

DL-CAMS : Deep Learning Based Children Activity Monitoring System

by

Ashiful Nahar Bithi (181014006)
Mitu Akter (201014008)
Zeenat Akter Mim (201014098)

*Capstone project report (CSE499) submitted in partial fulfillment of the
requirements for the degree of*

Bachelor of Science in Computer Science and Engineering

Under the supervision of

Md. Ferdous Bin Hafiz



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF LIBERAL ARTS BANGLADESH

SUMMER 2024

© *Ashiful Nahar Bithi, Mitu Akter, and Zeenat Akter Mim*
All rights reserved

DECLARATION

Project Title	DL-CAMS : Deep Learning Based Children Activity Monitoring System
Authors	Ashiful Nahar Bithi, Mitu Akter, and Zeenat Akter Mim
Student IDs	181014006, 201014008, and 201014098
Supervisor	Md. Ferdous Bin Hafiz

We declare that this capstone project report entitled *DL-CAMS : Deep Learning Based Children Activity Monitoring System* is the result of our own work except as cited in the references. The capstone project report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Ashiful Nahar Bithi
181014006

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Mitu Akter
201014008

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Zeenat Akter Mim
201014098

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Date: October 14, 2024



Department of Computer Science and Engineering
University of Liberal Arts Bangladesh
Mohammadpur, Dhaka - 1207

CERTIFICATE

This is to certify that the capstone project report entitled **DL-CAMS : Deep Learning Based Children Activity Monitoring System**, submitted by **Ashiful Nahar Bithi** (Student ID: 181014006), **Mitu Akter** (Student ID: 201014008) and **Zeenat Akter Mim** (Student ID: 201014098) are undergraduate students of the **Department of Computer Science and Engineering** has been examined. Upon recommendation by the examination committee, we hereby accord our approval of it as the presented work and submitted report fulfill the requirements for its acceptance in partial fulfillment for the degree of *Bachelor of Science in Computer Science and Engineering*.

Md. Ferdous Bin Hafiz, Lecturer

Dept. of CSE, University of Liberal Arts Bangladesh

Shakib Mahmud Dipto, Lecturer

Dept. of CSE, University of Liberal Arts Bangladesh

Sovon Chakraborty, Lecturer

Dept. of CSE, University of Liberal Arts Bangladesh

Md. Aktaruzzaman Pramanik, Lecturer

Dept. of CSE, University of Liberal Arts Bangladesh

Place: Dhaka

Date: October 14, 2024

ACKNOWLEDGEMENTS

We would like to express our deep and sincere gratitude to our research supervisor, *Md. Ferdous Bin Hafiz*, for giving us the opportunity to conduct research and providing invaluable guidance throughout this work. His dynamism, vision, sincerity and motivation have deeply inspired us. He has taught us the methodology to carry out the work and to present the works as clearly as possible. It was a great privilege and honor to work and study under his guidance.

We are greatly indebted to our honorable teachers of the Department of Computer Science and Engineering at the University of Liberal Arts Bangladesh who taught us during the course of our study. Without any doubt, their teaching and guidance have completely transformed us to the persons that we are today.

We are extremely thankful to our parents for their unconditional love, endless prayers and caring, and immense sacrifices for educating and preparing us for our future. We would like to say thanks to our friends and relatives for their kind support and care.

Finally, we would like to thank all the people who have supported us to complete the project work directly or indirectly.

Ashiful Nahar Bithi, Mitu Akter and Zeenat Akter Mim

University of Liberal Arts Bangladesh

Date: October 14, 2024

Dedicated to
my loving Mother

– *Ashiful Nahar Bithi*

To my beloved parents
the source of my inspiration

– *Mitu Akter*

To ...
my mother
a good soul.

– *Zeenat Akter Mim*

ABSTRACT

Accidents and falls present a significant safety risk for children, often resulting in severe injuries. While existing fall detection systems have mainly focused on adults, children's unique physiological characteristics and activity patterns require specialized solutions. Therefore, we developed a dedicated system to meet these children's specific requirements. We have explored the potential of keypoint-based approaches using OpenPose and different RNN architectures such as - LSTM, GRU, Bi-LSTM, and Attention-based LSTM. By a comparative analysis, we identified the most effective model, achieving an impressive accuracy rate of 99%. We designed and implemented a functional software solution using our developed model that reliably identifies falls and alerts the caregiver. Additionally, we collected a dataset of children performing activities, including fall, to train our model on real-world scenarios. This dedicated dataset allowed us to fine-tune the model's performance and enhance its ability to specialize in children's activities. Our research contributes to the development of innovative technologies for child safety and well-being. We aim to reduce the risk of injuries and improve the overall quality of life for children by providing a reliable and effective fall detection system. Our system not only enhances digital safety but also aligns with the broader goal of sustainable development by prioritizing the well-being of future generations.

Keywords: Fall detection, Activity detection, Child safety, Deep learning, LSTM, GRU, Keypoint-based approach

Contents

1	Introduction	1
1.1	Background Literature	1
1.2	Problem statement	2
1.3	Aims, objectives and Motivation	3
1.3.1	Aims	3
1.3.2	Objective	3
1.3.3	Motivation	4
1.4	Project Specification	5
1.4.1	System Requirements	5
1.4.1.1	Functional Requirements	5
1.4.1.2	Non-Functional Requirements	6
2	Literature Review	7
2.1	Literature Review	9
2.2	Performance Evaluation Criterion	10
2.3	Justification of the Adopted Performance Evaluation Criterion	12
3	Methodology	17
3.1	Data Collection and Preprocessing	18
3.1.1	Initial Dataset Details	20
3.2	Model Configuration	21
3.3	Training, Testing and Validation	24
3.4	Limitation of Existing Dataset	25

4	Design and Development	26
4.1	Project Management and Financial Activities	26
4.1.1	Dataset Creation	26
4.1.1.1	Dataset Description	27
4.1.1.2	Creation Procedures	28
4.1.1.3	Creation Cost Analysis	28
4.1.2	System Development	29
4.1.2.1	Video Acquisition and Processing	30
4.1.2.2	OpenPose-Based Keypoint Extraction	30
4.1.2.3	Activity and Fall Detection	32
4.1.2.4	Alert System	34
4.1.2.5	System Cost Analysis	35
4.2	Adopted Tools and techniques	37
4.2.1	Project Management Tools and Techniques	37
4.2.1.1	Project Management Software	37
4.2.1.2	Team Communication	37
4.2.1.3	Development Methodology	38
4.2.2	Design Tools and Techniques	39
4.2.2.1	Canva	39
4.2.2.2	Draw.io	39
4.2.2.3	Dribbble	39
4.2.3	Development Tools and Techniques	40
4.2.3.1	Frontend	40
4.2.3.2	Backend	40
4.2.3.3	Database	41
4.2.3.4	Model and Dataset	42
4.2.3.5	Testing	42
4.3	Implementation	43
5	Result analysis	50
5.1	Experimental Setup and Metrics	50
5.2	Initial Result Investigation	51
5.3	Final Result Investigation	53
5.3.1	LSTM	53

5.3.2	GRU	54
5.3.3	Bi-LSTM	55
5.3.4	Attention-Based LSTM	56
5.3.5	Final Model Selection	57
6	Conclusions	61
6.1	Societal, health, safety, legal and cultural aspects	61
6.1.1	Societal Impact	61
6.1.2	Health and Safety Impact	61
6.1.3	Legal and Cultural Impact	62
6.1.4	Course of Action to Mitigate Concerns	62
6.2	Impact on Environment and Sustainability	62
6.3	Ethical and professional principles	62
6.3.1	Data Privacy and Security	62
6.3.2	Accuracy and Reliability	63
6.3.3	Ethical Use of Technology	63
6.3.4	Transparency and Consent	63
6.4	Brief Summary	63
6.5	Future works	64
	References	65

List of Figures

1.1	Analysis of Accidental Injuries Among Children at Home	5
2.1	Single person pose estimation using YOLO NAS Pose	14
2.2	Single person pose estimation using MediaPipe Pose	14
2.3	Single person pose estimation using OpenPose	15
2.4	Multiple person pose estimation using YOLO NAS Pose	15
2.5	Multiple person pose estimation using MediaPipe Pose	15
2.6	Multiple person pose estimation using OpenPose	16
2.7	Comparative Literature Summary	16
3.1	Project Methodology	17
3.2	Video Dataset Breakdown by Activity	18
3.3	Keypoint Annotation Workflow	19
3.4	LSTM Model Architecture	22
3.5	GRU Model Architecture	22
3.6	Bi-directional LSTM Model Architecture	23
3.7	Attention-Based LSTM Model Architecture	23
4.1	Screenshot of the consent form used in our study.	27
4.2	List of Recorded Children’s Activities	28
4.3	Overall system workflow	29
4.4	Total frames were reduced to 60 instead of 300 using the frame ex- traction algorithm	30
4.5	Visual representation of how does Openpose work	31
4.6	Comparison of Original Video Frame and Skeleton Representation .	31
4.7	Inference Time Comparison of Pose Estimation Libraries	32
4.8	In-depth LSTM cell architecture	33
4.9	In-depth GRU cell architecture	34

4.10 Visual Representation of Monthly Software Development Cost Break-down	36
4.11 Notion	37
4.12 Whatsapp and Google Meet	38
4.13 Scrum Sprint	38
4.14 Canva	39
4.15 Draw IO	39
4.16 Dribble	40
4.17 Frontend Development Tools and IDE	40
4.18 Backend Development Tools and IDE	41
4.19 MongoDB	42
4.20 Model and Dataset Tools	42
4.21 Postman	43
4.22 Key Features Design Workflow	43
4.23 Login Page	44
4.24 Home Page	44
4.25 Selected a Video	45
4.26 Activity Detection Example 1	45
4.27 Activity Detection Example 2	46
4.28 Activity Detection Example 3	46
4.29 Activity Detection Example 4	47
4.30 Fall Detection Example	47
4.31 No Child Detection Example	48
4.32 Profile Page	48
4.33 Setting Page	49
4.34 Setting Page in Night Mode	49
5.1 Confusion Matrix for LSTM Model on Initial Dataset	52
5.2 Confusion Matrix for Bi-LSTM Model on Initial Dataset	52
5.3 Confusion Matrix for GRU Model on Initial Dataset	52
5.4 Confusion Matrix for Attention-Based LSTM Model on Initial Dataset	53

List of Tables

2.1	Summarized feature extractions	11
2.2	Comparative study of model's performance	11
2.3	Comparative study of the used dataset	12
2.4	Comparative study of pose estimation tools	13
3.1	Breakdown of Dataset by Activity	20
3.2	Breakdown of Dataset into Training and Test Sets	20
3.3	Comparison of Trainable and Non-Trainable Parameters Across Four Models	24
4.1	Cost analysis table for dataset creation	28
4.2	One-Time Hardware Investment Breakdown	35
4.3	Monthly Software Development Cost Breakdown	36
5.1	Models Performance Analysis on Initial Dataset	51
5.2	Comparison of LSTM Model Performance with and without Fine-Tuning	54
5.3	Confusion Matrix comparison of LSTM Models on Our Dataset . . .	54
5.4	LSTM Model Performance Analysis: F1-Score Comparison	54
5.5	Comparison of GRU Model Performance with and without Fine-Tuning	55
5.6	Confusion Matrix comparison of GRU Models on Our Dataset	55
5.7	GRU Model Performance Analysis: F1-Score Comparison	55
5.8	Comparison of Bi-LSTM Model Performance with and without Fine-Tuning	55
5.9	Bi-LSTM Model Performance Analysis: F1-Score Comparison	56
5.10	Confusion Matrix comparison of Bi-LSTM Models on Our Dataset . .	56
5.11	Comparison of Attention-Based LSTM Model Performance with and without Fine-Tuning	56

5.12	Confusion Matrix comparison of Attention Based LSTM Models on Our Dataset	57
5.13	Attention-Based LSTM Model Performance Analysis: F1-Score Com- parison	57
5.14	Model Comparision for Final Selection	58
5.15	ROC Curve Analysis of Finalized Models	58
5.16	Train and Validation Accuracy Analysis of Finalized Models	59
5.17	Comparative Analysis of Accuracy Metrics for Selected Model and Existing Skeleton-Based Approaches	60

Chapter 1

Introduction

1.1 Background Literature

Human Activity Recognition (HAR), a field dedicated to deciphering human movement through technology, has its origins in the pursuit of more intuitive human-computer interactions and intelligent systems. Initially focusing on video analysis for simple activity recognition, HAR has evolved with the advent of new technologies, harnessing richer data sources to understand human behavior with greater sophistication. Fall detection, a prime example of HAR's capabilities, demonstrates its proficiency in accurately identifying patterns indicative of falls within the complex tapestry of human movement. Recently many people focused on fall detection, specifically because fall-related injuries can be both fatal or non-fatal, with the majority being non-fatal [Organization \(2021\)](#). According to the World Health Organization (WHO), falls pose a high risk of death or severe injury to two specific age groups: older individuals and children [Organization \(2021\)](#). Young children's natural curiosity often leads them to try things that can be dangerous, like climbing or exploring [Organization \(2021\)](#). This can cause them to fall and get hurt, from small scratches to serious broken bones. According to a BBC news report, children under the age of 12 generally lack the maturity to effectively respond to emergencies and should not be left unsupervised at home for extended periods [Morrison \(2016\)](#). If we look into the traditional child supervision methods, they often involve the presence of a caregiver or guardian. However, even with constant supervision, children are susceptible to accidents, such as falls [Ahmed et al. \(2022\)](#). Though careful attention can mitigate the risk of severe injuries, the need for alternative solutions is also crucial, particularly for parents who cannot be with their children

continuously. To address this, we propose an analytical system that can detect falls and monitor children’s activities simultaneously.

In this study, we propose using a deep learning model to develop a reliable and accurate solution for detecting children’s activities and falls. Our skeleton-based approach uses Openpose [Cao et al. \(2019\)](#) to extract key points from video data rather than directly processing raw images. This technique improved the system’s efficiency and preserved privacy.

As a detection model, we selected a Recurrent Neural Network (RNN) architecture, which is ideal for processing sequential data derived from videos. We processed the videos using our custom frame-skipping technique to enhance computational efficiency. The extracted keypoints were used as input for an RNN architecture, which included Long Short-Term Memory (LSTM) [Graves \(2013\)](#), Gated Recurrent Unit (GRU) [Cho \(2014\)](#), Bidirectional LSTM (Bi-LSTM) [Schuster & Paliwal \(1997\)](#), and an Attention Mechanism incorporated into LSTM (Attention-based LSTM) [Bahdanau \(2014\)](#), to classify activities in real-time.

To train our model on a dataset of children, we have collected a custom dataset engaging in activities such as sitting, walking, standing, bending, and falling. This ensures that our model can effectively handle a wide range of children’s movements. Furthermore, we developed the system with a user-friendly interface to make it accessible and easy to use.

1.2 Problem statement

The goal of this research is to design and develop a deep learning-based activity detection and monitoring system with high accuracy. The existing children’s activity and monitoring systems lack real-time alerts, particularly in incidents such as falls [Suzuki et al. \(2012\)](#); [Hosseini et al. \(2018\)](#). Additionally, their range of activities is very limited [Al-Azani et al. \(2022\)](#). To mitigate this problem, our proposed system can detect five activities and alert the caregiver in case of fall detection. The lack of publicly available datasets designed for children’s activities is also very prominent in the studies [Al-Azani et al. \(2022\)](#). Therefore, we collected a children’s activity dataset, including fall, to train our model and accurately reflect the nuances of child activity.

1.3 Aims, objectives and Motivation

The key features we aim to deliver with our system are -

- Real-time Monitoring
- Fall Detection
- Activity Tracking
- Alert System
- Remote Access

1.3.1 Aims

Develop a real-time deep learning-based activity monitoring system to detect children's activities and falls accurately. The system should send immediate alerts to caregivers or parents via mobile devices, enabling timely intervention and ensuring the safety of children.

1.3.2 Objective

- **Enhance Child Safety:** Accurately detect fall and monitor regular activities of children in real-time and enable caregivers to respond promptly to incidents, minimizing risks.
- **Support Caregivers:** By providing timely notifications of any incidents, enabling caregivers to prioritize their responses and allow caregivers to customize the system to meet their specific needs and preferences.
- **Contribute to Research:** Collect valuable data on children's activities, contributing to research on child safety and drive the development of more accurate and robust deep learning algorithms for activity recognition [Al-Azani et al. \(2022\)](#).
- **Implement Interactive Web App:** Develop a comprehensive and interactive web application using React, Python, Pandas and .Net Web API to enable in-depth exploration and analysis of children's activity data facilitating informed decision-making and supporting child development.

1.3.3 Motivation

Insufficient adult supervision, a notable concern flagged by the World Health Organization, significantly impacts the frequency of unintentional falls among young children at home and on playgrounds [Organization \(2021\)](#). Because children spend extended periods at home, they become more susceptible to home-related injuries, particularly falls. Young children are especially vulnerable due to their ongoing developmental stage, which affects their ability to perceive and judge accidents [Banerjee et al. \(2022\)](#). The high prevalence and severity of childhood falls worldwide underscore the critical need for effective prevention strategies.

In the United States, a survey found unintentional falls are a leading cause of emergency department visits among children, often resulting in traumatic brain injuries [Peterson & Thomas \(2021\)](#). Similar trends have been noted in other nations, such as Oman, where falls contribute significantly to accidental injuries in children [Rumhi et al. \(2020\)](#). According to researchers, falls commonly occur in the home environment, particularly in stairs, playgrounds, and bedrooms [Boegemann \(2021\)](#). Additionally, as we can see in Figure 1.1, the presence of monitoring is crucial in home accidents because higher rates of injuries occur when parents are absent or distracted [Ahmed et al. \(2022\)](#). The possibility of severe consequences, including traumatic brain injuries, further increases the demand to detect falls using advanced technology and mitigate their impact on children's health and well-being. Figure 1.1 summarizes how accidental injury affects children at home.

