nawaz

Department of Mechatronics Engineering

College of Electrical and Mechanical Engineering

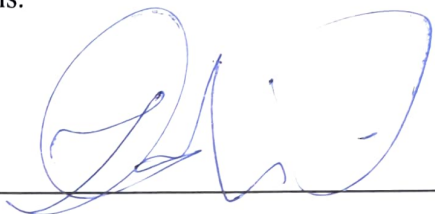
National University of Sciences & Technology (NUST)

Islamabad, Pakistan

(2024)

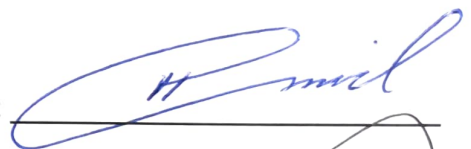
## THESIS ACCEPTANCE CERTIFICATE

It is to certify that the final version of MS Thesis written by Mr. **Mubashir Shah** (Registration No. **00000364602**), of **College of Electrical and Mechanical Engineering** has been reviewed and accepted. The submitted thesis complies with all NUST Statutes/ Regulations/ Masters Policy, is free of plagiarism, errors, and mistakes and is therefore accepted as partial fulfillment of the Master's degree requirements. Furthermore, it is certified that the revisions as pointed out by the GEC members and evaluators have also been incorporated into the final version of MS Thesis.

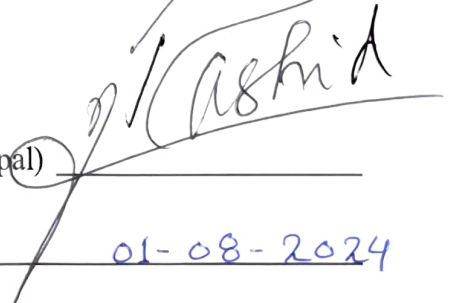
Signature: 

Name of Supervisor: **Dr. Tahir Habib Nawaz**

Date: 01 August 2024

Signature (HOD): 

Date: 01-08-2024

Signature (Dean/ Principal) 

Date: 01-08-2024

**Dedicated to my Parents.**

## ACKNOWLEDGEMENTS

In the name of Allah, the Compassionate and the Merciful. I express my gratitude to Almighty Allah for granting me the opportunity and abilities to complete my thesis.

I would like to express my profound appreciation to my esteemed supervisor, Dr. Tahir Habib Nawaz, for guiding, supporting, and motivating me throughout my thesis. His knowledge and advice were incredibly helpful.

I am deeply indebted to the esteemed members of my thesis committee, Dr. Nasir Rashid and Dr. Muhammad Osama Ali, for their constructive feedback and insightful suggestions, which have greatly enhanced the quality and rigor of my thesis.

Furthermore, I would like to extend my gratitude to NESCOMRAC Directorate and my Co-Principal Investigator, Dr. Rab Nawaz, for their generous funding support enabling the completion of this thesis research. Additionally, I express my sincere appreciation for his invaluable guidance and unwavering support throughout this research study.

I am also thankful to all my friends and colleagues at the Department of Mechatronics Engineering for their guidance, cooperation, and support during my studies. My special thanks to Hassan Akbar, Fahad ul Hassan and Muhammad Ahmed for their assistance in my research.

Lastly, I extend my sincere appreciation to my parents and siblings for their unwavering love, support, and encouragement during this journey. They have been my source of inspiration and motivation.

## ABSTRACT

Interaction recognition, a sub-domain of human activity recognition that primarily focuses on recognizing actions occurring between two subjects, which could be human-human or human-object interactions. In this area, researchers have concentrated on tasks such as object or person detection, tracking, and recognizing actions performed between subjects in videos. However, recognizing such activities in videos poses challenges, resulting in a limited availability of methods overall. Significant improvements have been made in recognizing solo actions, research on recognizing complex activities involving multiple subjects is ongoing. In this research study, we propose a novel keypoints-based deep learning model called 'InterAcT', that focuses on recognizing solo actions and interactions between two individuals in grayscale aerial videos. InterAcT is inspired from Action Transformer (AcT) model that captures spatial and temporal information using pose data. It features a lightweight architecture with 0.0795 million parameters and 0.0389 giga flops, distinguishing it from the AcT models. The pipeline primarily consists of a preprocessing stage and a pose-based deep learning transformer model. The preprocessing stage includes data augmentation, person detection, keypoints extraction, and data transformation modules. The transformer stage comprises six components: Linear Projection of Features, Class Token Embedding, Position Embedding, Transformer Encoder Layers, MLP Head, and Predicted Class Label. The transformer model utilizes sequential 2D pose data for training and outputs the recognized class. For performance evaluation, we have used two public datasets: the Drone Action dataset and UT-Interaction dataset, totaling 18 classes (13 solo actions and 5 interaction classes). The model was trained on 80% of the train set, validated on 10% of the validation set, and tested on 10% of the test set, achieving an

accuracy of 99.23%. On the same preprocessed data, we compared our model with benchmark models. It outperformed the AcT models (micro: 93.53%, small: 98.93%, base: 99.07%, and large: 95.58%). 2P-GCN achieved an accuracy of 93.37%, LSTM achieved 97.74%, 3D-ResNet achieved 99.21%, and 3D CNN achieved 99.20%. Our novel framework has the strength to recognize a large number of solo actions and two-person interaction classes in aerial videos, as well as fixed camera videos (grayscale and RGB). Due to its lightweight architecture, it can be utilized in real-world applications such as security and surveillance.

**Keywords:** Action Recognition, Interaction Recognition, Skeletal-based Transformer, Pose-based Classification, Aerial Videos, Grayscale Videos

# TABLE OF CONTENTS

<b>THESIS ACCEPTANCE CERTIFICATE</b>	<b>I</b>
<b>ACKNOWLEDGEMENTS</b>	<b>III</b>
<b>ABSTRACT</b>	<b>IV</b>
<b>TABLE OF CONTENTS</b>	<b>VI</b>
<b>LIST OF TABLES</b>	<b>IX</b>
<b>LIST OF FIGURES</b>	<b>X</b>
<b>LIST OF SYMBOLS, ABBREVIATIONS AND ACRONYMS</b>	<b>XI</b>
<b>CHAPTER 1: INTRODUCTION</b>	<b>1</b>
<b>1.1 Motivation, Scope, and Background</b>	<b>1</b>
<b>1.2 Activity Categories</b>	<b>2</b>
1.2.1 Gestures	2
1.2.2 Solo Actions	3
1.2.3 Interactions	4
1.2.4 Group Activity	5
1.2.5 Event Analysis	6
<b>1.3 Types of Activity Recognition Approaches</b>	<b>6</b>
1.3.1 Sensor-based Approaches	7
1.3.2 Vision-based Approaches	9
1.3.2.1 Action Representation	10
1.3.2.2 Action Classification	12
1.3.2.3 Deep Learning Approaches	15
1.3.2.4 Pose-based Approaches	17
<b>1.4 Challenges in Activity Recognition</b>	<b>18</b>
<b>1.5 Research Objectives</b>	<b>19</b>
<b>CHAPTER 2: LITERATURE REVIEW</b>	<b>20</b>
<b>2.1 Traditional and Deep Learning Approaches for Solo Actions and Interactions Recognition</b>	<b>20</b>
<b>2.2 Available Datasets for Activity Recognition</b>	<b>34</b>
<b>2.3 Scope of the Research Study</b>	<b>40</b>
<b>2.4 Research Gap Analysis and Contributions of the Proposed Methodology</b>	<b>41</b>
<b>CHAPTER 3: PROPOSED METHODOLOGY</b>	<b>42</b>
<b>3.1 Description of Proposed InterAcT Framework</b>	<b>44</b>
3.1.1 Linear Projection of Pose Features	45
3.1.2 Class Token Embedding	45
3.1.3 Position Embedding	46

3.1.4	Transformer Encoder Layer and MLP Head	46
<b>3.2</b>	<b>Implemented Framework - Flowchart</b>	<b>48</b>
<b>3.2.1</b>	<b>Pre-processing Block</b>	50
3.2.1.1	<i>Data Augmentation</i>	50
3.2.1.2	<i>Color Conversion</i>	51
3.2.1.3	<i>Pose Extraction</i>	51
3.2.1.4	<i>Data Transformation</i>	53
<b>3.2.2</b>	<b>Data Splits and DNN Block</b>	55
<b>3.2.3</b>	<b>Performance Evaluation</b>	56
3.2.3.1	<i>True Positives (TP)</i>	57
3.2.3.2	<i>True Negatives (TN)</i>	57
3.2.3.3	<i>False Positives (FP)</i>	57
3.2.3.4	<i>False Negatives (FN)</i>	57
3.2.3.5	<i>Model Parameters</i>	60
3.2.3.6	<i>Floating-point Operations (Flops)</i>	61
3.2.3.7	<i>Frames Per Seconds (FPS)</i>	61
3.2.3.8	<i>Inference Time</i>	62
<b>3.2.4</b>	<b>Inference Block</b>	62
<b>CHAPTER 4:</b>	<b>IMPLEMENTATION</b>	<b>63</b>
<b>4.1</b>	<b>Preprocessing of Datasets</b>	<b>63</b>
<b>4.2</b>	<b>Proposed InterAcT Model Training Parameters</b>	<b>67</b>
<b>4.2.1</b>	<b>Hyperparameters</b>	<b>67</b>
4.2.1.1	<i>Epoch</i>	68
4.2.1.2	<i>Learning Rate</i>	68
4.2.1.3	<i>Weight Decay</i>	69
4.2.1.4	<i>Batch Size</i>	70
4.2.1.5	<i>Sequence Length</i>	70
4.2.1.6	<i>Sequences Type</i>	71
<b>4.2.2</b>	<b>Activation Functions</b>	<b>72</b>
4.2.2.1	<i>Gaussian Error Linear Unit (GeLu)</i>	73
<b>4.2.3</b>	<b>Optimizers</b>	<b>73</b>
4.2.3.1	<i>Adaptive Moment Estimation with Weight Decay</i>	74
<b>4.3</b>	<b>Resources Utilized</b>	<b>75</b>
<b>4.3.1</b>	<b>Hardware Resources</b>	<b>75</b>
<b>4.3.2</b>	<b>Software Resources</b>	<b>75</b>
<b>4.4</b>	<b>Training Setup</b>	<b>76</b>
<b>CHAPTER 5:</b>	<b>EXPERIMENTS, RESULTS AND DISCUSSION</b>	<b>78</b>
<b>5.1</b>	<b>Experiments on Proposed InterAcT Framework</b>	<b>78</b>
<b>5.1.1</b>	<b>Experiments on Architectural Parameters</b>	<b>78</b>
5.1.1.1	<i>Effect of Number of Encoder Layers</i>	80
5.1.1.2	<i>Effect of Embedded Dimensions</i>	81
5.1.1.3	<i>Effect of Dropout Percentage</i>	82
5.1.1.4	<i>Effect of Dimensions of MLP Heads</i>	83
<b>5.1.2</b>	<b>Experiments on Hyperparameters, Optimizers and Activation Functions</b>	<b>84</b>
5.1.2.1	<i>Experiments on Optimizers with Learning Rates and Weight Decays</i>	<b>84</b>



<i>5.1.2.1.1 Experiments on AdamW Optimizer with Different Learning Rates and Weight Decays</i>	85
<i>5.1.2.1.2 Experiments on LAMB Optimizer with Different Learning Rates</i>	87
<i>5.1.2.1.3 Experiments on LazyAdam Optimizer with Different Learning Rates</i>	88
<i>5.1.2.1.4 Experiments on RAdam Optimizer with Different Learning Rates</i>	89
<i>5.1.2.1.5 Experiments on SGDW Optimizer with Different Learning Rates and Weight Decays</i>	90
<b>5.1.2.2 Experiments on Activation Functions</b>	<b>92</b>
<b>5.1.2.3 Experiments on Batch Size</b>	<b>93</b>
<b>5.1.2.4 Experiments on Sequences Type</b>	<b>94</b>
<b>5.2 InterAcT Model - Performance Evaluation Results</b>	<b>95</b>
<b>5.3 InterAcT Benchmarking with State-of-the-art Deep Learning Models</b>	<b>99</b>
<b>CHAPTER 6: CONCLUSIONS AND FUTURE RECOMMENDATIONS</b>	<b>101</b>
<b>REFERENCES</b>	<b>103</b>

## LIST OF TABLES

Table 2.1: Summary of literature review. ....	30
<b>No table of figures entries found.</b>	
Table 3.1: Parameters for four variants of AcT [118]. ....	47
Table 3.2: YOLO v8 pose keypoints indices. ....	53
Table 3.3: Classification performance metrics. ....	58
Table 4.1: Statistics of selected datasets. ....	63
Table 4.2: List of hyperparameters. ....	67
Table 4.3: List of activation functions. ....	72
Table 4.4: List of optimizers. ....	74
Table 4.5: Hardware specifications. ....	75
Table 5.1: Architectural parameters for four variants of AcT [118]. ....	79
Table 5.2: List of architectural parameters. ....	79
Table 5.3: InterAcT model comparison with AcT models. ....	99
Table 5.4: InterAcT model comparison with SOTA models. ....	99

## LIST OF FIGURES

Figure 1.1: Hand-Gestures, Image taken from HaGRID [4] dataset. ....	3
Figure 1.2: Solo Actions, Image taken from Drone-Action [2] dataset.....	3
Figure 1.3: Human-Human Interaction, Image taken from UT-Interaction [1] dataset.....	4
Figure 1.4: Human-Object Interaction, Image taken from HICO-DET [5] dataset.....	5
Figure 1.5: Group Activity, Image taken from Collective Activity (CAD) [6] dataset.....	5
Figure 1.6: Accident Analysis, Image taken from CADP [7] dataset.....	6
Figure 2.1: Image taken from UT-Interaction dataset [1]. ....	37
Figure 2.2: Image taken from Mini-drone dataset [127]. ....	37
Figure 2.3: Image taken from Drone-action dataset [2]. ....	37
Figure 2.4: Image taken from Okutama dataset [128]. ....	38
Figure 2.5: Image taken from Aeriform in-action dataset [129]. ....	38
Figure 2.6: Image taken from UCF Aerial Action dataset [91]. ....	39
Figure 3.1: Flowchart of InterAcT architecture. ....	44
Figure 3.2: InterAcT architecture - Transformer encoder layer (top) and Multi-head self-attention block (bottom). ....	47
Figure 3.3: Flowchart of Implemented Framework. ....	49
Figure 3.4: A General 2x2 Confusion Matrix. ....	58
Figure 4.1: Data Augmentation for Drone-Action “clapping” class. ....	65
Figure 4.2: Color Conversion for Ut-Interaction “handshaking” class and Drone-Action “kicking” class. ....	66
Figure 4.3: YOLO v8 Pose Keypoints for Ut-Interaction “handshaking” class and Drone-Action “waving hands” class. ....	66
Figure 5.1: Number of encoder layers versus validation accuracy. ....	80
Figure 5.2: Embedded dimensions versus validation accuracy. ....	81
Figure 5.3: Dropout percentage versus validation accuracy. ....	82
Figure 5.4: Dimensions of MLP heads versus validation accuracy. ....	83
Figure 5.5: AdamW optimizer - Weight decay versus validation accuracy. ....	85
Figure 5.6: AdamW optimizer - Learning rate versus validation accuracy. ....	86
Figure 5.7: LAMB optimizer - Learning rate versus validation accuracy. ....	87
Figure 5.8: LazyAdam optimizer - Learning rate versus validation accuracy. ....	88
Figure 5.9: RAdam optimizer - Learning rate versus validation accuracy. ....	89
Figure 5.10: SGDW optimizer: Weight decay versus validation accuracy. ....	90
Figure 5.11: SGDW optimizer: Learning versus validation accuracy. ....	91
Figure 5.12: Optimizers versus validation accuracy. ....	92
Figure 5.13: Activation functions versus validation accuracy. ....	92
Figure 5.14: Batch-size versus validation accuracy. ....	93
Figure 5.15: Sequences type versus validation accuracy. ....	94
Figure 5.16: Training and validation accuracy-loss curve. ....	95
Figure 5.17: Confusion matrix evaluated on test-set. ....	96
Figure 5.18: Class-wise accuracies. ....	97
Figure 5.19: Class-wise classification report. ....	98

# LIST OF SYMBOLS, ABBREVIATIONS AND ACRONYMS

3D Convolution	Attention Interactive Graph Convolutional
C3D, 23	Network
3D Convolutional Neural Network	(AIGCN), 23
(3DCNN), 22	Attention Residual 3D Network
3D Expanding Architecture	(AR3D), 26
X3D, 21	AZTR Auto Zoom and Temporal Reasoning
3D Expanding Architecture-Medium	AZTR, 21
X3D-M, 21	Backpropagation Through Time
3D Pose	(BPTT), 26
3DP, 36	Bidirectional Encoder Representations from
3D Pseudo	Transformers
P3D, 23	BERT, 45
Aalborg University Person Detection Thermal	Bi-directional Long-Short Term Memory
AAU-PD-T, 35	(Bi-LSTM), 24
Action Transformer	Car Accident Detection and Prediction
(AcT), xx	CADP, 6
Aerial-Rooftop-Ground Cameras	Channel-wise Topology Refinement Graph
ARG, 32	Convolution Network
Application Programming Interface	(CTR-GCN), 23
API, 75	Chinese Academy of Sciences Institute of
Artificial Neural Network	Automation
ANN, 7	CASIA, 35
Artificial Neural Network	Classification
(ANN), 7	CLS, 45

Collective Activity	Dmlp, 79
(CAD), 5	Dimensions of Model
Convolutional 3D	Dmodel, 79
C3D, 15	Donghua University
Convolutional Network	DHU, 35
ConvNet, 16	Double Data Rate
Convolutional Neural Network	DDR, 75
(CNN), 7	Error Function
CNN, xxi, 8	erf(x), 73
CNNs, 9, 15	False Negatives
Convolutional Neural Networks	(FN), 57
(CNNs), 8	False Positives
(CNNS), 15	(FP), 57
CNNs, 9, 17	Fisheye Modality
Cumulative distribution function of a standard	Fe, 36
Gaussian distribution	Floating-point Operations
$\Phi$ , 73	(Flops), 61
Deep Belief Networks	Frames Per Seconds
(DBNs), 8	(FPS), 61
Deep Neural Network	Gated Recurrent Units
(DNN), 7	(GRUs), 42
DNN, 7	Gaussian Error Linear Unit
Depth Features Extractor	GeLu, 46
DFE, 33	Generative Adversarial Network
Depth Modality	GAN, 24
D, 36	Graph Convolutional Networks
Dimensions of MLP Head	(GCNs), 21

Graphics Processing Unit	IR, 36
GPU, 76	Interaction Recognition Action Transformer
Hand Gestures Recognition Image Dataset	InterACT, xx
HaGRID, 3	Interactive Attention Encoding Graph
Hierarchical Long Short-Term Concurrent	Convolutional Network
Memory	(IAE-GCN), 23
(H-LSTCM), 26	Interactive-Attention Mask Temporal
High Resolution Network	Convolutional Network
HRNet, 52	(IAM-TCN), 23
Histogram of Optical Flow	Joint Modality
HOF, 11	J, 36
Histograms of Optical Flow	Joint-annotated Human Motion Data Base
(HOF), 11	J-HMDB, 35
Histograms of Oriented Gradients	Kinetic 3D Human Interactions
(HOG), 11	K3HI, 35
HOG, 11	Knowledge-embedded Graph Convolution
Human Activity Recognition	Network
(HAR), 7	(K-GCN), 25
HAR, 7, 8, 9	Lasso
Human Interacting with Common Objects-	L1, 60
Detection	Layer-wise Adaptive Moments optimizer for
HICO-DET, 5	Batch training
Human Mask Modality	LAMB, 74
HM, 36	Lazy Adaptive Moment Estimation Model
Inflated 3D Convolutionals Network	LazyAdam, 74
I3D, 24	Linköping Thermal InfraRed
Infrared Modality	LTIR, 35

Long Short-Term Memory  
(LSTM), 9  
LSTM, xxi, 16

Mean Average Precision  
mAP, 34

MicroSoft Research  
MSR, 32

Minimum Noise Fractions  
MNF, 24

Modular Integrated Video and Image Analysis  
MIVEA, 32

Motion Boundary Histograms  
(MBH), 11  
MBH, 11

Motion Energy Image  
(MEI), 10  
MEI, 10

Motion History Image  
(MHI), 10  
MHI, 10

Motion History Volume  
(MHV), 10

Multi-Layer Perceptron  
MLP, xx

Mutual Information based Temporal Feature  
Alignment and Sampling  
MITFAS, 21

Nanyang Technological University  
NTU, 23

Natural Language Processing  
(NLP), 17  
NLP, 17

Neighborhood Watch-University of California,  
Los Angeles  
NW-UCLA, 34

Night vision  
Nv, 36

Number of MLP Heads  
H, 79

Number of Transformer Encoder Layers  
L, 79

ontext Aware Vision using Image-based Active  
Recognition  
CAVIAR, 35

Peking University Multimodal Data  
PKU-MMD, 35

Principal Component Analysis  
PCA, 13

Random Access Memory  
RAM, 75

Random Sample Consensus  
RANSAC, 12

Ray Tracing Texel eXtreme  
RTX, 75

Rectified Lazy Adaptive Moment Estimation	Skinned Multi Person Linear fits
Model	SMPL, 36
RAdam, 74	Space-Time Interest Points
Recurrent Neural Network	(STIPs), 11
(RNN), 7	STIPs, 11
RNN, 9	Spatial Features Extractor
Recurrent Neural Networks	SFE, 33
(RNNs), 9	Spatio-Temporal Graph Convolutional Network
RNNs, 9, 17	ST-GCN, 22
Red Green Blue	Stacked Autoencoders
RGB, xxi, 2	(SAEs), 8
Red Green Blue-Depth	SAEs, 8, 9
RGB-D, 12, 14	Stacked Dense Flow Difference Image
Residual Network	SDFDI, 36
ResNet, xxi	Stacked Saliency Difference Image
Restricted Boltzmann Machine	SSDI, 36
(RBM), 7	State-Of-The-Art
RBM, 8	(SOTA), 23
Restricted Boltzmann Machines	Stochastic Gradient Descent
(RBMs), 8	(SGD), 74
RBMs, 8, 9	Stony Brook University
Ridge	SBU, 23
L2, 60	Temporal Convolutional Network
Semantics Guided Network	TCN, 22
(SGN), 23	Temporal Relation Network
Skeleton Modality	TRN, 23
S, 36	Temporal Segment Networks



(TSN), 16

True Negatives

(TN), 57

True Positives

(TP), 57

Two Persons Graph Convolutional Network

(2P-GCN), 20

2P-GCN, xxi

University of Central Florida

UCF, 31

Unmanned Aerial Vehicle

UAV, 21

Video Random Access Memory

VRAM, 75

Vision Transformers

(ViTs), 27

Wasserstein Generative Adversarial Network

with Gradient Penalty

(WCGAN-GP), 24

Weighted Adaptive Moment Estimation Model

AdamW, 74

Weighted Stochastic Gradient Descent

SGDW, 74

You Only Look Once

YOLO, 1, 20



# CHAPTER 1: INTRODUCTION

Activity recognition in videos is a prominent research study in computer vision and machine learning/deep learning. It involves developing algorithms and techniques to automatically analyze and comprehend human activities captured in video sequences. With the increasing availability of datasets and the advancements in computational power, action recognition has become an important study and has found applications in areas such as security and surveillance, human-computer interaction, sports analysis, and autonomous systems. There are numerous challenges that are faced by researchers in this domain, these particularly includes data-modalities, activity type to recognize, subjects involved, action classes, weather/illumination conditions, background variations, variations in subject appearance, camera altitude, and viewpoints changes. Researchers have proposed a wide range of methodologies using traditional and advanced machine learning approaches as well as computer vision algorithms to recognize activities under these challenges.

This research study proposes a novel deep learning model that utilizes pose data to recognize solo actions as well as human-human interactions in grayscale aerial videos. The proposed approach uses two public datasets namely Ut-Interaction [1] and Drone-Action [2] datasets to recognize 18 classes (13 solo actions and 5 human-human interactions classes). The pipeline incorporates YOLO v8 [3] Person Detection and Pose Extraction algorithm to extract poses of subjects in the videos and transform the sequential pose-data to fed into the transformer model to recognize the undergoing activity class in the videos.

## **1.1 Motivation, Scope, and Background**

Activity recognition is the task of detecting the subject(s) and recognizing the associated activity by using computer vision and machine learning/deep learning algorithms. This research has potential applications including Security and Surveillance, Sports Analysis, Education, Healthcare and Augmented Reality. The demand for video-based activity recognition is increasing due to availability of public video datasets, sharing

of videos on social media and development of advanced machine learning, deep learning, and computer vision algorithms.

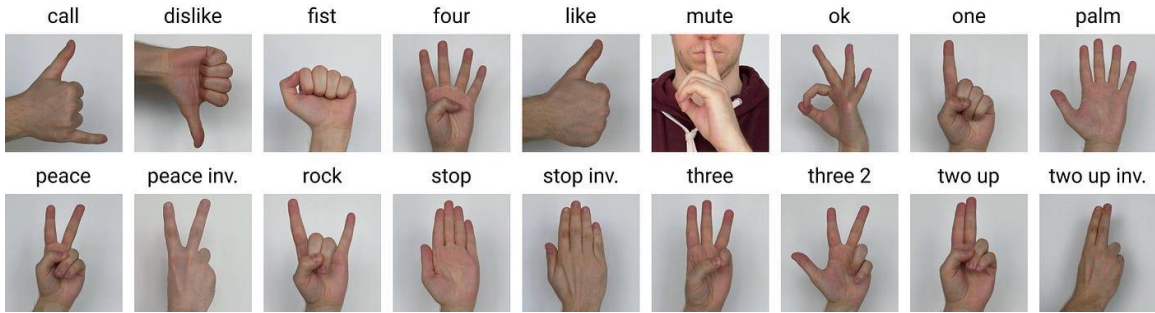
This research study primarily focused on utilizing RGB video datasets to perform the desired activity recognition which includes solo-actions recognition and human-human interactions. Here, it is crucial to mention that the low-altitude aerial dataset is to be considered containing moving target(s) that can be reliably detected and their pose reliably extracted; hence the assumption is that the image resolution is satisfactory. The solution could therefore possibly involve the use of extracted pose information because different activities, in principle, are distinguishable based on target's bodily movements. As part of the investigation, the model's evaluation will be performed on the data that is converted from RGB (24-bit) to grayscale (8-bit).

## **1.2 Activity Categories**

The choice of activity to recognize is a challenge as the range of human activities spans from simple, fine-grained actions to complex activities. The choice of selecting desired activity type can be made easy by categorizing the activities in five common groups. The categories of activity types are Gestures, Solo-actions, Interactions, Group Activity and Event Analysis.

### ***1.2.1 Gestures***

Gesture recognition involves the identification and interpretation of hand or body movements to understand user intentions or commands. This type of activity recognition plays a vital role in giving instructions or commands to other subject(s). Gestures offer a non-verbal means of communication, making them particularly valuable in scenarios where speech may be impractical or impossible. As such, gesture recognition holds promise in fields such as traffic control, autonomous vehicles, gaming, augmented reality, and healthcare, enhancing user experiences and expanding the possibilities of interaction with technology.



**Figure 1.1:** Hand-Gestures, Image taken from HaGRID [4] dataset.

### 1.2.2 Solo Actions

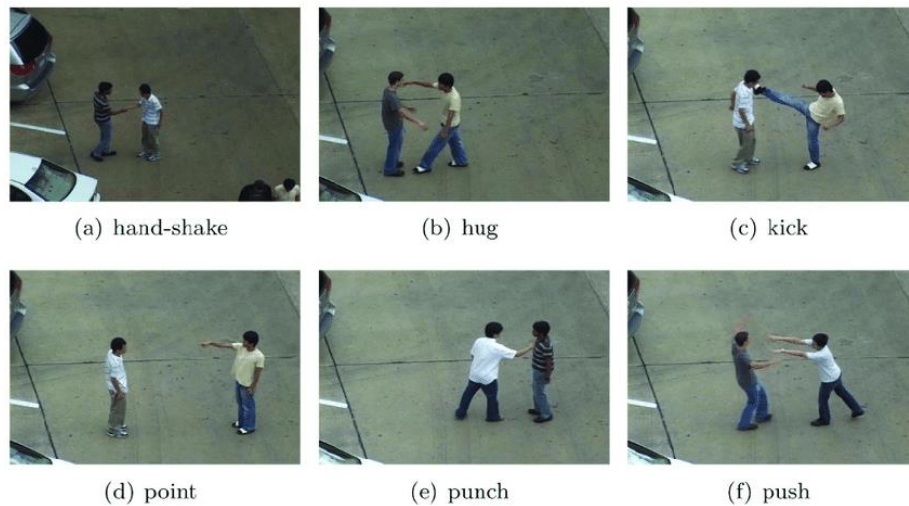
Solo action recognition pertains to the identification and understanding of individual movements or actions executed by a single subject. This form of activity recognition is essential in various contexts, including sports analysis, rehabilitation, and gesture-based interfaces. Solo actions examples are walking, running, or specific exercises. Solo action recognition systems often rely on advanced algorithms and sensor or visual data to accurately detect and classify these actions. The insights gained from recognizing solo actions can inform personalized training routines, track progress in rehabilitation programs, and even contribute to enhancing sports performance analysis. Consequently, solo action recognition holds significant potential in improving physical well-being, performance, and overall quality of life for individuals.



**Figure 1.2:** Solo Actions, Image taken from Drone-Action [2] dataset.

### 1.2.3 Interactions

Interaction recognition focuses on understanding the activities performed between multiple subjects which could be multiple humans or humans and objects. This category has two groups, Human-Human Interaction and Human-Object Interaction. This type of action recognition is important in numerous applications, including sports analysis, human-computer interaction, security and surveillance. By analysing how people interact with each other or with objects, the interaction model can recognize activities like shaking hands or opening a door. These insights are valuable in designing and development of interfaces which understand the interactions and enhances its usage in real-world applications. Interaction recognition ultimately aims to improve communication between multiple subjects making it more efficient and effective across multiple applications.



**Figure 1.3:** Human-Human Interaction, Image taken from UT-Interaction [1] dataset.



**Figure 1.4:** Human-Object Interaction, Image taken from HICO-DET [5] dataset.

### 1.2.4 Group Activity

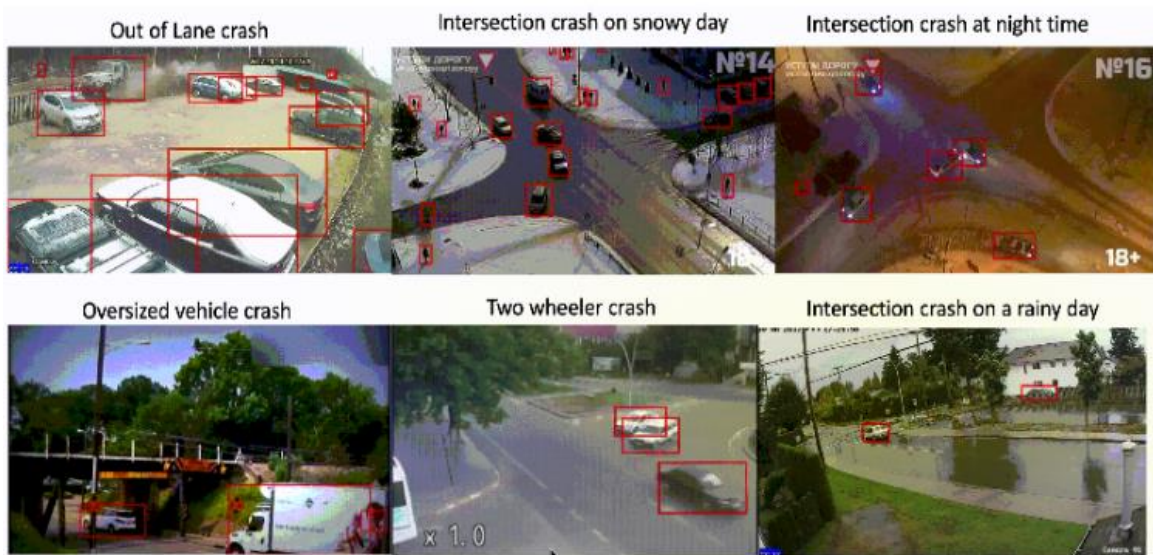
Group activity recognition involves identifying and understanding the collective behaviours and interactions of multiple individuals within a scene or environment. This activity type found its applications in crowd management, security and surveillance. By analysing the movements, positions, and interactions of individuals within a group, the group activity model can infer group activities including meetings, conversations, or crowd events like protests or gatherings. The knowledge acquired through group activity recognition can enhance decision-making across different fields, leading to enhanced safety, productivity, and comprehension of human behaviours within collective environments.



**Figure 1.5:** Group Activity, Image taken from Collective Activity (CAD) [6] dataset.

### 1.2.5 Event Analysis

Event analysis in activity recognition involves the detection and understanding of specific occurrences or incidents within an environment. This type of activity recognition is crucial in various domains, including surveillance, sports analysis, and event detection in videos. By analysing patterns and sequences of actions, technology can identify and classify events such as accidents, sports plays, or social gatherings. The knowledge acquired through event analysis can guide decision-making, improve awareness of situations, and streamline response strategies across diverse real-world scenarios.



**Figure 1.6:** Accident Analysis, Image taken from CADP [7] dataset.

### 1.3 Types of Activity Recognition Approaches

Activity recognition types depend on the nature of activity recognition category as discussed in Section 1.2, the available data modalities, and methods of either traditional or deep learning approaches. However, they can be divided into two main categories: sensor-based approaches and vision-based approaches.



### *1.3.1 Sensor-based Approaches*

In sensor-based approaches, there are mainly two categories: wearable sensors [8] and non-wearable sensors [9]. Wearable sensors are devices integrated into clothing or accessories worn by individuals to capture physiological signals or motion data directly from the body. Wearable sensors comprise of accelerometer, magnetometer, gyroscope, smart watches, bands, glasses and helmets. Alternatively, non-wearable sensors are stationary or mobile devices that capture visual or environmental data from the surroundings without direct attachment to the body. Non-wearable sensors include accelerometer, gyroscopes, sound sensors, pressure sensors and temperature sensors. The sensor-based approaches are explained in [10]. These approaches include Artificial Neural Network (ANN) and Deep Neural Network (DNN) [11], Convolutional Neural Network (CNN) [12]), Autoencoder [13], Restricted Boltzmann Machine (RBM) [14], Recurrent Neural Network (RNN) [15], and Hybrid Models [16]. They are explained as follows:

Deep Neural Network (DNN) is an Artificial Neural Network (ANN) which consists of multiple layers. Unlike simpler ANN models, DNNs utilizes large amounts of data to learn from, they are more effective compared to ANN because of their depth. In Human Activity Recognition (HAR), DNNs are often used. Early HAR approaches used DNNs mainly for classification tasks after extracting features manually. But recent studies have shown that deeper DNNs can perform better, especially when dealing with complex activities and multidimensional data.

Convolutional Neural Networks (CNNs) strengths lies in extracting features from signals, making them useful for activity recognition. They work well with time series data because they can capture local patterns and ignore irrelevant details. In HAR, CNNs applies pooling techniques and share weights effectively. Input adaptation transforms the data to a format that CNNs can understand. Pooling helps to extract important features and speed up training. Weight sharing techniques [17], which includes partial weight-sharing methods [12], can enhance the performance of CNNs by reducing redundancy and speeding up learning.

Autoencoders, including Stacked Autoencoders (SAEs) [18], learn to represent input data in a more compact form through hidden layers. In HAR, SAEs can automatically learn useful features from the data, which is helpful when labelled data is less. Pre-training methods, like layer-wise pretraining, help SAEs learn better representations. Adding constraints, such as sparsity, can further improve their performance. However, finding the right configuration and activation functions for SAEs can be challenging.

Restricted Boltzmann Machines (RBMs) [19], often used in Deep Belief Networks (DBNs), can also learn features from HAR data without labels. They work by finding patterns in the data and building a hierarchical representation. RBMs are trained layer by layer, starting with simpler features and gradually learning more complex ones. Techniques like pooling and weight updates help improve their efficiency and performance. RBM-based models are valuable for unsupervised feature learning in HAR, especially in environments with limited resources.

Recurrent Neural Networks (RNNs) [15] comprises of Long Short-Term Memory (LSTM) cells, plays a vital role in capturing temporal dependencies in video sequences. While not as common as CNNs, RNN-based models are optimized for learning speed and resource efficiency. They are particularly suitable for real-world applications where processing speed and memory usage are critical. Researchers have explored various techniques to optimize RNN architectures for HAR tasks, ensuring both accuracy and efficiency.

Hybrid models [16] is combination of spatial and temporal approaches, like combinations of CNNs and RNNs, offer a powerful approach for HAR. By integrating both spatial and temporal information, these models can achieve better performance. Combining CNNs with generative models like SAEs or RBMs also speeds up training and improves feature extraction. Hybrid models have shown more promising recognition of human activities in diverse environments.

### ***1.3.2 Vision-based Approaches***

Vision-based action recognition is defined as the algorithms that identify and comprehend human activities using visual information extracted from video or image data. These approaches use computer vision algorithms, such as feature extraction and pattern recognition, to analyze temporal and spatial cues within video frames or images and classify observed movements into predefined action classes.

A survey on vision-based approach is presented in [20]. Vision-based approaches mainly comprise of two components [21]: action representation and action classification. Action representation is the process of converting action videos into

feature vectors [22] whereas action classification refers to deduction of class labels from the feature vectors [23]. The evolution of deep neural architectures [24] has combined these components into their complex model structure and has enhanced the classification performance. Recent studies focus on pose-based methods which are presented in [25], they rely on human poses to understand actions. These methods involve detecting key body joints extracted from images or videos, often using techniques like pose estimation. These methods have demonstrated promising results in tasks of human action recognition.

This section provides insights into action representation, action classification and deep architectures approaches. Each approach is discussed in detail as follows:

#### ***1.3.2.1 Action Representation***

Action representation is the extraction of representative and discriminative features and patterns that help in recognition of different action classes. The action representation methods discussed in this section are known as hand-crafted methods as in these methods, the model's parameters are predefined. The action representations include holistic representations and local representations.

Holistic representation in video action recognition encompasses a variety of methods aimed at encoding spatial and dynamic information of human actions comprehensively. Motion Energy Image (MEI) and Motion History Image (MHI) [26] provide single-image representations that emphasize both spatial and temporal aspects of motion, with MEI highlighting the spatial distribution and MHI indicating the temporal evolution. The 3D Motion History Volume (MHV) [27] addresses the

challenge of viewpoint changes by utilizing 3D voxel data from multiple camera views and applying Fourier transform to ensure viewpoint-invariant features. Additionally, methods such as the utilization of the Poisson equation [28] and spatio-temporal volume analysis [29] extract shape properties and features, contributing to a holistic understanding of human actions. Optical flow algorithms [30] capture apparent motion between frames, offering insights into both horizontal and vertical motion patterns. Channel-splitting techniques [31] further enhance the representation of detailed motion information for human body and body parts. Collectively, these methods provide a rich and expressive representation of human actions, enabling robust action recognition systems capable of understanding diverse actions in video data.

Local representations in video action recognition focus on identifying salient motion information within specific regions, offering a solution to the limitations of holistic approaches. Techniques such as space-time interest points (STIPs) [32] and motion trajectories [33] have demonstrated robustness to translation and appearance variation. STIPs extend the Harris corner detector [34] to the spatio-temporal domain, detecting large motion changes, while dense interest point detection utilizes Gaussian smoothing and Gabor filtering [35]. Descriptors like histograms of optical flow (HOF) [36], histograms of oriented gradients (HOG) [37] and motion boundary histograms (MBH) [38] capture motion and appearance information within these regions. However, spatiotemporal interest points are limited in capturing long-term duration information, addressed by feature trajectories which track interest points over time. Trajectories are described by concatenating features such as HOG, HOF, and MBH, [33] or employing hierarchical context information for accuracy and robustness.

Techniques to mitigate camera motion effects, such as finding correspondences between frames and estimating homography using RANSAC, are also employed [39]. These local representation methods provide detailed insights into motion patterns within specific regions, enhancing the accuracy and robustness of action recognition.

### ***1.3.2.2 Action Classification***

Action classification involves training the model to learn patterns from training data to determine the probabilities boundaries to discriminate and classify numerous action classes. Classification methods can be categorized into direct classification, sequential classification, space-time classification, part-based approaches, manifold learning approaches, mid-level feature classification. feature fusion techniques, classification approaches for interaction recognition and approaches for RGB-D modality, each approach is discussed as follows:

Direct Classification [39] approaches summarize action videos into feature vectors, which are then classified into action classes using pre-existing classifiers such as support vector machine and k-nearest neighbor. These methods characterize action dynamics holistically using action shape or the bag-of-words model, capturing local motion patterns.

Sequential approaches [40] capture the temporal dependencies of appearance or pose using sequential models like hidden Markov models, conditional random fields, and structured support vector machines. These methods treat a video as a temporal segments or frames known as video sequences, modeling the trajectory or key poses to represent human actions.

Space-time approaches [41] take into account the spatiotemporal correlations between local features and the global spatio-temporal distribution of interest points. Methods like global Gaussian mixture models, Directional Pyramid Cooccurrence Matrix, and context-dependent graph kernels capture the geometrical distribution and relationships among local features.

Part-based approaches [42] model human actions by utilizing motion data from both the entire body and individual body parts. These methods inherently capture the spatial relationships among body parts, employing models like constellation models and hierarchical part-based models.

Manifold learning approaches [43] decrease the dimensionality of silhouette representations and map them onto nonlinear low-dimensional dynamic shape manifolds. Techniques like kernel PCA and optimal manifold embedding aim to efficiently represent temporally variational human silhouettes for action recognition.

Mid-level feature approaches [44] learn additional layers of representations from low-level features to better abstract features for classification. Hierarchical approaches and semantic descriptions are explored to learn better action representations.

Feature fusion approaches [45] combine multiple types of features from videos to improve action recognition. Methods like maximum margin distance learning, Multi-Task Sparse Learning, and multi-feature max-margin hierarchical Bayesian model fuse various features by exploiting their correlations and inter-relationships.

Human interaction recognition involves understanding actions performed among multiple subjects, for instance "handshaking" or "talking" between two individuals. Early methods [36] treated interactions as holistic motions, failing to capture individual actions within the group. Recent advancements focus on explicit modeling of action co-occurrence and spatial relationships between interacting individuals. These approaches utilize techniques like coupling motion states, body part tracking, and structured learning to better understand complex interactions. Hierarchical representation models offer a comprehensive understanding by categorizing atomic actions, interactions, and collective actions. Additionally, part-based approaches focus on modeling interactions using key poses and spatial relationships between body parts, providing detailed insights into interaction dynamics. Patch-aware models address the ambiguity in feature-to-person assignments, improving recognition accuracy, especially in interactions with close physical contact.

Action recognition from RGB-D videos [46] leverages depth information to capture 3D structural details, reducing noise and simplifying motion variations. Effective feature extraction techniques like histogram of oriented 4D normals and depth spatiotemporal interest points enhance recognition accuracy. These features capture spatio-temporal information, contributing to an effective recognition. Fusion of RGB and depth data enhances classification performance by leveraging shared features between modalities. Methods explore both shared and private features, improving recognition accuracy even when one modality is missing. Integration of auxiliary information, such as skeleton data from Kinect sensors, further enhances recognition



performance by learning shared feature spaces and leveraging auxiliary databases for improved action reconstruction.

### ***1.3.2.3 Deep Learning Approaches***

Deep learning techniques have revolutionized action recognition by enabling the automatic learning of powerful features from raw data. In contrast to hand-crafted features, which demand considerable human effort and expertise, deep learning methods can automatically learn hierarchical representations that generalize well across diverse datasets. Deep architectures comprise of Convolutional Neural Networks (CNNS), Temporal Modeling, Multistream Networks and Hybrid Approaches. Recent deep architectures are Transformer based architectures relying on attention mechanisms.

In CNNs domain, the prominent architecture for action recognition is the C3D (Convolutional 3D) network [47] purpose-built for analyzing video data. C3D networks consist of layers of convolutions and max-pooling, followed by fully connected layers, enabling the extraction of spatiotemporal features from video sequences efficiently. On the other hand, traditional 2D Convolutional Networks [48] although efficient for capturing spatial features from individual frames, lack explicit temporal modeling capabilities.

Temporal modeling in deep architectures involves various techniques to capture temporal patterns in videos effectively. Three-Dimensional Convolution (3D Convolution) directly captures temporal trajectories by convolving over a video sequence, allowing for the creation of hierarchical representations of spatiotemporal

data. Additionally, temporal pooling and aggregation methods [49] such as LSTM networks and temporal pooling layers, facilitate the modeling of temporal dependencies and long-range interactions in videos. Temporal Segment Networks (TSN) offer an efficient way to analyze short video snippets to capture long-range dynamics effectively.

Multi-Stream Networks [50] such as Two-Stream Networks, incorporate separate spatial and temporal streams to capture appearance and motion information independently. These streams, typically comprising a spatial ConvNet and a temporal ConvNet, are fused at the decision level to achieve robust action recognition. Fusion strategies like spatial fusion functions and residual connections enhance interaction between spatial and temporal streams, improving the extraction of spatiotemporal features.

Hybrid Networks [51] merge the capabilities of convolutional layers with recurrent layers, like LSTMs, to efficiently capture both spatial and temporal information. By leveraging the capabilities of both architectures, hybrid networks can model complex spatiotemporal patterns and long-term dependencies present in videos. In pose-based action recognition, hybrid neural networks are developed to model structured body joints and temporal information obtained from skeleton data. Methods like recurrent neural networks, temporal CNNs, and graph convolution networks are utilized to comprehensively capture both the spatial and temporal features of body movements.

A survey on vision transformers in action recognition tasks are presented in [52] represents an advancement in deep learning, with its significant impact spanning various fields. Many architectures of vision transformers have been applied for activity recognition tasks and each has its unique pros and cons. Initially recognized for its remarkable success in natural language processing (NLP), transformers have outperformed traditional methods like recurrent and convolutional neural networks (RNNs and CNNs). The primary feature of this architecture is its attention mechanism, enabling models to concentrate on relevant parts of input sequences. Expanding beyond NLP, transformers have been used in computer vision tasks, showing better performance than standard CNNs for recognizing images on large datasets. Vision transformers have demonstrated high effectiveness in various computer vision tasks. Additionally, recent efforts have extended transformer architectures to action recognition tasks, dealing with the complexities of extracting spatial and temporal information from video sequences and has shown promising results.

#### ***1.3.2.4 Pose-based Approaches***

A survey on pose-based methods is presented in [53]. The survey explores recent advancements in understanding human actions through analyzing body poses in images or videos. There are two main categories of pose-based action recognition, these are 3D skeleton-based action recognition, and 2D skeleton-based action recognition.

The survey explores recent advancements in pose extraction and its application in activity recognition using poses data extracted from RGB videos. It covers common datasets used for evaluation, comprehensive reviews of pose extraction methods (both

deep learning-based and traditional), recent works integrating pose estimation into action recognition, and discussions on the challenges and future directions in the field. It provides insights into different frameworks for pose extraction, compares skeleton-based and video-based activity recognition algorithms, and highlights the importance of improving pose extraction for effective activity recognition.

#### **1.4 Challenges in Activity Recognition**

Vision-based activity recognition encounters numerous challenges, including anthropometric differences [54], multi-view variation [55], cluttered backgrounds [56], intraclass variability [57], occlusions [58], illumination changes [59], low quality videos [60], camera motion [61], data scarcity [62], adverse weather [63], and computational constraints [64]. Moreover, multi-target tracking is also a challenging task in surveillance applications [65]. Anthropometric variation refers to the diversity in body size and shape among individuals, which impact activity recognition. Multiview variation arises from different perspectives capturing activities, adding complexity. Cluttered and dynamic backgrounds with distractions hinder recognition, the difficulty of distinguishing between similar actions (intraclass variability) and similar-looking actions (interclass similarity) in the videos also poses a challenge in activity recognition. Low-quality videos with poor resolution or noise and occlusions, where objects or body parts block views also present a challenge to activity recognition. Illumination variations, shadows, and scale changes also affect recognition. The challenge of camera motion introduces blurriness or distortion into video data which makes recognition a challenging task. Insufficient data availability and limited diversity in datasets restrict model training. Poor weather conditions can degrade video quality, and computational constraints challenge resource-intensive processing needs. Tracking multi-targets effectively in surveillance applications is also a challenging task. Addressing these challenges is crucial for improving the performance of vision-based human activity recognition models. For further insights into these challenges, we recommend referring to the following sources [66], [67], [68] which offer comprehensive discussions on the challenges in activity recognition.

## 1.5 Research Objectives

The research objectives of this study are as follows:

1. To recognize Human-Human Interactions as well as Solo actions in aerial grayscale videos.
2. To identify suitable dataset(s) for evaluation.
3. To propose and implement a state-of-the-art pose-based deep learning framework that recognizes human-human interactions as well as solo actions.
4. To perform experimentations on proposed pipeline to enhance model's performance and obtain an optimized variant of the model.
5. To perform comparative analysis of the proposed model with other state-of-the-art models.

## CHAPTER 2: LITERATURE REVIEW

Activity recognition in videos is an important research study in computer vision and machine learning/deep learning. It involves developing algorithms and techniques to automatically analyze and comprehend human activities captured in video sequences. With the increasing availability of video data and the advancements in computational power, it has gained significant attention in diverse applications such as surveillance and security [69] human-computer interaction, sports analysis, and autonomous systems. This literature review aims to explore and summarize the recent advancements, methodologies, and datasets employed specifically in the field of interactions and solo actions recognition for security and surveillance applications, shedding light on the progress made in this exciting field of study.

### **2.1 Traditional and Deep Learning Approaches for Solo Actions and Interactions Recognition**

An effective Transformer-based method [70] for human activity recognition in aerial videos is proposed which uses skeletal keypoints extracted with YOLOv8 [3]. The proposed method has a two-stage network, first extracting skeletal keypoints and then feeding them into a Transformer network for training and testing. This approach offers a promising solution for efficient and effective solo human activity recognition in aerial videos. [70]

Two-persons Graph Convolutional Network (2P-GCN) [71] is skeleton-based human-human interaction recognition, that aims to improve the accuracy of action recognition in human-to-human interaction sequences. It addresses the limitation of current interaction recognition methods that treat two-person interactions as isolated individuals, potentially losing crucial interactive information. 2P-GCN introduces a unified two-person graph to represent both intra-body and inter-body correlations, enhancing spatial modeling. The model achieves significant accuracy improvements in recognizing both human-human interactions and solo actions [71].

MITFAS [72] employs the expanding architecture X3D, which encompasses five different variants. Each variant of X3D is characterized by its computational complexity, measured in terms of flops/G-flops. Within the MITFAS architecture, the baseline model is X3D-M. This model is enhanced with temporal feature alignment and frame sampling techniques, enabling it to identify individual actions performed by a person in UAV videos. The temporal feature alignment method employs the concept of mutual information to extract features that emphasize human actions rather than background areas. Meanwhile, the frame sampling method is responsible for selecting the most informative and distinct frames from video sequences. The purpose of the MITFAS model is to effectively handle challenges such as variations in human resolution, significant alterations in the positions of individuals across frames, and the partial occlusion of key action points due to the continuous movement of UAVs [72].

AZTR [73] is a novel deep learning method that incorporates an auto-zoom model to identify and scale the subject, optimizing device memory and processor usage. The auto-zoom model focuses on spatial information, reduces interference and anomalies caused by UAV movement, and keeps the target object centered in the video. It enhances feature extraction and robustness. Additionally, the method employs a temporal reasoning algorithm, utilizing convolutions (2D+1 & 3D) and attention mechanisms for action prediction on high-end desktops, edge devices, robots, and UAVs. The 2D+1 convolutions fuse spatial features from each frame, while the 3D convolutions handle spatio-temporal information simultaneously. The attention mechanism consists of cross-attention and self-attention, providing spatio-temporal representations with linear computational complexity. Cross-attention adjusts input sequences to a new sequence size as needed for computational efficiency, while self-attention is a fundamental component of transformers. AZTR is developed to effectively utilize available hardware resources, including high-end desktops and low-power devices, for action recognition. It aims to optimize the model's performance on both types of hardware platforms [73].

Graph Convolutional Networks (GCNs) [74] use skeleton data for action recognition, including single-person actions and human-to-human interactions. Message passing relies on a dynamic adjacency matrix, facilitating meaningful propagation. A frame

importance calculation module reduces traditional convolution impact. Multidimensional features capture spatial-temporal relationships, measuring similarity with different metrics. Hand-crafted spatial and temporal features like 3D coordinates, bone information, and velocity assess relative limb positions. Graph diffusion connects joints within the body and between interacting individuals, guiding network learning and accelerating convergence. Dynamic convolution extends message transmission, while a searching scheme suppresses noise and enables adaptive segmentation. Feature-specific similarity metrics guide message propagation. Dynamic Temporal Convolution (Dynamic TCN) calculates frame importance. The network combines ST-GCN, diffusion-guided GCN, and dynamic TCN, with extracted features classified using a standard classifier. Graph diffusion and dynamic convolution improve message propagation and frame importance. The model achieves high-precision recognition by utilizing multidimensional features and appropriate similarity metrics. It accurately predicts human-to-human interactions by integrating graph diffusion, dynamic convolution, and feature-based similarity analysis [74].

The 3D Convolutional Neural Network (3DCNN) [75] architecture is commonly utilized for video classification, particularly in complex and medical images. It utilizes 3DCNN layers, MaxPooling3D, batch normalization, dense layers, and a flatten layer to achieve accurate results. The 3DCNN excels in analyzing moving 3D images by considering temporal object positions [75].

A comprehensive review of aerial surveillance tasks in computer vision and pattern recognition tasks is presented in [76]. The authors perform a comprehensive analysis of the present status of aerial surveillance, employing drones, UAVs, and other airborne platforms. The main emphasis is on detecting, identifying, tracking, re-identifying, and analyzing human activity from aerial perspectives. This paper discusses the unique challenges that arise in performing these tasks compared to ground-based settings for each specific objective. Additionally, it thoroughly examines publicly available aerial datasets and investigates the approaches proposed in existing literature to tackle these challenges. The authors propose multiple methods for aerial action recognition, including single-frame classification utilizing established 2D networks such as ResNet, InceptionNet, MobileNet, and DenseNet. They also explore fusion approaches, LSTM-based techniques, and two-



stream CNNs that combine appearance and motion, incorporating pose estimation methods. The effectiveness of 3D CNNs, particularly the widely used I3D architecture, is examined for aerial action recognition. Furthermore, the paper discusses the exploration of alternative models such as C3D, P3D, and TRN, as well as the application of lightweight MobileNet with self-attention mechanisms in transformers. The authors address various challenges in aerial action recognition, such as low-resolution imagery, limited data availability, diverse viewing angles, and the use of fish-eye cameras. Overall, this paper presents a comprehensive analysis of activity recognition in aerial surveillance tasks, making remarkable contributions in the field of computer vision and pattern recognition [76].