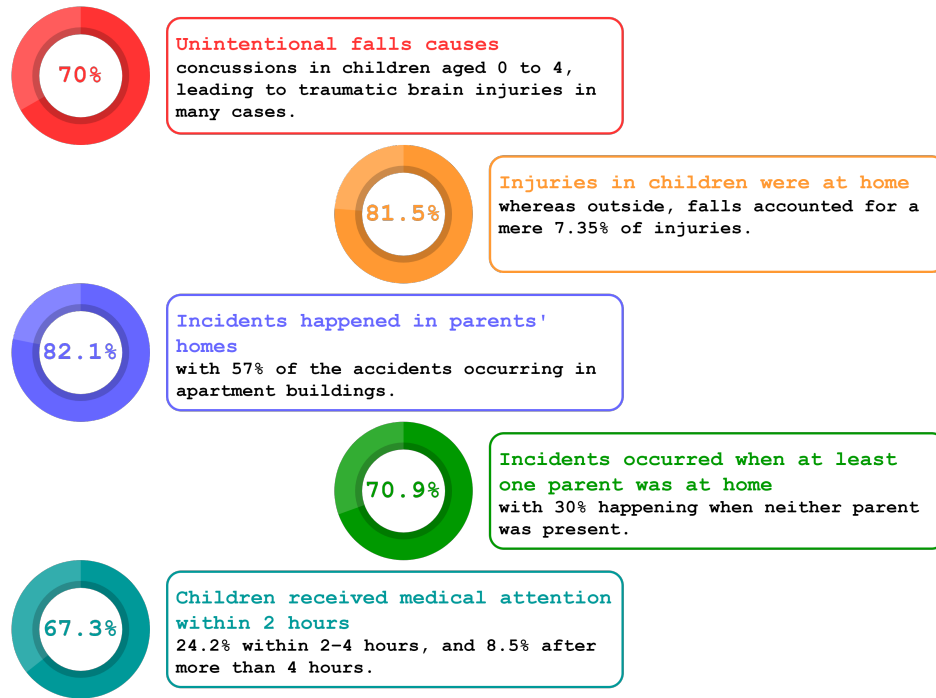


Figure 1.1: Analysis of Accidental Injuries Among Children at Home

1.4 Project Specification

The proposed system to detect children's falls and activities uses advanced deep learning algorithms and an NVIDIA Jetson Nano to analyze real-time video data from an IP camera. Furthermore, there is a strong emphasis on minimizing false positives and false negatives in our deep learning model to ensure reliable and accurate performance. The system's user-friendly interface further mainstreams the operations and its practicality.

1.4.1 System Requirements

System requirements are the features we promised to deliver after the development. Therefore, we have provided our targeted functional and non-functional requirements below.

1.4.1.1 Functional Requirements

- Real-time Fall Detection

- **Activity and Fall Classification**
- **Alert Generation**
- **User-Friendly Interface**

1.4.1.2 Non-Functional Requirements

- **Accuracy**
- **Security**
- **Reliability**
- **Cost-Effectiveness**

Chapter 2

Literature Review

For over a decade, the fall and activity detection system has mainly focused on adult populations, which created a gap in understanding and addressing the challenges of fall detection in children. While existing systems may be effective for adults, they often struggle to detect falls and activities in children due to their different physiological characteristics. Since they have distinct bone structures and activity patterns, they need specialized solutions [Figaji \(2017\)](#).

[Hendry et al. \(2023\)](#) reviewed 20 studies that analyzed the posture and movement of young children (0-5 years) using machine learning and other algorithmic approaches based on accelerometer data. It revealed that most of the studies focused on either infant activities or daily living activities. However, many research lacked dedicated fall detection or alarm mechanisms. Additionally, the studies often suffered from inadequate data description, which hindered reproducibility and comparability. This suggests that child-centric fall and activity detection still has significant room for improvement, particularly in developing systems that can accurately identify falls and provide timely alerts.

To further identify the research gap in this field, we conducted a thorough review of existing literature, which included examining various methods [Mubashir et al. \(2013\)](#), including:

- **Wearable sensor-based**
- **Ambient sensor-based**
- **Vision-based**

Wearable sensor-based methods utilize specialized devices attached to the human body to detect variations in multimedia tools and applications. Accelerometers, gyroscopes, or combinations thereof are commonly employed for fall detection. However, wearable devices are susceptible to environmental noise and may cause discomfort or be forgotten by the wearer [Khraief et al. \(2020\)](#). Additionally, shoe-based sensors can experience significant stress during user activity, increasing the risk of breakage [Yadav et al. \(2021\)](#). Inertial sensors, while valuable for motion tracking, have limitations such as sensor drift and sensitivity to location on the body. This can lead to inaccuracies in prolonged use and when fewer sensors are used. Traditional inertial sensor systems often require multiple sensors worn on different body parts, which can be intrusive and hinder natural movement of children [Chen et al. \(2017\)](#). Pervasive sensing devices like smartphones offer a less noticeable and more convenient alternative [Amor & James \(2015\)](#). Smartphones and smartwatches are well-suited for human activity recognition (HAR) due to their integrated sensors, communication capabilities, processing power, and ability to provide user feedback [Galán-Mercant et al. \(2019\)](#). Smartwatches may offer advantages over smartphones, as smartphones are typically ‘on the user’ for only 23% of the day and their position relative to the user’s body can be inconsistent [Van Laerhoven et al. \(2015\)](#); [Rawassizadeh et al. \(2014\)](#). A study involving 494 smartwatch users (52.8% female, average age = 35.89, age range = 18-76) highlights the potential for using wearable devices for HAR [Kang & Jung \(2021\)](#). However, children may be less likely to adopt wearable devices, especially sensor-based shoes or watches. On the other hand, mobile-based systems often face limitations in computational power, hindering the implementation of complex pattern detection algorithms on wearables. Power efficiency is another critical concern, as wearables must have a reasonable battery life to ensure user adoption for HAR [Yadav et al. \(2021\)](#). Ambient sensor systems, such as those utilizing vibration or pressure sensors under the bed or floor, offer a cost-effective and non-intrusive approach to human activity monitoring. However, they often face challenges in accurately detecting falls and may generate a higher rate of false alarms [Khraief et al. \(2020\)](#).

We can categorize vision-based approaches into two types - RGB and Depth camera-based. Depth cameras can capture depth information and are less sensitive to changes such as - color, texture, and lighting conditions. However, their range is often limited, and specialized depth sensors, such as the Microsoft Kinect, are

powerful but may not be practical for widespread use outside controlled laboratory settings due to their cost and equipment requirements [Yadav et al. \(2021\)](#).

Finally, since our target is detecting activities in indoor settings, using RGB video as input would be more practical, easier, and convenient for widespread application.

2.1 Literature Review

Lately, many studies proposed systems that use traditional RGB cameras to accurately recognize human actions and detect falls for the adult domain. [He et al. \(2021\)](#) developed a model to capture spatial, short-term, and long-term temporal patterns by analyzing video frames and optical flow maps using a Sample Representation Learner (SRL). They used a sparse sampling strategy that efficiently captured long-range temporal relationships. [Muhammad et al. \(2021\)](#) introduced a system that captures high-level spatial details and long-term temporal dependencies by combining a Dilated Convolutional Neural Network (DCNN) for spatial feature extraction with a Bi-LSTM network for temporal modeling. They added an attention mechanism to enhance performance by focusing on critical spatiotemporal features during classification. On the other hand, [Huu et al. \(2021\)](#) presented an approach to recognizing posture by combining machine learning and deep learning techniques. [Shu & Shu \(2021\)](#) employed a multi-camera-based system to monitor elderly individuals in care settings for fall detection. Their system showed flexibility by working with minimal calibration and accommodating up to eight cameras or twelve with a more powerful microcomputer like the Raspberry Pi 4. By utilizing conventional, affordable surveillance cameras, the system extracted key parameters including velocity, acceleration, also head and leg movements to correlate with fall events. [Vaiyapuri et al. \(2021\)](#) presented a system based on IoT incorporating SqueezeNet and the Salp Swarm Optimization (SSO) algorithm for fall detection. Once a fall event is detected, the system sends alerts to caretakers and hospital management via smartphones. [Núñez-Marcos & Arganda-Carreras \(2024\)](#) introduced a transformer-based model, specifically the Uniformer architecture, for real-time fall detection within video streams. [Chang et al. \(2021\)](#) proposed a modular system combining human detection and fall detection modules, utilizing OpenPose-light for joint recognition. The system employed human tracking, spine

offset, distance, and speed calculations to filter out non-fall actions, followed by LSTM analysis for fall judgment. [Gao et al. \(2023\)](#) integrated human pose estimation with a lightweight neural network, leveraging OpenPose for key point extraction and MobileNetV2 for fall detection. [Beddiar et al. \(2022\)](#) introduced a geometric approach, analyzing body angles and distances to classify falls, employing LSTM and Support Vector Machine (SVM) for classification. [McCall et al. \(2024\)](#) proposed a novel transformer-based framework for fall detection and prediction. The model was initially trained on a vast dataset of 2D human poses using transfer learning and then adapted for fall-specific applications. This approach demonstrated more efficiency than traditional machine learning and deep learning methods, including LSTM, CNN, ST-GCN, and PoseC3D.

2.2 Performance Evaluation Criterion

To assess the effectiveness of the proposed systems, we began by summarizing the feature extraction techniques employed in these studies, showed in table 2.1. This helped us to streamline and comprehend the processes they utilized to attain high performance.

Paper	Feature Extraction	Segmentation	Dimensionality Reduction
Shu & Shu (2021)	Isolated regions of motion using background subtraction, computed HOG features across each frame, utilized gradients and orientations from segmented regions to describe local patterns effectively.	Utilized temporal segmentation.	Implemented a two-stage feature selection - filter-based and wrapper-based methods.
He et al. (2021)	Extracted features using custom SLR, which employed DenseNet backbone networks that effectively learned hierarchical features encoding visual appearance and motion dynamics.	N/A	N/A
Muhammad et al. (2021)	Employed a DCNN to maintain comprehensive receptive fields and capture detailed spatial features, introduced Upgrade Features Learning Blocks, incorporating residual connections with DCNN layers, batch normalization, and Leaky ReLU activation functions, and used residual blocks with skip connections.	N/A	N/A
Vaiyapuri et al. (2021)	Used SqueezeNet, convolutional layers with filters to extract local features, followed by activation functions for non-linear mappings and the pooling layers. The final convolution layer's global features were converted to vectors for the fully connected layer.	N/A	Used the Salp Swarm Optimization (SSO) algorithm.
Chang et al. (2021)	Used OpenPose-light to detect an 18-joint 2D human skeleton key points, used spine line orientation, calculated speed and acceleration of the head based on the movement of the center of gravity of the head in consecutive frames, and aspect ratio of the bounding box.	N/A	N/A

Huu et al. (2021)	Used OpenPose algorithm with a MobileNetV2 network to identify and extract key points from the human body in the frame.	N/A	N/A
Gao et al. (2023)	Used OpenPose to extract and label human keypoints in images, and annotated these keypoints in the original images.	N/A	N/A
Beddiar et al. (2022)	Created feature vectors by concatenating calculated angles and distances from video frames, standardized these vectors through padding, converted the angles and distances into RGB macro images, with each channel representing different features, and processed images by pre-trained ResNet50 and AlexNet models to extract significant features.	Video frames used segmentation and localized body parts.	N/A
McCall et al. (2024)	Used OpenPose algorithm to extract 2D coordinates of 25 key points corresponding to different body parts for each frame, computed the position and velocity of all key points, and concatenated the position and velocity vectors for all keypoints to form a comprehensive feature vector for each frame.	N/A	N/A
Núñez-Marcos & Arganda-Carreras (2024)	Used a transformer-based approach where Dynamic Position Embedding provided a lightweight position encoding based on depthwise convolution, adaptable to varying sequence lengths, used Multi-Head Relation Aggregator, and used a token affinity matrix to capture the relationships between tokens.	N/A	N/A

Table 2.1: Summarized feature extractions

Next, in table 2.2 we summarized the models employed by the authors in their system and evaluated their overall accuracy.

Paper	Classification Model	Accuracy (Overall)
Shu & Shu (2021)	RVM	89%
He et al. (2021)	DB-LSTM	89.25%
Muhammad et al. (2021)	Bi-LSTM with attention	92.53%
Vaiyapuri et al. (2021)	IMEFD-ODCNN	99.76%
Chang et al. (2021)	LSTM with OpenPose-Light	98%
Huu et al. (2021)	MobilenetV2 + LSTM	92%
Gao et al. (2023)	MobileNetV2 + Pooling + Two FC + Keypoints Figure	99.10%
Beddiar et al. (2022)	Feature images + Resnet50 + SVM	96.10%
McCall et al. (2024)	Transformer model	97.95%
Núñez-Marcos & Arganda-Carreras (2024)	Transformer model with Feedforward Network	Binary - 99.17%, Multi - 93.17%

Table 2.2: Comparative study of model's performance

The performance of these models was evaluated using datasets that included individuals of varying ages. Many of these models were trained on publicly available

datasets, while only one-third of the studies collected their own data to either train or evaluate their model’s performance. As we aim to collect our own dataset, we examined the data collection methods used in these studies and summarized their dataset details in the table 2.3.

Paper	Dataset (Existing)	Dataset (Custom)	Activities	Included Fall	Data Access
Shu & Shu (2021)	No	Yes	2 activities (Fall, Non-Fall)	Yes	Public
He et al. (2021)	HMDB51 Kuehne et al. (2011), UCF101 Soomro (2012)	No	51 activities, 101 activities	No	Public
Muhammad et al. (2021)	UCF11 Liu et al. (2009), UCF Sports Shao et al. (2013), J-HMDB Jhuang et al. (2013)	No	11 activities, 10 activities, 21 activities	No	Public
Vaiyapuri et al. (2021)	URFD Kwolek & Kepski (2014), Multiple Cameras Fall Auvinet et al. (2010)	No	2 activities (Fall, Non-fall)	Yes	Public
Chang et al. (2021)	Multiple Cameras Fall Auvinet et al. (2010), URFD Kwolek & Kepski (2014), Le2i FD Charfi et al. (2012)	Yes for testing (10 subjects)	2 activities (Fall, Non-fall)	No	Public, not mentioned (testing data)
Huu et al. (2021)	MPII Andriluka et al. (2014)	Yes	7 activities	Yes	Public, not mentioned (testing data)
Gao et al. (2023)	URFD Kwolek & Kepski (2014), Le2i FD Charfi et al. (2012)	No	2 activities (Fall, Non-fall)	Yes	Public
Beddiar et al. (2022)	URFD Kwolek & Kepski (2014), Le2i FD Charfi et al. (2012)	No	2 activities (Fall, Non-fall)	Yes	Public
McCall et al. (2024)	CAUCAFall Guerrero et al. (2022), Gait Kour et al. (2022)	No	2 activities (Fall, Non-fall)	Yes	Public
Núñez-Marcos & Arganda-Carreras (2024)	UPFD Martínez-Villaseñor et al. (2019), URFD Kwolek & Kepski (2014)	No	2 activities, (Binary - 2 activities, Multi - 11 activities)	Yes	Public

Table 2.3: Comparative study of the used dataset

2.3 Justification of the Adopted Performance Evaluation Criterion

In our work with children, data privacy is of paramount importance. To ensure the utmost security and accuracy, we have carefully evaluated various approaches and selected a 2d keypoints-based methodology McCall et al. (2024); Chang et al. (2021); Huu et al. (2021); Gao et al. (2023). This approach involves extracting essential data points, known as keypoints, from each frame of a video McCall et al. (2024); Huu et al. (2021); Al-Azani et al. (2022). By transmitting only these keypoints to our detection model, we significantly reduce the amount of sensitive information

shared, thereby safeguarding children’s privacy while maintaining the effectiveness of our system.

Given the prevalence of OpenPose in the existing literature [McCall et al. \(2024\)](#); [Chang et al. \(2021\)](#); [Huu et al. \(2021\)](#); [Gao et al. \(2023\)](#), keeping that in mind, we conducted a comprehensive evaluation of alternative keypoints detection methods. Accurate fall detection is a critical component of our system, as false alarms could cause unnecessary distress for caregivers and parents. Recognizing the importance of precise keypoint detection in this context, we prioritized accuracy over efficiency and conducted a comparative analysis of existing 2D keypoint detection models to identify the most reliable pose estimation tool. As suggested by many researchers, we have planned to implement our system for multi-person detection [Muhammad et al. \(2021\)](#); [Yadav et al. \(2021\)](#). To achieve this, we have carefully selected the most suitable tools to detect the 2D keypoints of multiple individuals within a single frame. Table 2.4 presents a comparison between OpenPose, MediaPipe, AlphaPose, and YOLOv8 Pose.

Feature	OpenPose	MediaPipe Pose	AlphaPose	YOLOv8 (2D Pose)
2D Pose Estimation	Yes	Yes	Yes	Yes
Number of Body Keypoints	25 (body), 21 (each hand), 70 (face)	33	25	Varies
Multi-person Detection	Yes (Up to 25)	No	Yes (Up to 20)	No
Accuracy	High	High in single-person, moderate on multi-person	Low on complex scenes and multi-person, High in single-person	Varies
Speed	Moderate	Very fast	Fast	Faster than OpenPose, slower than MediaPipe Pose
Availability	Open-source, separate installation	Open-source, embedded in MediaPipe framework	Open-source	Open-source

Table 2.4: Comparative study of pose estimation tools

To get a practical experience and check performance on a dataset specific to children, we implemented OpenPose [Cao et al. \(2019\)](#), MediaPipe Pose [Google AI \(2020\)](#), and a YOLO variant called YOLO NAS Pose [Aharon et al. \(2021\)](#). We chose these tools because OpenPose is frequently used in the literature we reviewed, MediaPipe is renowned for its accuracy, and YOLO NAS Pose is a newly developed variant of YOLO.



Figure 2.1: Single person pose estimation using YOLO NAS Pose

As shown in the figures 2.1, 2.2, 2.3, OpenPose and MediaPipe got better results than YOLO NAS Pose in single-person detection, with MediaPipe showing the best performance in detecting single individuals.



Figure 2.2: Single person pose estimation using MediaPipe Pose

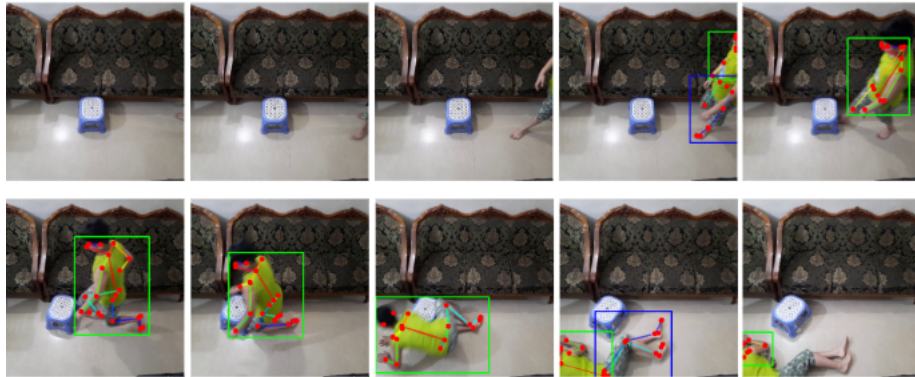


Figure 2.3: Single person pose estimation using OpenPose

Next, we evaluated all three tools' performance on multi-person detection by using frames with multiple individuals.



Figure 2.4: Multiple person pose estimation using YOLO NAS Pose



Figure 2.5: Multiple person pose estimation using MediaPipe Pose

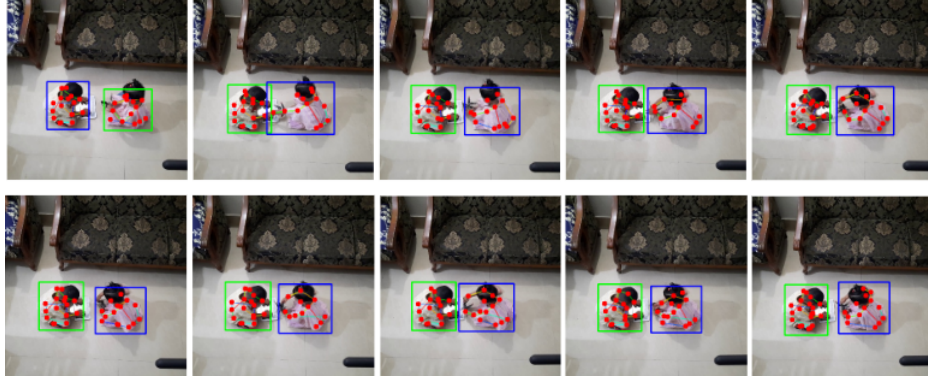


Figure 2.6: Multiple person pose estimation using OpenPose

The figures 2.4, 2.5, 2.6 show that OpenPose outperformed the others in this scenario. Therefore, based on its consistent performance in both single-person and multi-person detection in our real-world dataset, we selected OpenPose as the keypoint detection tool for our system.

Videos are sequential data, making Recurrent Neural Networks (RNNs) a natural choice [Keren & Schuller \(2016\)](#). Our literature review revealed a prevalence of LSTM models [He et al. \(2021\)](#); [Muhammad et al. \(2021\)](#); [Chang et al. \(2021\)](#); [Huu et al. \(2021\)](#), prompting us to explore various RNN architectures, including LSTM variants and GRUs, to identify the most effective model for our system.

Paper	Skeletal Based	Activity Detection	Fall Detection	Children Specific	Presented Dataset
Muhammad et al. (2021)	✗	✓	✗	✗	✗
He et al. (2021)	✗	✓	✗	✗	✗
Chang et al. (2021)	✓	✓	✗	✗	✓
Huu et al. (2021)	✓	✓	✓	✗	✓
McCall et al. (2024)	✓	✓	✓	✗	✗
Núñez-Marcos & Arganda-Carreras (2024)	✗	✓	✓	✗	✗
Gao et al. (2023)	✓	✓	✓	✗	✗
Beddiar et al. (2022)	✗	✓	✓	✗	✗
Ours	✓	✓	✓	✓	✓

Figure 2.7: Comparative Literature Summary

Lastly, Figure 2.7 summarizes our literature review and highlights the areas where we significantly outshine others.

Chapter 3

Methodology

To develop our system, we adopted a concurrent approach, working simultaneously on our dataset and system design. The cornerstone of our system is a sophisticated deep learning model designed to accurately detect falls in children. To train this model, we adhered to the methodological framework outlined in Figure 3.1.

In the following sections, we will delve into each step of the system development process, providing a detailed explanation of our approach.

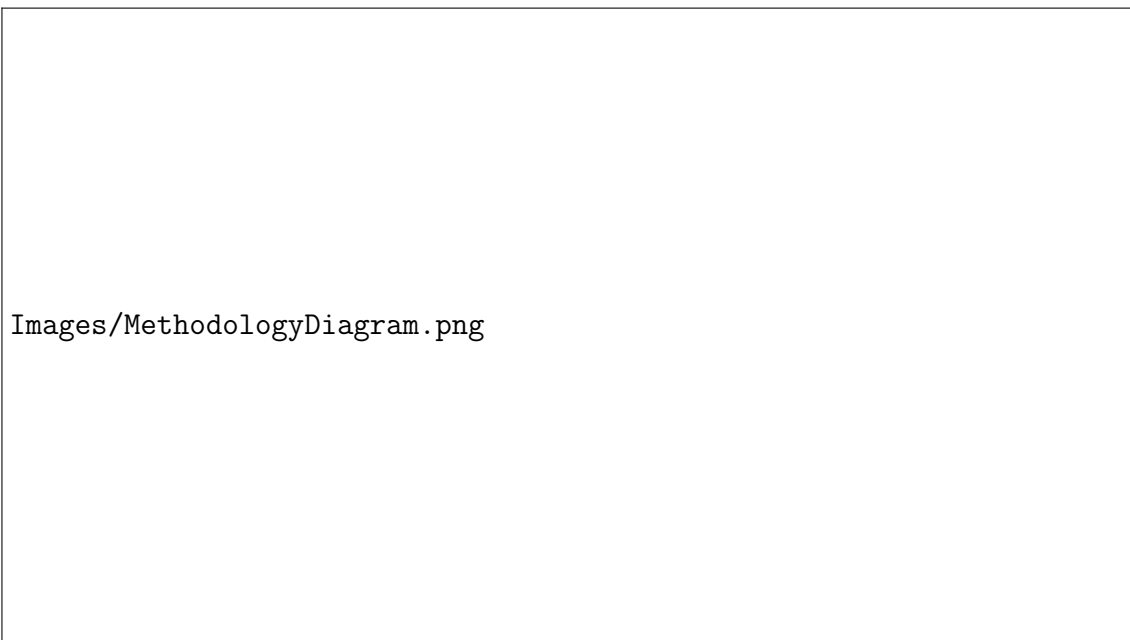


Figure 3.1: Project Methodology

3.1 Data Collection and Preprocessing

To train our model effectively for fall and activity detection, we needed a comprehensive dataset. While we were diligently collecting our own data, we initially utilized an existing activity dataset as a preliminary step. This allowed us to assess the model’s baseline performance and identify areas for improvement before focusing solely on our custom-collected data.

After collecting our dataset, which included the total number of videos as detailed in Figure 3.2, we proceeded with data preprocessing. This essential step involved transforming the raw data to align with our system’s specific requirements.

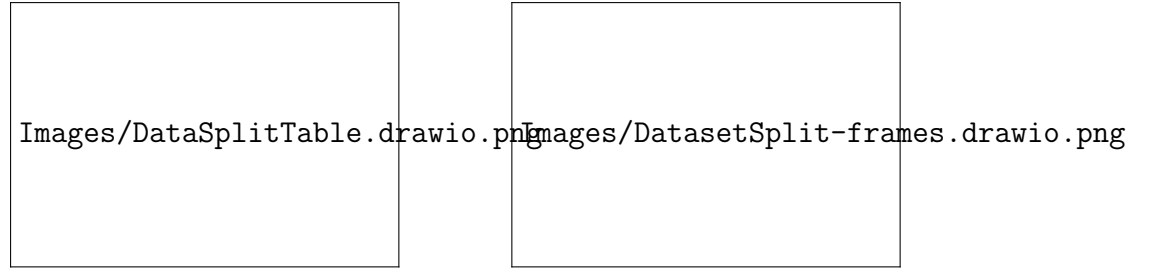
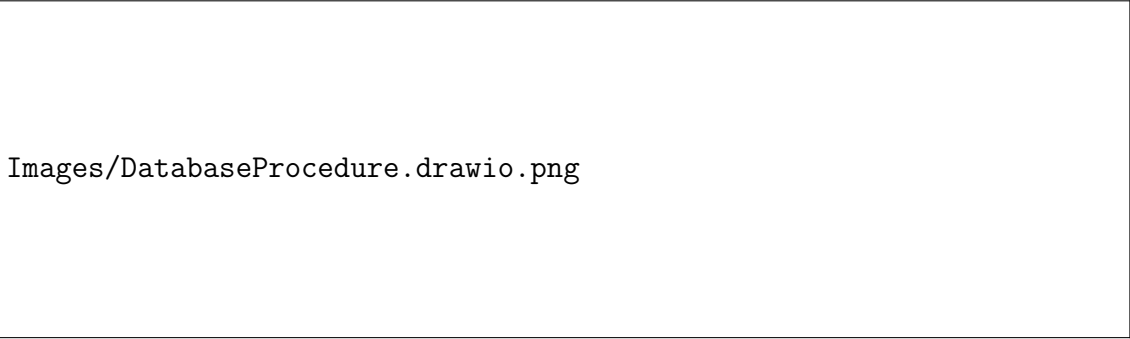


Figure 3.2: Video Dataset Breakdown by Activity

To streamline our video annotation process, we first categorized them by individual children, making it easier to track specific participants. Next, we organized the videos based on the activities they depicted. This hierarchical structure facilitated efficient annotation.

During the annotation process, we implemented our proposed frame skipping technique, processing every fifth frame to ensure the model was trained on a representative pattern of video data. All frames were standardized to a resolution of 368x368 pixels for consistent training data. To mitigate the imbalance in the dataset, we employed a flexible sliding window approach during preprocessing. The window length varied from 100 to 200 frames, adapting to the specific activity. After each processing cycle, we incrementally shifted the window by 1 or 3 frames for underrepresented activities and by 7 to 13 (odd numbers only) frames for overrepresented activities. This adaptive strategy ensured a balanced dataset for training, with varying frame sequences for each data point. The keypoint annotation process is visually summarized in Figure 3.3.



Images/DatabaseProcedure.drawio.png

Figure 3.3: Keypoint Annotation Workflow

By utilizing OpenPose, we obtained a comprehensive set of keypoint annotations for each frame. OpenPose detects 25 keypoints per person, providing x, y coordinates and a confidence score for each point. For our analysis, we focused solely on the x and y coordinates, representing each frame as a two-dimensional array with shape of (25, 2). This structure effectively captured the spatial information of the detected keypoints. To represent the complete sequence of keypoints, we concatenated the individual frame data into a single array. For example, a video sequence with 23 frames would result in a final array of length 23, where each element corresponds to the keypoint data for a specific frame. To facilitate analysis, we created a Pandas DataFrame that incorporated key information from each video sequence and consisted of three columns: 'Total Frames', 'Keypoints', and 'Activity'. The 'Activity' column indicated the class label for each sequence, specifying the type of activity being performed. The 'Keypoints' column within the Pandas DataFrame represents the keypoint annotations for a specific frame sequence, while the 'Total Frames' column indicates the total number of frames being analyzed.

The final dataset consisted of 14,066 data points, each representing a sequence of varying length, with up to 40 frames. The following Table 3.1 provides a breakdown of the class distribution within the dataset.

To optimize the training process for our LSTM or GRU models, which are capable of handling variable-length sequences but may benefit from standardized input formats, we implemented a post padding strategy [Cho \(2014\)](#). By adding zeros to sequences that were shorter than the longest sequence, we ensured that all sequences had a consistent length, potentially improving training efficiency

and stability. Additionally, we flattened the keypoints array from (25, 2) to (1, 50), transforming the feature representation into a one-dimensional format.

Activity	Data Count
Stand	2888
Sit	3066
Walk	3080
Bend	2665
Fall	2367
Total	14066

Table 3.1: Breakdown of Dataset by Activity

After preprocessing, the input shape for the deep learning models became (14066, 40, 50), where 14066 represents the number of samples, 40 denotes the timestamps, and 50 signifies the feature count. To handle the categorical nature of the activity labels, we employed label encoding followed by one-hot encoding. This transformed the output into a suitable format, with a shape of (14,066, 5), aligning it with the model’s expected output structure.

As outlined in Figure 3.1, we adopted a standard 80/20 split for training and testing. To further refine the model, we divided the training set into an additional 80/20 split for training and validation. Table 3.2 summarizes the distribution of the dataset into training and testing sets, providing the total number of data points and the count for each set.

Train	Test	Total
11252	2814	14066

Table 3.2: Breakdown of Dataset into Training and Test Sets

3.1.1 Initial Dataset Details

The baseline model was trained and evaluated using a dataset consists of 1438 videos capturing 12 subjects performing 6 actions (jumping, jumping jacks, boxing, waving with two hands, waving with one hand, and clapping hands) from 4 different angles, repeated 5 times each. The videos are composed of 211200 individual frames, and the 2D positions of 18 joints are tracked across these frames. The dataset was minimally preprocessed, involving running OpenPose on individual frames to obtain joint positions and accuracies, converting JSON outputs to text format,

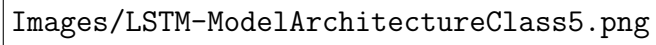
and retaining only relevant information (x and y positions, action performed, and frame order). They did not apply any normalization or preprocessing, resulting in raw 2D joint positions. The missing joint positions are represented in the dataset as [0.0,0.0]. The dataset is divided into training and testing sets, with an 80/20 split and a 26-frame overlap between training sequences [Eiffert \(2018\)](#).

3.2 Model Configuration

To explore different modeling approaches, we implemented four distinct architectures based on a common foundation. Starting with a baseline LSTM model, we systematically replaced layers to create variations such as bidirectional LSTM, GRU, and LSTM with attention.

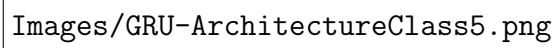
Our baseline model employs a multi-layered LSTM structure with dropout layers to address overfitting and improve generalization. The initial LSTM layer processes the input sequence, capturing temporal dependencies with 128 units. Subsequent LSTM layers further refine the representation, extracting relevant features from the sequence with 64 units each. The final dense layers with softmax activation classify the input sequence into one of the activity classes.

To identify the optimal model for our activity detection system, we initially explored and evaluated the mentioned architectures using the existing dataset. Subsequently, we trained these models on our annotated dataset to select the top-performing model based on its ability to effectively capture the unique patterns and characteristics of our data. The architectures of the models considered for final selection are depicted in Figures 3.4, 3.5, 3.6, and 3.7.



Images/LSTM-ModelArchitectureClass5.png

Figure 3.4: LSTM Model Architecture



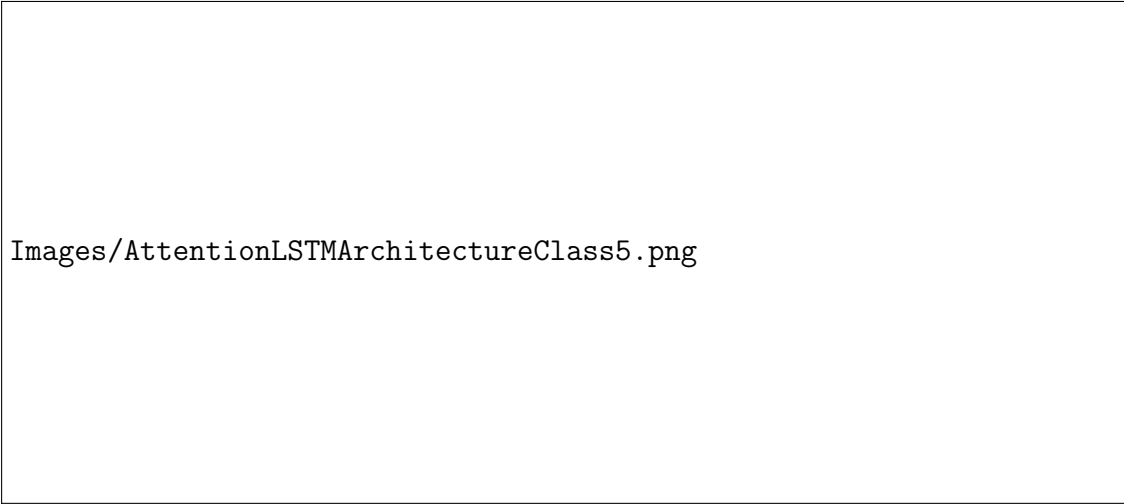
Images/GRU-ArchitectureClass5.png

Figure 3.5: GRU Model Architecture



Images/BiLSTMArchitectureClass5.png

Figure 3.6: Bi-directional LSTM Model Architecture



Images/AttentionLSTMArchitectureClass5.png

Figure 3.7: Attention-Based LSTM Model Architecture

In our attention-based model, we included a custom attention layer to improve the model's capacity to concentrate on the most important parts of the input sequence. The custom attention layer computes self-attention by taking the dot product of the input tensor with its transpose to create a scoring matrix and indicate similarities between time steps. Then, the layer normalizes the scores with softmax

to produce attention weights, highlighting important elements in the sequence. Finally, these weights generate a context vector as a weighted sum of the inputs, allowing the model to focus on relevant features over time. In this model, we made an architectural modification by replacing batch normalization with global average pooling after the attention layer and before the flattening layer to reduce computational overhead.

For a more comprehensive comparison, Table 3.3 presents a detailed analysis of the trainable and non-trainable parameters for each of the four models. Understanding the distribution of these parameters provides valuable insights into the model’s capacity for learning and generalization.

Model Name	Trainable Parameters	Non-Trainable Parameters	Total Parameters
LSTM	178,693	128	178,821
Bi-Directional LSTM	455,301	256	455,557
GRU	135,941	128	136,069
LSTM with Attention	178,565	0	178,565

Table 3.3: Comparison of Trainable and Non-Trainable Parameters Across Four Models

3.3 Training, Testing and Validation

To evaluate the initial performance of each model, we trained them on the existing dataset using categorical cross-entropy as the loss function and the Adam optimizer. All models were trained for 80 epochs, with a 20% validation split and shuffling enabled. To enhance the performance of the GRU model, we implemented a slower learning rate. We employed a learning rate reduction strategy to enhance our model’s training process. The ReduceLROnPlateau callback automatically decreased the learning rate when validation loss plateaued, preventing the model from becoming stuck in local minima.

To train on our dataset, we trained each model for 100 epochs. We explored two distinct training approaches: one incorporating a learning rate optimization strategy and another without. For the former, we strategically employed the ReduceLROnPlateau callback in Keras, initially setting the learning rate to 0.0010

and the patience to 20 while monitoring the validation loss. If the validation loss failed to improve within 20 epochs, the learning rate was reduced by a factor of 0.1. Throughout our training endeavors, we adopted a batch size of 32 and enabled shuffling of the training data to ensure robust generalization and prevent the model from becoming overly reliant on specific patterns within the data.

To optimize the model’s performance, we experimented with different patience values in our dynamic learning rate callback. However, upon analyzing the training and validation loss curves, we observed signs of overfitting when using lower patience values. Consequently, we ultimately selected a patience of 20 to strike a balance between model performance and preventing overfitting.

We evaluated the models using the designated test dataset and selected the model that demonstrated the highest F1-score for fall detection. This alignment with our primary objective of accurately identifying fall events guided our choice of the best-performing model for our system.

3.4 Limitation of Existing Dataset

Existing research on children’s activities often faces limitations due to the availability of private or limited datasets. Therefore, we selected a dataset that, while not containing fall activities, offered a diverse range of activities similar to our target domain. The pre-collected dataset mainly focused on upper body movements, which were different from our target activities. However, we found it valuable as a starting point for developing the baseline model structure. By using this dataset, we established the foundation for our model architecture before incorporating our dataset.