An Attention Interactive Graph Convolutional Network (AIGCN) [77] is utilized for efficient extraction of spatial-temporal relationships in interaction recognition. Skeletons are encoded using two semantic vectors to derive both dynamic and static adjacency matrices. An Interactive Attention Encoding GCN (IAE-GCN) module is employed to extract interactive spatial structures, capturing the influence of joints on each other through joint position encoding. The IAE-GCN model utilizes static and dynamic graph convolutions, represented by a Mirror graph (static adjacency matrix) and an Attention-encoding graph (dynamic adjacency matrix) that dynamically symbolizes connections using joints' semantic codes and self-attention. Furthermore, an Interactive-Attention Mask TCN (IAM-TCN) is used to extract temporal interactive features by representing the temporal attentional excitation signal among different joints over time. The attention mechanism is utilized to generate a mask that activates features at various times from the past or future. The proposed model achieves state-of-the-art (SOTA) performance on interaction datasets namely SBU-Interaction, NTU-RGB+D and NTU-RGB+D 120 [77].

In activity recognition, skeleton-based methods have shown remarkable progress, but there is a lack of focus on action recognition from UAV views, and traditional models face challenges due to drastic changes in viewpoints. To address this, two novel methods, namely Channel-wise Topology Refinement Graph Convolution Network (CTR-GCN) and Semantics Guided Network (SGN), have been introduced in [78]. These methods leverage

graph-based modeling to capture relationships between joints and their neighbors, with the adjacency matrix representing the correlations between joints. CTR-GCN dynamically infers skeleton topology using pairwise relationships between joints, while SGN incorporates high-level semantics like joint types and frame index for enhanced feature representation. The fusion of CTR-GCN and SGN prediction scores is achieved through a weighted sum, ensuring that the two methods complement each other without interference. Overall, these approaches offer robustness to complex backgrounds, consume fewer computational resources, and have shorter training periods compared to RGB-based or video stream processing methods [78].

To address the limitation of real data and annotations, paper [79] proposes a framework that utilizes game action videos and GAN-generated features to enhance activity recognition in real-time aerial videos. The GTA and FIFA gaming engines were chosen for their ability to provide highly realistic simulations, accurately depicting real-world environments, objects, and human actions. GAN-generated video examples are generated using conditional Wasserstein GAN with gradient penalty (WCGAN-GP) to produce discriminative features that mimic real aerial data. These features are then used to train soft-max classifiers for action classification. Disjoint multitask learning is introduced to handle the action labels in both the game and real datasets, allowing the model to simultaneously learn multiple related tasks. Deep features are extracted from real aerial and game videos using 3D convolutional neural networks, namely MNF-3D, I3D, and ResNet3D models. The training process involves branches for classification, where available real and game labels are used to train corresponding branches, and labels for game data and real data are inferred from the predictions of other branches. By combining real, GAN-generated, and game videos within this framework, the proposed approach tackles the constraints of traditional multitask learning and facilitates the effective classification of aerial videos [79].

A deeply coupled ConvNet with RGB frames and bi-directional LSTM (Bi-LSTM) is introduced in [80]. At the bottom layer, a CNN model trained with a single dynamic motion image (DMI) is employed. The RGB frames refine the pre-trained CNN features via end-to-end learning, whereas the dynamic image stream is fine-tuned to capture

temporal information. Both streams' features are fused using late fusion techniques, with maximum fusion achieving high accuracy. The model incorporates the Inception-v3 deep architecture and utilizes Bi-LSTM for sequential data processing. The concept of compact Dynamic Motion Image (DMIs) is introduced, capturing long-term dynamics and motion patterns. The two-stream architecture is connected in parallel, with scores fused at the decision level. The Inception-v3 architecture includes input blocks, Inception Modules, grid size reduction blocks, auxiliary classifiers, and output blocks. Bi-LSTM addresses sequential data analysis, and the proposed approach samples frames from RGB videos and fine-tunes the CNN architecture. DMIs represent the entire video sequence using rank pooling, and approximate rank pooling is introduced for efficiency. Late fusion techniques combine features from different channels, and maximum fusion is applied in this approach. Overall, the proposed method achieves excellent action recognition results [80].

For interaction recognition among between two individuals, a GCN [81] is utilized by incorporating knowledge graphs. Two types of graphs are proposed: a knowledge-given graph and a knowledge learned graph. The knowledge-given graph captures direct correlations between joints of two persons based on common sense knowledge, while the knowledge-learned graph adaptively learns connections between joints from the dataset. These graphs are combined with a naturally connected graph to form the knowledge embedded graph convolution network (K-GCN), which extracts discriminative features for interaction recognition. The K-GCN consists of multiple graph convolution blocks that perform spatial and temporal convolutions. Additionally, a multi-level scheme for modelling joint-level and part level information simultaneously is proposed, that enhances the model's performance. It also explores a two-stream network that combines joint-level and part-level inputs using a weighted fusion approach [81].

An improved human action recognition through the combination of a 3D CNN, a residual structure, and an attention mechanism is proposed in [82]. The researchers introduced a shallow feature extraction module and a customized deep feature extraction module to enhance the extraction of temporal and spatio-temporal features. A 3D spatio-temporal attention mechanism is proposed to capture global action features by considering the inter-frame relationship. The fusion of the residual structure and attention mechanism

led to the development of the Attention Residual 3D Network (AR3D), with two fusion strategies. These strategies integrated the attention mechanism either into the identity transformation connection or the output of the residual structure. The proposed models aimed to address limitations in existing 3D CNN approaches and improve the performance of human action recognition systems [82].

Dual-stream HAR model as proposed in [83] combines the advantages of human pose information and scene images. By utilizing spatio-temporal graph convolution, the model extracts motion features from human skeleton data, which provides robustness to illumination and scene changes. Moreover, a scene recognition model employing a video-level consensus strategy processes the visual scene information captured within the video frames. The fusion of skeleton-based motion features and scene images enhances the accuracy and robustness of action recognition. This comprehensive approach leverages the strengths of both modalities to overcome limitations and improve the comprehension and recognition of human activities, providing a promising solution in activity recognition [83].

A novel architecture called Hierarchical Long Short-Term Concurrent Memory (H-LSTCM) [84] is proposed to recognize human interactions. It utilizes Single-Person LSTMs to model individual dynamics and a Concurrent LSTM to capture inter-related dynamics among multiple persons. The Concurrent LSTM comprises of sub-memory units, a cell gate, and a co-memory cell, allowing selective integration and storage of motion information. A loss function is employed for training the model and can be optimized using Backpropagation Through Time (BPTT) algorithm [84].

In vision-based action recognition, recent models are based on Vision Transformer. A survey of transformer-based models is presented in paper [52]. The paper provides a comprehensive review on four fundamental action recognition tasks (classification, detection, segmentation, and anticipation), data modalities (uni-modal and multimodal data), and three types of dominant architectures (CNN based Transformer, Transformer based CNN and Pure Transformer models). RNNs tend to forget earlier inputs in long sequences, while Transformers address this issue by using attention mechanisms to capture relevant information of the sequence, allowing better handling of long-range dependencies

without recurrence. CNN-Integrated Transformers uses 2D and 3D features to extract meaningful spatio-temporal, convolution has proven ability to learn visual features, these models are easy to replicate and integrate in other architectures. CNNs face challenges in effectively handling positional and temporal information in videos, particularly when objects of interest exhibit variations in rotation and scaling. This weakness affects subsequent processes in recognition tasks. Moreover, during striding, CNNs can lose crucial compositional and positional details, potentially leading to misclassification of actions. Although data augmentation techniques like flipping or rotating training data help mitigate these issues to some extent, they fall short in addressing the complexities of real-world scenarios, leaving CNN-based approaches vulnerable to adversarial attacks. Additionally, CNNs can be slower due to operations such as max pooling. Pure Transformer-based architectures, like Vision Transformers (ViTs), are becoming popular in action recognition tasks. They're good at capturing complex patterns in images and focusing on important details using attention maps. ViTs handle positional and temporal information well and are robust against image distortions. However, they're harder to train than CNNs because they lack some built-in shortcuts. While ViTs often perform very well, they may not always beat CNNs, especially on simpler tasks, and they can be challenging to use on devices with limited resources. CNNs have been around longer and have proven effective in many situations, while ViTs are still being explored [52].

More recent studies are utilizing Transformers for pose-based activity recognition. An extensive review of skeleton-based transformers architectures is presented in [85]. The paper discusses the traditional methods based on prior knowledge and the deep learning models which mainly includes RNN-based methods, CNN-based methods, GCN-based methods, and Transformer-based methods. Graph convolution and Transformer are popular technologies for action recognition, known for their strong capabilities in interacting with skeleton data and extracting topological structures. Many methods improve traditional structures like ST-GCN by adding self-attention mechanisms or correlation matrix auxiliary networks for better spatio-temporal processing. The paper explores Transformer-style graph convolution and standard Transformer methods for skeleton action recognition, as well as unsupervised and transfer learning techniques. The survey summarizes the cutting-edge skeleton transformers, proposes a classification

taxonomy, and discusses the introduction, innovation, and improvement of related papers. Moreover, challenges and future research directions in transformer-based skeleton action recognition are also discussed [85].

Both sensor-based and vision-based approaches possess distinct strengths and weaknesses, and the selection of approach relies on the requirements of the application and constraints. Sensor-based approaches are accurate and robust to environmental conditions and occlusions however they are expensive and extracting desired features from raw sensor data is a tedious and challenging task. On the other hand, vision-based approaches are versatile and cost-effective. However, they can be affected by environmental conditions like noise, lightning, cluttered scenes, and occlusions. Among all approaches, pose-based approaches are often preferred over traditional and other deep learning methods because they capture rich information about the actions being performed allowing accurate recognition. Pose-based methods focus on body's skeleton and movements patterns making them robust to environmental factors as they filter-out the irrelevant cluttered background and occlusions. Moreover, they required standard cameras or depth sensors which are accessible and cheap hence making them cost-effective.

Among vision-based approaches, shallow approaches have their unique pros and cons. Direct approaches are straightforward to implement but often exhibit limited performance. Sequential models have evolved by adding temporal dimension however they are sensitive to noises. Space-time methods are good at capturing spatio-temporal structures, but they are limited to datasets of small sizes. Part-based models learn the patterns at a finer level, but its limitations lie in the use of small size datasets. Manifold methods have shown promising results, but they rely on human-silhouettes. The limitation of mid-level feature models is their requirement of large annotations. Feature fusion methods enhance accuracy by fusing two or more feature extraction methods, but they are computationally expensive. Alternatively, deep learning models in vision-based recognition have unified frameworks that automatically extract relevant features and classify the predefined class. Among them, space-time models have short temporal intervals, multi-stream networks are limited by complexity, data synchronization challenges, feature fusion difficulties, data imbalances, and interpretability issues. Whereas

hybrid approaches have shown greater accuracy, however they are computationally expensive, difficult to fine-tune and are not suitable to employ in real-time applications.

In recent research's, Vision Transformers (ViTs) are becoming popular, because of their capability to capture complex patterns in images and effectively handle long-range dependencies without recurrence. Unlike traditional recurrent neural networks (RNNs), which tend to forget earlier inputs in long sequences, Transformers utilize attention mechanisms to focus on relevant information, enabling them to handle long sequences more efficiently. Recent models based on Vision Transformers integrate both 2D and 3D features to extract meaningful spatio-temporal information, allowing them to better represent temporal dynamics in videos. While CNN-based approaches have shown effectiveness in learning visual features, they may struggle with positional and temporal information and can be vulnerable to adversarial attacks. Pure Transformer-based architectures, such as Vision Transformers (ViTs), excel in capturing spatio-temporal information. However, ViTs may be more challenging to train compared to CNNs and may not always outperform them, especially on simpler tasks. Despite their advantages, ViTs may also pose challenges in deployment on devices with limited computational resources. Both CNNs and ViTs have their unique pros and cons, ViTs have shown promising results compared to CNNs.

Skeleton-based transformers are preferred over other deep learning models for action recognition due to their robustness, compactness, and noise immunity. Skeleton data is lightweight and ideal for resource-constrained environments. Graph Convolutional Networks (GCNs) have traditionally been used for skeleton-based action recognition particularly for human-human interactions. However, traditional GCNs and recent variants of GCNs have complex architectures making them computationally expensive and are not suitable for real-time applications. Moreover, skeleton-based transformers have shown promising results in terms of computational complexity, performance, and real-time applications. Hence, in this study, we proposed a skeleton-based action transformer model which is lightweight, computationally effective for recognizing solo actions and human-to-human interactions in aerial videos.

**Table 2.1:** Summary of literature review.

Reference	Input	Description	Method	Dataset/Results	Remarks	Limitations	Future work
[70]	Keypoints Data	A light-weight Transformer model designed for solo-actions recognition in aerial videos.	Action Transformer	Drone-Action: Accuracy: 75.42%	Efficient method for recognizing solo actions in aerial videos	Single-person action recognition	Nil
[71]	Keypoints Data	A novel graph convolution method, capturing inter-body and intra-body correlations between two individuals to recognize human-to-human interactions.	2P-GCN (Baseline Model:ST-GCN)	Accuracy: SBU: 98.90%, NTU-RGB+D: 97.05% (Xsub)& 98.80% (Xview), NTU-RGB+D 60: 93.47% (Xsub)& 93.73% (Xview)	A unified framework that recognizes human-to-human interactions effectively	Two-person recognition	Extension to human-object interactions and group activity recognition
[72]	Video frames	Mutual information extracts useful temporal features for human action focus. Frame sampling selects informative frames for training. Integrated with X3D model for improved action prediction.	Integrated X3D + Softmax	Improved Top-1 Accuracy: UAV Human/18.9%, Drone Action/7.3%, NEC Drones/7.16%	Flexible method handles occlusion and viewpoint changes, Novel sampling method to extract most informative frames	No spatial relationship b/w actor and background, Single-person action recognition	Extend to multi-human and multi-actions videos
[73]	Video frames	A novel architecture for action prediction on different platforms, using auto-zoom and temporal reasoning algorithms.	AZTR: Movinets, MobileNet V2(light-weight platforms) & X3D-M, Cascaded Mask RCNN (high-end platforms) as backbone architecture & localization respectively. + Softmax	Improved Top-1 Accuracy: RoCoG-v2/6.1-7.4%, UAV-Human/8.3-10.4%, Drone Action/3.2%	Model run on edge/mobile platforms, Robust auto-zoom algorithm for key spatial feature extraction, Temporal reasoning algorithm for key temporal feature extraction making accurate predictions	Performance depends on localization method, Single-person action recognition	Extend to multi-human, multi-action and dynamic environmental conditions
[74]	Two-person skeleton	A novel GCN that utilizes	Baseline model of ST-GCN, combin	Accuracy: NTU-RGB+D 60/88.6%, NTU-	Graph Diffusion embedded into	Model confusion in Inter-	Extend to group activity recognition,



	sequence graph; Joint, velocity and bones features	graph diffusion mechanism to recognize human-to-human interactions in videos.	ed with Diffusion guided GCN and Dynamic TCN+ Softmax	RGB+D 120/85.2%, Kinetics-Skeleton 400/33.8%, SBU-Interaction/-	GCNs, Dynamic construction of adjacency matrix, Frame importance calculation module for dynamic convolution, Multidimensional features extraction and distinct metrics for similarity measure in motions.	class actions	Need of efficient pattern extraction method
[75]	Video frames	A modified 3DCNN to classify human activities	Modified 3DCNN + Softmax	Accuracy: UCF YouTube Action dataset/85.2%, UCF-101/79.9%	Non-standard behavior recognition in public spaces. Satisfactory overall accuracy: 79.9% for UCF101 dataset, 85.2% for UCF YouTube Action dataset.	Model effectiveness may be limited beyond non-standard behavior in public places.	Nil
[76]	Video frames & Posture data	Survey of action recognition including activity types, challenges and approaches in each activity domain	Multiple techniques including single-frame classification, Two-stream CNNs and 3D CNNs + Softmax for action recognition, Other activity types have multiple classifiers	Multiple datasets for different activity types, along with various performance metrics and results for each approach.	The paper provides a survey of activity types, its challenges, datasets and methods to recognize each activity.	The paper discusses open issues from two perspectives; Data & Model development and model deployment	Multiple future work is suggested to address limitations in data, model development, and model deployment.
[77]	Two Persons Skeleton Data	AIGCN captures spatial-temporal relationships for interaction recognition using skeleton encoding, dynamic/static adjacency matrices, IAE-GCN for spatial structures, IAM-TCN for temporal features.	AIGCN: IAE-GCN-Spatial features, IAM-TCN-Temporal features + Softmax	Accuracy: SBU Interaction/99.1%, NTU-RGB+D/95.34%, NTU-RGB+D 120/90.71%	Skeleton encoding, dynamic/static adjacency using IAE-GCN, Enhanced temporal coactions using IAM-TCN, AIGCN achieve SOTA performance on SBU-Interaction, NTU-RGB+D and NTU-RGB+D 120.	Nil	Nil
[78]	Skeleton data	Two novel models, CTR-GCN and SGN, for action	CTR-GCN & SGN + FCN followed by Argmax function	Accuracy: UAV-Human/45.84%	Dynamic topology modeling using CTR-GCN,	Low accuracy on NTU-RGB+D dataset,	Exploration and implementation of robust models to

		recognition from UAV views, leveraging graph-based modeling and incorporating high-level semantics for enhanced feature representation.			Semantic feature integration using SGN, Combined strengths of both models using Fusion module.	Lower performance compared to regular approaches in UAV videos	address UAV challenges and recognize actions accurately.
[79]	(Few real aerial videos + Game videos) OR (Few real aerial videos + GAN generated videos); Video deep visual feature	A framework that combines game action videos and GAN-generated features to improve human action recognition in real-world aerial videos when few aerial videos are available.	Video features: MNF-3D, I3D & ResNet3D; Conditional Wasserstein GAN; Disjoint Multitask learning framework + Softmax	Accuracy: UCF-ARG-Aerial/35.9%, 16.3%, 15.1%; YouTube-Aerial/68.2%, 67.0%, 58.6%	Useful approach for training with few aerial videos, First method that utilize aerial game videos, Use of Disjoint Multitask Learning framework to learn an accurate classifier, Two novel datasets proposed	Game-videos are specific actions biased, Difficult to generate GAN based videos for all actions	Attention-based spatio-temporal localization, Low-power algorithms
[80]	Video frames	A two-stream spatio-temporal network followed by a late fusion technique to recognize actions in videos.	Deeply coupled ConvNet: Two streams network of spatial (pre-trained with Inception-v3 architecture followed by Bi-LSTM) & temporal (pre-trained with Inception-v3 architecture) features followed by late fusion technique (Maximum fusion); Upper stream input: RGB images; Bottom stream input: Dynamic motion image + Softmax followed by late fusion (maximum fusion) technique	SBU Interaction/98.70%, MIVIA Action/99.41%, MSR Action Pair/98.30%, MSR Daily Activity/98.37%.	SOTA accuracy as compared to other deep architectures	Nil	To utilize depth & skeleton modality in the model; To add complex multi-view classes in the training data
[81]	Skeleton data	A novel GCN named K-GCN is proposed	K-GCN & Multi-level K-GCN + K-	Accuracy: K-GCN: SBU/96.55%,	Proposed two novel graphs (Knowledge given graph &	Nil	Nil

		along with Multi-level K-GCN for two-person action recognition.	GCN: Softmax; Multi-level K-GCN: Softmax followed by weighted fusion method	NTU-RGB+D/92.70% Multi-level K-GCN: SBU/97.2%, NTU-RGB+D/92.70%	Knowledge learned graph); Proposed two novel networks - Knowledge embedded graph convolution graphs (K-GCN & Multi-level K-GCN)		
[82]	RGB Images/ Video frames	Novel AR3D models combines 3D CNN, residual structure, and attention mechanism for enhanced feature extraction and representation to address limitations of existing 3D CNNs	R3D: [3D SFE+DFE (3D Residual Modules)] AR3D: - AR3D_V1: R3D with 3D Attention fused in identify transformation connection within 3D Residual structure of DFE AR3D_V2: R3D with 3D Attention fused after the 3D Residual Modules of DFE + Softmax	Accuracy: UCF-101/87.89, 88.39, 89.28, HMDB51/50.27, 51.53, 52.51	Separate extraction of shallow & deep features in R3D, 3D convolution decoupled residual module, 3D attention-mechanism for handling background changes, Two novel AR3D fusion models, High-accuracy without pre-training on large-datasets	Nil	Nil
[83]	Video frames	A dual-stream model combining pose data and scene image information for improved action recognition through spatio-temporal graph convolution and fusion of motion and visual features.	Dual stream network: - Skeleton stream: Human skeleton extraction+ ST-GCN + Max.Pool + FCN, Image stream: Random frames sampling + ResNet101 + Segmental consensus; Late fusion (weighted & maximum) of both streams + Softmax	Accuracy: UCF-101/96.6%, HMDB51/73.1%	Model incorporates skeleton and scene images modalities, Dual-stream fusion of skeleton stream & image/scene stream	Nil	To extend & enhance the proposed model incorporating multi-modalities streams & overcome occlusion changes
[84]	Video frames	A novel Hierarchical Long Short-Term Concurrent Memory (H-LSTCM) model to capture long-term inter-related dynamics	H-LSTCM: Single-person LSTMs + Co-LSTM units + Softmax	Accuracy: - (H-LSTCM): BIT/94.03, UT-Interaction/98.33%, CAD/83.75%; (E-H-LSTCM): VD/88.4%	Proposed a novel H-LSTCM model, Proposed a novel Co-LSTM unit	Nil	To extend the model by incorporating hypergraph architecture

		among a group of persons for human interactions recognition.					
[52]	Video frames	A survey of Vision Transformer based architectures for action recognition	CNN based Transformer, Transformer based CNN and Pure Transformer model	Top-1 Accuracy, F1 Score, mAP for top performing models on popular datasets	A comprehensive review on transformers-based models	Each models pros and cons are discussed	To enhance motion modeling in pure-transformer based models, To design a multimodel transformer model, To advance the training strategies
[85]	Pose data	A survey of skeleton-based transformer architectures for recognition actions in videos	Variants of Transformers utilizing GCNs	Accuracies reported on various models using NTU-RGB+D 60, NTU-RGB+D 120, Kinetics, NW-UCLA and UAV Human	Novel models utilizing Transformers with GCNs	Each model pros and cons are discussed.	Nil

## 2.2 Available Datasets for Activity Recognition

There are number of actions which are categorized in gestures, single-person/solo actions, interactions (human-human & human-object), group/crowd analysis, behaviors and events analysis. The target of this research study is to recognize human interactions (human-to-human) and solo actions. So, from literature review, for interactions and solo actions recognition, there are number of datasets which were captured using indoor cameras, outdoor ground truth cameras, rooftops cameras and very limited datasets were captured using UAVs. Table 2.2 shows a few human-human interactions, human-object interactions and solo actions datasets which have been utilized in number of interactions and solo actions recognition approaches.

**Table 2.2:** Publicly available interactions-solo actions datasets.

<b>Dataset (Year) [Ref]</b>	<b>No. of Actions</b>	<b>Modality</b>
<b>SBU</b> (2012) [59]	8	RGB+D+S
<b>K3HI</b> (2013) [86]	8	S
<b>CAD-60</b> (2011) [87]	12	RGB+D+S
<b>CAD-120</b> (2013) [87]	20	RGB+D+S
<b>PKU-MMD</b> (2017) [88]	51	RGB+D+S+IR
<b>NTU-RGB+D/NTU-RGB+D 60</b> (2016) [89]	60	RGB+D+S+IR
<b>NTU+RGB+D 120</b> (2019) [90]	120	RGB+D+S+IR
<b>Kinetics/Kinetics-400</b> (2017) [91]	400	RGB
<b>Kinetics-600</b> (2018) [92]	600	RGB
<b>Kinetics-700</b> (2019) [93]	700	RGB
<b>MSR Action3D</b> (2010) [94]	20	D+S
<b>BEHAVE</b> (2004) [95]	Unspecified	RGB+D+SMPL
<b>TV Human Interaction</b> (2010) [96]	4	RGB
<b>UT Interaction</b> (2010) [1]	240+	RGB
<b>CAVIAR</b> (2012) [97]	7	RGB
<b>CASIA Action</b> (2013) [98]	15	RGB
<b>Hollywood</b> (2009) [99]	8	RGB
<b>Hollywood2</b> (2009) [100]	12	RGB
<b>HMDB51</b> (2011) [101]	51	RGB
<b>UCF 101</b> (2013) [102]	101	RGB
<b>J-HMDB</b> (2013) [103]	21	Joints, Pose, Dense Opticalflow
<b>UCF 50</b> (2013) [104]	50	RGB
<b>CASIA C</b> (2006) [105]	1	IR
<b>KTH</b> (2004) [106]	6	RGB+Grayscale
<b>Drive&amp;Act</b> (2019) [107]	83	RGB+D+IR+3DP
<b>UAV Human</b> (2021) [108]	119	Fe+Nv+RGB+IR+D+J
<b>DHU Night</b> (2021) [109]	4	IR
<b>LTIR</b> (2015) [110]	20	IR
<b>AAU-PD-T</b> (2020) [111]	9	IR
<b>IITR Infrared Action Recognition-(IITR-IAR)</b> (2019) [112]	21	IR+SDFDI+SSDI
<b>IITR+Thermal Simulated Fall-(TSF)</b> (2022) [113]	38	IR

<b>IIAR-30</b> (2022) [114]	8	IR
-----------------------------	---	----

The abbreviations for the modalities listed in Table 2.2 are RGB-Color, D-Depth, S-Skeleton, IR-Infrared, HM-Human Mask, SMPL-Skinned Multi Person Linear fits, 3DP-3D Pose, Fe-Fisheye, Nv-Night vision, J-Joint, SDFDI- Stacked Dense Flow Difference Image, SSDI- Stacked Saliency Difference Image.

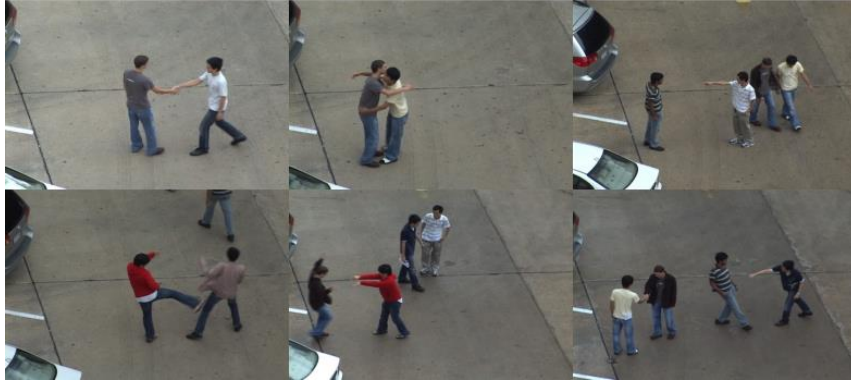
As a part of our investigation study is to recognize solo actions as well as human-human interactions in grayscale videos, we will utilize the (RGB modality) videos captured from UAVs but in our study, we will convert the RGB videos to grayscale videos as per the requirements from the end user who has funded the project. Some relevant aerial datasets are listed in Table 2.3.

**Table 2.3:** Relevant aerial datasets.

<b>Dataset (Year) [Ref]</b>	<b>No. of Actions</b>	<b>Modality</b>
<b>UT-Interaction</b> (2010) [1]	240+	RGB
<b>Mini-Drone</b> (2015) [115]	38	RGB
<b>Drone-Action</b> (2019) [2]	13	RGB
<b>UCF ARG &amp; UCF Aerial Action</b> (2009) [79]	10+9	RGB
<b>Okutama-Action</b> (2017) [116]	12	RGB
<b>Aeriform in-action</b> (2023) [117]	13	RGB

Among the datasets listed in Table 2.3, UT-Interaction dataset [1] is particularly suitable for recognizing human-human interactions due to its resolution of 720x480 pixels. The videos in this dataset are recorded using a low altitude camera with static camera movement, as depicted in Figure 2.1. Similarly, the Mini-drone dataset [115],

with a resolution of 1920x1080 pixels, features videos recorded using a low altitude camera (Figure 2.2), which also exhibits dynamic camera movement. Both datasets are relevant for the task of human-human interactions recognition.