Our work is unique because it focuses on both children’s activities and falls using a single dataset. Our combined dataset fills a critical gap in the existing literature and provides a valuable resource for researchers interested in exploring children’s key points data. We also analyzed the dataset using different models to gain insights into the dataset’s performance.

By providing a well-structured and annotated dataset, we aim to contribute to the field of children’s activity and fall detection.

Chapter 4

Design and Development

4.1 Project Management and Financial Activities

We divided the project work into two primary phases:

- Dataset creation.
- System development.

In the following sections, we will outline the project structure and a budget breakdown for completed tasks and future development.

4.1.1 Dataset Creation

Our goal was to build an ethical children’s activity dataset. Therefore, we first obtained permission from parents and children. We ensured that all participants fully understood the goals of our work and their role in it [Al-Azani et al. \(2022\)](#). We maintained a cooperative and supportive environment to make the data collection process enjoyable for the children. Furthermore, using a structured procedure, we maintained consistency and reliability in our data collection work. Additionally, we created a Google Form to obtain digital parental consent by allowing parents to review and sign the consent agreement and ensuring a secure record of their permission. See Figure 4.1 for a screenshot of the digital consent form obtained from parents.



Figure 4.1: Screenshot of the consent form used in our study.

4.1.1.1 Dataset Description

Our dataset includes videos of five activities performed by children, as outlined in Figure 4.2. We gathered video data from 25 children, including 12 girls and 13 boys, aged 3 to 10. Since we planned to monitor falls and other activities in a home environment, we captured videos in various indoor settings with different lighting conditions: daylight, nighttime, bright light, and medium light. Our video dataset also includes real-world scenarios with furniture and potential occlusions of children’s bodies. The videos in our dataset range in duration from 5 to 65 seconds and each was recorded at 30 frames per second (FPS).



Figure 4.2: List of Recorded Children’s Activities

4.1.1.2 Creation Procedures

In our dataset, we prioritized recording videos from normal and CCTV angles. We used mobile cameras to simultaneously capture footage from two to three perspectives within the same room. Two or three individuals held the cameras in various positions to ensure diverse angles. After reviewing all captured videos, we carefully categorized them based on the activities performed. In the final stage, we refined the dataset by removing videos that inadvertently included individuals not participating in the activities. Additionally, we adjusted the video frames to exclude any unwanted persons, ensuring data consistency.

4.1.1.3 Creation Cost Analysis

To minimize expenses, we relied on human assistance and utilized readily available mobile cameras for video capture, eliminating the need for additional equipment such as tripods or professional cameras. While our primary objective was to include as many children as possible, this required traveling to various locations. To create a welcoming and engaging environment for the children, we invested in treats to encourage enthusiastic participation. A detailed breakdown of all associated costs is provided in Table 4.1.

Purpose	Per Child (BDT)	Total Children	Total (BDT)
Travels	200	25	5000
Treats	50	25	1250
		Total	6250

Table 4.1: Cost analysis table for dataset creation

4.1.2 System Development

A system is composed of interconnected processes working together to achieve a specific purpose. Our system's primary goal is to detect activities and promptly alert caregivers or parents if a fall is detected. To accomplish this, we divided the system into several subsystems that collectively contribute to the overall objective. These subsystems are -

1. Video Acquisition and Processing
2. OpenPose-Based Keypoint Extraction
3. Activity and Fall Detection
4. Alert System



Figure 4.3: Overall system workflow

Figure 4.3 provides a comprehensive overview of our system, illustrating how these subsystems interact to achieve the desired outcome. In the subsequent sections, we will delve into the details of each subsystem and present a thorough cost analysis.

4.1.2.1 Video Acquisition and Processing

Our system has two input options: direct video upload and video streaming. To expedite the processing, we implemented a frame-skipping technique. Instead of analyzing every frame, we skipped five frames between each processed frame. This approach significantly reduced the number of frames required to detect an activity while maintaining sufficient accuracy. Research indicates that five to seven keyframes are generally adequate for activity recognition, and skipping frames does not compromise the reliability of our results [Schindler & van Gool \(2008\)](#). We maintained consistent frame dimensions by standardizing all frames to 368x368 pixels. Figure 4.4 shows the significant reduction in frame count achieved using our simple frame extraction algorithm, which decreases the number of frames in a 10-second, 30 FPS video from 300 to 60.

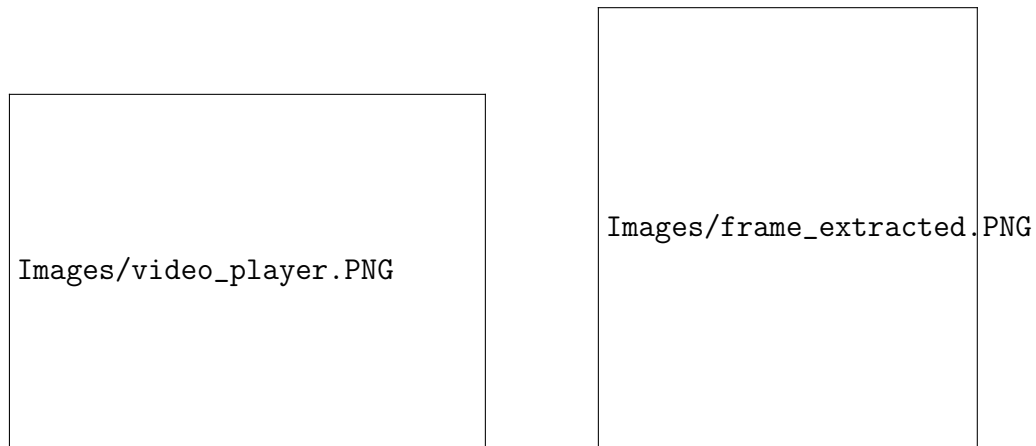


Figure 4.4: Total frames were reduced to 60 instead of 300 using the frame extraction algorithm

4.1.2.2 OpenPose-Based Keypoint Extraction

OpenPose is a real-time multi-person 2D pose estimation library. It can detect and track the poses of multiple people in an image or video sequence. OpenPose uses

a deep neural network architecture, primarily the Convolutional Pose Machine (CPM), which integrates various components, such as -

- **Convolutional Layers**
- **Stagewise Refinement**
- **Part Affinity Fields (PAFs)**
- **Confidence Maps**



Figure 4.5: Visual representation of how does Openpose work

Figure 4.5 illustrates the workflow of OpenPose. After receiving an input image, convolutional layers extract features. Then, the CPM network processes these features through multiple stages, where the pose estimation is progressively refined at each stage. Subsequently, PAFs are computed to predict the likelihood of connections between body parts. Confidence maps are then generated to represent the probability of each keypoint being located at a specific position. Finally, the final pose estimates are obtained by combining the confidence maps and PAFs.

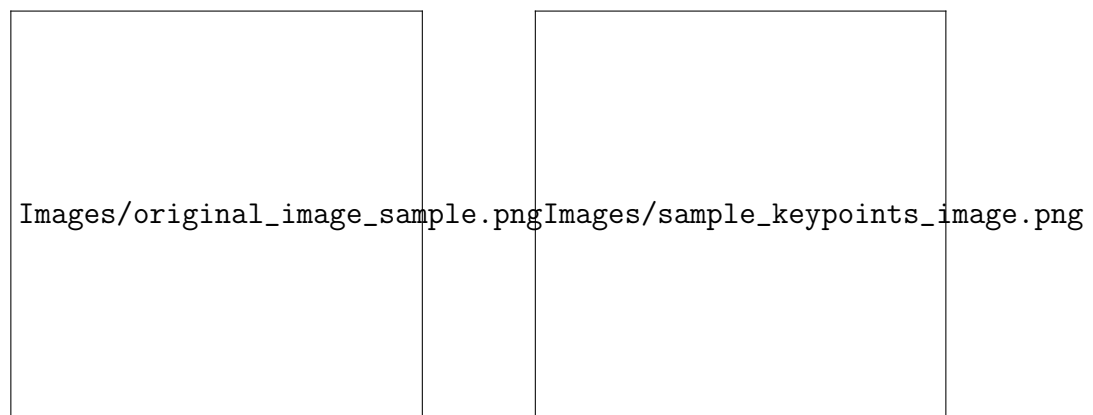


Figure 4.6: Comparison of Original Video Frame and Skeleton Representation

Figure 4.6 demonstrates how OpenPose processes a frame from our collected video dataset, extracting keypoints and representing the child's pose as a skeletal structure.

Figure 4.7 presents a visual comparison of inference time among three pose estimation libraries (OpenPose, Alpha-Pose, and Mask R-CNN) under identical hardware and conditions. While OpenPose exhibits a consistent inference time, the runtime of Alpha-Pose and Mask R-CNN increases linearly with the number of individuals in the image.

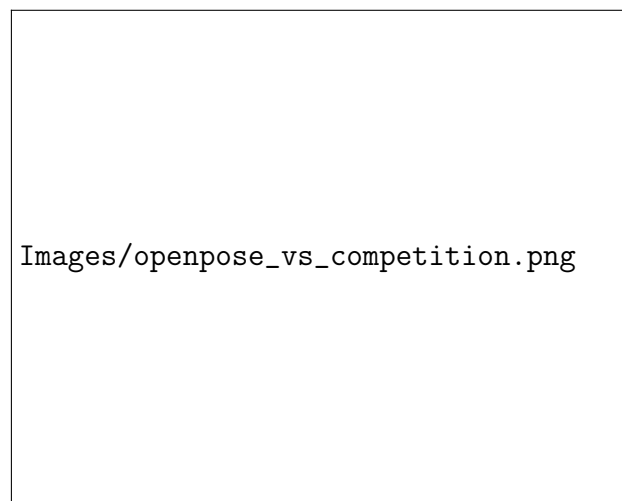


Figure 4.7: Inference Time Comparison of Pose Estimation Libraries

4.1.2.3 Activity and Fall Detection

Deep learning is a subfield of machine learning that utilizes artificial neural networks with multiple layers to learn intricate patterns from data.

RNN is one of the frequent choices for analyzing sequential data. RNNs have internal memory that enables them to retain information from previous inputs, making them suitable for tasks involving sequential data [Keren & Schuller \(2016\)](#). The components of an RNN model are:

1. **Hidden State**
2. **Weights**
3. **Activation Function**

To explore advanced RNN architectures, we selected LSTM and GRU models for our research. We also explore different variations of LSTM, such as Bi-LSTM and Attention-based LSTM, as the LSTM model demonstrated promising performance in our initial results.

- **LSTM:** LSTM addresses the vanishing gradient problem that occurs in traditional RNNs when learning long-term dependencies. The problem arises as the error signal gradients can become very small or vanish when they propagate back through the network. LSTM has two main components: a vector called cell state that stores information from previous time steps, and gates including the input gate, forget gate and output gate to control the flow of information into and out of the cell state.

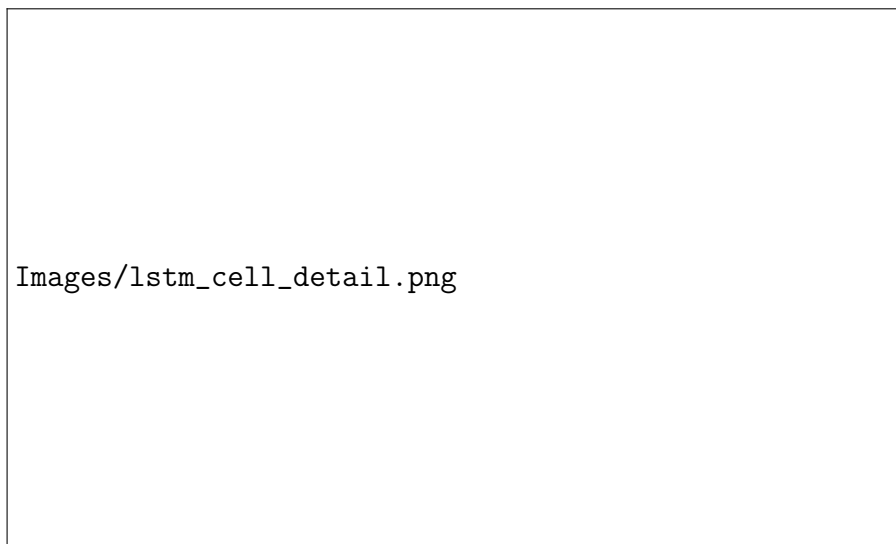


Figure 4.8: In-depth LSTM cell architecture

In figure 4.8, the input gate determines the amount of new input that should be added to the cell state, the forget gate helps in quantifying the amount the previous cell state should be forgotten, output gate determines the output of the cell state.

- **Multi-layer LSTMs:** Multi-Layer LSTMs stack multiple LSTM layers on top of each other [Graves \(2013\)](#). This allows the model to learn more complex patterns and representations from the input data. Each layer in the stack

processes the output of the previous layer, enabling the model to capture information at different levels of abstraction.

- **Bidirectional LSTMs:** Bidirectional LSTMs process the input sequence in both directions (forward and backward), allowing the model to capture information from both past and future contexts [Schuster & Paliwal \(1997\)](#).
- **Attention-Based LSTMs:** Attention-Based LSTMs incorporate an attention mechanism that allows the model to focus on different parts of the input sequence at different times [Bahdanau \(2014\)](#).
- **GRUs (Gated Recurrent Units):** GRUs are a simplified variant of LSTMs, designed to reduce the computational complexity while maintaining comparable performance. They introduce a reset gate and an update gate to control the flow of information through the network [Cho \(2014\)](#).

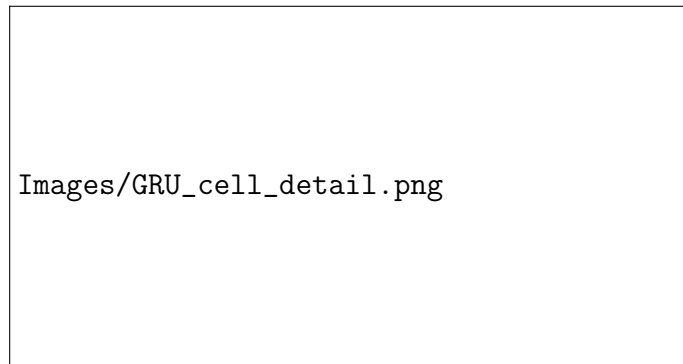


Figure 4.9: In-depth GRU cell architecture

As shown in figure 4.9, a GRU cell has three gates including an input gate similar to the LSTM cell, a reset gate to determine how much of the previous hidden state should be forgotten, and an update gate to control how much of the previous hidden state should be updated with the new candidate hidden state [Abboush et al. \(2023\)](#).

4.1.2.4 Alert System

The primary function of the alert system is to notify caregivers when their children fall. We considered various notification methods, including text messages and

dedicated mobile applications. Given the advantages of a tailored mobile app, we opted for this approach. Our system will trigger a notification upon detecting a fall, requiring parents to install the app on their devices and maintain an internet connection to receive alerts.

4.1.2.5 System Cost Analysis

To fully support our system, we must consider both software and hardware requirements. For deployment in a household environment, hardware components such as an IP camera, a powerful processor capable of running OpenPose on-premise for faster performance, and a stable internet connection are essential. To host our services and model, we chose a cloud server, which offers cost-effective scalability through a pay-as-you-go pricing model. To maintain and further develop the system, we require an engineering team composed of full-stack developers and quality assurance professionals.

While OpenPose officially supports the NVIDIA Jetson TX2, it can also be successfully deployed on the more affordable NVIDIA Jetson Nano with 4GB of RAM. This provides a cost-effective option to utilize OpenPose in embedded applications. Additionally, the Jetson TX2 has been superseded by the NVIDIA Jetson Xavier NX, which offers enhanced performance, power efficiency, and features.

Table 4.2 provides a detailed breakdown of the one-time hardware investment required for users considering NVIDIA Jetson Nano as a primary embedded system.

Components	Price (BDT)
IP Camera	3000
Jetson Nano 4GB Developer Kit	46000
Total	49000

Table 4.2: One-Time Hardware Investment Breakdown

To efficiently run our system on the cloud, we considered utilizing an AWS EC2 c5.xlarge instance in shared mode, paired with a 100GB EBS gp2 volume and 100GB of S3 storage. EBS is essential for storing the EC2 instance's operating system, application files, and temporary data, while S3 is ideal for storing large datasets, model checkpoints, and training results. The EC2 instance provides the necessary processing power to execute the system's functions. Assuming continuous system operation, the total active time per month would be 24 hours/day * 30 days/month = 720 hours. Based on hourly rates of \$0.354 for the EC2 c5.xlarge instance, \$0.114

for the 100GB EBS gp2 volume, and \$0.025 per GB for S3 storage, Table 4.3 presents a detailed breakdown of the minimum monthly cost for software development [Amazon \(2024\)](#). Figure 4.10 shows visual representation for further comprehension.

Components	Single Unit Price (BDT)	Required	Cost/Month (BDT)
EC2 instance (c5.xlarge)	30960	1	30960
EBS Volume (100GB gp2)	1366	1	1366
S3 Storage (100GB)	300	1	300
Software Developer (Mid Level)	70000	2	140000
AI Engineer (Mid Level)	80000	1	80000
SQA	40000	1	40000
		Total	292626

Table 4.3: Monthly Software Development Cost Breakdown

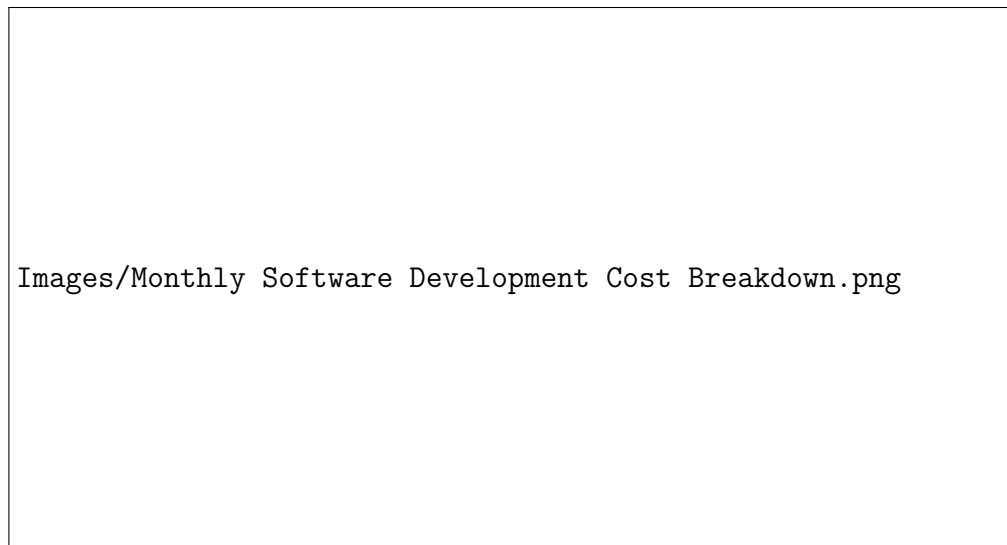


Figure 4.10: Visual Representation of Monthly Software Development Cost Breakdown

During the development phase, we leveraged the free Google Colab platform to conduct our development efforts, eliminating the need for any additional hardware investment.