**Figure 2.1:** Image taken from UT-Interaction dataset [1].



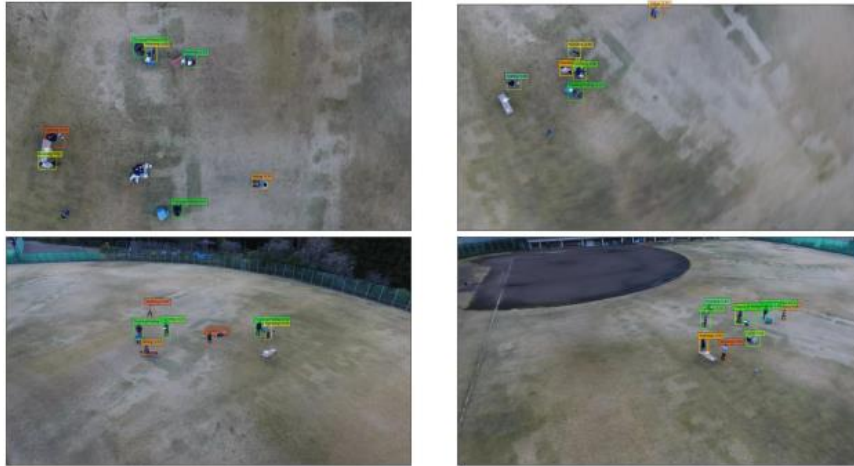
**Figure 2.2:** Image taken from Mini-drone dataset [115]

While the Drone-action dataset [2] (Figure 2.4) also possesses a resolution of 1920x1080 pixels and includes low altitude camera footage, this dataset is well-suited for solo-actions recognition.



**Figure 2.3:** Image taken from Drone-action dataset [2].

On the other hand, the Okutama-action dataset [116] boasts a high resolution of 4k. However, it involves high altitude UAV footage (Figure 2.5), which is required in our study.



**Figure 2.4:** Image taken from Okutama dataset [116].

The most recent dataset of captured unmanned aerial vehicle (UAV) footage, demonstrated in Figure 2.6, is referred to as "Aeriform in-action" [117]. It possesses a high resolution of 4k and encompasses 14 distinct action classes, including both solo activities and interactions. However, the drawback of this dataset lies in its restricted accessibility, as it is not publicly accessible.



**Figure 2.5:** Image taken from Aeriform in-action dataset [117].



Among all the datasets, the UCF datasets [79] suffer from poor resolution and high-altitude camera shots with noticeable jitters (Figure 2.7), making action recognition a complex and challenging task.



**Figure 2.6:** Image taken from UCF Aerial Action dataset [79].

For this research study, the most appropriate datasets are the Ut-Interaction dataset [1], which is ideal for recognizing human-human interactions, and the Drone-action dataset [2], which is well-suited for recognizing solo actions. These datasets are particularly advantageous due to their low-altitude videos and high resolution, facilitating easy detection of individuals.

### **2.3 Scope of the Research Study**

The agreed scope of the thesis is to focus on human activity recognition, where activity could involve solo actions or interactions. The datasets to be considered for evaluation of human-human interactions and solo actions recognition are UT-Interaction [1] and Drone-Action [2]. It is important to note that the low-altitude aerial videos are being considered containing moving target(s) that can be reliably detected and their pose reliably extracted; hence the assumption is that the image resolution is satisfactory. The solution therefore involves the use of extracted pose information because different activities, in principle, are distinguishable based on target's bodily movements. The detection/tracking problem is assumed to have been solved a priori for this study. As part of the investigation, the evaluation will be performed on the data that is converted from RGB (24-bit) to grayscale (8-bit) as per the requirements from the funding body. It is also relevant to mention that, recently, some approaches (including transformer-based networks) have shown encouraging performance for human activity recognition (but not with aerial videos) for solo actions; however, there appears a lot of room to devise effective methods for aerial human activity recognition, particularly to recognize and classify 'interactions' that is the main goal of this research study.

## 2.4 Research Gap Analysis and Contributions of the Proposed Methodology

The current state-of-the-art methods have made significant progress, however there is a lack of generic methods capable of recognizing both solo actions as well as interactions in aerial videos. Due to diverse interactions classes and challenges in videos, limited methods are proposed to recognize them, Additionally, existing methods can be improved in terms of computational performance to make them more suitable for real-time applications. Specifically, for low altitude grayscale aerial videos, no generic method is proposed that is computationally efficient for real-time applications.

The contributions of our research study are as follows:

1. We have proposed a state-of-the-art skeletal-based transformer 'InterAcT' model capable of recognizing solo actions and human-human interactions in aerial grayscale videos.
2. We have performed experiments to select optimized parameters of the proposed model.
3. We have proposed a novel framework which is computationally efficient and accurate for real-time applications.
4. We have performed performance evaluation and comparative analysis of our model with other state-of-the-art methods.

## CHAPTER 3: PROPOSED METHODOLOGY

Human Action Recognition (HAR) is an interesting research study in computer vision and pattern recognition and it encompasses the identification and classification of human activities. Previous HAR studies generally utilized datasets featuring extended temporal durations, treating HAR as a post-processing task, categorizing intricate and prolonged human actions by leveraging past and future information. In contrast, Action Transformer [118] focuses on short-time HAR, continuously classifying actions within brief past time steps, typically up to a second. This approach is essential for real-time applications. The Action Transformer (AcT), drawing inspiration from the straightforward and architecture-independent design of the Vision Transformer [119]. The Transformer architecture [120] stands out as a significant breakthrough in the field of natural language processing (NLP) over the past few years. Furthermore, multi-head self-attention has demonstrated its effectiveness across various tasks beyond NLP, including applications in image classification [121], image super-resolution [122] and speech recognition [123]. Additionally, refined editions of the Transformer have emerged, tailored for real-time and embedded applications [124], demonstrating the adaptability of this architecture for Edge AI objectives. In recent times, numerous models aimed at enhancing the accuracy of Human Action Recognition have suggested incorporating attention mechanisms within convolutional and recurrent blocks. Nevertheless, solutions exclusively reliant on self-attention blocks for this task have not yet been explored. AcT employs a purely Transformer encoder-based architecture for action recognition, resulting in an accurate and low-latency model suitable for real-time applications.

The choice for utilizing Transformer models over other deep neural networks such as Convolutional Neural Networks (CNNs) [125], Recurrent Neural Networks (RNNs) that includes Long Short-Term Memory (LSTMs) and Gated Recurrent Units (GRUs) [59], and Graph Convolutional Networks (GCNs) [126] rests upon several distinct advantages. 2D CNNs are effective in extracting spatial features, Transformers can complement CNNs by capturing not just spatial but also temporal dependencies. For temporal dependencies, there are 3D CNNs, but they are computationally expensive and are not suitable for inference.

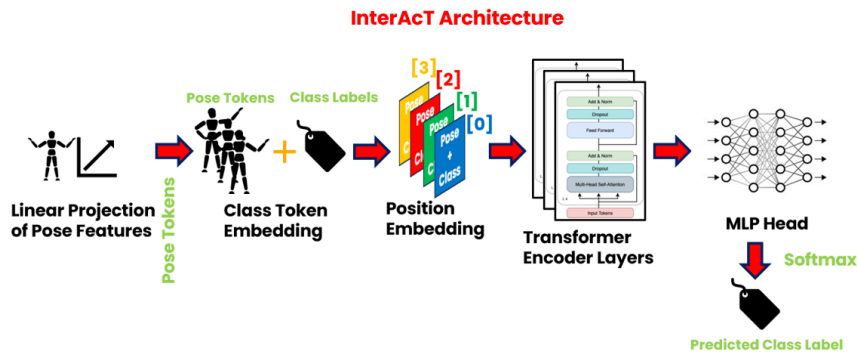
RNN-based models that process data sequentially, Transformers, with their attention mechanism, offer a significant edge in processing sequential data, enabling efficient capture of long-range dependencies without suffering from the vanishing gradient problem seen in recurrent networks. Unlike RNNs that process sequences sequentially, the parallelization capability of Transformers accelerates training and inference, making them well-suited for large datasets. Moreover, their modular structure facilitates scalability, adaptability to various tasks, and the availability of pre-trained models for effective transfer learning. With more interpretable representations due to the attention mechanism, Transformers stands out as a compelling choice for tasks reliant on sequential data processing, leveraging their efficiency and robust performance.

The Action Transformer (AcT) [118] is a state-of-the-art (SOTA) model of Transformer family that exclusively leverages keypoints data in recognizing actions classes. AcT utilizes a Transformer encoder-based architecture for action recognition, yielding an accurate and low-latency model ideal for real-time applications. It possesses the robust capability to identify action classes even when the dataset is limited in sample size. Offering four variants—micro, small, base, and large—the model exhibits a spectrum of architectures, ranging from simpler to more complex designs. Each architectural variant bears distinct characteristics in comprehending data patterns. Notably, the model's complexity directly correlates with the requirement for more extensive data to effectively learn underlying patterns. Compared to alternative deep neural networks, the Action Transformer demonstrates significant potential and yields promising outcomes in its applications of activity recognition.

To the best of our knowledge, the use of AcT has been used for recognition of ‘solo actions’ (that involve just a single person), but not well explored for recognizing ‘interactions’ (involving more than one person e.g., hand shaking, hitting someone, etc.) along with solo actions, which is envisaged to be the expected contribution of this work.

### 3.1 Description of Proposed InterAcT Framework

Inspired from the Action Transformer (AcT) [118] model, InterAcT model incorporates a Transformer-encoder based architecture to process video input sequences. Initially, the model pre-processes the input data from a multi-person 2D pose estimation network, generating 2D poses for each frame. These poses are subsequently utilized to create a 1D sequence of token embeddings. The Transformer architecture then receives these token embeddings and processes them using a standard Transformer encoder. Through a sequence of layers, which incorporate self-attention mechanisms and feed-forward blocks, the model transforms and processes the input information. Notably, the model leverages a class token to facilitate self-attention aggregation, enabling a high-dimensional representation to discern different action classes. The resulting network can predict actions for multiple individuals within a video stream with high accuracy. The transformer architecture within the model employs a multi-layer design with alternating self-attention and feed-forward blocks, optimizing the sequence of operations for efficient processing. Additionally, the network is designed with a focus on reducing hyperparameters, scaling the model's dimension, and facilitates optimal performance with reduced complexity and parameters, making it adaptable and effective for real-time applications. The proposed InterAcT architecture as explained above is illustrated in Figure 3.1 below. To understand the architecture, it's important to understand each element in the pipeline. There are six components in the architecture that includes Linear Projection of Features, Class Token Embedding, Position Embedding, Transformer Encoder Layers, MLP Head and Predicted Class Label.



**Figure 3.1:** Flowchart of InterAcT architecture.

### ***3.1.1 Linear Projection of Pose Features***

It serves as a crucial step in transforming input pose matrices into a higher-dimensional space, necessary for the model's operations. The primary objectives behind this transformation include increasing the model's capacity to grasp the patterns and relationships within pose data, allowing for the discernment of subtle pose variations. Additionally, linear projection aids in feature learning, extracting meaningful insights from pose data to highlight crucial aspects pertinent to action recognition. Moreover, this projection facilitates improved interactions during self-attention, enabling the model to better capture dependencies between different elements in the input sequence. Integrating with positional embeddings, the linear projection further enriches the input tokens, enhancing the model's comprehension of the significance of various keypoints within the sequence. Overall, this process significantly augments the model's ability to comprehend essential features and relationships, ultimately advancing the accuracy of action recognition within video streams.

### ***3.1.2 Class Token Embedding***

Class token, a concept inspired by BERT [127] and Vision Transformer [119], plays a unique role in input sequences by serving as a special token. Its primary function is to amalgamate information across the sequence, facilitating the model's comprehension of the broader context. In this architectural design, the input sequence of pose matrices incorporates the [CLS] token. The core objective behind including the [CLS] token is to stimulate the self-attention mechanism within the Transformer, enabling the creation of a high-dimensional representation that encapsulates crucial features pertinent to distinct action categories. With this distinct token, the model becomes proficient at distinguishing between different actions with greater effectiveness.

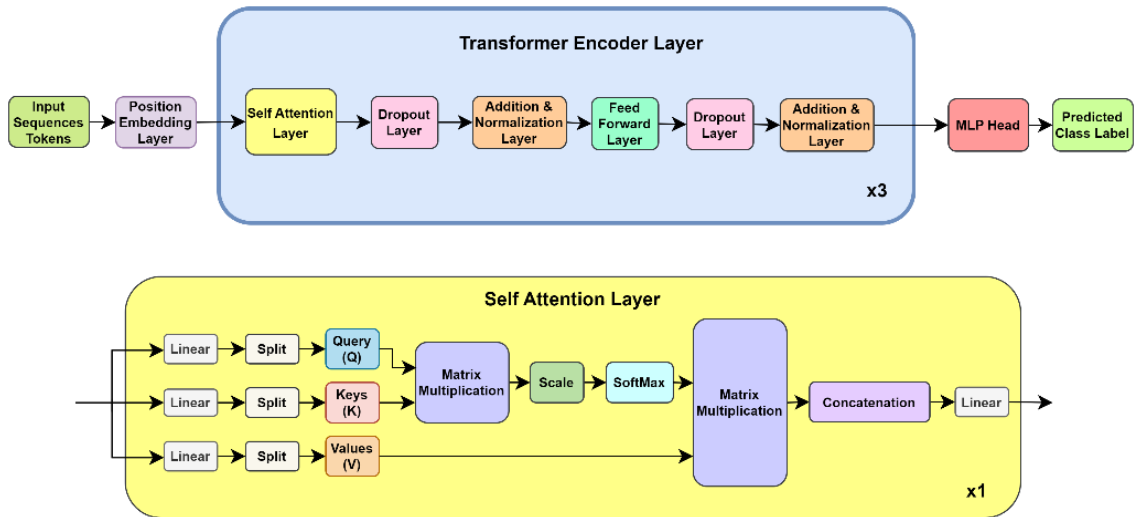
### ***3.1.3 Position Embedding***

Positional information is crucial because the model does not inherently understand the order or position of the tokens (pose+class) in the input sequence. This allows the model to distinguish the position of each token within the sequence and understand the temporal relationships between the frames in the video.

### ***3.1.4 Transformer Encoder Layer and MLP Head***

The Transformer encoder as shown in Figure 3.2, consists of multiple layers employing alternating multi-head self-attention and feed-forward blocks. After each block, Dropout, Layer-norm, and residual connections are applied. Each feed-forward block operates as a multi-layer perceptron, using GeLU non-linearity. The self-attention mechanism involves computing queries, keys, and values, then calculating attention weights based on pairwise similarities, resulting in weighted sums. These operations are performed for all heads, concatenated, and then linearly projected back to the initial dimension. The attention mechanism operates in the time domain, allowing the representation of a global class embedding by correlating different time windows. To manage hyperparameters and model dimensions, the grid search method was employed to select the optimal parameters that gives efficient and effective performance. The Action Transformer (AcT)[118] architectural parameters for its four variants are given in Table 3.1. For this study, inspired by the 'micro' architecture, we have proposed a lightweight InterAcT architecture 'nano' that is effective and efficient based on its performance making it well-suited for real-time applications.





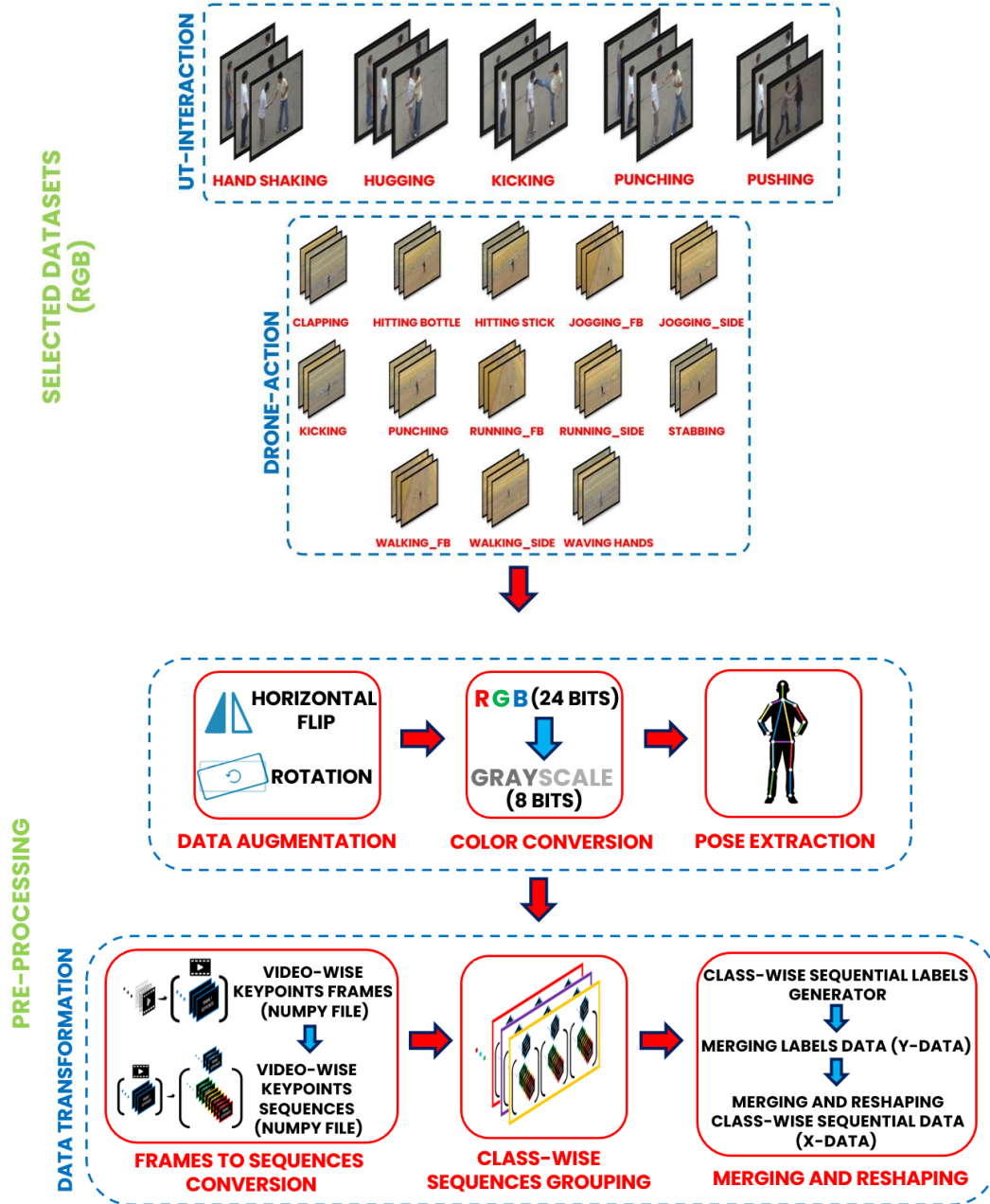
**Figure 3.2:** InterAcT architecture - Transformer encoder layer (top) and Multi-head self-attention block (bottom).

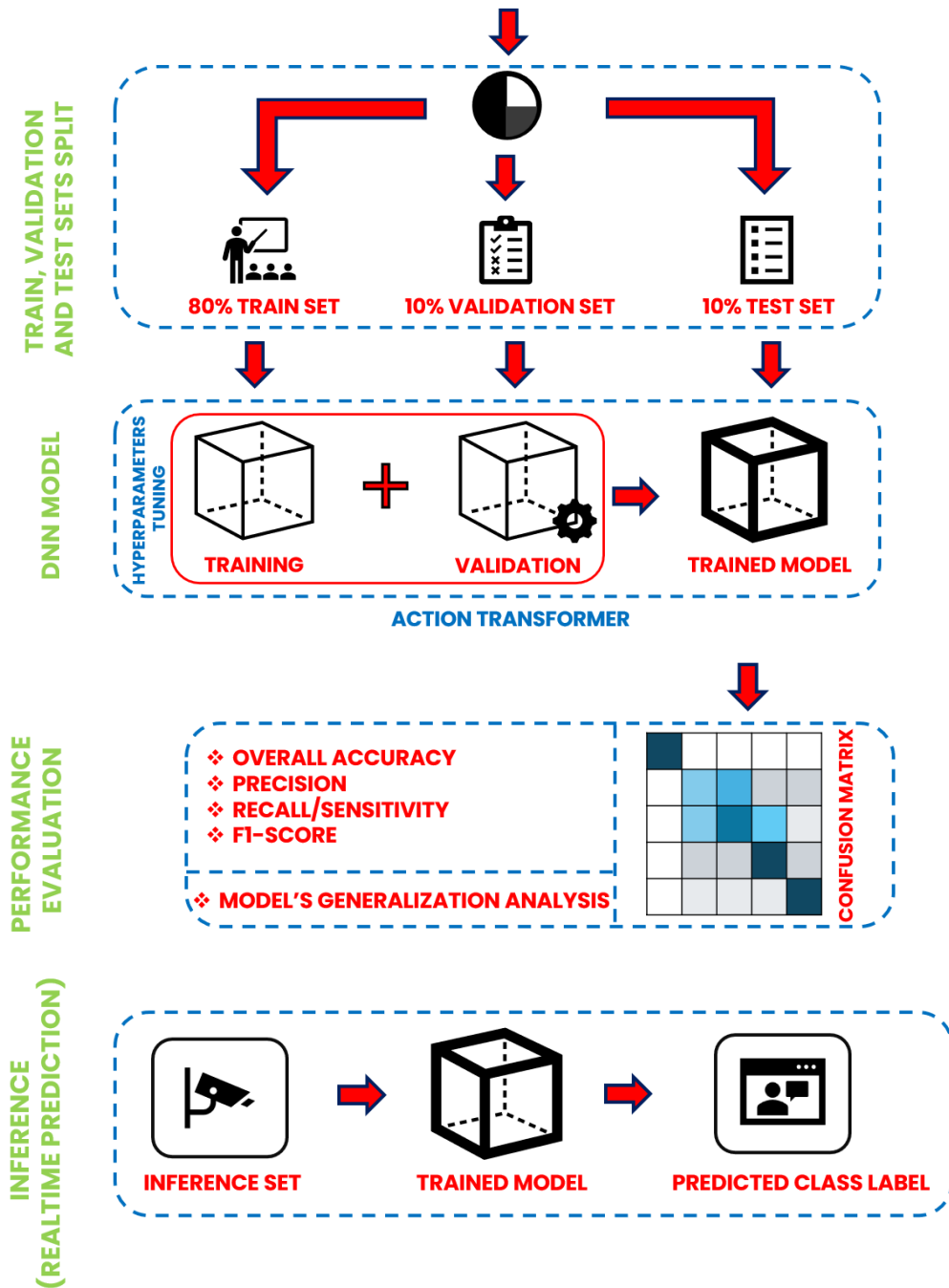
**Table 3.1:** Parameters for four variants of AcT [118].

Model	Number of Heads (H)	Dimensions of Model ( $D_{model}$ )	Dimensions of MLP Head ( $D_{mlp}$ )	Number of Transformer Encoder Layers (L)	Parameters
micro	1	64	256	4	227.7k
small	2	128	256	5	1041.9k
base	3	192	256	6	2742.6k
large	4	256	512	6	4905.2k

### 3.2 Implemented Framework - Flowchart

Figure 3.3 below illustrates the steps undertaken to attain the intended outcomes.





**Figure 3.3:** Flowchart of Implemented Framework.

The proposed framework commences by utilizing two RGB modality videos datasets named Ut-Interaction [1] and Drone-Action [2]. The Ut-Interaction dataset comprises of five human-human interactions classes and the Drone-Action dataset comprises of 13 solo-action classes. The videos from both datasets were fed into the pre-

processing block that outputs the sequential data and the labelled data (X-data and Y-data) that were further fed into the data splits block to splits the sequential data and the labelled data (X-data and Y-data) into training set, validation set and test set in order to train, validate, tune hyperparameters and perform performance evaluations of the proposed model.

### ***3.2.1 Pre-processing Block***

The pre-processing block consists of multiple modules. The detail of each module is explained as follows:

#### ***3.2.1.1 Data Augmentation***

The pre-processing block commences by the data-augmentation module. Data-augmentation techniques are utilized when videos are fewer than the required number of videos to extract desired number of sequential data to train the model. Moreover, it also helps in making the model generalizable. For extracting a greater number of sequential data from the videos, it's important to have a greater number of videos for each class to obtain an equal number of sequential samples per class. To increase number of videos per class, two types of augmentation techniques were employed namely horizontal flip and rotation of 30° and 45° about the center.

### ***3.2.1.2 Color Conversion***

The second stage of pre-processing is the conversion of videos color. Color videos consist of three channels known as RGB (Red, Green and Blue channels). The RGB videos contains more information and can help us in detection and extraction of desired data. For this study, as a part of investigation, our goal is to recognize solo-actions and human-human interactions in grayscale videos. Thus, the videos were converted from RGB to Grayscale videos and were fed into the pose extraction module to extract the pose data.

### ***3.2.1.3 Pose Extraction***

Pose estimation in computer vision is the process of identifying the positions of specific points in an image, commonly known as keypoints. These keypoints typically denote different parts of the object, such as joints, landmarks, or other notable features. The coordinates of these keypoints are often represented as a set of 2D  $[x, y]$  or 3D  $[x, y, visible]$  coordinates. In the context of human pose estimation, the goal is to identify the locations of key body joints such as elbows, knees, wrists, and ankles, as well as the overall body orientation. Pose estimation algorithms often rely on deep learning techniques, particularly convolutional neural networks (CNNs), which are trained on large datasets containing annotated images or videos of human poses. These networks learn to detect and localize key body joints by analysing patterns and features within the input data. Once trained, the pose estimation model can

process new images or video frames to estimate the pose of the subjects present. This typically involves identifying key points, connecting them to form skeletal structures, and inferring the spatial relationships between these points to determine the pose.

There are variety of pose estimation algorithms developed by researchers. Some of the pose estimation algorithms includes OpenPose [128], HRNet [129], AlphaPose [130], YOLO Pose [3] and Mediapipe ([131]).

In this study, we have utilized the latest Ultralytics YOLO v8 Pose Estimation model [3]. YOLO v8 pose estimation model has six variants, each differ in terms of input resolution, performance, and computation complexity. In our study, we have utilized the light-weight model named “yolov9n-pose” that accepts a maximum input resolution of 640x640, and it has the minimum model complexity of 3.3M parameters and 9.2B flops among all six pose estimation models. The default confidence score of detection offer by this model is 0.25. Any higher resolution as input to the model resizes by default and extract 17 2D keypoints per person. The details of indices and name of extracted keypoints for a person is given in Table 3.2.

**Table 3.2:** YOLO v8 pose keypoints indices.

<b>Index</b>	<b>Keypoints</b>
0	Nose
1	Left-eye
2	Right-eye
3	Left-ear
4	Right-ear
5	Left-shoulder
6	Right-shoulder
7	Left-elbow
8	Right-elbow
9	Left-wrist
10	Right-wrist
11	Left-hip
12	Right-hip
13	Left-knee
14	Right-knee
15	Left-ankle
16	Right-ankle

#### ***3.2.1.4 Data Transformation***

The final module of pre-processing block is transformation of extracted data. By utilizing Ultralytics YOLO-v8 pose model for each video, we extracted the keypoints of person(s) involved in each frame of the video. For each class in the datasets, frames keypoints data for each video were saved to numpy files (.npz files) for further processing. Each video numpy file have a shape of (Number of Frames in the Video, Number of Keypoints per Person\*Number of Person(s), Number of Coordinates per Keypoint). As our study focus on recognizing two persons interaction as well as solo actions. The frames with no single person detection, we have appended an array of zeros in the numpy files so that the shape of numpy files remains consistent for further transformation.

The transformation module commences with frames to sequences conversion. Depending upon required temporal or sequence length, the frames keypoints data per video were transformed into sequences to form the sequential keypoints data for each video. The sequential keypoints data for all videos were also saved to numpy files (.npy files), each having a shape of (Number of Sequences in the Video, Sequence Length, Number of Keypoints per Person\*Number of Person(s), Number of Coordinates per Keypoint). These video-wise sequential keypoints data were merged according to the respective class to form class-wise sequential keypoints data that has shape of (  $\sum_{n=1}^N$  Number of Sequences of Video\_(n), Sequence Length, Number of Keypoints per Person\*Number of Person(s), Number of Coordinates per Keypoint ). In this conversion module, there was the problem of selection of sequences transformation type, i.e. fixed window sequences or sliding window sequences. In case of fixed window sequences, during conversion, some last frames of each video fail to form a complete sequence of required sequence length. To address this problem, there are multiple ways, one way is to drop those frames, another is to duplicate those last frames, some methods use approximation techniques or use sliding window sequences instead of fixed window sequences. Class-imbalance was another problem in this conversion, when class-wise sequential keypoints data are generated, one class has high number of sequences than the other class. To address the class imbalance problem, the technique of data slicing was employed. In this study, for each video, we



have generated sequential keypoints data with temporal or sequence length of 30 frames and to remove the class imbalance, among all classes sequences samples, we selected the minimum number of samples and used that number of samples per class to balance all classes sequential data.

In the last stage of transformation module, class-wise sequential labelled data were generated based on the class-wise sequential keypoints data as generated in the prior stage and all class-wise sequential labelled data were merged to form the overall labelled data (Ydata) that has shape of (Total Classes\*Number of Sequences per Class). By merging the class-wise sequential keypoints data, it resulted in a data shape of (Total Classes\*Number of Sequences per Class, Sequence Length, Number of Keypoints per Person\*Number of Person(s), Number of Coordinates per Keypoint). This data was reshaped to (Total Classes\*Number of Sequences per Class, Sequence Length, Number of Keypoints per Person\*Number of Person(s)\* Number of Coordinates per Keypoint) and this reshaped data is known as overall sequential keypoints data (Xdata). Both overall labelled data (Ydata) and overall sequential keypoints data (Xdata) were then fed into the data splitting block of the pipeline.

### ***3.2.2 Data Splits and DNN Block***

When data (Xdata and Ydata) was prepared, it was split into three parts namely training set, validation set and test set. The training set is used to train the model so that the model learns the patterns of the data to make it work for the

desired task. For validation and hyperparameters tuning of the model, a validation set was used to check the model's generalizability (underfitting and overfitting) and to set the model on best settings. After training and validation, the performance of the model was evaluated on unseen test data. The performance of the model on the test set tells us how it will work for the desired task and its suitability for deployment. The data splits percentages depend on the available size of the prepared data (Xdata and Ydata), the choice of the model and the choice of the desired task. The percentage range of training-set varies from 60% to 80%, validation-set from 10% to 20% and remaining 10% to 20% for test-set respectively. In this study we have used 80% of data for training, 10% for validation and hyperparameters tuning and 10% for testing the model.

### ***3.2.3 Performance Evaluation***

The process of assessing how well a trained model performs on unseen data is known as performance evaluation. Depending on the recognition task and choice of the model, there are different techniques and metrics which helps us in measuring the model's effectiveness and generalization ability of the model.

For understanding the classification performance metrics, it's important to understand the basics of Confusion Matrix. A confusion matrix also known as Error Matrix is a table used to describe the performance of a classification model. A general binary classification confusion matrix is shown in Figure 3.4 which is a 2x2 matrix to understand the terminologies which is useful when finding the performance metrics. A confusion matrix is a  $n \times n$  matrix where  $n$  represents the

number of labels in the data. It presents a comprehensive summary of the model's predictions against the actual outcomes in a tabular format. The general 2x2 matrix is composed of four different combinations of predicted and actual classes:

#### ***3.2.3.1 True Positives (TP)***

Instances where the model correctly predicts the positive class. It means that the true class label case as in ground truth has correctly predicted by the model as true.

#### ***3.2.3.2 True Negatives (TN)***

Instances where the model correctly predicts the negative class. It means that the negative class label case as in ground truth has correctly predicted by the model as negative.

#### ***3.2.3.3 False Positives (FP)***

Instances where the model predicts the positive class incorrectly. It's a misclassification case where a true class label as in ground truth is predicted as negative class label by the model.

#### ***3.2.3.4 False Negatives (FN)***

Instances where the model predicts the negative class incorrectly. It's a misclassification case where a negative class label as in ground truth is predicted as positive class label by the model.

A confusion matrix is useful in finding classification performance metrics that includes overall accuracy, precision, recall/sensitivity, and F1-Score. The details of each performance metric are explained in Table 3.3.

		Predicted Class	
		True Positive (TP)	False Negative (FN)
True Class	True Positive (TP)		
	False Positive (FP)		True Negative (TN)

**Figure 3.4:** A general 2x2 confusion matrix.

**Table 3.3:** Classification performance metrics.

Performance Metric	Definition	Purpose	Formula
<b>Overall Accuracy</b>	The ratio of correct predictions to the total number of predictions.	Offers a comprehensive evaluation of the model's accuracy across all predictions.	$\frac{TP + TN}{TP + TN + FP + FN}$
<b>Precision</b>	The ratio of true positive predictions out of the total predicted positive instances.	Evaluates the model's precision in predicting positive instances, minimizing false positives.	$\frac{TP}{TP + FP}$
<b>Recall/Sensitivity</b>	The ratio of true positives to the sum of true positives and false negatives.	Evaluates the model's sensitivity to identifying actual positive	$\frac{TP}{TP + FN}$

		instances, minimizing false negatives.	
<b>F1-Score</b>	The harmonic means of precision and recall, providing a balanced measure between the two metrics.	Balances precision and recall, yielding a single metric that accounts for both false positives and false negatives.	$2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

Generalization is defined as the model's ability to perform well on new, unseen data. There are two types of generalization problems, and both are undesirable in evaluating model performance on unseen data, these include underfitting and overfitting.

Underfitting occurs when a model is too basic to understand the patterns in the training data. It fails to learn the complexity of the data and performs poorly, both on the training and unseen data. It may arise from using a model that's too basic or lacks the capacity to represent the underlying patterns in the data. Additionally, insufficient training or a lack of suitable features can contribute to underfitting. The model in underfitting performs poorly on both the training and test datasets, showing high errors and low accuracy. To address underfitting, one can try using more complex models, adding more features, increasing model capacity, or extending training time.

Overfitting occurs when a model is too complex, capturing noise or random changes in the training data rather than learning the underlying patterns. An overfit

model performs very well on the training data but poorly on unseen data. Complex models or models trained for too long can capture noise instead of general patterns, leading to overfitting. Additionally, when a dataset is small or the model is too tailored to the training data, overfitting can occur. High performance on the training data but a significant drop in performance on the test or validation data indicates overfitting. The model essentially memorizes the training data rather than learning the underlying trends. To address overfitting, there are several ways including simplifying the model, reducing its complexity through regularization techniques (like dropout, L1/L2 regularization), increasing the dataset size, or using cross-validation to mitigate the effects of overfitting.

The number of computational resources required to execute and deploy the model is also important and to access the computational complexity of model, metrics like number of model parameters, flops, fps and inference time are used that helps us in selecting the appropriate resources in order to train the model and deploy it in real-time applications. Balancing computational complexity and performance metrics is essential in designing efficient models. These metrics are explained as follows:

### ***3.2.3.5 Model Parameters***

The total number of fixed parameters, also known as non-trainable parameters and updatable parameters (weights and biases) that are learned by the model during the training process, is known as the parameters of the model. The more the number of parameters, the more complex is the model, such

complex model helps in capturing all complex patterns of the data and increases the accuracy however it increases the computational complexity that results in high utilizing of hardware resources and such model may not be preferable to deploy for real-time applications. On the other hand, the smaller number of parameters makes the model less computationally expensive, it utilizes less hardware resources. However, in such a model the accuracy of model may sacrifice but well suitable for deployment in real-time applications.

#### ***3.2.3.6 Floating-point Operations (Flops)***

The number of arithmetic operations performed by the model during training and inference is known as floating-point operations (flops). The more the number of flops, the more complex is the model and vice versa. Models with high flops require more hardware resources and time to execute as compared to models with less flops.

#### ***3.2.3.7 Frames Per Seconds (FPS)***

The number of frames processed by a model per second is known as frames per second (fps). This metric is commonly used in real-time applications to check whether a model is suitable for deployment. The more fps indicates faster processing which is crucial for applications where quick response is needed. For more fps, a complex model requires more computation power and vice versa.

### ***3.2.3.8 Inference Time***

The amount of time taken by a trained model to make predictions on new data is known as inference time. Lower inference time is often desirable, making the model well suitable to deploy for real-time applications.

### ***3.2.4 Inference Block***

Inference, also known as real-time prediction is a stage where a trained model is deployed in a real-time application without further training or testing the model. The trained model at inference stage takes new unseen data as input, processes it, make quick predictions and a decision is made by the system on the predictions.



## CHAPTER 4: IMPLEMENTATION

This chapter provides insights into the implementation of the proposed InterAct model. It presents the details of implementation of each section as discussed in chapter 3. The chapter starts with pre-processing of public datasets, discusses common characteristics of model, and provides the details of training process.

### 4.1 Preprocessing of Datasets

We have selected two public datasets for our study, Ut-Interaction [1] which is a human-human interaction dataset and Drone-Action [2] which is a solo action dataset. The statistics of these datasets are as given in Table 4.1.

**Table 4.1:** Statistics of selected datasets.

<b>Dataset</b>	<b>Number of Classes</b>	<b>Number of Videos</b>	<b>Number of Person</b>	<b>Video Resolution</b>	<b>Environment</b>
<b>Ut-Interaction</b> [1]	5	20	2	720x480	Outdoor (Plain Road + Forest)
<b>Drone-Action</b> [2]	13	240	1	1920x1080	Outdoor (Unpaved Road)

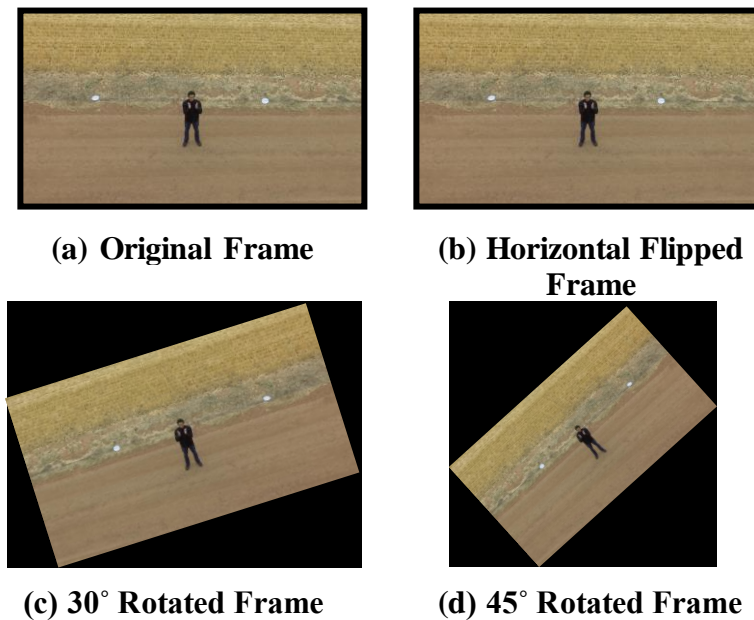
Ut-Interaction dataset offers two types of video data: original and segmented. Based on environment, there are two sets of original data: original set1 (plain-road) and original set2 (forest). Similarly, there are two sets of clipped data: clipped set1 (plain-road) and clipped set2 (forest). Original data contains 10 videos per set which includes all five classes and has resolution of 720x480 while clipped data contains videos that are clipped to the duration of class and are cropped. Since clipped videos are cropped, all videos have

different resolution. The dataset has five human-human interactions classes namely handshaking, hugging, kicking, punching, and pushing.

Drone-Action dataset is a solo human action dataset comprises of 13 classes including clapping, hitting bottle, hitting stick, jogging front-back-view, jogging side-view, kicking, punching, running front-back-view, running side-view, stabbing, walking front-back-view, walking side-view and waving hands. The videos are shot on an unpaved road environment. Each class has 20 videos except clapping and waving hands, both classes have 10 videos. The resolution of the videos is 1920x1080.

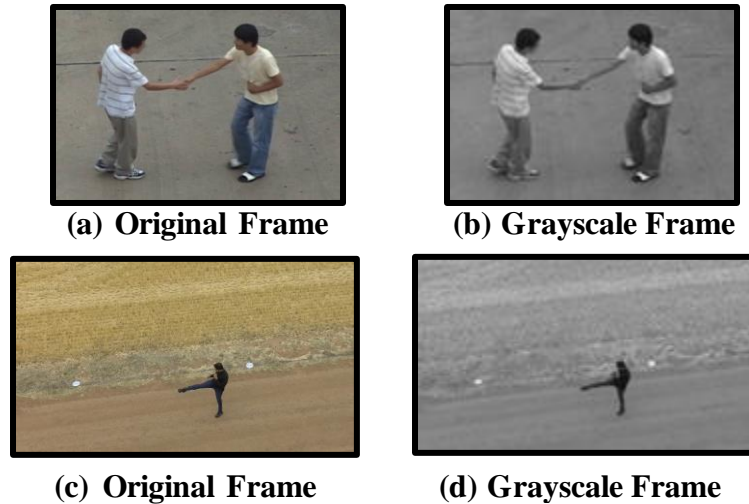
It is to note that both datasets have different video resolutions which can be problematic in extracting the poses because YOLO v8 pose extractor model (yolov9n-pose) accepts a max resolution of 640x640 to extract poses. It resizes videos that have resolution higher than the max resolution. Resizing the resolution can affect the pose model to extract the poses effectively or may not extract poses if subject(s) size is very small. Another problem is the change of scale of extracted pose also known as coordinate scaling, different resolutions have different aspect ratios which affect the scale of extracted pose. To tackle these issues, the possible solution to resizing problem is to crop the original videos of both datasets into a fixed resolution and then employ the pose extractor model to extract poses, this will solve both issues. Another solution is to crop the videos of Drone-Action dataset to a resolution that are equivalent to that of Ut-Interaction clipped dataset, this will help model to adapt to equivalent resolutions. The scale change can be addressed by use of a scaling technique, one way is to multiply either axis (width or height) by the ratio of width over height. For equivalent resolutions, this will maintain the scale of both coordinates of the extracted pose.

From the analysis of selected datasets, there is an imbalance in the number of videos per class. To address this problem, data augmentation was employed for those classes that have a smaller number of videos. Another advantage of employing data augmentation is to increase the generalizability of the model. Two types of augmentation techniques were employed: horizontal flip and rotation of  $30^\circ$  and  $45^\circ$  about the center. Figure 4.1 shows the data-augmentation for clapping class of Drone-Action dataset.



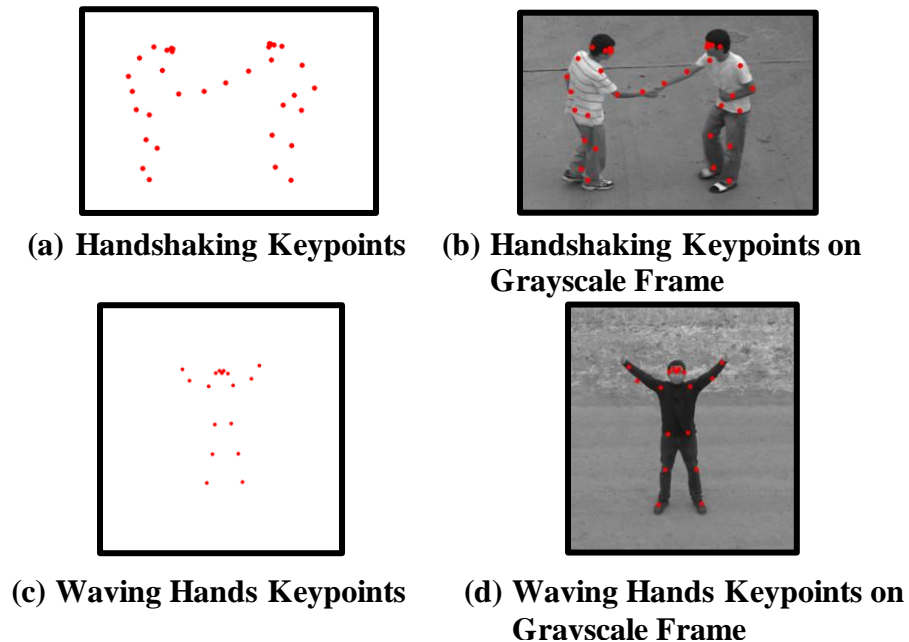
**Figure 4.1:** Data augmentation for Drone-Action “clapping” class.

The next stage of pre-processing is color conversion in which all RGB videos were converted into Grayscale videos. Figure 4.2 shows the color conversion for Ut-Interaction “handshaking” class and Drone-Action “kicking” class respectively.



**Figure 4.2:** Color conversion for Ut-Interaction “handshaking” class and Drone-Action “kicking” class.

After conversion from RGB to Grayscale, YOLO v8 pose model was employed to extract the poses of subject(s) involved in the videos. YOLO v8 pose model extracts 17 2D keypoints for each subject(s) in the video. The results of YOLO v8 pose model are depicted in Figure 4.3.



**Figure 4.3:** YOLO v8 pose keypoints for Ut-Interaction “handshaking” class and Drone-Action “waving hands” class.

After extracting keypoints of subject(s) in the videos, the keypoints data were transformed into sequential data (Xdata) along with labelled data (Ydata) as discussed in Section 3.2.1.4 of Chapter 3. The sequential data (Xdata) and labelled data (Ydata) were further split into three sets: training- set, validation-set and test-set with percentages of 80%, 10% and 10% respectively.

## 4.2 Proposed InterAcT Model Training Parameters

This section presents the characteristics of the proposed InterAcT model during training it. This section gives an overview of hyperparameters, activation functions and optimizers that were experimented and selected during this study. The details of each parameter are explained as follows:

### 4.2.1 Hyperparameters

The parameters that are chosen prior to training are known as hyperparameters. Their values can be adjusted prior to training and remain constant during training. The hyperparameters used in our model are mentioned in Table 4.2.

**Table 4.2:** List of hyperparameters.

<b>Hyperparameter</b>	<b>Set of Values</b>	<b>Selected Value</b>
<b>Epoch</b>	500	500
<b>Learning Rate</b>	0.1, 0.01, 0.001, 0.0001, 0.00001,0.000001	0.001
<b>Weight Decay</b>	0.1, 0.01, 0.001, 0.0001, 0.00001,0.000001	0.000001
<b>Batch Size</b>	1, 2, 4, 8, 16, 32, 64, 96, 128, 160, 192, 224, 256	192
<b>Sequence Length</b>	30	30
<b>Sequences Type</b>	Fixed Window, Sliding Window	Sliding Window

#### ***4.2.1.1 Epoch***

The number of iterations taken by the model to learn the entire training dataset is known as epoch. Training the model with a smaller number of epochs takes less time to train the model. However, the model may not learn the training set well resulting in underfitting problem. On the other hand, if the model is trained for many epochs, the model will take more time to learn, and training loss decreases but sometimes training model for many epochs leads to overfitting problem. It is important to select a balanced number of epochs which make the training process efficient and to overcome the generalizability problem. To select a balanced number of epochs, plot the training and validation loss over each epoch. Initially, both losses should decrease as the model learns. However, if the validation loss starts to increase while the training loss continues to decrease, it indicates overfitting. Choose the number of epochs just before the validation loss begins to increase.

#### ***4.2.1.2 Learning Rate***

In machine learning, learning rate is a tuning parameter that determines the size of steps taken in optimization process while moving toward a minimum of a loss function. A high learning rate may cause the algorithm to overshoot the minimum which leads to divergence. On the other hand, a very low learning rate may cause slow convergence or may be stuck in local minima. Therefore, choosing an appropriate learning rate is

essential to train the model efficiently and effectively. To select an appropriate learning rate, numerous techniques are used including grid search, learning rate schedulers, learning rate warmup, and visualizing learning rate vs loss curve. It's a good practice to start with a small learning rate and gradually increase it until a value that trains the model effectively without causing divergence. Grid search is often a good choice to look for an appropriate learning rate. By selecting a set of values, the model is trained on each value and the one that performs best on validation set is selected.

#### ***4.2.1.3 Weight Decay***

In machine learning, weight decay also known as L2 regularization is a method that prevents overfitting of models and optimizes the model. It is used in weighted optimizers. It adds a regularization term to loss function that penalizes large weights. Choosing large weight decay reduces overfitting but may cause underfitting. Conversely, choosing a very small weight decay weakens the regularization and increases the risks of overfitting. Therefore, choosing an appropriate weight decay is essential for training model effectively without generalization problems. To select an appropriate weight decay, grid search is a good technique. By selecting a set of values, the model is trained on each value and the one that performs best on validation set is selected.

#### ***4.2.1.4 Batch Size***

In machine learning, batch size is the number of samples that are utilized in one iteration during training. By using batch size, the training set is divided into subsets called batches and are utilized in each training iteration. The number of iterations taken during training are determined by dividing the total number of training samples by batch size. Choosing a larger batch size results in faster convergence and less training time. Selecting an appropriate batch size is essential in training the model effectively. By using grid search, a batch size that gives good performance on validation set is selected.

#### ***4.2.1.5 Sequence Length***

In sequential based models, sequence length is the number of samples or frames per sequence. In action recognition, it provides the temporal dimension of video sequence. It determines the amount of context a model can consider at once. Large sequence length provides more context to the model but requires more memory and computation as compared to small sequence length. The choice of sequence length depends on data characteristics, model architectures, computational resources, and nature of task. To select an optimal sequence length, grid search is often employed to select the value that gives good performance on validation set.



#### ***4.2.1.6 Sequences Type***

Sequences type or Sequences formation type is defined as the nature of extracting desired number of frames for a video having  $n$  frames. There are mainly two types of sequence formation, fixed window sequencing and sliding window sequencing.

In fixed window sequencing, desired number of frames (sequence length) are taken subsequently. In this type, for a video having  $n$  frames and desired sequence length of  $S$ , we get  $n/S$  sequences. In fixed window sequencing, there is problem of complete sequence formation in the last sequence. To address this issue, the frames in the last sequence are either duplicated or dropped.

In sliding window sequencing, desired number of frames (sequence length) are taken consecutively. In this type, for a video having  $n$  frames and desired sequence length of  $S$ , we get  $n-S+1$  sequences. In sliding window sequencing, we always get complete sequences.

For selection of best sequencing type, grid search is often employed to select the type that gives good performance on validation set.

### 4.2.2 Activation Functions

The mathematical operations applied to output of a neuron in a neural network are known as activation functions. They introduce non-linearity in the model to make model learn complex patterns of data. It decides whether a neuron should be activated or not by calculating the weighted sum and adding bias to it. There are numerous activation functions that can be utilized in models however, the choice of activation functions depends on architecture of model, nature of data and specific task at hand. In the case of action recognition using deep learning models, the most popular activation function is Gaussian Error Linear Unit (GeLu). Other activation functions that are used in machine learning and deep learning tasks are sigmoid, relu, leaky-relu, selu, silu, elu, softmax, softplus, swish and mish functions. To select an activation function, choose a set of activation functions and perform grid search. The activation function that gives good performance on validation set is selected for the desired task. Activation functions that were experimented under this study are given in Table 4.3.

**Table 4.3:** List of activation functions.

<b>Set of Activation Functions</b>	<b>Selected Activation Function</b>
selu, silu, elu, gelu, swish, mish	gelu

### **4.2.2.1 Gaussian Error Linear Unit (GeLu)**

Gaussian Error Linear Unit (GeLu) activation function weights the input by its probability under a Gaussian distribution. The operation performed by GeLU is given by equation 4.1.

$$GeLu(x) = x \cdot \Phi(x) = x \cdot \frac{1}{2} \left( 1 + erf \left( \frac{x}{\sqrt{2}} \right) \right) \quad (4.1)$$

Where  $\Phi(x)$  represents the cumulative distribution function of a standard Gaussian distribution, and  $erf(x)$  is the error function.

GeLU is suitable to use in deep learning models because it's a smooth function (continuously differentiable), it approaches linearity for large positive and negative values of  $x$  which is advantageous for capturing linearity, it is zero centered which helps mitigate issues related to shifting distributions in the network's activations, leading to faster convergence during training. Based on its performance on validation set, it outperforms the activation functions that are given in Table 4.3.

### **4.2.3 Optimizers**

An algorithm that is used to adjust the weights and biases iteratively during the training process to minimize the loss function is known as an optimizer. An optimizer is used to adjust the weights and biases that best fit the training data and generalize well on unseen data. The optimizers that were experimented on the proposed model are presented in Table 4.4.

**Table 4.4:** List of optimizers.

<b>Set of Optimizers</b>	<b>Selected Optimizer</b>
AdamW, LAMB, LazyAdam, RAdam, SGDW	AdamW

#### *4.2.3.1 Adaptive Moment Estimation with Weight Decay*

AdamW (Adaptive Moment Estimation with Weight Decay) is a variant of the traditional Adaptive Moment Estimation (Adam) optimizer that incorporates weight decay regularization directly into the update rule. Weight decay penalizes large weights in the model to prevent overfitting. The update rule for AdamW is given by equation 4.2.

$$\theta_{t+1} = \theta_t - \alpha \frac{\hat{m}}{\sqrt{\hat{v} + \epsilon}} - \alpha \cdot \lambda \cdot \theta_t \quad (4.2)$$

Where  $\theta$  represents the model parameters (weights and biases).  $\alpha$  is the learning rate,  $\hat{m}$  and  $\hat{v}$  are the bias-corrected first and second moment estimates, respectively.  $\epsilon$  is a small value to prevent division by zero.  $\lambda$  is the weight decay coefficient.  $\lambda$  is the weight decay coefficient.

### 4.3 Resources Utilized

This section presents the hardware and software resources that were utilized to implement the proposed framework.

#### 4.3.1 Hardware Resources

For deep learning models, it is important to use a powerful system that takes less time to train the model and to test the model for inference. The specifications of hardware that was used for this study are given in Table 4.5.

**Table 4.5:** Hardware specifications.

<b>Component</b>	<b>Specifications</b>
Processor Name	12 <sup>th</sup> Generation Intel (R) Core (TM) i7-12700
Processor Clock Speed	4.9 GHz
Processor Cores	12
Processor Threads	24 (2 Threads per Core)
Graphics Card Name	NVIDIA GeForce RTX 3090
Graphics Memory (VRAM)	24 GB
RAM Size	62 GB
RAM Technology	DDR5

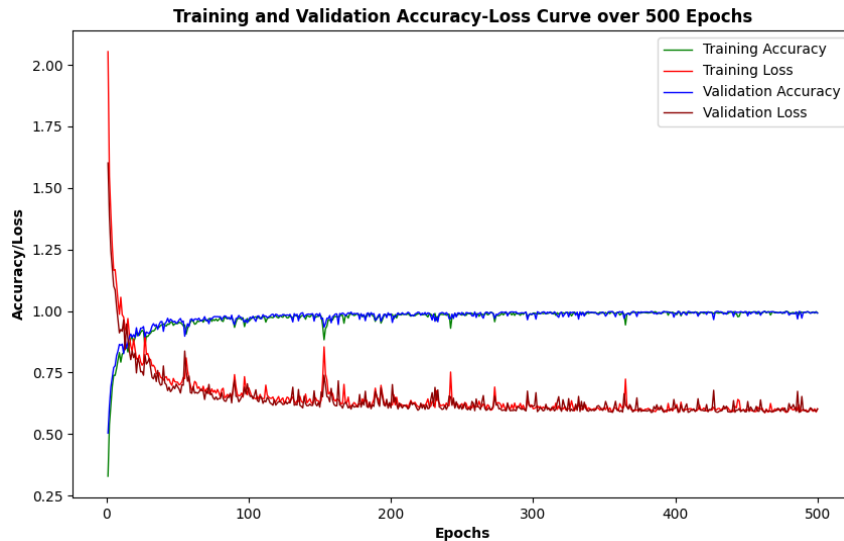
#### 4.3.2 Software Resources

For this study, we have used Python 3.11.5. Python is a popular programming language that is well suited for machine learning and deep learning programming. It offers a wide range of libraries and is a user-friendly language. For implementation of proposed framework, we have used TensorFlow 2.15.0 which is an open-source framework developed by Google providing high-level APIs to build deep learning models. Moreover, we have implemented our model on Linux Operating System

(Ubuntu 22.04.1, Linux Kernel version 6.5.0-25-generic) due to its flexibility, performance, and strong support for development tools and GPU acceleration.