4.2 Adopted Tools and techniques

Development tools and techniques refer to the software, methodologies, and practices used in the process of creating software applications. They improve productivity, quality, and collaboration. By automating tasks, streamlining workflows, and ensuring quality, these tools help teams create high-quality software applications more effectively. The tools and techniques employed in our system are detailed in the subsequent sections, along with a comprehensive explanation of their respective purposes.

4.2.1 Project Management Tools and Techniques

In this section, we will delve into the tools and methodologies employed to orchestrate the entire development process of our system. We will explore how these tools facilitated project workflow management, communication, and the successful attainment of our system objectives.

4.2.1.1 Project Management Software

- **Notion:** Notion is a versatile workspace where we can organize thoughts, manage projects, and collaborate with others [Notion Labs \(2024\)](#). We used Notion as our primary project management tool, instead of traditional task management software. This versatile platform performed as a centralized hub where we planned project procedures, documented critical research insights, and effectively tracked the progress of our project works (Figure 4.11).

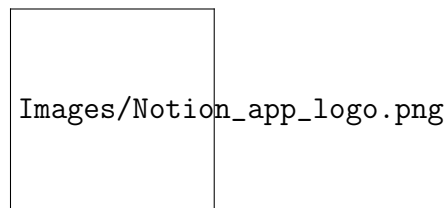


Figure 4.11: Notion

4.2.1.2 Team Communication

We maintained team communication using various platforms. We primarily used WhatsApp for ongoing discussions and Google Meet for real-time collaboration

and remote project meetings (Figure 4.12).

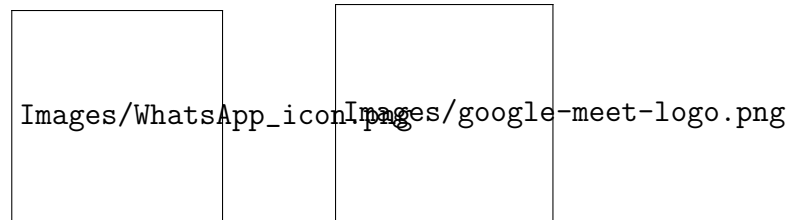


Figure 4.12: Whatsapp and Google Meet

4.2.1.3 Development Methodology

- **Agile Methodology (Scrum):** We adopted an Agile methodology, specifically Scrum (Figure 4.13), as our development process. This iterative and incremental approach allowed us to break down our project into manageable sprints, enabling collaboration, flexibility, and continuous improvement. However, instead of daily scrum meetings, a key feature of Scrum, we scheduled weekly scrum meetings to stimulate progress reporting among team members.

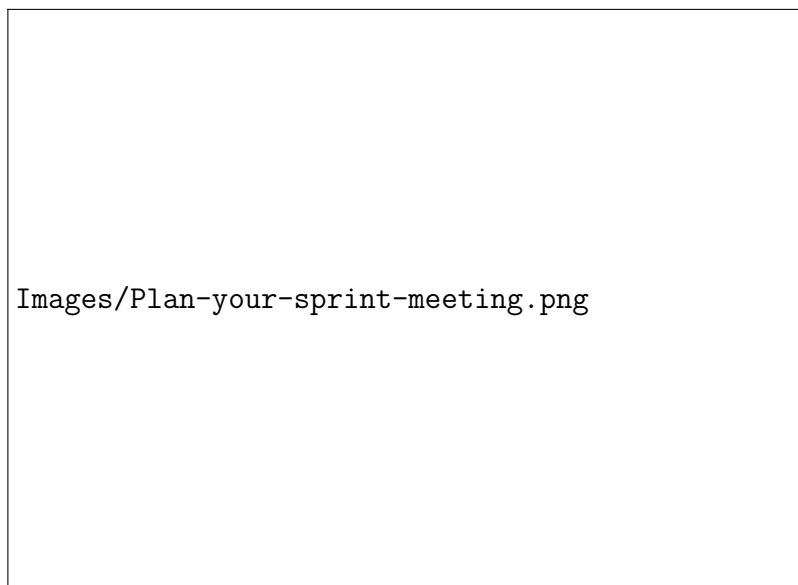


Figure 4.13: Scrum Sprint

4.2.2 Design Tools and Techniques

4.2.2.1 Canva

Canva (Figure 4.14) is a user-friendly online design platform that allows for creating visually appealing graphics, presentations, and mockups. We utilized Canva to create visually appealing project timelines and other essential graphic assets.

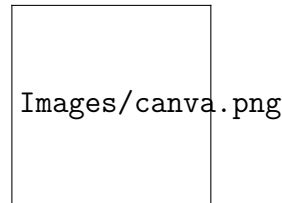


Figure 4.14: Canva

4.2.2.2 Draw.io

Draw.io (Figure 4.15) is a powerful diagramming and flowcharting tool that facilitates the creation of wireframes, user flows, and system architecture diagrams. We utilized this to create intricate model diagrams and other crucial visual representations.

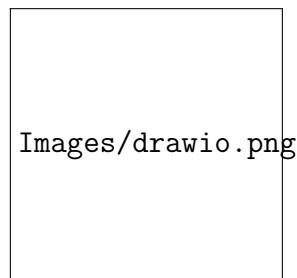


Figure 4.15: Draw IO

4.2.2.3 Dribbble

Dribbble (Figure 4.16) is not a design tool itself, but rather a platform for discovering design trends and inspiration. To enhance the visual appeal and user experience of our system, we incorporated contemporary UI design trends and inspiration from Dribbble.

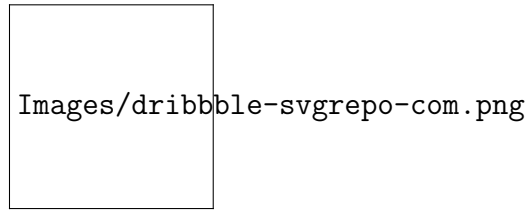


Figure 4.16: Dribbble

4.2.3 Development Tools and Techniques

4.2.3.1 Frontend

We employed the React library to build our frontend application. ReactJS is a versatile open-source JavaScript library developed and maintained by Meta. We selected ReactJS because it creates a single-page web application that is quicker and more efficient than vanilla JavaScript. In our system, ReactJS mainly focuses on the view layer, known as "V" in the model-view-controller (MVC) pattern. React employs a virtual DOM, a lightweight in-memory representation of the actual DOM, and updates the actual DOM selectively. The intelligent DOM manipulation approach results in faster rendering and provides a more responsive user experience. We utilized Chakra UI, a component library, to make an interactive user interface faster. Finally, we used Visual Studio Code, a popular open-source code editor, for our front-end development. All frontend development tools and technologies are illustrated in Figure 4.17.

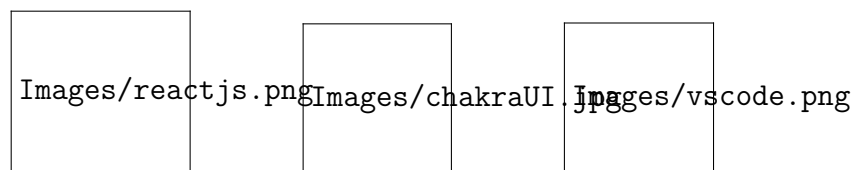


Figure 4.17: Frontend Development Tools and IDE

4.2.3.2 Backend

The backend services are the core of our application that handles data interactions and business logic. We have used loose coupling by implementing the repository pattern. This pattern creates an abstraction layer between data access and

business logic to encapsulate data-related operations within dedicated repositories [Shekhawat \(2024\)](#). We used dependency injection to inject repositories into controllers, decoupling them from the underlying data access technology.

By employing a Model-First approach with Entity Framework, we prioritized conceptual design over existing database structures. This allowed us to define the application's entities, relationships, and attributes within the Entity Framework designer, generating the necessary database schema. This approach ensured the perfect alignment of the database with our system's requirements, fostering a cohesive and efficient development process [DevIQ \(2024\)](#).

We developed our backend services using ASP.NET Web API, a robust framework for building HTTP-based services. ASP.NET Web API emphasizes data exchange through HTTP requests and responses and allows flexibility to develop a comprehensive suite of services without worrying about the platform. Lastly, we used Visual Studio as the development environment for our backend service. Figure 4.18 presents the backend tools of our system.

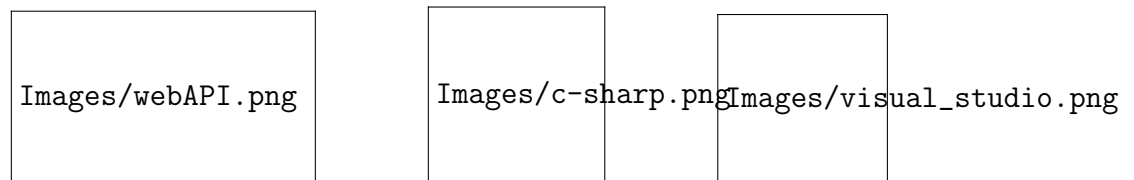


Figure 4.18: Backend Development Tools and IDE

4.2.3.3 Database

Selecting a database was a critical aspect of our system development. Therefore, we opted for a flexible database like MongoDB (Figure 4.19), a leading open-source and document-oriented database, due to its horizontal scalability and flexible schema. It is a NoSQL database and offers a dynamic and adaptable approach to data storage, making it well-suited for modern applications. MongoDB stores data as documents in compressed BSON files that can be directly retrieved in JSON format, offering several advantages such as - natural data representation, human readability, and nested JSON structures [MongoDB \(2024\)](#).

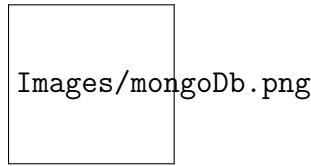


Figure 4.19: MongoDB

4.2.3.4 Model and Dataset

We used Google Colab for model development and dataset creation. To simulate cloud calling during development, we established a tunnel using ngrok from our Colab notebook. This allowed us to test API interactions between the frontend and the model directly within the Colab environment. Ultimately, Colab served as our model service. Our final model was constructed using TensorFlow as the underlying framework and Keras as the high-level API. We incorporated LSTM, GRU, and BiLSTM layers to process sequential data effectively. Figure 4.20 illustrates the tools used to develop our model and dataset.

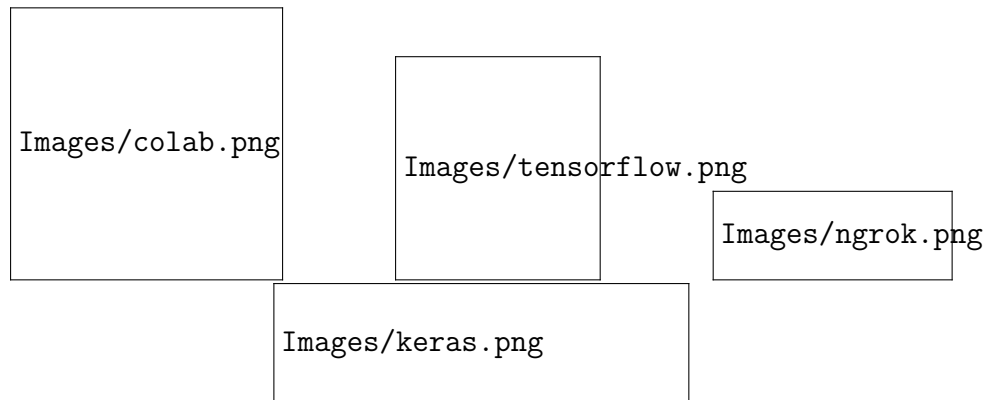


Figure 4.20: Model and Dataset Tools

4.2.3.5 Testing

To ensure the proper functioning of our API and responses, we rigorously tested them using Postman (Figure 4.21), a popular API development and testing tool. Postman provides a user-friendly interface for sending HTTP requests, inspecting responses, and managing API collections. Through thorough manual testing, we verified the proper functioning of the entire system.

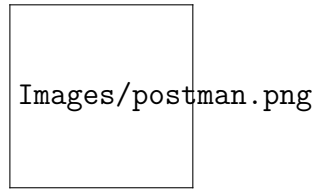


Figure 4.21: Postman

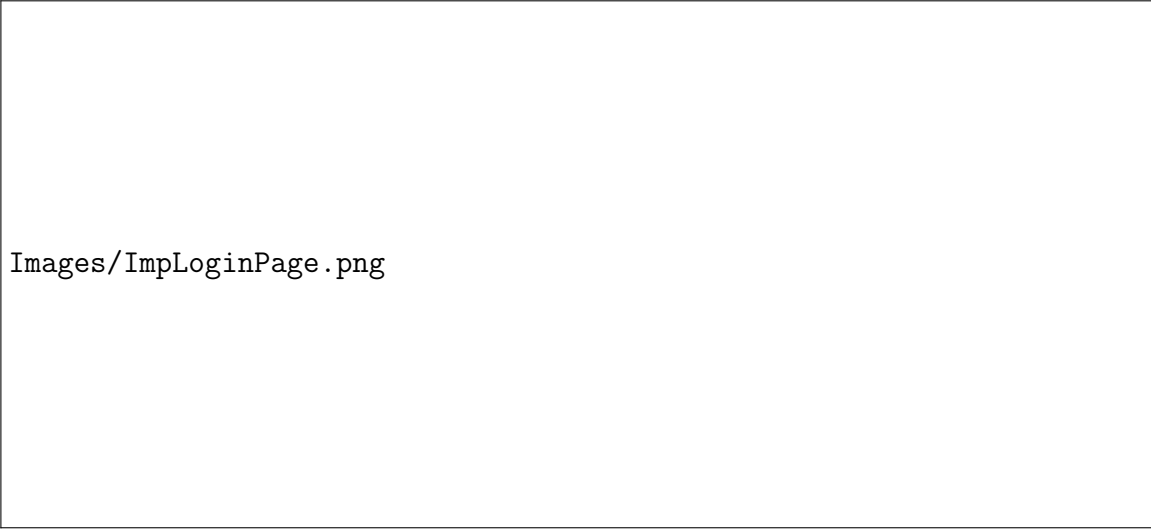
4.3 Implementation

Figure 4.22 illustrates the key features and components that constitute our system's architecture and collectively bring our system's vision to life.



Figure 4.22: Key Features Design Workflow

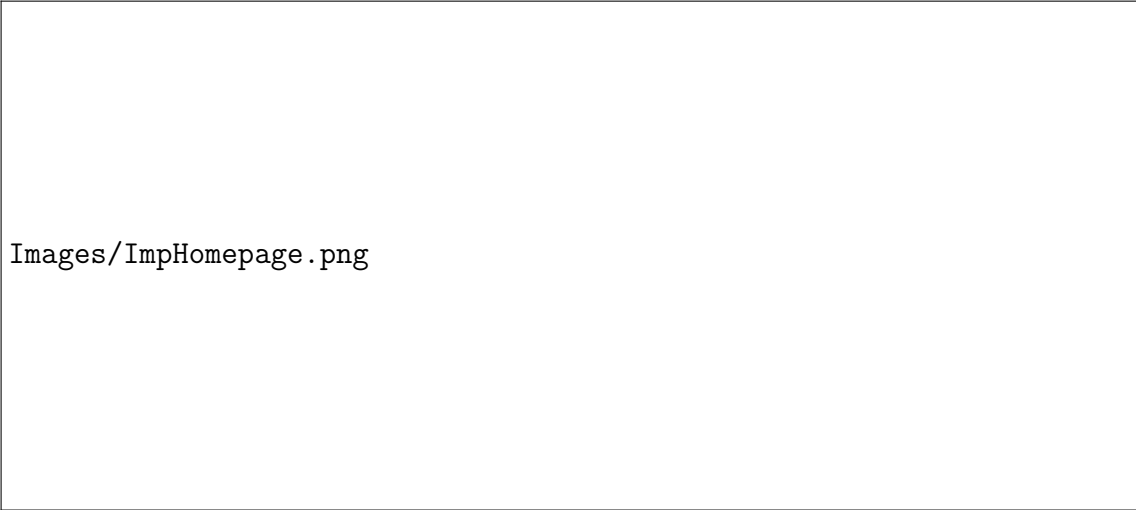
Upon initial system access, users are presented with a login page, as depicted in Figure 4.23. This login page interacts with the backend services to retrieve user data and authenticate their identity. With successful authentication users can have access to the system's features and functionalities.

The image shows a login page with a light blue background. At the top, there is a header with the text "Login" in a bold, dark blue font. Below the header, there is a large, empty rectangular area with a light blue background, which appears to be a placeholder for a login form or a large image. The text "Images/ImpLoginPage.png" is located in the bottom left corner of the image.

Images/ImpLoginPage.png

Figure 4.23: Login Page

After logging in, users will find themselves on the home page, illustrated in Figure 4.24. While our system can detect falls in real time from live feeds and alert users immediately, we've streamlined things for the demo purpose. Users can still experience the system's capabilities by uploading a video from their device.

The image shows a home page with a light blue background. At the top, there is a header with the text "Home" in a bold, dark blue font. Below the header, there is a large, empty rectangular area with a light blue background, which appears to be a placeholder for a home page content or a large image. The text "Images/ImpHomepage.png" is located in the bottom left corner of the image.

Images/ImpHomepage.png

Figure 4.24: Home Page

By clicking 'Select Video,' users can choose a video from their device. Once selected, clicking 'Detect Activity' will send the video to our analysis service for further processing, presented in Figure 4.25.