#### 4.4 Training Setup

The selected hyperparameters, activation function and optimizer are given in Table 4.2, Table 4.3, and Table 4.4 respectively. With these selected parameters, we trained the model and tested its performance on the test set. The training and validation accuracy-loss curve is illustrated in Figure 4.4.



**Figure 4.4:** Training and validation accuracy-loss curve.

Figure 4.4 illustrates the training and validation accuracies and losses of the proposed InterAcT model, trained with selected hyperparameters as given in Table 4.2. The model was trained for 500 epochs. It shows that both training and validation accuracies are increasing, and losses are decreasing over time which means that the model is learning and improving. Moreover, the validation accuracy and validation loss are closer to the training accuracy and training loss respectively. This behavior indicates that the model is neither underfit nor overfit which is a positive sign in terms of model's generalizability.

After training the model with 500 epochs, it reports an accuracy of 99.23% and evaluation time of 0.2013 seconds on test set.

## CHAPTER 5: EXPERIMENTS, RESULTS AND DISCUSSION

This chapter presents the detailed experiments performed on the proposed InterAcT framework, performance evaluation on test set and benchmarks comparison. This chapter commences with experimentations results of hyperparameters, optimizers and activation functions. After selection of parameters, the InterAcT model was evaluated on test set and at the end, it was compared with other state-of-the-arts benchmark models in terms of model complexity, evaluation time and test accuracy.

### 5.1 Experiments on Proposed InterAcT Framework

This section commences with the experimentation results of architectural changes of the proposed InterAcT model. After selection of architectural parameters, different parameters including hyperparameters, activation functions and optimizers as given in Table 4.2, Table 4.3 and Table 4.4 respectively were experimented on our model.

#### *5.1.1 Experiments on Architectural Parameters*

The original action transformer (AcT)[118] has proposed four variants of action transformer including micro, small, base, and large. The architectural parameters of these models are given in Table 5.1.



**Table 5.1:** Architectural parameters for four variants of AcT [118].

<b>Model</b>	<b>Number of Heads H</b>	<b>Dimensions of Model Dmodel</b>	<b>Dimensions of MLP Head Dmlp</b>	<b>Number of Transformer Encoder Layers L</b>	<b>Dropout Percentage</b>	<b>Parameters</b>
<b>micro</b>	1	64	256	4	0.30	0.2277M
<b>small</b>	2	128	256	5	0.30	1.0419M
<b>Base</b>	3	192	256	6	0.30	2.7426M
<b>large</b>	4	256	512	6	0.30	4.9052M

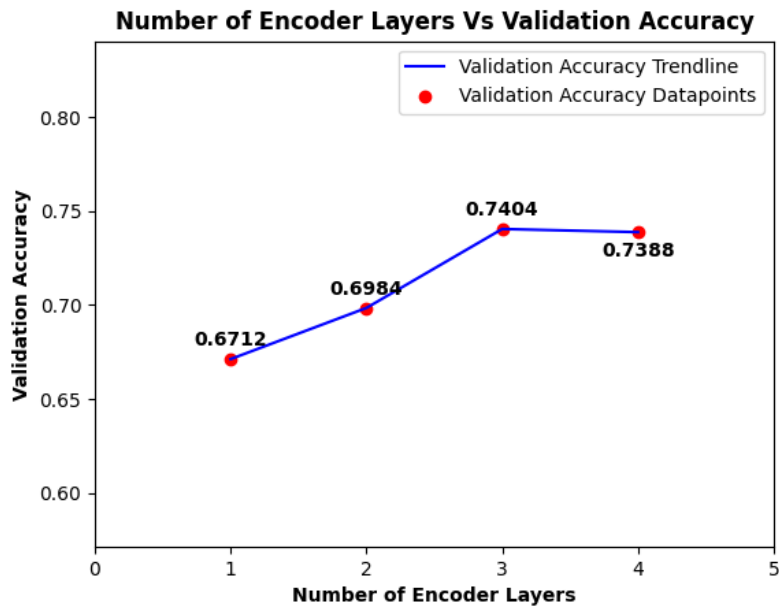
Inspired from the ‘micro’ architecture of AcT [118], we have proposed a light-weight model by performing experiments on architectural parameters such as dimensions of model (Dmodel) also known as embedded dimensions, dimensions of MLP Head (Dmlp) and number of transformer encoder layers (L). The set of values taken for experiments on these architectural parameters are given in Table 5.2. The experiments with these architectural parameters were performed with fixed window sequences, 500 epochs, batch-size of 32, gelu activation function, AdamW optimizer with learning rate of 0.0001 and weight decay of 0.00001. These architectural parameters were selected sequentially.

**Table 5.2:** List of architectural parameters.

<b>Architecture Parameter</b>	<b>Set of Values</b>	<b>Selected Value</b>
Number of Heads (H)	1	1
Number of Transformer Encoder Layers (L)	1,2,3,4	3
Embedded Dimensions (Dmodel)	1, 2, 4, 8, 12, 16, 24, 32, 40, 48, 56, 64	56
Dropout Percentage	0.05, 0.1, 0.15, 0.20, 0.25, 0.30	0.10
Dimensions of MLP Heads (Dmlp)	1, 2, 4, 8, 16, 32, 64, 96, 128, 160, 192, 224, 256	96

### 5.1.1.1 Effect of Number of Encoder Layers

Our aim is to propose a light-weight model, so we have chosen four values for embedded dimensions to perform experiments for it. During these experiments, other architectural parameters were set to their default values. The result of the number of encoder layers is shown in Figure 5.1. The final value was selected based on highest validation accuracy.



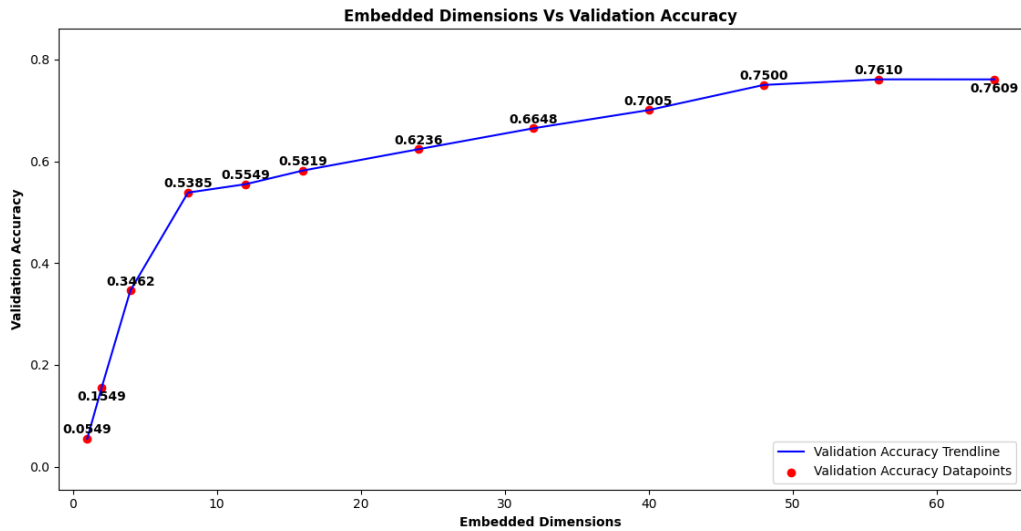
**Figure 5.1:** Number of encoder layers versus validation accuracy.

Figure 5.1 illustrates the effect of number of encoder layers versus validation accuracy. The plot shows an increasing validation accuracy over number of encoder layers which means that increasing the number of encoder layers increases the model performance however, if number of encoder layers are increased too much, the performance drops which is evident from the plot. Increasing the number of encoder layers results in a more complex model having increased number of flops and training

parameters of the proposed model. The model with a greater number of encoder layers will take more time in training and inference. In this study, we have selected 3 number of encoder layers for our proposed model so that there is a balance between performance and model complexity.

### 5.1.1.2 Effect of Embedded Dimensions

To perform experiments for embedded dimensions, we have chosen twelve values. During these experiments, the number of encoder layers was set to 3 (selected from previous experiment), while other architectural parameters were set to their default values. The result of embedded dimensions is shown in Figure 5.2. The value with the highest validation accuracy is selected for our model.

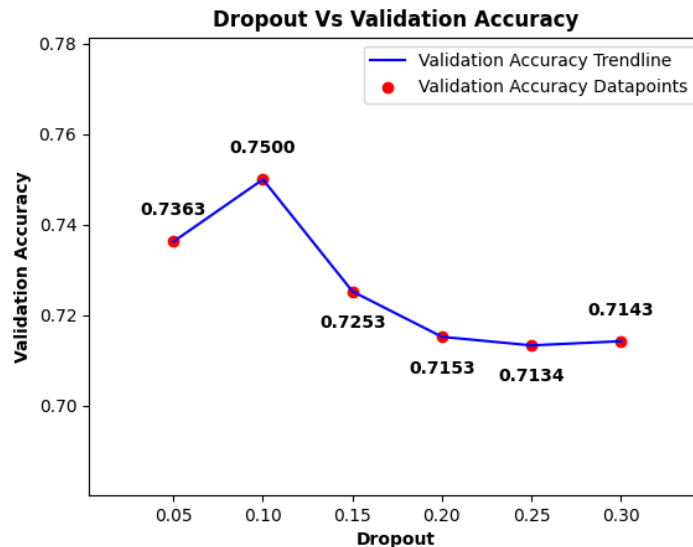


**Figure 5.2:** Embedded dimensions versus validation accuracy.

Figure 5.2 illustrates the effect of embedded dimensions versus validation accuracy. The plot shows an increase in validation accuracy for increase in embedded dimensions. The behavior after embedded dimensions of 48 becomes closer and steady. However, increasing embedded dimensions increases the complexity of the model and reduces the performance at higher values. In this study, we have used embedded dimensions of 56 for our model.

### 5.1.1.3 Effect of Dropout Percentage

In machine learning, dropout is a regularization technique used to prevent the model from overfitting and increase generalizability by removing a percentage of neurons which do not contribute to the forward or backward pass during training. The set of values for this experiment are listed in Table 5.2. The result of dropout percentage is illustrated in Figure 5.3.

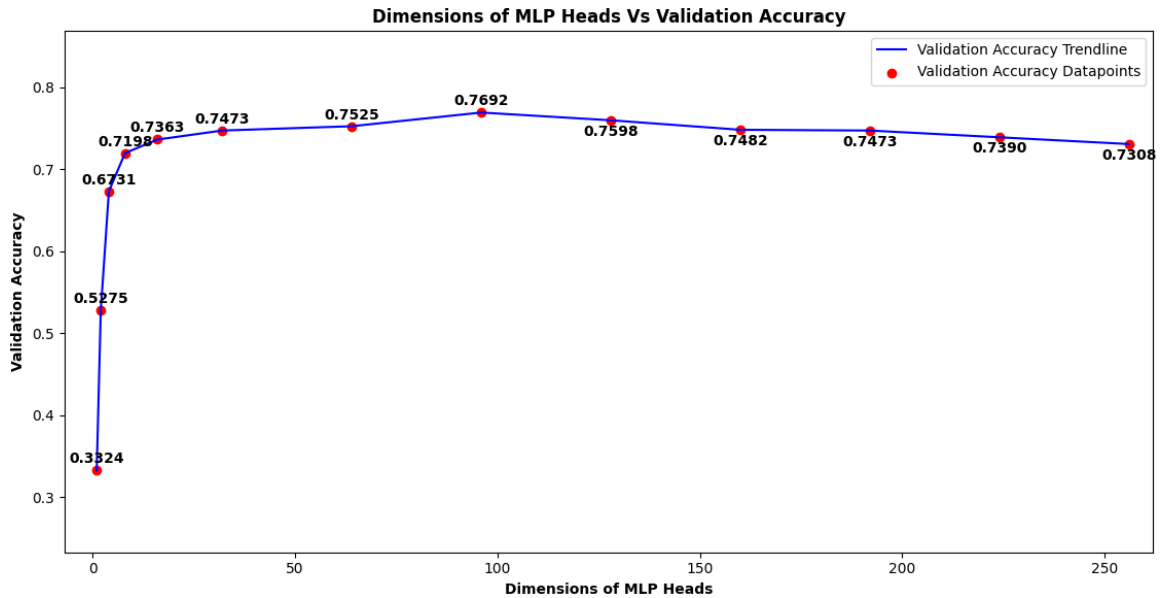


**Figure 5.3:** Dropout percentage versus validation accuracy.

Figure 5.3 illustrates the effect of dropout percentage versus validation accuracy. The plot shows that the validation accuracy decreases with an increase in dropout percentage. For our proposed model, we have used a dropout percentage of 10%.

#### 5.1.1.4 Effect of Dimensions of MLP Heads

The set of values for this experiment are listed in Table 5.2. The result for this experiment is shown in Figure 5.4.



**Figure 5.4:** Dimensions of MLP heads versus validation accuracy.

Figure 5.4 illustrates an increase in validation accuracy over increase in dimensions of MLP Heads. After MLP Heads dimensions of 96, the validation accuracy decreases. Selecting a larger value result in a complex model which requires more computational resources and time to both training and inference. From the plot, we get maximum validation

accuracy at MLP Heads dimensions of 96. Thus, for our proposed model, we have used MLP Heads dimensions of 96.

After performing and analyzing these experiments, the selected values of these parameters for our study are listed in Table 5.2.

### ***5.1.2 Experiments on Hyperparameters, Optimizers and Activation Functions***

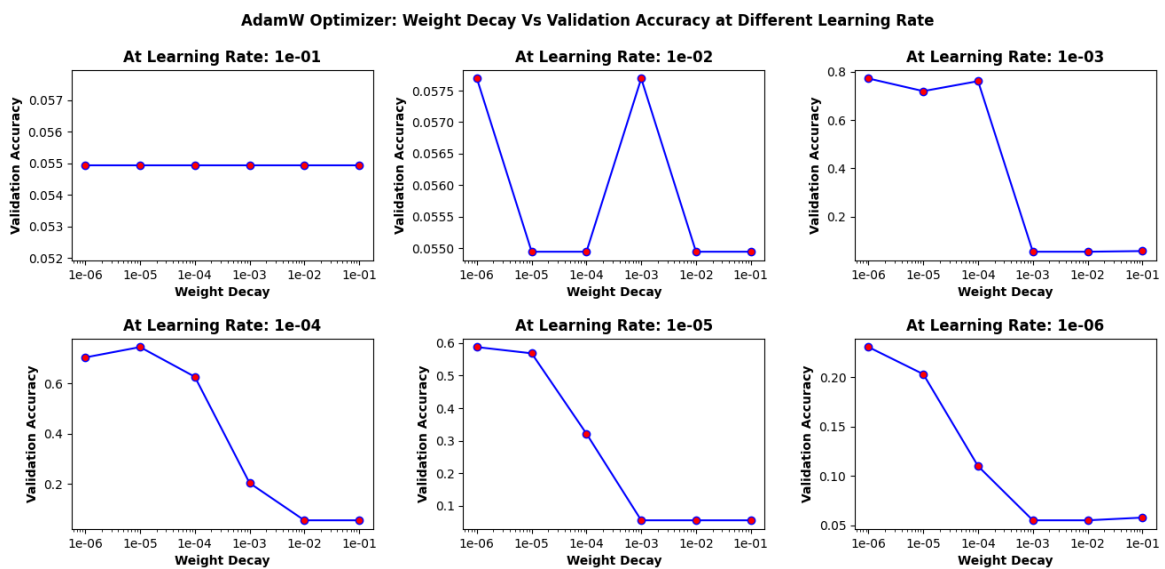
This section presents the experiments performed on hyperparameters, optimizers and activation functions as given in Table 4.2, Table 4.4, and Table 4.3 respectively.

#### ***5.1.2.1 Experiments on Optimizers with Learning Rates and Weight Decays***

This section presents the experiments performed on different optimizers as given in Table 4.4 at different learning rates and weight decays as given in Table 4.2. The optimizer along with learning rate and weight decay was selected based on validation accuracy.

### 5.1.2.1.1 Experiments on AdamW Optimizer with Different Learning Rates and Weight Decays

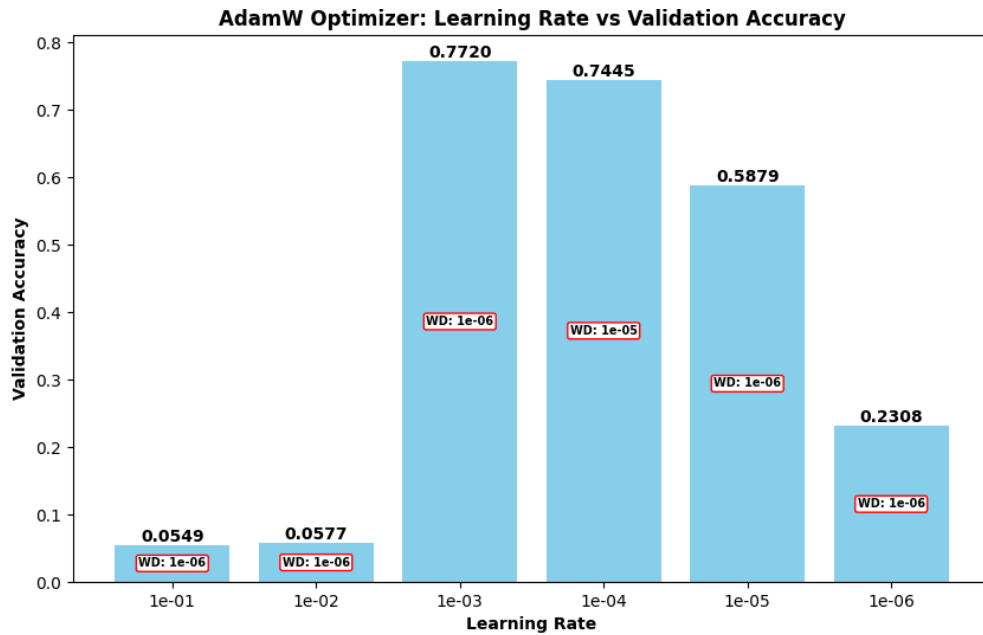
In this experiment, the AdamW (Adaptive Moment Estimation with Weight Decay) optimizer was experimented with different learning rates and weight decays. The result of each learning rate and weight decay are illustrated in Figure 5.5.



**Figure 5.5:** AdamW optimizer - Weight decay versus validation accuracy.

Figure 5.5 illustrates the validation accuracy of the model with change in weight decay at different learning rates. At learning rate 0.1, the validation accuracy remains constant for all values of weight decay, this shows that at learning rate of 0.1, the model is not learning effectively. At the learning rate of 0.01, the model shows improvement. At learning rate of 0.001 and smaller, the validation accuracy decreases with increase in weight decay. Figure 5.6

illustrates the maximum validation accuracy bar plot for AdamW at respective learning rates and weight decays.



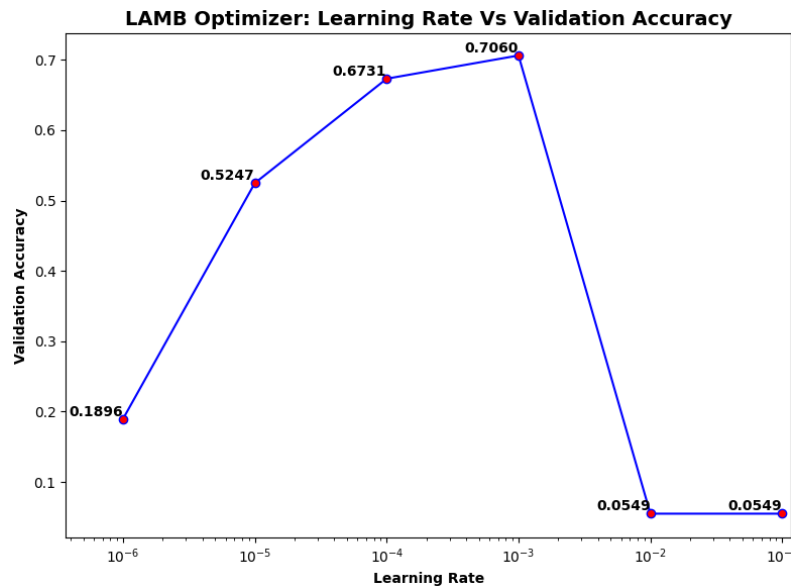
**Figure 5.6:** AdamW optimizer - Learning rate versus validation accuracy.

From Figure 5.6, For AdamW optimizer, the model gives a maximum validation of 77.20% at learning rate of 0.001 and weight decay of 0.000001.



### 5.1.2.1.2 Experiments on LAMB Optimizer with Different Learning Rates

In this experiment, the LAMB (Layer-wise Adaptive Moments optimizer for Batch training) optimizer was experimented with change of learning rates only, because LAMB optimizer does not use weight decay in its operation and is only dependent on the learning rate. The experimentation results for this optimizer are shown in Figure 5.7.

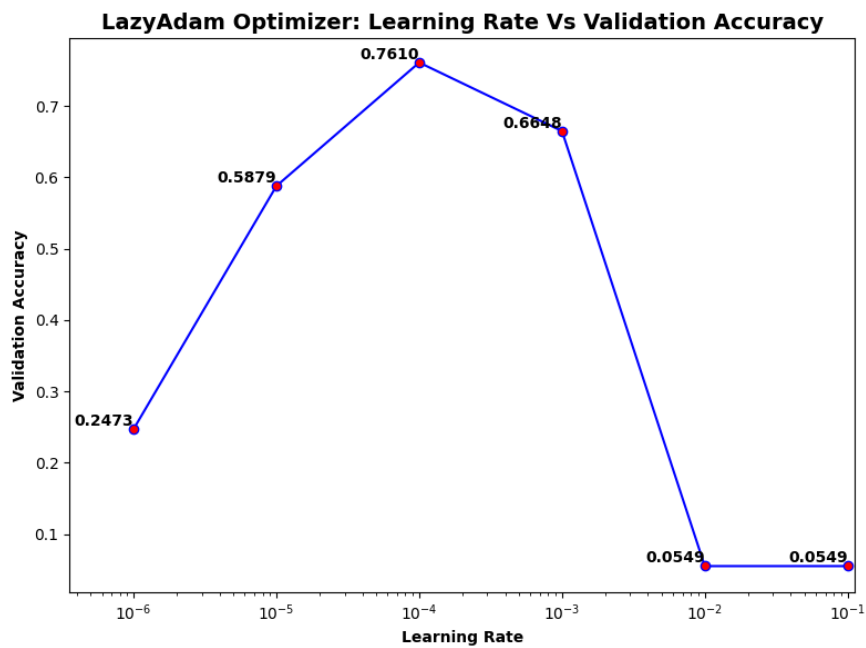


**Figure 5.7:** LAMB optimizer - Learning rate versus validation accuracy.

From Figure 5.7, as learning rate increases, the validation accuracy increases however if it is increased too much, the validation accuracy tends to drop. For LAMB optimizer, the model gives the maximum validation accuracy (70.60%) at a learning rate of 0.001.

### 5.1.2.1.3 Experiments on LazyAdam Optimizer with Different Learning Rates

In this experiment, the LazyAdam optimizer was experimented with change of learning rates as this optimizer is dependent on the learning rate. The result of change of learning rate on this optimizer is shown in Figure 5.8.

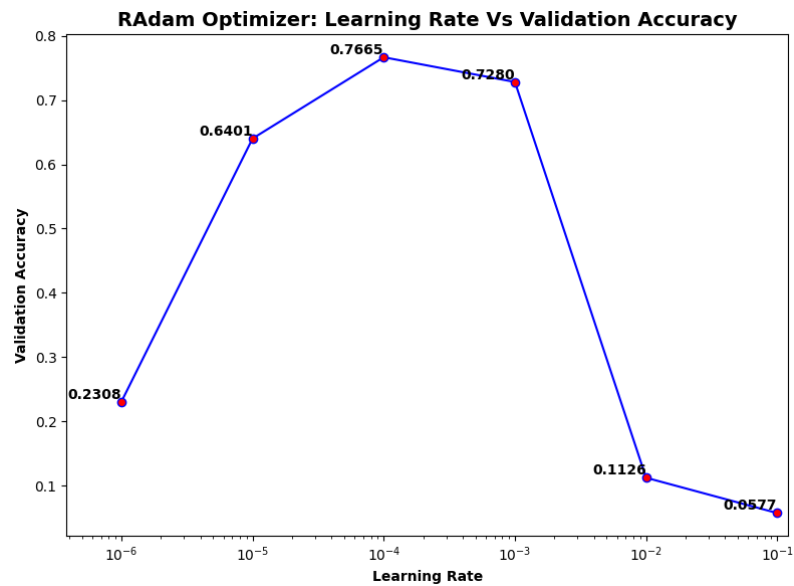


**Figure 5.8:** LazyAdam optimizer - Learning rate versus validation accuracy.

From Figure 5.8, as the learning rate increases, the validation accuracy also increases. After the learning rate of 0.0001, the validation accuracy drops with an increase in the learning rate. For LazyAdam optimizer, the model gives the maximum validation accuracy (76.10%) at a learning rate of 0.0001.

#### 5.1.2.1.4 Experiments on RAdam Optimizer with Different Learning Rates

In this experiment, the RAdam (Rectified Adam) optimizer was experimented with change of learning rates. Figure 5.9 illustrates the results of change of learning rate on RAdam optimizer.

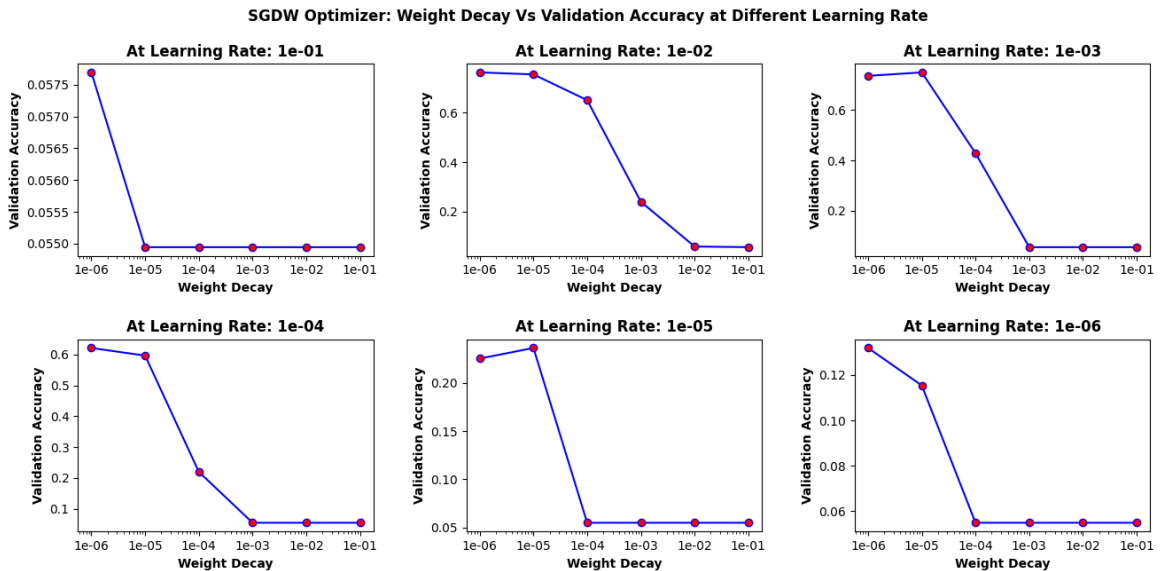


**Figure 5.9:** RAdam optimizer - Learning rate versus validation accuracy.

From Figure 5.9, as the learning rate increases, the validation accuracy also increases. After the learning rate of 0.0001, the validation accuracy drops with an increase in the learning rate. For RAdam optimizer, the model gives the maximum validation accuracy (76.65%) at a learning rate of 0.0001.

### 5.1.2.1.5 Experiments on SGDW Optimizer with Different Learning Rates and Weight Decays

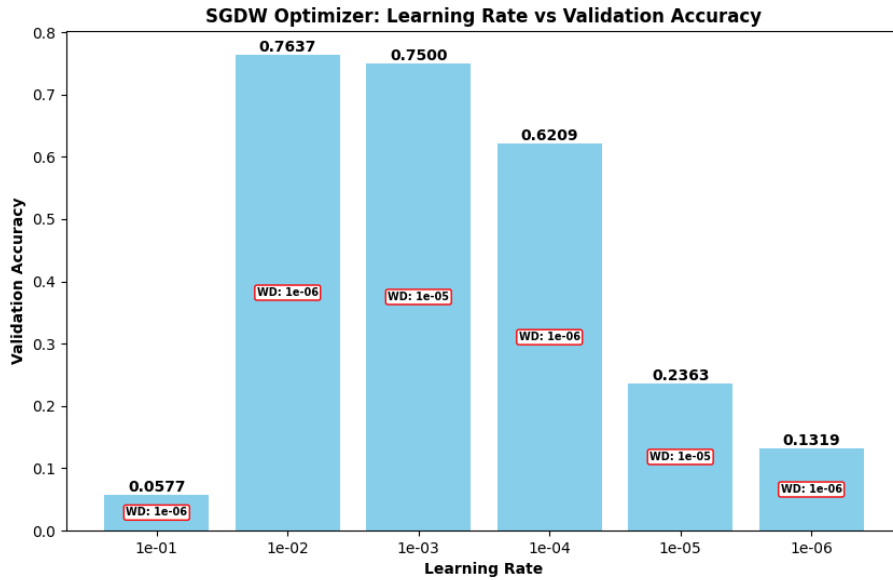
In this experiment, SGDW (Stochastic Gradient Descent with Weight Decay) optimizer was experimented with change of learning rate as well as weight decay. The results of change of weight decay versus validation accuracy at different learning rates are shown in Figure 5.10.



**Figure 5.10:** SGDW optimizer: Weight decay versus validation accuracy.

Figure 5.10 illustrates that at a learning rate of 0.1, as the weight decay increases, the validation accuracy comparatively remains constant, this shows that the model is not learning effectively. At other learning rates, the model gives similar behavior for change in weight decay. The model comparatively gives high validation accuracy for a smaller weight decay and decreases as weight decay increases. Figure 5.11 illustrates the maximum

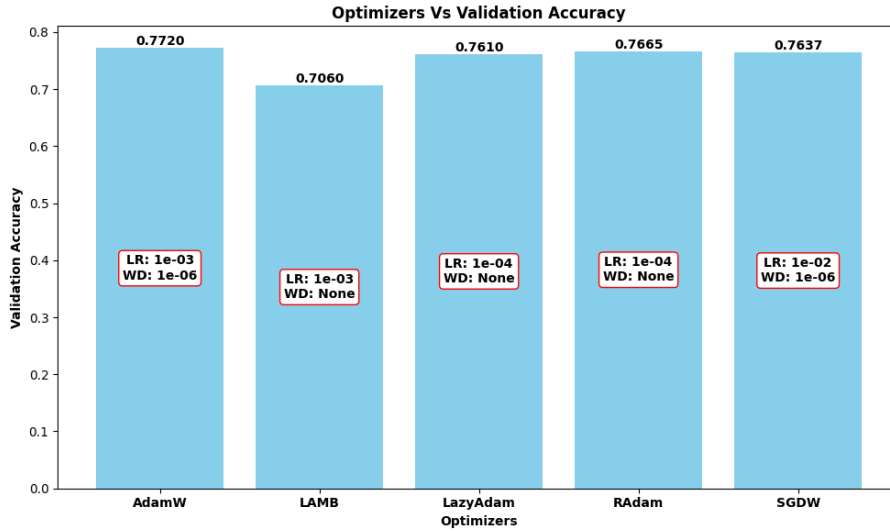
validation accuracy bar plot for SGDW optimizer at respective learning rates and weight decays.



**Figure 5.11:** SGDW optimizer: Learning versus validation accuracy.

Figure 5.11 shows the max validation accuracy at different learning rates and weight decays. Among all the learning rates, at a learning rate of 0.01, the model gives the maximum validation accuracy (76.37%) for the SGDW optimizer.

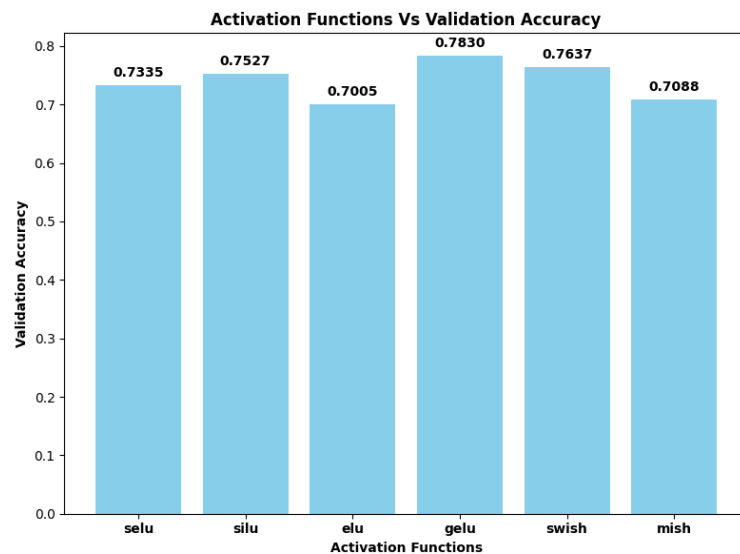
After performing the above experiments, a bar plot of max validation accuracy for each optimizer was made as shown in Figure 5.12 to perform their comparative analysis. From the bar plot, the validation accuracies are comparable to each other. Among all optimizers, AdamW with a learning rate of 0.001 and a weight decay of 0.000001 gives the max validation accuracy of 77.20%. So, we have selected AdamW optimizer for our proposed model.



**Figure 5.12:** Optimizers versus validation accuracy.

### 5.1.2.2 Experiments on Activation Functions

This section presents experiments performed for selection of activation function for our proposed model. Different activation functions are given in Table 4.3. The result of this experiment is illustrated in Figure 5.13.

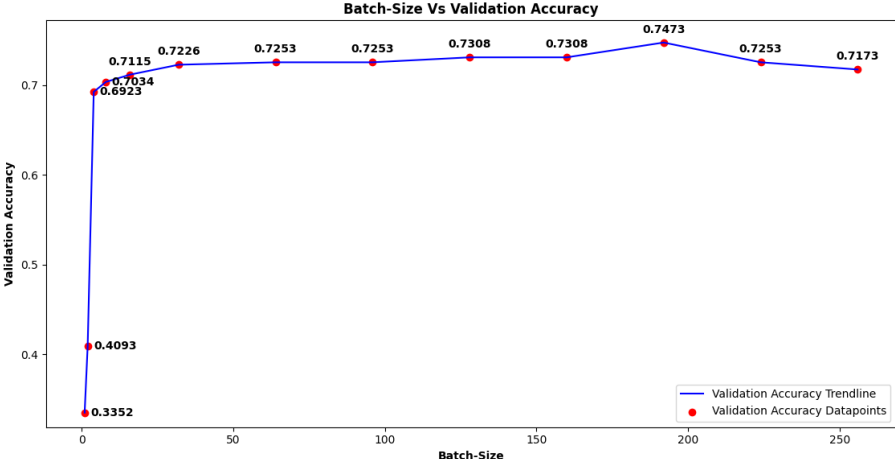


**Figure 5.13:** Activation functions versus validation accuracy.

Figure 5.13 illustrates the validation accuracy reported by the model on three different activation functions including selu, silu, elu, gelu, swish and mish. Among all activation functions, gelu has the highest validation accuracy and hence this activation function is utilized in our proposed model.

**5.1.2.3 Experiments on Batch Size**

This section presents experiments performed for selection of optimal value of batch size for our proposed model. The result of change of batch size is shown in Figure 5.14.



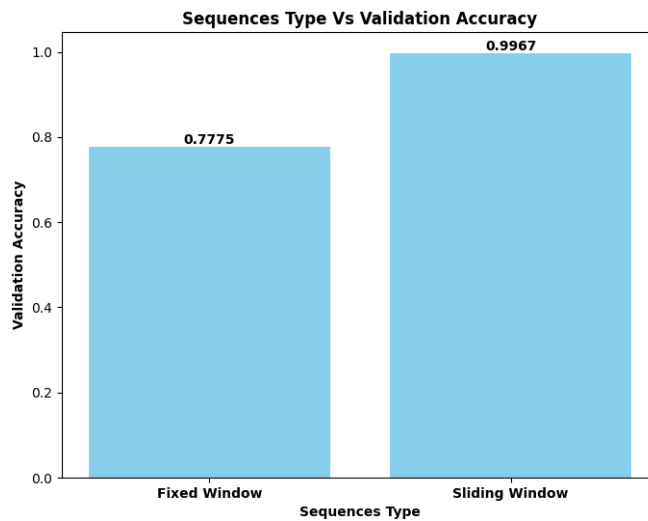
**Figure 5.14:** Batch-size versus validation accuracy.

Figure 5.14 illustrates the effect of change of batch-size on model’s validation accuracy. Generally, a smaller batch size takes longer time to train the model as compared to a larger batch size. Moreover, the performance at smaller batch sizes may not be as efficient as at larger batch sizes. This behavior is evident from the plot, at smaller batch-sizes the validation accuracy is very less and as batch-size increases the validation accuracy also increases. However, if batch-size is increased too much, the validation accuracy tends to drop, this behavior is also

shown in the plot for the large number of batch-sizes. From the plot, the maximum validation accuracy occurs at a batch-size of 192 and we have utilized this batch-size in our proposed model.

#### 5.1.2.4 Experiments on Sequences Type

This section presents experiments performed for selection of sequences type to perform training and performance evaluation of our proposed model. The result of sequences type is illustrated in Figure 5.15.



**Figure 5.15:** Sequences type versus validation accuracy.

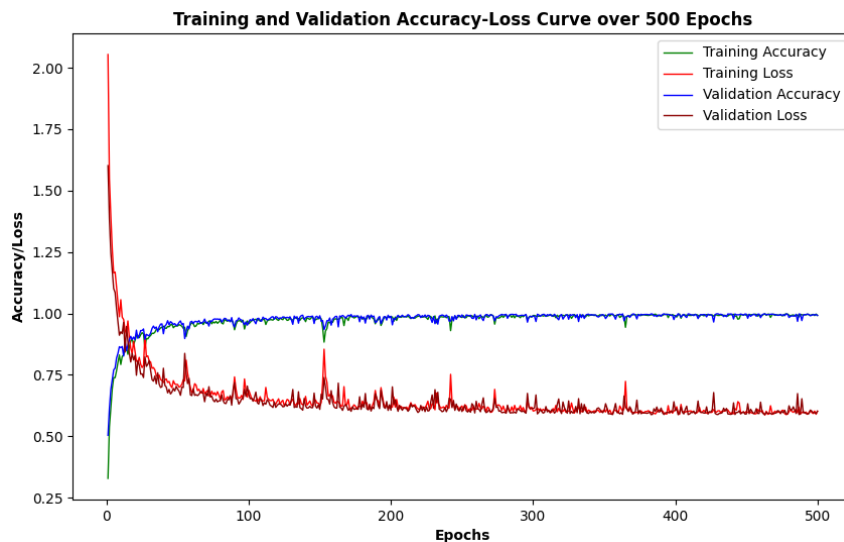
From Figure 5.15, Sliding Window Sequences has the maximum validation accuracy, it is due to the increased number of sequences that includes consecutive temporal patterns. It enhances the training of the model and hence increases the performance of the model. Alternatively, Fixed Window Sequences consists of subsequent temporal patterns that results in fewer sequences and less sequential data for training the model, hence the model's performance drops as compared to Sliding Window Sequences.



## 5.2 InterAcT Model - Performance Evaluation Results

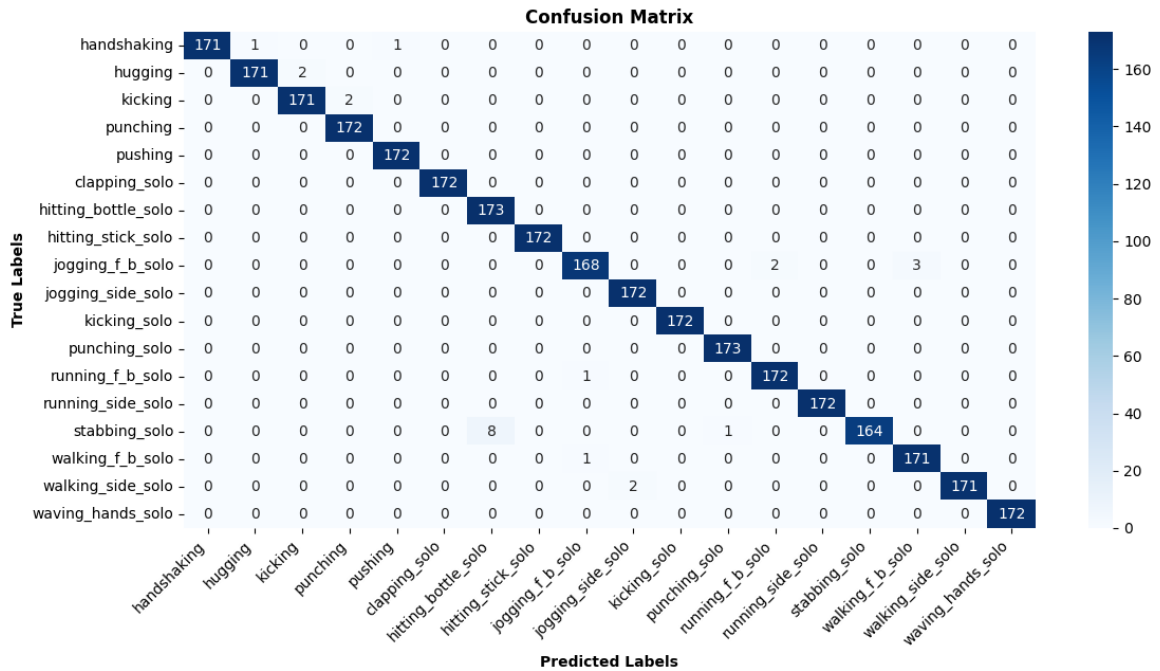
This section presents the results of the proposed InterAcT model with selected parameters as discussed in the previous section. This section includes training and validation accuracy-loss curve followed by the confusion matrix and performance metrics as listed in Table 3.3.

After running the proposed model with tuned parameters, the training and validation accuracy-loss curve is shown in Figure 5.16. From the plot, the training accuracy is improving over epochs which shows that the model is learning. To check the model's generalizability problems (underfitting and overfitting), validation accuracy-loss curve was obtained. From validation accuracy-loss curve, the validation accuracy is increasing while validation loss is decreasing. The behavior depicted by validation accuracy-loss curve is closer to that of training accuracy-loss curve which shows that the model is learning effectively and is generalizable.



**Figure 5.16:** Training and validation accuracy-loss curve.

After training the model, the trained model was evaluated on the test set. The confusion matrix for the test set is given in Figure 5.17.

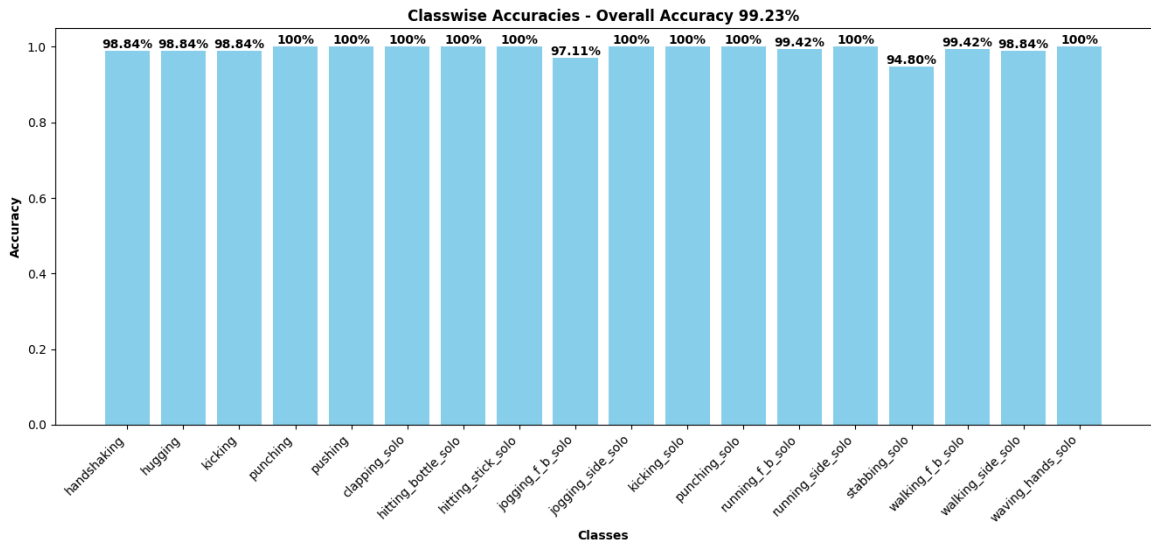


**Figure 5.17:** Confusion matrix evaluated on test-set.

The confusion matrix as illustrated in Figure 5.17 depicts the classification results for a test set consisting of 18 classes. For the 'handshaking' class, among 173 instances, 171 instances are accurately predicted while one instance misclassified with 'hugging' and one instance misclassified with 'pushing'. 'Hugging' sequences, totaling 173, has 171 correctly identified, with 2 instances misclassified as 'kicking'. 'Kicking' sequences (173) has 171 accurate predictions while 2 instances misclassified as 'punching'. All sequences of 'punching' (172), 'pushing' (172), 'clapping\_solo' (172), 'hitting\_bottle\_solo' (173) and 'hitting\_stick\_solo' (172) are accurately predicted. 'Jogging\_f\_b\_solo' sequences (173) has 168 correct predictions, with 2 sequences misclassified as 'running\_f\_b\_solo' and 3 sequences as 'walking\_fb\_solo'. All sequences of 'jogging\_side\_solo' (172), 'kicking\_solo' (172) and 'puching\_solo' (173) are correctly predicted. 'Running\_f\_b\_solo' has a total of

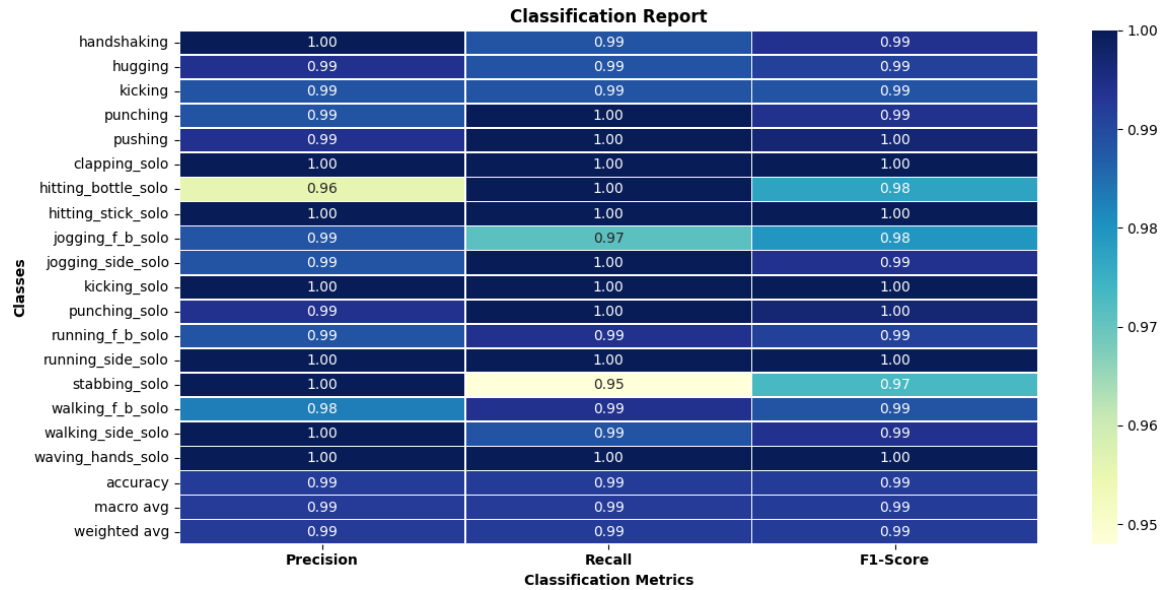
173 sequences out of which 172 are predicted correctly while one instance misclassified as 'jogging\_f\_b\_solo'. All sequences of 'running\_side\_solo' (172) are accurately predicted. Among 173 sequences of 'stabbing\_solo', 164 sequences are correctly identified while 8 sequences misclassified as 'hitting\_bottle\_solo' and one sequence misclassified as 'punching\_solo'. Out of 172 sequences of 'walking\_f\_b\_solo', 171 sequences are accurately predicted while one sequence misclassified as 'jogging\_f\_b\_solo'. 'Walking\_side\_solo' sequences (173) has 171 correct predictions while 2 sequences misclassified as 'jogging\_side\_solo'. All sequences of 'waving\_hands\_solo' sequences (172) are correctly predicted. In summary, the confusion matrix provides detailed insights into the classification performance across all 18 classes, highlighting correct predictions as well as instances of misclassification.

Using the confusion matrix, we obtained the class-wise accuracies which are shown in Figure 5.18. The model reports an accuracy of 99.31% for interactions classes, 99.20% for solo-actions and overall accuracy of 99.23%.



**Figure 5.18:** Class-wise accuracies.

Further classification metrics including precision, recall and f1-score are also obtained for each class. Figure 5.19 illustrates the classification report containing precision, recall and f1-score for each class.



**Figure 5.19:** Class-wise classification report.

### 5.3 InterAcT Benchmarking with State-of-the-art Deep Learning Models

This section gives the benchmark comparison tables of the proposed InterAcT model with state-of-the-art models. First, we have compared our proposed model with the models proposed in Action Transformer (AcT) given in Table 5.3. Our proposed model was then compared with other models including 2P-GCN [71], LSTM [132], 3D-ResNet [133] and 3D-CNN [134] along with the Action Transformer (AcT) [118] models as given in Table 5.4.

**Table 5.3:** InterAcT model comparison with AcT models.

Model Architecture		Multi-head Attention Layers	Encoder Layers	Embedded Dimensions	Drop-out Percentage	MLP Heads Dimensions
AcT [118]	micro	1	4	64	0.30	256
	small	2	5	128	0.30	256
	base	3	6	192	0.30	256
	large	4	6	256	0.40	512
<b>Ours</b>		<b>1</b>	<b>3</b>	<b>56</b>	<b>0.10</b>	<b>96</b>

**Table 5.4:** InterAcT model comparison with SOTA models.

Model		Model Parameters (M)	Model Flops (G)	Test Evaluation Time (s)	Test Accuracy	
GCN	2PGCN [71]	4.0300	2.5100	8.900	0.9337	
	LSTM [132]	4.1566	0.1644	7.6194	0.9774	
CNN	3D ResNet [133]	33.1567	0.3241	1.2067	0.9921	
	3D CNN [134]	0.3524	0.0266	0.9509	0.9920	
Transformer	AcT [118]	micro	0.2277	0.0701	1.0520	0.9353
		small	1.0419	0.1752	1.4881	0.9893
		base	2.7426	0.3153	2.1741	0.9907
		large	4.9052	0.4208	3.0388	0.9558
	<b>Ours</b>		<b>0.0795</b>	<b>0.0389</b>	<b>0.2013</b>	<b>0.9923</b>

From Table 5.3 and Table 5.4, our proposed model outperforms the other state-of-the-art models in terms of model complexity and evaluation time. Our model is lightweight and efficient, having 0.0795M training parameters and 0.0389G flops. Our model reports an accuracy of 99.23% with an evaluation time of 0.2013 seconds on the test set. From the comparative analysis, our model is well-suited for deployment in real-time applications. We call our proposed model 'InterAcT' because it has the capabilities to recognize both human-human interactions and solo-actions and the model architecture is named as 'nano' because it offers the least model complexity as compared to other models of Action Transformer (AcT) [118].

## **CHAPTER 6: CONCLUSIONS AND FUTURE RECOMMENDATIONS**

In this research study, we have proposed an efficient and effective novel skeletal-based transformer called 'InterAcT' model which is capable of recognizing solo actions as well as human-human interactions in Aerial Grayscale videos. We have utilized the action transformer (AcT) as a baseline model and developed an architecture 'nano' that offers less model complexity and computational cost. For keypoints extraction, we have utilized Ultralytics YOLO v8 pose estimation model on Ut-Interaction and Drone-Action datasets to extract 2D keypoints data that are fed into the proposed model. Our model gives an overall accuracy of 99.23% with an evaluation time of 201.3 milli seconds. Compared with other state-of-the-art models, our model outperforms 2P-GCN (93.37%, 8900 milli seconds), LSTM (97.74%, 7619.4 milli seconds), 3D ResNet (99.21%, 1206.7 milli seconds), 3D CNN [134] (99.20%, 950.9 milli seconds) and AcT models (micro: 93.53%, 1052 milli seconds – small: 98.93%, 1488.10 milli seconds – base: 99.07%, 2174.10 milli seconds – large: 95.58%, 3038.8 milli seconds). Our model has the potential to work well with RGB videos as well. The key strengths of our model are its lightweight architecture and performance making it deployable for real-time applications.

In future, our model can be extended to gestures recognition, human-human interactions between more than two persons, human-object interactions and group analysis. Our model classification performance can be improved by utilizing more keypoints extracted on finer level i.e. to utilize 3D keypoints and to increase number of person(s) keypoints (for example: adding facial keypoints for emotions/gestures recognition, adding

hands keypoints for human-human or human-object interactions). Person and object tracking algorithms can be employed to track each person(s)/object(s) to easily extract respective person(s)/object(s) keypoints which will help in keypoints concatenation module to overcome the multi-persons/multi-objects/person(s)-object(s) keypoints mixing and to make an ordered sequential data which will help the model to understand the patterns more effectively. Our model can be utilized for recognition using multi-modal data.

Our future recommendations can be summarized as follows:

1. Our model can be extended to Gesture recognition, Multi-persons Human-Human Interactions, Human-Object Interactions and Group Analysis.
2. Improving classification performance by utilizing finer level keypoints, such as 3D keypoints or increasing the number of keypoints for individuals (e.g., facial keypoints for emotions/gestures recognition, hand keypoints for interactions).
3. Employing person and object tracking algorithms to track each person/object to extract respective keypoints. It also helps to overcome multi-person/multi-object keypoints mixing.
4. Utilizing the model for recognition using multi-modal data.