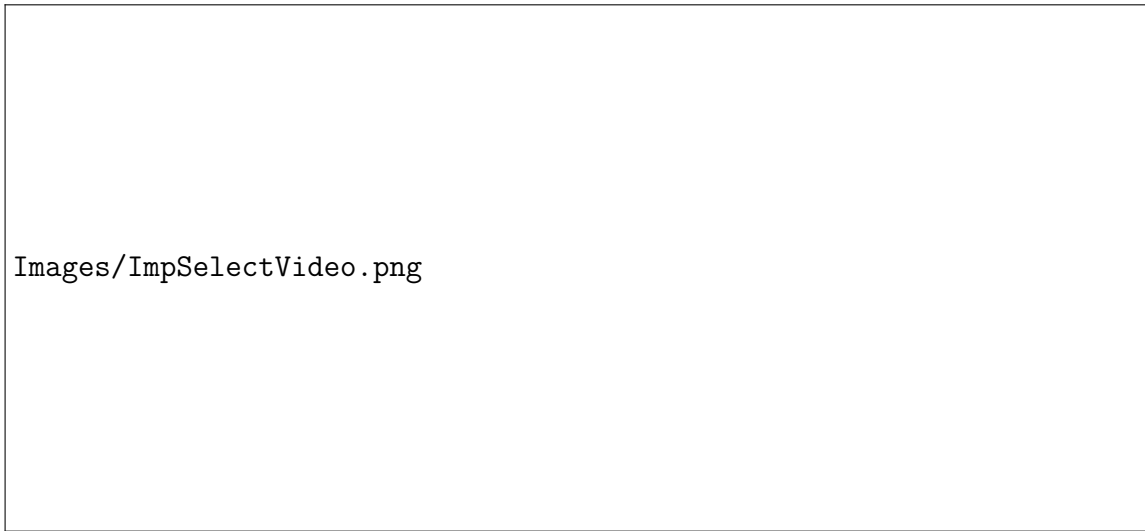


Figure 4.25: Selected a Video

Once our analysis service receives a video, it undergoes a preprocessing stage. This involves preparing the video data into batches suitable for feeding into our detection model. While our demonstration visually illustrates the entire workflow, in a real-time setting, the preprocessing would occur locally on a device like a Jetson Nano, ensuring faster processing times. The system would then transmit only the processed keypoint data for prediction.

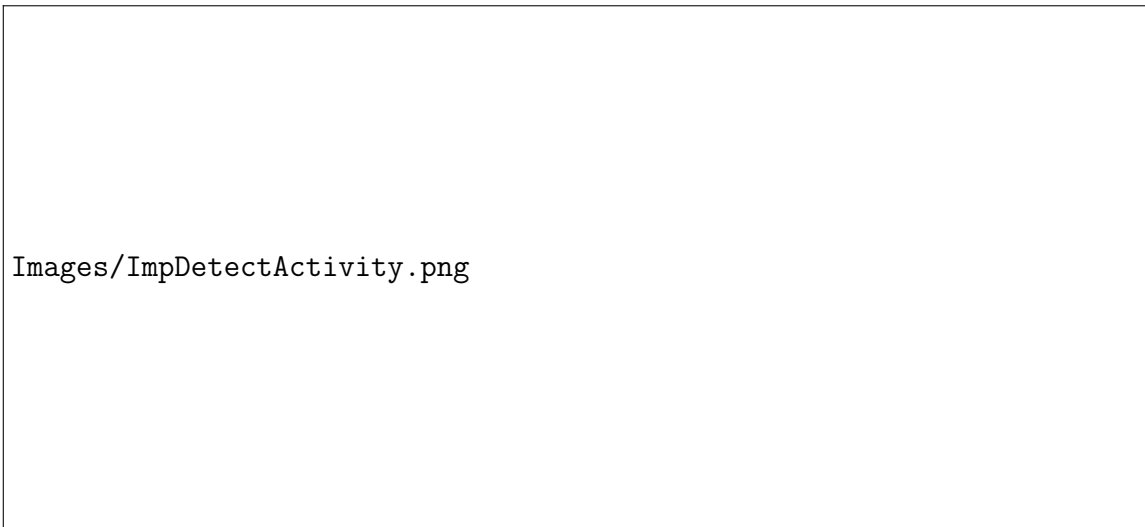


Figure 4.26: Activity Detection Example 1

Given that the video data is already being sent, our analysis service responds to

the API call by returning the annotated and activity-labeled video. This enables us to visually showcase the results within our system.

We have included examples using random videos from the internet and one of our fall videos that was not used for model training. Our model successfully predicted all activities, as demonstrated in Figures 4.26, 4.27, 4.28 and 4.29.

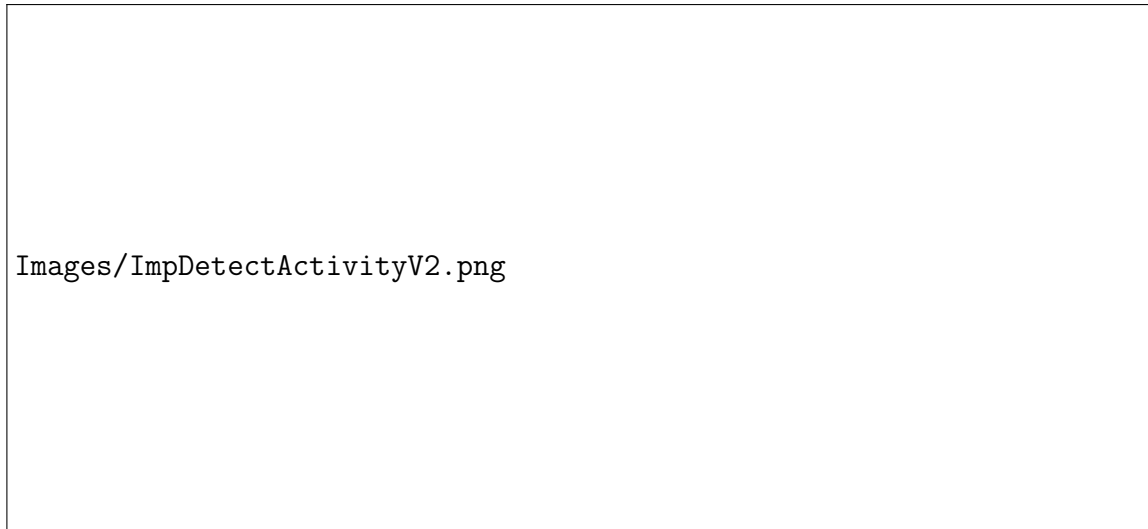


Figure 4.27: Activity Detection Example 2

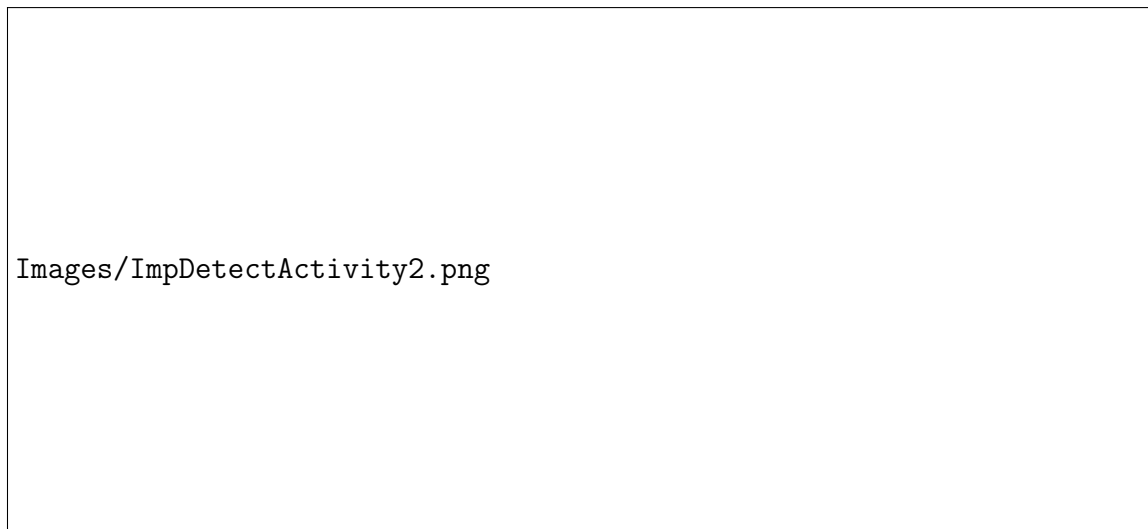


Figure 4.28: Activity Detection Example 3

If our system detects a fall, the analysis service notifies the backend service,

which in turn triggers an immediate notification to the user. For visualization purposes, after detecting a fall, the analysis service returns a snapshot of the video that was identified as a fall and displays a warning on the web application 4.30.

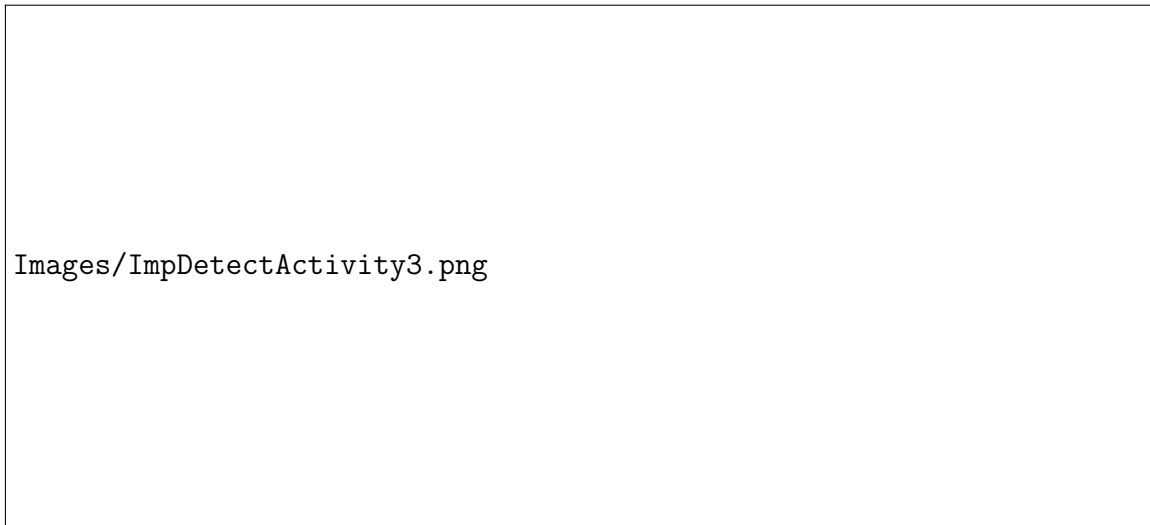


Figure 4.29: Activity Detection Example 4

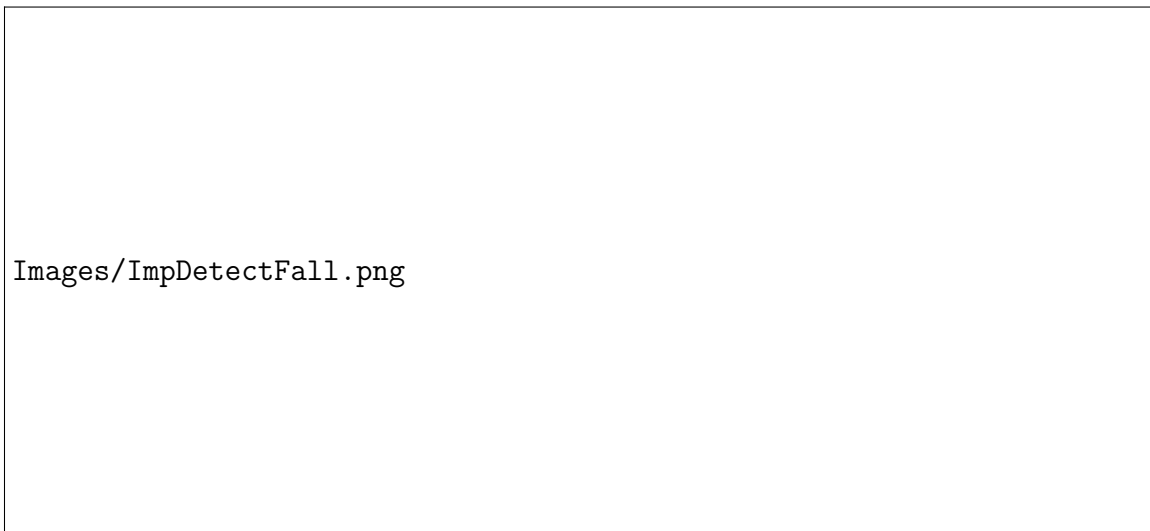
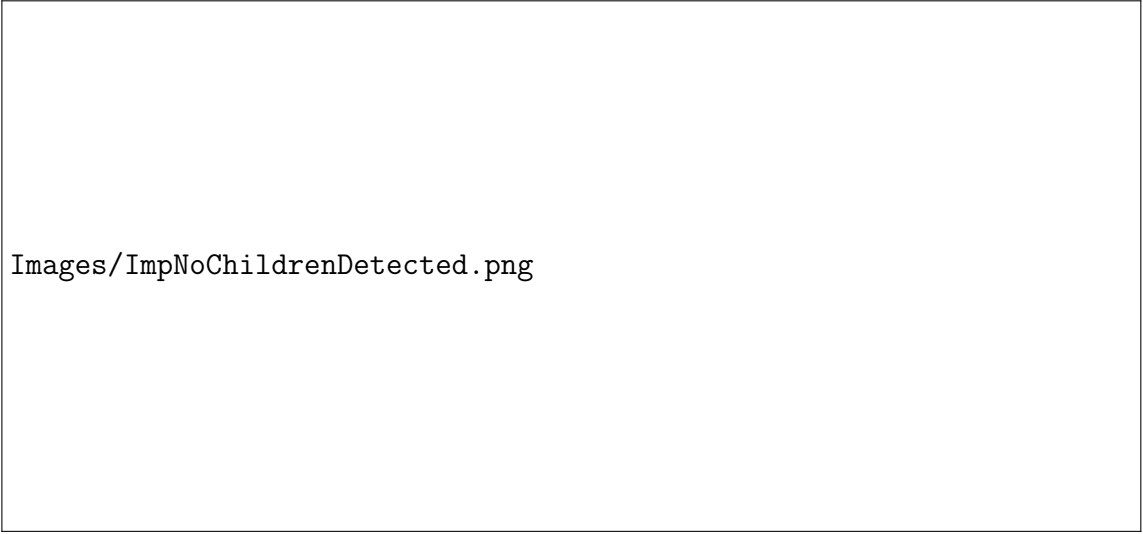


Figure 4.30: Fall Detection Example

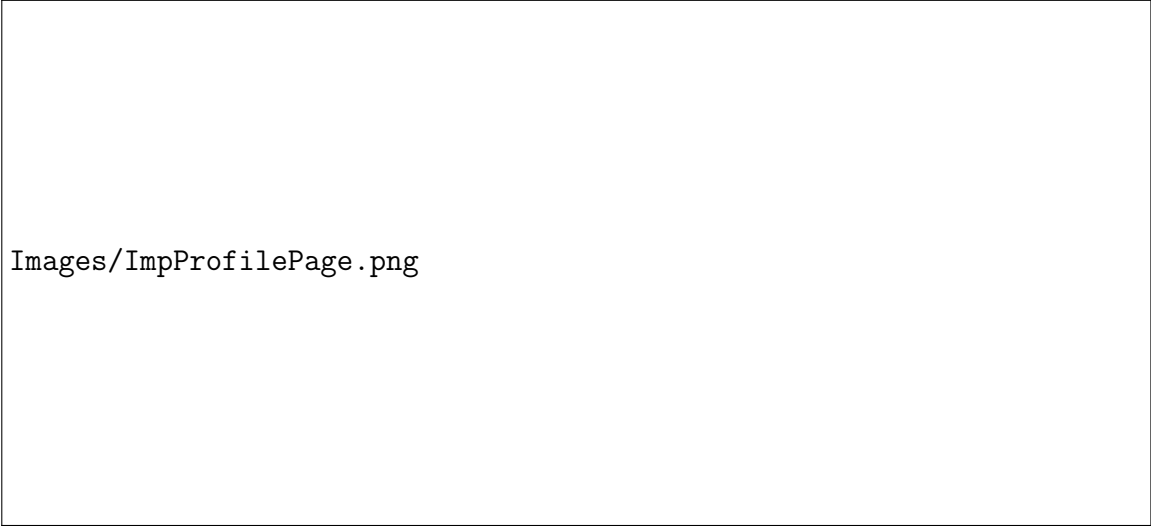
If the system encounters empty frames or detects the absence of children in the video, it will identify the situation and display a warning within the web application, as seen in Figure 4.31.

A screenshot of a user interface showing a message. The text 'Images/ImpNoChildrenDetected.png' is visible in the top left corner of the image area.

Images/ImpNoChildrenDetected.png

Figure 4.31: No Child Detection Example

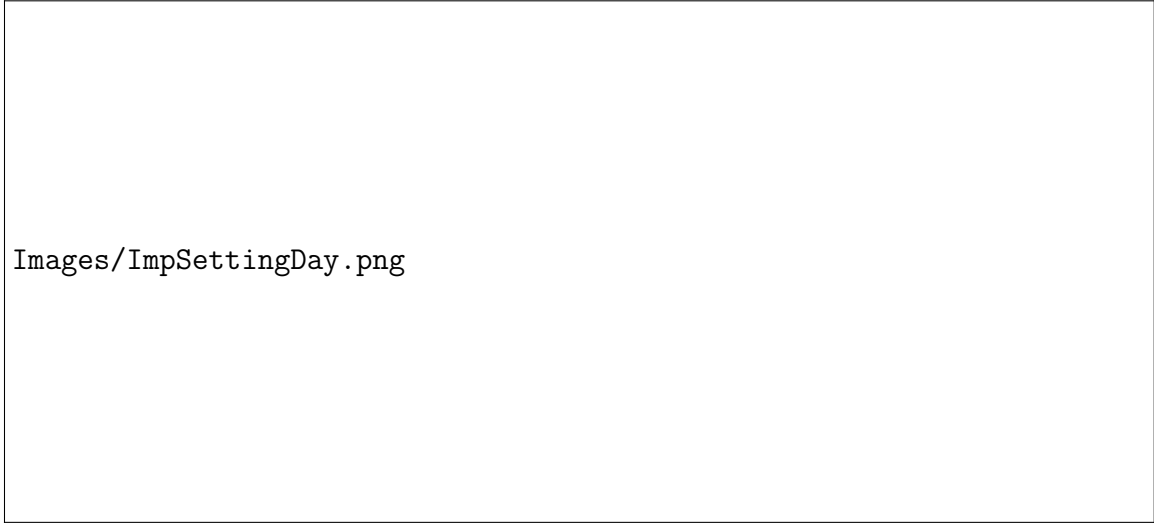
The profile page presented in 4.32 dynamically retrieves and displays user information, including personal details, contact information, and emergency contact details from the backend services. Users can also modify their personal information as needed.

A screenshot of a user profile page. The text 'Images/ImpProfilePage.png' is visible in the top left corner of the image area.