## REFERENCES

- [1] M. S. Ryoo and J. K. Aggarwal, "UT-Interaction Dataset, ICPR contest on Semantic Description of Human Activities (SDHA)," 2010.
- [2] A. G. Perera, Y. W. Law, and J. Chahl, "Drone-action: An outdoor recorded drone video dataset for action recognition," *Drones*, vol. 3, no. 4, pp. 1–16, Dec. 2019, doi: 10.3390/drones3040082.
- [3] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLO," 2023. [Online]. Available: <https://ultralytics.com>
- [4] A. Kapitanov, A. Makhlyarchuk, and K. Kvanchiani, "HaGRID – HAnd Gesture Recognition Image Dataset," Mar. 2022.
- [5] Y.-W. Chao, Z. Wang, Y. He, J. Wang, and J. Deng, "HICO: A Benchmark for Recognizing Human-Object Interactions in Images," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015.
- [6] W. Choi, K. Shahid, and S. Savarese, "What are they doing?: Collective Activity Classification Using Spatio-Temporal Relationship Among People," in *9th International Workshop on Visual Surveillance (VSWS09) in conjunction with ICCV 2009*, 2009. [Online]. Available: [http://www.cis.upenn.edu/~jshi/ped\\_html/Papers/WChoi\\_VSWS\\_ICCV09.pdf](http://www.cis.upenn.edu/~jshi/ped_html/Papers/WChoi_VSWS_ICCV09.pdf)

- [7] A. Shah, J. B. Lamare, T. N. Anh, and A. Hauptmann, "CADP: A Novel Dataset for CCTV Traffic Camera based Accident Analysis," in *International Workshop on Traffic and Street Surveillance for Safety and Security*, Nov. 2018.
- [8] Y. Chen and Y. Xue, "A deep learning approach to human activity recognition based on single accelerometer," in *Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on*, 2015, pp. 1488–1492.
- [9] R. Chavarriaga *et al.*, "The opportunity challenge: A benchmark database for on-body sensor-based activity recognition," *Pattern Recognit Lett*, vol. 34, pp. 2033–2042, 2013.
- [10] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A survey," *Pattern Recognit Lett*, vol. 119, pp. 3–11, 2019, doi: <https://doi.org/10.1016/j.patrec.2018.02.010>.
- [11] P. Vepakomma, D. De, S. K. Das, and S. Bhansali, "A-wristocracy: Deep learning on wrist-worn sensing for recognition of user complex activities," in *2015 IEEE 12th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*, 2015, pp. 1–6.
- [12] M. Zeng *et al.*, "Convolutional neural networks for human activity recognition using mobile sensors," in *Mobile Computing, Applications and Services (MobiCASE), 2014 6th International Conference on*, 2014, pp. 197–205.

- [13] B. Almaslukh, J. AlMuhtadi, and A. Artoli, “An effective deep autoencoder approach for online smartphone-based human activity recognition,” *International Journal of Computer Science and Network Security*, vol. 17, p. 160, 2017.
- [14] G. E. Hinton, S. Osindero, and Y. W. Teh, “A fast learning algorithm for deep belief nets,” *Neural Comput*, vol. 18, pp. 1527–1554, 2006.
- [15] N. Y. Hammerla, S. Halloran, and T. Ploetz, “Deep, convolutional, and recurrent models for human activity recognition using wearables,” in *IJCAI*, 2016.
- [16] F. J. Ordoñez and D. Roggen, “Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition,” *Sensors*, vol. 16, p. 115, 2016.
- [17] T. Zebin, P. J. Scully, and K. B. Ozanyan, “Human activity recognition with inertial sensors using a deep learning approach,” in *SENSORS*, 2016, pp. 1–3.
- [18] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, “Exploiting multichannels deep convolutional neural networks for multivariate time series classification,” *Front Comput Sci*, vol. 10, pp. 96–112, 2016.
- [19] C. Liu, L. Zhang, Z. Liu, K. Liu, X. Li, and Y. Liu, “Lasagna: towards deep hierarchical understanding and searching over mobile sensing data,” in *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*, 2016, pp. 334–347.

- [20] Y. Kong and Y. Fu, "Human Action Recognition and Prediction: A Survey," *Int J Comput Vis*, vol. 130, no. 5, pp. 1366–1401, 2022, doi: 10.1007/s11263-022-01594-9.
- [21] C. Schüldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local SVM approach," in *IEEE ICPR*, 2004.
- [22] H. Wang, D. Oneata, J. Verbeek, and C. Schmid, "A robust and efficient video representation for action recognition," *IJCV*, 2015.
- [23] C. Sminchisescu, A. Kanaujia, Z. Li, and D. Metaxas, "Conditional models for contextual human motion recognition," in *International Conference on Computer Vision*, 2005.
- [24] C. Feichtenhofer, A. Pinz, and R. P. Wildes, "Spatiotemporal multiplier networks for video action recognition," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 7445–7454.
- [25] S. N. Boualia and N. Essoukri Ben Amara, "Pose-based Human Activity Recognition: a review," in *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*, 2019, pp. 1468–1475. doi: 10.1109/IWCMC.2019.8766694.
- [26] A. Bobick and J. Davis, "The recognition of human movement using temporal templates," *IEEE Trans Pattern Analysis and Machine Intelligence*, vol. 23, no. 3, pp. 257–267, 2001.

- [27] D. Weinland, R. Ronfard, and E. Boyer, “Free viewpoint action recognition using motion history volumes,” *Computer Vision and Image Understanding*, vol. 104, no. 2–3, pp. 249–257, 2006.
- [28] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri, “Actions as space-time shapes,” *Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2247–2253, 2007.
- [29] A. Yilmaz and M. Shah, “Actions sketch: A novel action representation,” in *CVPR*, 2005.
- [30] B. D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *Proceedings of imaging understanding workshop*, 1981.
- [31] A. Efros, A. Berg, G. Mori, and J. Malik, “Recognizing action at a distance,” in *ICCV*, 2003, pp. 726–733.
- [32] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, “Behavior recognition via sparse spatio-temporal features,” in *ICCV VSPETS*, 2005.
- [33] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, “Action Recognition by Dense Trajectories,” in *IEEE conference on computer vision & pattern recognition*, Colorado Springs, United States, 2011, pp. 3169–3176. [Online]. Available: <http://hal.inria.fr/inria-00583818/en>

- [34] C. Harris and M. Stephens, "A combined corner and edge detector," in *Alvey vision conference*, 1988.
- [35] M. Bregonzio, S. Gong, and T. Xiang, "Recognizing action as clouds of space-time interest points," in *CVPR*, 2009.
- [36] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *CVPR*, 2008.
- [37] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, 2005.
- [38] H. Wang, M. M. Ullah, A. Kläser, I. Laptev, and C. Schmid, "Evaluation of local spatio-temporal features for action recognition," in *BMVC*, 2009.
- [39] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *IEEE International Conference on Computer Vision*, Sydney, Australia, 2013. [Online]. Available: <http://hal.inria.fr/hal-00873267>
- [40] Q. Shi, L. Cheng, L. Wang, and A. Smola, "Human action segmentation and recognition using discriminative semi-Markov models," *IJCV*, vol. 93, pp. 22–32, 2011.
- [41] B. Wu, C. Yuan, and W. Hu, "Human action recognition based on context-dependent graph kernels," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 2609–2616.

- [42] Y. Wang and G. Mori, “Hidden part models for human action recognition: Probabilistic vs. max-margin,” *PAMI*, 2010.
- [43] L. Wang and D. Suter, “Recognizing human activities from silhouettes: Motion subspace and factorial discriminative graphical model,” in *CVPR*, 2007.
- [44] W. Choi, K. Shahid, and S. Savarese, “Learning context for collective activity recognition,” in *CVPR*, 2011.
- [45] C. Yuan, W. Hu, G. Tian, S. Yang, and H. Wang, “Multi-task sparse learning with beta process prior for action recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 423–429.
- [46] J. Wang, Z. Liu, J. Chorowski, Z. Chen, and Y. Wu, “Robust 3D action recognition with random occupancy patterns,” in *ECCV*, 2012.
- [47] S. Ji, W. Xu, M. Yang, and K. Yu, “3D convolutional neural networks for human action recognition,” in *ICML*, 2010.
- [48] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, “Large-scale video classification with convolutional neural networks,” in *CVPR*, 2014.
- [49] R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. Russell, “ActionVLAD: Learning spatio-temporal aggregation for action classification,” in *CVPR*, 2017.

- [50] C. Feichtenhofer, A. Pinz, and R. P. Wildes, "Spatiotemporal multiplier networks for video action recognition," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 7445–7454.
- [51] J. Donahue *et al.*, "Long-term recurrent convolutional networks for visual recognition and description," in *CVPR*, 2015.
- [52] A. Ulhaq, N. Akhtar, G. Pogrebna, and A. Mian, "Vision Transformers for Action Recognition: A Survey," arXiv, United States, Sep. 2022.
- [53] S. N. Boualia and N. Essoukri Ben Amara, "Pose-based Human Activity Recognition: a review," in *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*, 2019, pp. 1468–1475. doi: 10.1109/IWCMC.2019.8766694.
- [54] D. Weinland, R. Ronfard, and E. Boyer, "Free viewpoint action recognition using motion history volumes," *Computer Vision and Image Understanding*, vol. 104, no. 2–3, pp. 249–257, Nov. 2006, doi: 10.1016/j.cviu.2006.07.013.
- [55] S. Yu, D. Tan, and T. Tan, "A Framework for Evaluating the Effect of View Angle, Clothing and Carrying Condition on Gait Recognition," 2006.
- [56] X. Chen, H. Guo, G. Wang, and L. Zhang, "Motion Feature Augmented Recurrent Neural Network for Skeleton-based Dynamic Hand Gesture Recognition," in *2017 IEEE International Conference on Image Processing (ICIP)*, Aug. 2017. doi: 10.1109/ICIP.2017.8296809.



- [57] A. Shahroudy, T. T. Ng, Q. Yang, and G. Wang, "Multimodal Multipart Learning for Action Recognition in Depth Videos," *IEEE Trans Pattern Anal Mach Intell*, vol. 38, no. 10, pp. 2123–2129, Oct. 2016, doi: 10.1109/TPAMI.2015.2505295.
- [58] A. Yilmaz, X. Li, and M. Shah, "Contour-Based Object Tracking with Occlusion Handling in Video Acquired Using Mobile Cameras," 2004.
- [59] P. Pareek and A. Thakkar, "A survey on video-based Human Action Recognition: recent updates, datasets, challenges, and applications," *Artif Intell Rev*, vol. 54, no. 3, pp. 2259–2322, Mar. 2021, doi: 10.1007/s10462-020-09904-8.
- [60] S. Rahman, J. See, and C. C. Ho, "Action recognition in low quality videos by jointly using shape, motion and texture features," in *2015 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, 2015, pp. 83–88. doi: 10.1109/ICSIPA.2015.7412168.
- [61] I. Jegham and A. Ben Khalifa, "Pedestrian detection in poor weather conditions using moving camera," in *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*, 2017, pp. 358–362. doi: 10.1109/AICCSA.2017.35.
- [62] S. Abu-El-Hajja and et al., "YouTube-8M: A Large-Scale Video Classification Benchmark," Sep. 2016, [Online]. Available: <http://arxiv.org/abs/1609.08675>
- [63] K. Chebli and A. Ben Khalifa, "Pedestrian Detection Based on Background Compensation with Block-Matching Algorithm," in *2018 15th International Multi-*

*Conference on Systems, Signals & Devices (SSD)*, 2018, pp. 497–501. doi: 10.1109/SSD.2018.8570499.

- [64] J.-B. Alayrac and others, “Self-Supervised MultiModal Versatile Networks,” Jun. 2020, [Online]. Available: <http://arxiv.org/abs/2006.16228>
- [65] L. Li, T. Nawaz, and J. Ferryman, “Performance analysis and formative assessment of visual trackers using PETS critical infrastructure surveillance datasets,” *J Electron Imaging*, vol. 28, no. 04, p. 1, Jul. 2019, doi: 10.1117/1.jei.28.4.043004.
- [66] G. Saleem, U. I. Bajwa, and R. H. Raza, “Toward human activity recognition: a survey,” *Neural Comput Appl*, vol. 35, no. 5, pp. 4145–4182, Feb. 2023, doi: 10.1007/s00521-022-07937-4.
- [67] F. Camarena, M. Gonzalez-Mendoza, L. Chang, and R. Cuevas-Ascencio, “An Overview of the Vision-Based Human Action Recognition Field,” *Mathematical and Computational Applications*, vol. 28, no. 2, p. 61, Apr. 2023, doi: 10.3390/mca28020061.
- [68] I. Jegham, A. Ben Khalifa, I. Alouani, and M. A. Mahjoub, “Vision-based human action recognition: An overview and real world challenges,” *Forensic Science International: Digital Investigation*, vol. 32, Mar. 2020, doi: 10.1016/j.fsidi.2019.200901.
- [69] T. Nawaz, A. Berg, J. Ferryman, J. Ahlberg, and M. Felsberg, “Effective evaluation of privacy protection techniques in visible and thermal imagery,” *J Electron Imaging*, vol. 26, no. 05, p. 1, Sep. 2017, doi: 10.1117/1.jei.26.5.051408.

- [70] S. Uddin, T. Nawaz, J. Ferryman, N. Rashid, Md. Asaduzzaman, and R. Nawaz, "Skeletal Keypoint-Based Transformer Model for Human Action Recognition in Aerial Videos," *IEEE Access*, vol. 12, pp. 11095–11103, 2024, doi: 10.1109/ACCESS.2024.3354389.
- [71] Z. Li, Y. Li, L. Tang, T. Zhang, and J. Su, "Two-Person Graph Convolutional Network for Skeleton-Based Human Interaction Recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 33, no. 7, pp. 3333–3342, 2023, doi: 10.1109/TCSVT.2022.3232373.
- [72] R. Xian, X. Wang, and D. Manocha, "MITFAS: Mutual Information based Temporal Feature Alignment and Sampling for Aerial Video Action Recognition," *arXiv preprint arXiv:2303.02575*, Mar. 2023, [Online]. Available: <http://arxiv.org/abs/2303.02575>
- [73] X. Wang and et al., "AZTR: Aerial Video Action Recognition with Auto Zoom and Temporal Reasoning," *arXiv preprint arXiv:2303.01589*, Mar. 2023, [Online]. Available: <http://arxiv.org/abs/2303.01589>
- [74] S. Li, X. He, W. Song, A. Hao, and H. Qin, "Graph Diffusion Convolutional Network for Skeleton Based Semantic Recognition of Two-Person Actions," *IEEE Trans Pattern Anal Mach Intell*, pp. 1–17, Jan. 2023, doi: 10.1109/tpami.2023.3238411.

- [75] R. Vrskova, R. Hudec, P. Kamencay, and P. Sykora, "Human Activity Classification Using the 3DCNN Architecture," *Applied Sciences (Switzerland)*, vol. 12, no. 2, Jan. 2022, doi: 10.3390/app12020931.
- [76] K. Nguyen *et al.*, "The State of Aerial Surveillance: A Survey," Jan. 2022, [Online]. Available: <http://arxiv.org/abs/2201.03080>
- [77] F. Gao, H. Xia, and Z. Tang, "Attention Interactive Graph Convolutional Network for Skeleton-Based Human Interaction Recognition," in *Proceedings - IEEE International Conference on Multimedia and Expo*, IEEE Computer Society, 2022. doi: 10.1109/ICME52920.2022.9859618.
- [78] B. Li, C. Tan, J. Wang, R. Qi, P. Qi, and X. Li, "Skeleton-Based Action Recognition with UAV Views," in *ACM International Conference Proceeding Series*, Nov. 2021, pp. 16–20. doi: 10.1145/3503961.3503964.
- [79] W. Sultani and M. Shah, "Human action recognition in drone videos using a few aerial training examples," *Computer Vision and Image Understanding*, vol. 206, May 2021, doi: 10.1016/j.cviu.2021.103186.
- [80] T. Singh and D. K. Vishwakarma, "A deeply coupled ConvNet for human activity recognition using dynamic and RGB images," *Neural Comput Appl*, vol. 33, no. 1, pp. 469–485, Jan. 2021, doi: 10.1007/s00521-020-05018-y.
- [81] J. Li, X. Xie, Y. Cao, Q. Pan, Z. Zhao, and G. Shi, "Knowledge embedded GCN for skeleton-based two-person interaction recognition," *Neurocomputing*, vol. 444, pp. 338–348, Jul. 2021, doi: 10.1016/j.neucom.2019.12.149.

- [82] M. Dong, Z. Fang, Y. Li, S. Bi, and J. Chen, "Ar3d: Attention residual 3D network for human action recognition," *Sensors*, vol. 21, no. 5, pp. 1–15, Mar. 2021, doi: 10.3390/s21051656.
- [83] Q. Xu, W. Zheng, Y. Song, C. Zhang, X. Yuan, and Y. Li, "Scene image and human skeleton-based dual-stream human action recognition," *Pattern Recognit Lett*, vol. 148, pp. 136–145, Aug. 2021, doi: 10.1016/j.patrec.2021.06.003.
- [84] X. Shu, J. Tang, G. J. Qi, W. Liu, and J. Yang, "Hierarchical Long Short-Term Concurrent Memory for Human Interaction Recognition," *IEEE Trans Pattern Anal Mach Intell*, vol. 43, no. 3, pp. 1110–1118, Mar. 2021, doi: 10.1109/TPAMI.2019.2942030.
- [85] W. Xin, R. Liu, Y. Liu, Y. Chen, W. Yu, and Q. Miao, "Transformer for Skeleton-based action recognition: A review of recent advances," *Neurocomputing*, vol. 537, pp. 164–186, 2023, doi: <https://doi.org/10.1016/j.neucom.2023.03.001>.
- [86] T. Hu, X. Zhu, S. Wang, and L. Duan, "Human interaction recognition using spatial-temporal salient feature," *Multimed Tools Appl*, vol. 78, no. 20, pp. 28715–28735, Oct. 2019, doi: 10.1007/s11042-018-6074-6.
- [87] "Cornell Activity Datasets: CAD-60 & CAD-120."
- [88] T. Ahmad, H. Mao, L. Lin, and G. Tang, "Action Recognition Using Attention-Joints Graph Convolutional Neural Networks," *IEEE Access*, vol. 8, pp. 305–313, 2020, doi: 10.1109/ACCESS.2019.2961770.

- [89] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, “NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016.
- [90] J. Liu, A. Shahroudy, M. Perez, G. Wang, L. Y. Duan, and A. C. Kot, “NTU RGB+D 120: A Large-Scale Benchmark for 3D Human Activity Understanding,” *IEEE Trans Pattern Anal Mach Intell*, vol. 42, no. 10, pp. 2684–2701, Oct. 2020, doi: 10.1109/TPAMI.2019.2916873.
- [91] W. Kay *et al.*, “The Kinetics Human Action Video Dataset,” May 2017, [Online]. Available: <http://arxiv.org/abs/1705.06950>
- [92] B. Ghanem *et al.*, “The ActivityNet Large-Scale Activity Recognition Challenge 2018 Summary,” Aug. 2018, [Online]. Available: <http://arxiv.org/abs/1808.03766>
- [93] J. Carreira, E. Noland, C. Hillier, and A. Zisserman, “A Short Note on the Kinetics-700 Human Action Dataset,” Jul. 2019, [Online]. Available: <http://arxiv.org/abs/1907.06987>
- [94] J. Yuan, Z. Liu, and Y. Wu, “Discriminative video pattern search for efficient action detection,” *IEEE Trans Pattern Anal Mach Intell*, vol. 33, no. 9, pp. 1000–9999, 2011, doi: 10.1109/TPAMI.2011.38.
- [95] B. L. Bhatnagar, X. Xie, I. A. Petrov, C. Sminchisescu, C. Theobalt, and G. Pons-Moll, “BEHAVE: Dataset and Method for Tracking Human Object Interactions,” Apr. 2022, [Online]. Available: <http://arxiv.org/abs/2204.06950>

- [96] T. Özzyer, D. S. Ak, and R. Alhajj, “Human action recognition approaches with video datasets—A survey,” *Knowl Based Syst*, vol. 222, Jun. 2021, doi: 10.1016/j.knosys.2021.106995.
- [97] R. Fisher, “The PETS04 surveillance ground-truth data sets,” Sep. 2004.
- [98] J. M. Chaquet, E. J. Carmona, and A. Fernández-Caballero, “A survey of video datasets for human action and activity recognition,” *Computer Vision and Image Understanding*, vol. 117, no. 6, pp. 633–659, 2013, doi: 10.1016/j.cviu.2013.01.013.
- [99] C. Demirdjian David and Varri, “Recognizing Gestures for Virtual and Real World Interaction,” in *Computer Vision Systems*, B. and P. J. H. Fritz Mario and Schiele, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 1–10.
- [100] M. Marszałek, I. Laptev, and C. Schmid, “Actions in context,” in *IEEE conference on computer vision & pattern recognition*, 2009.
- [101] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, “HMDB: a large video database for human motion recognition,” in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2011.
- [102] K. Soomro, A. Roshan Zamir, and M. Shah, “UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild,” 2012. [Online]. Available: <http://crcv.ucf.edu/data/UCF101.php>
- [103] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black, “Towards understanding action recognition.” [Online]. Available: <http://jhmdb.is.tue.mpg.de>.

- [104] K. K. Reddy and M. Shah, "Recognizing 50 human action categories of web videos," *Mach Vis Appl*, vol. 24, no. 5, pp. 971–981, 2013, doi: 10.1007/s00138-012-0450-4.
- [105] D. Tan, K. Huang, S. Yu, and T. Tan, "Efficient night gait recognition based on template matching," in *Proceedings - International Conference on Pattern Recognition*, 2006, pp. 1000–1003. doi: 10.1109/ICPR.2006.478.
- [106] KTH.se, "Recognition of Human Actions."
- [107] M. Martin *et al.*, "Drive&Act: A Multi-modal Dataset for Fine-grained Driver Behavior Recognition in Autonomous Vehicles." [Online]. Available: [www.driveandact.com](http://www.driveandact.com)
- [108] T. Li, J. Liu, W. Zhang, Y. Ni, W. Wang, and Z. Li, "UAV-Human: A Large Benchmark for Human Behavior Understanding with Unmanned Aerial Vehicles," *CoRR*, vol. abs/2104.00946, 2021, [Online]. Available: <https://arxiv.org/abs/2104.00946>
- [109] S. A. F. Manssor, S. Sun, and M. A. M. Elhassan, "Real-time human recognition at night via integrated face and gait recognition technologies," *Sensors*, vol. 21, no. 13, Jul. 2021, doi: 10.3390/s21134323.
- [110] A. Berg, J. Ahlberg, and M. Felsberg, "A thermal Object Tracking benchmark," in *2015 12th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2015, pp. 1–6. doi: 10.1109/AVSS.2015.7301772.



- [111] N. U. Huda, B. D. Hansen, R. Gade, and T. B. Moeslund, "The effect of a diverse dataset for transfer learning in thermal person detection," *Sensors (Switzerland)*, vol. 20, no. 7, Apr. 2020, doi: 10.3390/s20071982.
- [112] J. Imran and B. Raman, "Deep residual infrared action recognition by integrating local and global spatio-temporal cues," *Infrared Phys Technol*, vol. 102, p. 103014, Nov. 2019, doi: 10.1016/J.INFRARED.2019.103014.
- [113] P. Srihari and J. Harikiran, "Spatio-Temporal Information for Action Recognition in Thermal Video Using Deep Learning Model," *International journal of electrical and computer engineering systems*, vol. 13, pp. 669–680, Sep. 2022, doi: 10.32985/ijeces.13.8.7.
- [114] M. Ding, Y. Ding, L. Wei, Y. Xu, and Y. Cao, "Individual Surveillance Around Parked Aircraft at Nighttime: Thermal Infrared Vision-Based Human Action Recognition," *IEEE Trans Syst Man Cybern Syst*, vol. 53, no. 2, pp. 1084–1094, Feb. 2023, doi: 10.1109/TSMC.2022.3192017.
- [115] M. Bonetto, P. Korshunov, G. (Gianni) Ramponi, and T. Ebrahimi, "Privacy in mini-drone based video surveillance," Sep. 2015, pp. 1–6. doi: 10.1109/FG.2015.7285023.
- [116] M. Barekatin *et al.*, "Okutama-Action: An Aerial View Video Dataset for Concurrent Human Action Detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017, pp. 2153–2160. doi: 10.1109/CVPRW.2017.267.

- [117] S. Kapoor, A. Sharma, A. Verma, and S. Singh, “Aeriform in-action: A novel dataset for human action recognition in aerial videos,” *Pattern Recognit*, vol. 140, p. 109505, Aug. 2023, doi: 10.1016/J.PATCOG.2023.109505.
- [118] V. Mazzia, S. Angarano, F. Salvetti, F. Angelini, and M. Chiaberge, “Action Transformer: A self-attention model for short-time pose-based human action recognition,” *Pattern Recognit*, vol. 124, p. 108487, 2022, doi: <https://doi.org/10.1016/j.patcog.2021.108487>.
- [119] A. Dosovitskiy *et al.*, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale,” Oct. 2020, [Online]. Available: <http://arxiv.org/abs/2010.11929>
- [120] A. Vaswani *et al.*, “Attention is All you Need,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., Curran Associates, Inc., 2017. [Online]. Available: [https://proceedings.neurips.cc/paper\\_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf)
- [121] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jegou, “Training data-efficient image transformers & distillation through attention,” in *Proceedings of the 38th International Conference on Machine Learning*, M. Meila and T. Zhang, Eds., in *Proceedings of Machine Learning Research*, vol. 139. PMLR, Nov. 2021, pp. 10347–10357. [Online]. Available: <https://proceedings.mlr.press/v139/touvron21a.html>

- [122] C. Liu *et al.*, “A Novel Deep-Learning-Based Enhanced Texture Transformer Network for Reference Image Super-Resolution,” *Electronics (Basel)*, vol. 11, no. 19, 2022, doi: 10.3390/electronics11193038.
- [123] L. Dong, S. Xu, and B. Xu, “Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition,” in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, Institute of Electrical and Electronics Engineers Inc., Sep. 2018, pp. 5884–5888. doi: 10.1109/ICASSP.2018.8462506.
- [124] A. Berg, M. O’Connor, and M. T. Cruz, “Keyword Transformer: A Self-Attention Model for Keyword Spotting,” in *Proc. Interspeech 2021*, in Interspeech. ISCA, Aug. 2021, pp. 4249–4253. doi: 10.21437/Interspeech.2021-1286.
- [125] G. Yao, T. Lei, and J. Zhong, “A review of Convolutional-Neural-Network-based action recognition,” *Pattern Recognit Lett*, vol. 118, pp. 14–22, 2019, doi: <https://doi.org/10.1016/j.patrec.2018.05.018>.
- [126] T. Ahmad, L. Jin, X. Zhang, S. Lai, G. Tang, and L. Lin, “Graph Convolutional Neural Network for Human Action Recognition: A Comprehensive Survey,” *IEEE Transactions on Artificial Intelligence*, vol. 2, no. 2, pp. 128–145, 2021, doi: 10.1109/TAI.2021.3076974.
- [127] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for*

*Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, and T. Solorio, Eds., Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. doi: 10.18653/v1/N19-1423.

- [128] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh, “OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields,” *IEEE Trans Pattern Anal Mach Intell*, 2019.
- [129] K. Sun, B. Xiao, D. Liu, and J. Wang, “Deep High-Resolution Representation Learning for Human Pose Estimation,” in *CVPR*, 2019.
- [130] H.-S. Fang *et al.*, “AlphaPose: Whole-Body Regional Multi-Person Pose Estimation and Tracking in Real-Time,” *IEEE Trans Pattern Anal Mach Intell*, 2022.
- [131] C. Lugaresi *et al.*, “MediaPipe: A Framework for Building Perception Pipelines,” 2019.
- [132] S. Saeed, H. Akbar, T. Nawaz, H. Elahi, and U. Khan, “Body-Pose-Guided Action Recognition with Convolutional Long Short-Term Memory (LSTM) in Aerial Videos,” *Applied Sciences*, vol. 13, p. 9384, Mar. 2023, doi: 10.3390/app13169384.
- [133] H. Kataoka, T. Wakamiya, K. Hara, and Y. Satoh, “Would Mega-scale Datasets Further Enhance Spatiotemporal 3D CNNs?,” Apr. 2020, [Online]. Available: <http://arxiv.org/abs/2004.04968>

- [134] S. Ji, W. Xu, M. Yang, and K. Yu, "3D Convolutional Neural Networks for Human Action Recognition," *IEEE Trans Pattern Anal Mach Intell*, vol. 35, no. 1, pp. 221–231, 2013, doi: 10.1109/TPAMI.2012.59.