Images/ImpProfilePage.png

Figure 4.32: Profile Page

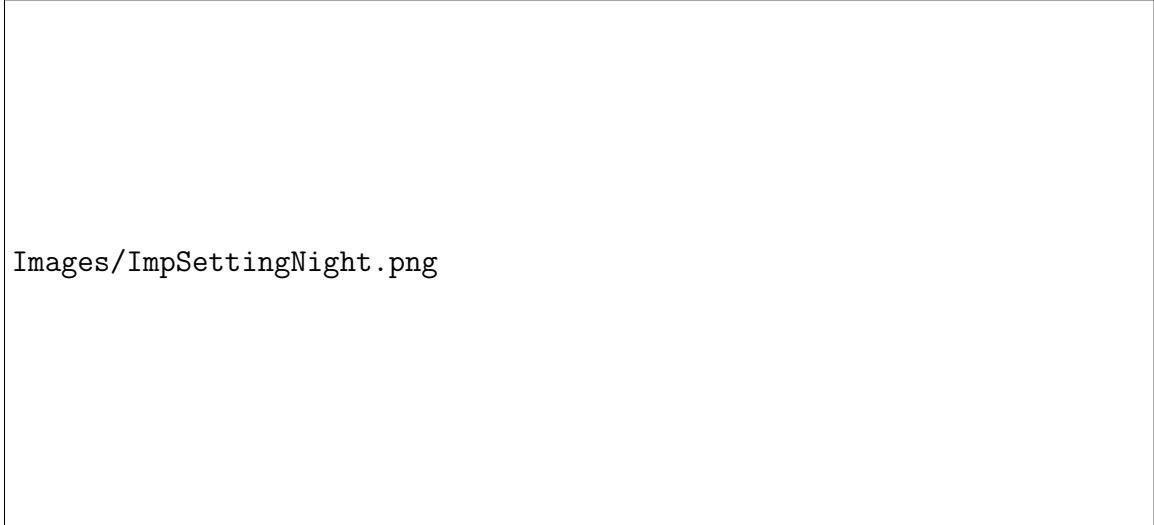
Our system currently offers two primary user settings: password modification and customization of the system's visual appearance as illustrated in 4.33.

The image shows a screenshot of a 'Setting Page' in a light theme. The page has a white background with a dark header bar at the top. The text 'Images/ImpSettingDay.png' is visible in the lower-left area of the page.

Images/ImpSettingDay.png

Figure 4.33: Setting Page

To enhance user experience and minimize visual strain, we have incorporated both light and dark mode options into our system. Users can seamlessly switch between these modes to match their preferred system appearance. Figure 4.34 visually demonstrates the system's dark mode interface.

The image shows a screenshot of the 'Setting Page' in a dark theme (night mode). The background is dark, and the text is light. The text 'Images/ImpSettingNight.png' is visible in the lower-left area of the page.

Images/ImpSettingNight.png

Figure 4.34: Setting Page in Night Mode

Chapter 5

Result analysis

5.1 Experimental Setup and Metrics

Evaluation metrics are quantitative measures used to assess the performance of a machine learning or deep learning model. They help us understand how well a model is performing on a given task and identify areas for improvement. To select the most effective model for our activity detection system, we conducted a thorough evaluation of the LSTM, Bi-LSTM, GRU, and attention-based LSTM models.

Our primary objective was to achieve high accuracy in detecting falls while minimizing false positives and false negatives. Therefore, we evaluated the model's performance using accuracy, precision, recall, and F1-score.

- **Accuracy:** Accuracy measures the probability of the model's correct predictions. We can calculate it by dividing the number of correct predictions (true positives + true negatives) by the total number of predictions.
- **Precision:** Precision statistically quantifies the proportion of correct positive predictions (true positives) out of all positive predictions (true positives + false positives).
- **Recall:** Recall quantifies the correct positive predictions (true positives) out of all actual positive instances (true positives + false negatives).
- **F1-score:** The F1-score is a metric that balances precision and recall, a common ground.

We used visual tools such as confusion matrices to understand the model’s performance and identify potential areas for improvement. These visualizations provided valuable insights into the model’s classification accuracy and helped us refine the model.

5.2 Initial Result Investigation

Table 5.1 provides a detailed comparison of the four models’ performance on the initial dataset, evaluated using the mentioned metrics.

Model	Accuracy	Precision	Recall	F1 - Score
LSTM	0.90	0.84	0.90	0.86
Bi-LSTM	0.88	0.83	0.88	0.85
Attention-based LSTM	0.89	0.84	0.89	0.86
GRU	0.87	0.81	0.87	0.84

Table 5.1: Models Performance Analysis on Initial Dataset

Our initial goal was to achieve an accuracy rate of at least 80% using the pre-collected dataset. We began experimenting with LSTM and GRU models, eventually selecting LSTM as the superior performer based on the evaluation metrics shown in 5.1. To further explore the potential of LSTM architectures, we implemented variations such as attention-based LSTM and Bi-LSTM. These models also achieved our desired accuracy and performance thresholds. Given the limitations of the initial dataset, which primarily focused on upper body movements and differed from our target activities, we refrained from extensive hyperparameter tuning. We reasoned that further adjustments might not yield significant improvements and could potentially introduce overfitting. Our focus was on selecting the most promising model architecture, recognizing that the dataset’s characteristics would play a crucial role in the model’s final performance.

As illustrated in Figures 5.1, 5.2, 5.3, and 5.4, the models demonstrated strong performance across all activities using the initial dataset, with the exception of the final class. Notably, the models exhibited a tendency to confuse ‘Clapping Hands’ and ‘Boxing’.



Figure 5.1: Confusion Matrix for LSTM Model on Initial Dataset



Figure 5.2: Confusion Matrix for Bi-LSTM Model on Initial Dataset



Figure 5.3: Confusion Matrix for GRU Model on Initial Dataset



Figure 5.4: Confusion Matrix for Attention-Based LSTM Model on Initial Dataset

Despite the observed confusion between certain activity classes, we were generally satisfied with the models' performance on the initial dataset. Encouraged by these results, we proceeded to train the selected model on our actual children's dataset.

5.3 Final Result Investigation

To investigate the potential benefits of dynamic learning rates, we trained all four models with and without this technique. Our initial experiments demonstrated that a dynamic learning rate strategy significantly enhanced the GRU model's performance on the existing dataset. Therefore, we conducted further evaluations with and without fine-tuning to determine the potential benefits for other models and to optimize performance on our children's dataset. To ensure the model's performance aligns with our system's primary objective of fall detection, we prioritized the model that achieved the highest F1-score for this specific task.

In the following sections, we will present a comprehensive analysis of each model's results and discuss the factors that influenced our selection of the most suitable model for our activity detection system.

5.3.1 LSTM

The LSTM model consistently demonstrated strong performance on our dataset, achieving an accuracy rate exceeding 95% both with and without fine-tuning. Table 5.2 provides a detailed comparison of these results.

Model	Accuracy	Precision	Recall	F1 - Score
LSTM	0.96	0.96	0.96	0.96
Fine-Tuned LSTM	0.97	0.97	0.97	0.97

Table 5.2: Comparison of LSTM Model Performance with and without Fine-Tuning

To gain deeper insights into the performance of each model, we visualized confusion matrices as shown in Table 5.3. Table 5.4 provides a detailed comparison of the F1-scores for both LSTM models across different activities.

LSTM	Fine-Tuned LSTM
 The image shows a placeholder for the confusion matrix of the standard LSTM model. The text 'Images/custmLSTM_CM.png' is visible within the box.	 The image shows a placeholder for the confusion matrix of the fine-tuned LSTM model. The text 'Images/customLSTM_LR_CM.png' is visible within the box.

Table 5.3: Confusion Matrix comparison of LSTM Models on Our Dataset

While the fine-tuned LSTM model demonstrated high overall F1-scores, a closer examination of individual class performance revealed that the standard LSTM model achieved a higher F1-score for the 'Fall' class. However, we observed a lower false negative rate of 'Fall' class in fine-tuned LSTM model compared to the regular LSTM model. This improvement in detecting actual fall events led us to prioritize the fine-tuned LSTM model for further analysis.

Model	Sit	Stand	Walk	Bend	Fall
LSTM	0.97	0.96	0.94	0.94	0.97
Fine-Tuned LSTM	0.98	0.97	0.96	0.96	0.96

Table 5.4: LSTM Model Performance Analysis: F1-Score Comparison

5.3.2 GRU

Both the standard and tuned GRU models achieved an impressive accuracy rate of 96%, indicating minimal variation in their overall performance, as summarized in Table 5.5. However, a closer analysis of the confusion matrices presented in Table 5.6 revealed that the fine-tuned GRU model detected fewer true positive fall cases.

Model	Accuracy	Precision	Recall	F1 - Score
GRU	0.96	0.96	0.96	0.96
Fine-Tuned GRU	0.96	0.96	0.96	0.96

Table 5.5: Comparison of GRU Model Performance with and without Fine-Tuning

GRU	Fine-Tuned GRU
Images/customGRU_CM.png	Images/customGRU_LR_CM.png

Table 5.6: Confusion Matrix comparison of GRU Models on Our Dataset

While both GRU models exhibited comparable F1 scores for fall detection in 5.7, a closer examination of their confusion matrices revealed a slight advantage for the non-tuned GRU model in terms of false negatives. Given the importance of minimizing false negatives in this application, we have selected the regular GRU model for our final comparison.

Model	Sit	Stand	Walk	Bend	Fall
GRU	0.97	0.97	0.93	0.94	0.97
Fine-Tuned GRU	0.97	0.96	0.95	0.95	0.97

Table 5.7: GRU Model Performance Analysis: F1-Score Comparison

5.3.3 Bi-LSTM

As summarized in Table 5.8, the tuned Bi-LSTM model consistently outperformed its regular counterpart and all other models considered, including LSTM and GRU.

Model	Accuracy	Precision	Recall	F1 - Score
Bi-LSTM	0.98	0.98	0.98	0.98
Fine-Tuned Bi-LSTM	0.99	0.99	0.99	0.99

Table 5.8: Comparison of Bi-LSTM Model Performance with and without Fine-Tuning

Given the critical importance of minimizing false negatives in fall detection, we prioritized minimizing false negatives over high accuracy. As shown in Table

5.9, both Bi-LSTM models achieved same F1 scores and based on Table 5.10, the regular Bi-LSTM model exhibited a marginally lower false negative rate compared to its tuned counterpart. Despite a 1% reduction in overall accuracy, we selected the non-tuned Bi-LSTM model for further comparison.

Model	Sit	Stand	Walk	Bend	Fall
Bi-LSTM	0.99	0.98	0.97	0.97	0.98
Fine-Tuned Bi-LSTM	0.99	0.98	0.99	0.99	0.98

Table 5.9: Bi-LSTM Model Performance Analysis: F1-Score Comparison

Bi-LSTM	Fine-Tuned Bi-LSTM
 Images/customBiLSTM_CM.png	 Images/customBiLSTM_LR_CM.png

Table 5.10: Confusion Matrix comparison of Bi-LSTM Models on Our Dataset

5.3.4 Attention-Based LSTM

As depicted in Table 5.11, both attention-based LSTM models exhibited identical overall performance, achieving identical accuracy, recall, precision, and F1-score values.

Model	Accuracy	Precision	Recall	F1 - Score
Attention-Based LSTM	0.97	0.97	0.97	0.97
Fine-Tuned Attention-Based LSTM	0.97	0.97	0.97	0.97

Table 5.11: Comparison of Attention-Based LSTM Model Performance with and without Fine-Tuning

Both attention-based LSTM models showed a remarkable ability to detect falls accurately, as evidenced by the minimal false negatives observed in their confusion matrices (Table 5.12). Furthermore, a comparative analysis of their F1 scores in Table

5.13 revealed that the regular attention-based LSTM model consistently achieved superior performance across all activities, although marginally.

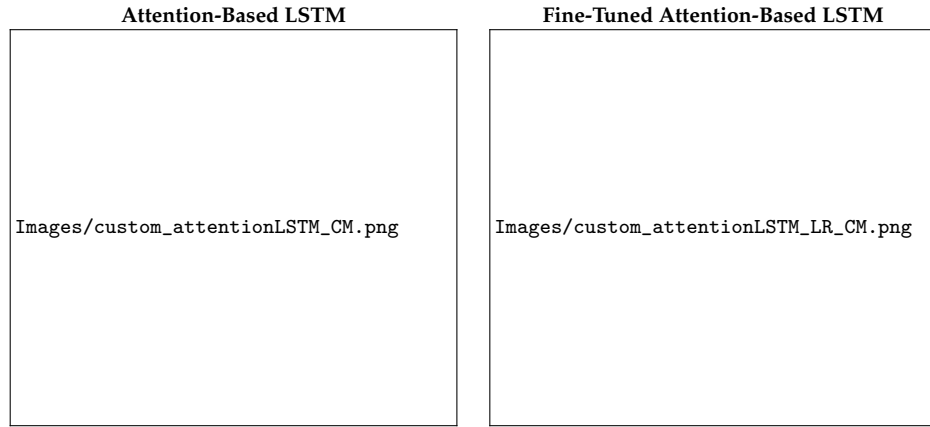


Table 5.12: Confusion Matrix comparison of Attention Based LSTM Models on Our Dataset

Based on the comparative analysis presented above, the non-tuned attention-based LSTM model was chosen for further analysis due to its consistently superior performance across all activities.

Model	Sit	Stand	Walk	Bend	Fall
Attention-Based LSTM	0.98	0.98	0.95	0.96	0.98
Fine-Tuned Attention-Based LSTM	0.98	0.97	0.95	0.95	0.98

Table 5.13: Attention-Based LSTM Model Performance Analysis: F1-Score Comparison

5.3.5 Final Model Selection

Based on our preliminary analysis, four models were shortlisted for further evaluation: the fine-tuned LSTM, GRU, Bi-LSTM, and Attention-based LSTM. To identify the optimal model for our application, we conducted a comprehensive comparison using various metrics.

A final comparison table 5.14 was constructed, incorporating accuracy and overall F1-score as key evaluation criteria. Additionally, we visualized the Receiver Operating Characteristic (ROC) curves of each model to assess their ability to discriminate between positive and negative instances.

Among the selected models, the Bi-LSTM demonstrated superior performance, achieving a remarkable 98% accuracy and F1-score. While it ranked second overall

in our analysis, it exhibited the highest sensitivity to fall events, correctly identifying the greatest number of falls and exhibiting the lowest false negative rate for the 'Fall' class.

Model	Accuracy	Precision	Recall	F1 - Score
Fine-Tuned LSTM	0.97	0.97	0.97	0.97
GRU	0.96	0.96	0.96	0.96
Bi-LSTM	0.98	0.98	0.98	0.98
Attention-Based LSTM	0.97	0.97	0.97	0.97

Table 5.14: Model Comparison for Final Selection

Table 5.15 presents the ROC curve analysis for the four selected models: fine-tuned LSTM, regular GRU, regular Bi-directional LSTM, and regular Attention-based LSTM. The ROC curve plots the true positive rate (sensitivity) against the false positive rate (specificity) for different classification thresholds.

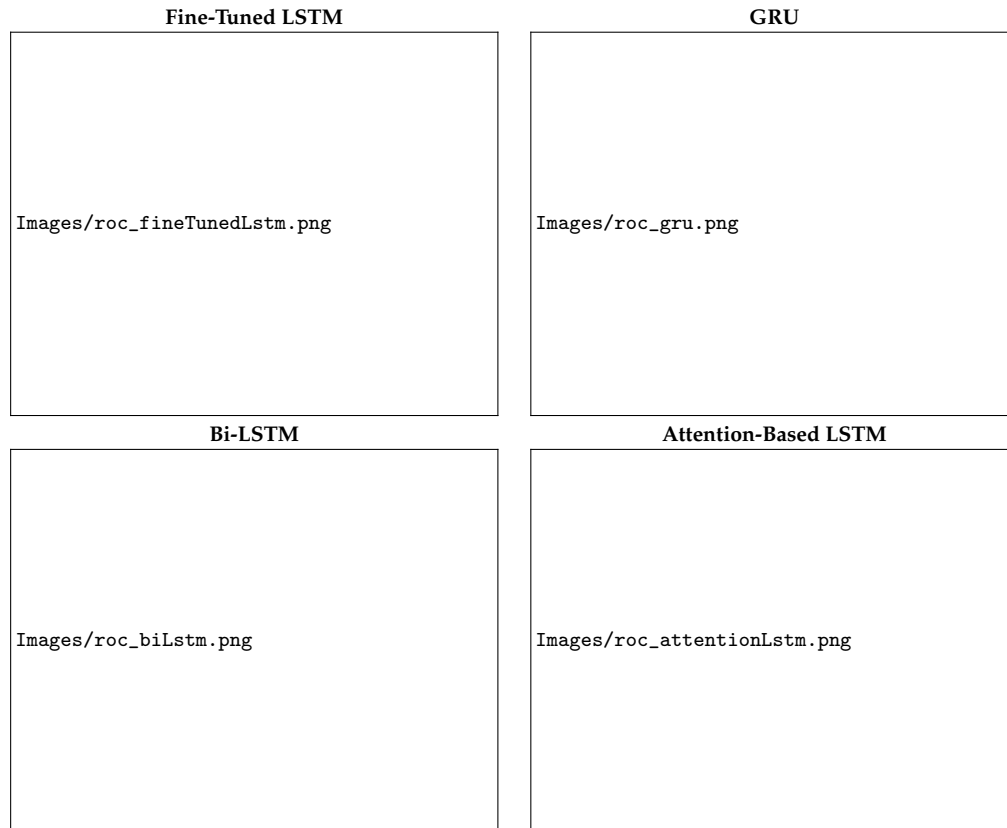


Table 5.15: ROC Curve Analysis of Finalized Models

Moreover, the training and validation accuracy trends were examined to identify any potential overfitting or underfitting issues.

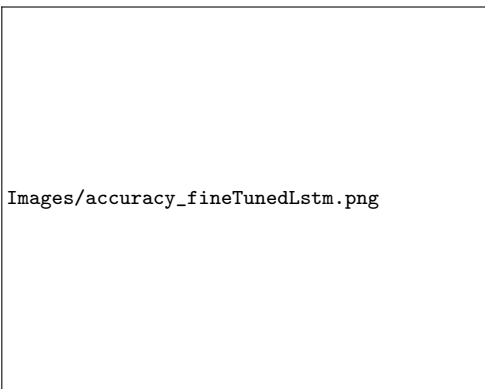
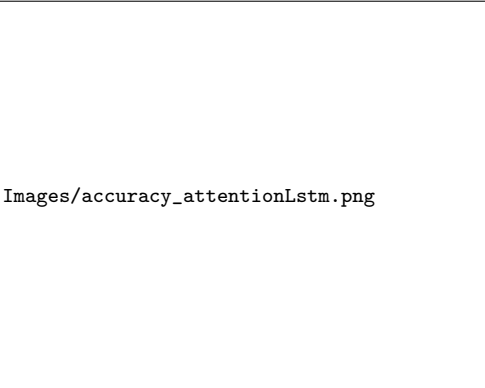
Fine-Tuned LSTM  Images/accuracy_fineTunedLstm.png	GRU  Images/accuracy_gru.png
Bi-directional LSTM  Images/accuracy_biLstm.png	Attention Based LSTM  Images/accuracy_attentionLstm.png

Table 5.16: Train and Validation Accuracy Analysis of Finalized Models

Table 5.16 presents the training and validation accuracy trends for the four selected models over 100 epochs. The training accuracy represents the model’s performance on the training data, while the validation accuracy reflects its performance on unseen data. By comparing these trends, we can assess the model’s susceptibility to overfitting or underfitting. A significant gap between training and validation accuracy suggests overfitting, while a consistently low validation accuracy indicates underfitting.

To evaluate the performance of our selected model in the context of existing literature, we conducted a comparative analysis of accuracy metrics. Table 5.17 presents the results of this comparison, highlighting the key differences and similarities between our model and previous skeleton-based approaches.

As shown in Table 5.17, our proposed approach demonstrated high performance comparable to state-of-the-art methods in the field of fall and activity detection. While our highest-performing model achieved an impressive 99% accuracy, sur-

passing existing benchmarks, we ultimately selected the second-best performing model (98% accuracy) to prioritize fall detection sensitivity. This decision aligned with the primary objective of our system, which is to minimize false negatives and ensure accurate identification of fall events. By selecting the model with the highest sensitivity to falls, even at a slight cost to overall accuracy, we have optimized the system’s ability to safeguard users from potential injuries associated with undetected falls.

Paper	Classification Model	Accuracy (Overall)
Muhammad et al. (2021)	Bi-LSTM with attention	92.53%
Chang et al. (2021)	LSTM with OpenPose-Light	98%
Huu et al. (2021)	MobilenetV2 + LSTM	92%
McCall et al. (2024)	Transformer model	97.95%
Núñez-Marcos & Arganda-Carreras (2024)	Transformer model with Feedforward Network	93.17%
Ours	Openpose + Bi-LSTM	98%

Table 5.17: Comparative Analysis of Accuracy Metrics for Selected Model and Existing Skeleton-Based Approaches

Chapter 6

Conclusions

6.1 Societal, health, safety, legal and cultural aspects

Societal, health, safety, legal, and cultural aspects are the various dimensions that need to be considered when evaluating the impact of a technology or system on society. Therefore, we critically analyzed our system to have a greater impact on these areas.

6.1.1 Societal Impact

Our proposed system offers an alternative way to reduce the risk of accidents and injuries in children, creating a more secure environment for child care. With a system like this, parents can have increased peace of mind and contribute to a stronger and more resilient community.

6.1.2 Health and Safety Impact

A safe and supportive environment can provide optimal child development and growth [Jamieson et al. \(2017\)](#). We are prioritizing fall and activity detection in children because this is a crucial developmental age for anyone. If we can help reduce serious injury by promptly alerting caregivers, we are essentially safeguarding the future generation.

6.1.3 Legal and Cultural Impact

The successful implementation of our system may require cultural adjustments and an openness to new technologies. Since we are working with children, we need to handle everything with extra care. Therefore, it is our responsibility to critically handle all the data related to the system and provide the best service.

6.1.4 Course of Action to Mitigate Concerns

We took informed consent from parents and caregivers during data collection to ensure transparency and trust in our procedures. We worked closely with parents, caregivers, and other key stakeholders to determine the system’s ability to meet their needs and priorities.

6.2 Impact on Environment and Sustainability

Aligning with the Sustainable Development Goals (SDGs) for 2030, our system can contribute in children’s safety and good health [Jamieson et al. \(2017\)](#). Detecting falls in time is crucial, as it allows immediate assistance, which can help reduce the risk of serious injuries or fatalities in children. In addition, our system can help to identify potentially risky areas, allowing caregivers to adjust their supervision and the environment to improve child safety.

Our dataset can also drive innovation in sustainable childcare and can be used to train models to develop robotic devices to enhance children’s accessibility [Neto et al. \(2021\)](#). Furthermore, real-time skeleton data can provide objective measurements of physical activity levels, postural analysis, and activity patterns, which contribute to more effective and personalized childcare strategies.

6.3 Ethical and professional principles

6.3.1 Data Privacy and Security

As we were working with children’s data, data privacy, and security were always our top priorities. We adopted a skeleton-based approach to ensure that only skeletal data is processed within our services, minimizing exposure to sensitive information. Our proposed system further enhances security by handling data

preprocessing directly on the local device. Additionally, we implemented user authentication for login, ensuring that unauthorized individuals cannot access the web application. This layered approach safeguards user information while maintaining the integrity and privacy of the data.

6.3.2 Accuracy and Reliability

To ensure the most accurate and reliable user experience, we selected the model that consistently identified falls with the highest F1-score and lowest false negative rate. We then conducted thorough testing to verify the system’s reliability. With the help of our developed model, the system can provide reliable predictions for children.

6.3.3 Ethical Use of Technology

Our system is designed to serve a noble purpose: enhancing child safety and well-being. We have ensured that our system does not collect or store sensitive personal information, and we have taken steps to prevent unauthorized access to user data.

6.3.4 Transparency and Consent

We have made transparency and informed consent a top priority throughout the data collection process. We cleared our objectives with parents, and made sure that children are fully informed about how their data will be used to train our model.

6.4 Brief Summary

While fall and activity detection systems have been widely studied for adults, the unique needs of children require specialized solutions. General-purpose systems may not be as effective for pediatric populations due to differences in physiology and movement patterns. To address this gap, we developed a system specifically designed for children. Recognizing the limited availability of public datasets focused on children’s activities and falls, we took the initiative to curate our dataset. This allowed us to capture authentic scenarios and train our deep learning model on data that accurately represents the target population. By training the model on

children’s dataset, we achieved excellent accuracy and F1-score in detection within the pediatric context.

6.5 Future works

Our research explored different RNN architectures, such as Bi-LSTM, Attention-based LSTM, LSTM, and GRU to identify the most effective model for our detection task. In the future, we plan to use a transformer-based model, which has shown promising performance in recent studies [Núñez-Marcos & Arganda-Carreras \(2024\)](#); [Vaswani et al. \(2023\)](#). After getting clearance of our video data, we will explore the potential benefits of using an image-based input, rather than relying solely on keypoints [Vaiyapuri et al. \(2021\)](#); [Ferrando et al. \(2023\)](#).

We will implement explainable AI techniques, such as LIME and SHAP, to identify the keyframes or data points that significantly contribute to the detection. With this integration, we can gain valuable insights into the model’s reasoning and potentially improve its performance and interpretability [Ribeiro et al. \(2016\)](#); [Samek et al. \(2017\)](#); [Yeo et al. \(2023\)](#). Lastly, we will strive to expand our dataset by adding a diverse range of fall scenarios. New data can help to improve the model’s generalization ability and enhance its overall effectiveness.

References

- Abboush, M., Knieke, C., & Rausch, A. (2023). Gru-based denoising autoencoder for detection and clustering of unknown single and concurrent faults during system integration testing of automotive software systems. *Sensors*, 23(14). Retrieved from <https://www.mdpi.com/1424-8220/23/14/6606> doi: 10.3390/s23146606
- Aharon, S., Louis-Dupont, Ofri Masad, Yurkova, K., Lotem Fridman, Lkdci, ... Eran-Deci (2021). *Super-gradients*. GitHub. Retrieved from <https://zenodo.org/record/7789328> doi: 10.5281/ZENODO.7789328
- Ahmed, A., Siam, M. H. B., Shojon, M., Hasan, M. M., Raheem, E., & Hossain, M. S. (2022). Accidental poisoning in children: a single centre case series study in bangladesh. *BMJ Paediatrics Open*, 6(1), 1097-1105. doi: 10.1136/bmjpo-2022-001541
- Al-Azani, S., Sait, S. M., & Al-Utaibi, K. A. (2022). A comprehensive literature review on children's databases for machine learning applications. *IEEE Access*, 10, 12262-12285. doi: 10.1109/ACCESS.2022.3146008
- Amazon. (2024). *Amazon ec2 c5 instances*. Retrieved from <https://aws.amazon.com/ec2/instance-types/c5/>
- Amor, J. D., & James, C. J. (2015). Setting the scene: Mobile and wearable technology for managing healthcare and wellbeing. In *2015 37th annual international conference of the ieee engineering in medicine and biology society (embc)* (pp. 7752–7755).
- Andriluka, M., Pishchulin, L., Gehler, P., & Schiele, B. (2014). 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 3686–3693).
- Auvinet, E., Rougier, C., Meunier, J., St-Arnaud, A., & Rousseau, J. (2010). Multiple cameras fall dataset. *DIRO-Université de Montréal, Tech. Rep*, 1350, 24.
- Bahdanau, D. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

- Banerjee, N., Sharma, N., Soni, K. D., Bansal, V., Mahajan, A., Khajanchi, M., . . . Roy, N. (2022). Are home environment injuries more fatal in children and the elderly? *Injury*, 53, 1987-1993. doi: doi.org/10.1016/j.injury.2022.03.050.
- Beddiar, D. R., Oussalah, M., & Nini, B. (2022). Fall detection using body geometry and human pose estimation in video sequences. *Journal of Visual Communication and Image Representation*, 82, 103407.
- Boegemann, B. N. (2021). *Reducing fall risk in children aged 0-4 years through caregiver education* (Unpublished doctoral dissertation). The University of Arizona.
- Cao, Z., Hidalgo Martinez, G., Simon, T., Wei, S., & Sheikh, Y. A. (2019). Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Chang, W.-J., Hsu, C.-H., & Chen, L.-B. (2021). A pose estimation-based fall detection methodology using artificial intelligence edge computing. *IEEE Access*, 9, 129965–129976.
- Charfi, I., Miteran, J., Dubois, J., Atri, M., & Tourki, R. (2012). Definition and performance evaluation of a robust svm based fall detection solution. In *2012 eighth international conference on signal image technology and internet based systems* (pp. 218–224).
- Chen, Z., Zhu, Q., Soh, Y. C., & Zhang, L. (2017). Robust human activity recognition using smartphone sensors via ct-pca and online svm. *IEEE transactions on industrial informatics*, 13(6), 3070–3080.
- Cho, K. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.
- DevIQ. (2024). *Repository pattern*. Retrieved from <https://deviq.com/design-patterns/repository-pattern>
- Eiffert, S. (2018). *Rnn for human activity recognition - 2d pose input*. <https://github.com/stuarteiffert/RNN-for-Human-Activity-Recognition-using-2D-Pose-Input>. ([GitHub repository])
- Ferrando, J., Gállego, G. I., Tsiamas, I., & Costa-jussà, M. R. (2023, July). Explaining how transformers use context to build predictions. In A. Rogers, J. Boyd-Graber, & N. Okazaki (Eds.), *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 5486–5513). Toronto, Canada: Association for Computational Linguistics. Retrieved from <https://aclanthology.org/2023.acl-long.301> doi: 10.18653/v1/2023.acl-long.301

- Figaji, A. A. (2017). Anatomical and physiological differences between children and adults relevant to traumatic brain injury and the implications for clinical assessment and care. *Frontiers in neurology*, 8, 685.
- Galán-Mercant, A., Ortiz, A., Herrera-Viedma, E., Tomas, M. T., Fernandes, B., & Moral-Munoz, J. A. (2019). Assessing physical activity and functional fitness level using convolutional neural networks. *Knowledge-Based Systems*, 185, 104939.
- Gao, M., Li, J., Zhou, D., Zhi, Y., Zhang, M., & Li, B. (2023). Fall detection based on openpose and mobilenetv2 network. *IET Image Processing*, 17(3), 722–732.
- Google AI. (2020). *Mediapipe pose*. Retrieved from https://ai.google.dev/edge/mediapipe/solutions/vision/pose_landmarker
- Graves, A. (2013). Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.
- Guerrero, J. C. E., España, E. M., Añasco, M. M., & Lopera, J. E. P. (2022). Dataset for human fall recognition in an uncontrolled environment. *Data in Brief*, 45, 108610. Retrieved from <https://www.sciencedirect.com/science/article/pii/S2352340922008162> doi: <https://doi.org/10.1016/j.dib.2022.108610>
- He, J.-Y., Wu, X., Cheng, Z.-Q., Yuan, Z., & Jiang, Y.-G. (2021). Db-lstm: Densely-connected bi-directional lstm for human action recognition. *Neurocomputing*, 444, 319–331.
- Hendry, D., Rohl, A. L., Rasmussen, C. L., Zabatiero, J., Cliff, D. P., Smith, S. S., ... Campbell, A. (2023). Objective measurement of posture and movement in young children using wearable sensors and customised mathematical approaches: A systematic review. *Sensors*, 23(24), 9661.
- Hosseini, A., Fazeli, S., Vliet, E. v., Valencia, L., Habre, R., Sarrafzadeh, M., & Bui, A. (2018). Children activity recognition: Challenges and strategies. In *2018 40th annual international conference of the ieee engineering in medicine and biology society (embc)* (p. 4331-4334). doi: 10.1109/EMBC.2018.8513320
- Huu, P. N., Thi, N. N., & Ngoc, T. P. (2021). Proposing posture recognition system combining mobilenetv2 and lstm for medical surveillance. *IEEE Access*, 10, 1839–1849.
- Jamieson, L., Richter, L., & Cavoukian, R. (2017). Striving for the sustainable development goals: What do children need to thrive. *ChildGauge*, 201733.
- Jhuang, H., Gall, J., Zuffi, S., Schmid, C., & Black, M. J. (2013). Towards understanding action recognition. In *Proceedings of the ieee international conference on computer vision* (pp. 3192–3199).

- Kang, H., & Jung, E. H. (2021). The smart wearables-privacy paradox: A cluster analysis of smartwatch users. *Behaviour & Information Technology*, 40(16), 1755-1768.
- Keren, G., & Schuller, B. (2016). Convolutional rnn: An enhanced model for extracting features from sequential data. In *2016 international joint conference on neural networks (ijcnn)* (p. 3412-3419). doi: 10.1109/IJCNN.2016.7727636
- Khraief, C., Benzarti, F., & Amiri, H. (2020). Elderly fall detection based on multi-stream deep convolutional networks. *Multimedia Tools and Applications*, 79(27), 19537–19560.
- Kour, N., Sunanda, & Arora, S. (2022). A vision-based gait dataset for knee osteoarthritis and parkinson's disease analysis with severity levels. In A. Khanna, D. Gupta, S. Bhattacharyya, A. E. Hassanien, S. Anand, & A. Jaiswal (Eds.), *International conference on innovative computing and communications* (pp. 303–317). Singapore: Springer Singapore.
- Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., & Serre, T. (2011). Hmdb: a large video database for human motion recognition. In *2011 international conference on computer vision* (pp. 2556–2563).
- Kwolek, B., & Kepski, M. (2014). Human fall detection on embedded platform using depth maps and wireless accelerometer. *Computer methods and programs in biomedicine*, 117(3), 489–501.
- Liu, J., Luo, J., & Shah, M. (2009). Recognizing realistic actions from videos “in the wild”. In *2009 ieee conference on computer vision and pattern recognition* (pp. 1996–2003).
- Martínez-Villaseñor, L., Ponce, H., Brieva, J., Moya-Albor, E., Núñez-Martínez, J., & Peñafort-Asturiano, C. (2019). Up-fall detection dataset: A multimodal approach. *Sensors*, 19(9). Retrieved from <https://www.mdpi.com/1424-8220/19/9/1988> doi: 10.3390/s19091988
- McCall, S., Kolawole, S. S., Naz, A., Gong, L., Ahmed, S. W., Prasad, P. S., ... Ardakani, S. P. (2024). Computer vision based transfer learning-aided transformer model for fall detection and prediction. *IEEE Access*.
- MongoDB. (2024). *Why use mongodb and when to use it?* Retrieved from <https://www.mongodb.com/resources/products/fundamentals/why-use-mongodb>
- Morrison, A. (2016). *At what age can children be left alone?* Retrieved from <https://www.bbc.com/news/uk-35803414>

- Mubashir, M., Shao, L., & Seed, L. (2013). A survey on fall detection: Principles and approaches. *Neurocomputing*, 100, 144–152.
- Muhammad, K., Ullah, A., Imran, A. S., Sajjad, M., Kiran, M. S., Sannino, G., ... others (2021). Human action recognition using attention based lstm network with dilated cnn features. *Future Generation Computer Systems*, 125, 820–830.
- Neto, I., Nicolau, H., & Paiva, A. (2021). Community based robot design for classrooms with mixed visual abilities children. In *Proceedings of the 2021 chi conference on human factors in computing systems*. New York, NY, USA: Association for Computing Machinery. Retrieved from <https://doi.org/10.1145/3411764.3445135> doi: 10.1145/3411764.3445135
- Notion Labs, I. (2024). *Notion*. Retrieved from <https://www.notion.so/> (Accessed: 2024-10-03)
- Núñez-Marcos, A., & Arganda-Carreras, I. (2024). Transformer-based fall detection in videos. *Engineering Applications of Artificial Intelligence*, 132, 107937.
- Organization, W. H. (2021). *Falls*. Retrieved from <https://www.who.int/news-room/fact-sheets/detail/falls>
- Peterson, A. B., & Thomas, K. E. (2021). Incidence of nonfatal traumatic brain injury-related hospitalizations - united states, 2018. In *Morbidity and mortality weekly report* (Vol. 70, p. 1664-1668). doi: 10.15585/mmwr.mm7048a3
- Rawassizadeh, R., Price, B. A., & Petre, M. (2014). Wearables: Has the age of smartwatches finally arrived? *Communications of the ACM*, 58(1), 45–47.
- Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (p. 1135–1144). New York, NY, USA: Association for Computing Machinery. Retrieved from <https://doi.org/10.1145/2939672.2939778> doi: 10.1145/2939672.2939778
- Rumhi, A. A., Awisi, H. A., Buwaiqi, M. A., & Rabaani, S. A. (2020). Home accidents among children: A retrospective study at a tertiary care center in oman. *Oman Medical Journal*, 35. doi: 10.5001/omj.2020.03
- Samek, W., Wiegand, T., & Müller, K.-R. (2017). *Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models*. Retrieved from <https://arxiv.org/abs/1708.08296>
- Schindler, K., & van Gool, L. (2008). Action snippets: How many frames does human action recognition require? In *2008 ieee conference on computer vision and pattern recognition* (p. 1-8). doi: 10.1109/CVPR.2008.4587730

- Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11), 2673–2681.
- Shao, L., Zhen, X., Tao, D., & Li, X. (2013). Spatio-temporal laplacian pyramid coding for action recognition. *IEEE Transactions on Cybernetics*, 44(6), 817–827.
- Shekhawat, S. S. (2024). *Crud using the repository pattern in mvc*. Retrieved from <https://www.c-sharpcorner.com/UploadFile/3d39b4/crud-using-the-repository-pattern-in-mvc/>
- Shu, F., & Shu, J. (2021). *An eight-camera fall detection system using human fall pattern recognition via machine learning by a low-cost android box*. *sci rep* 11, 2471 (2021).
- Soomro, K. (2012). Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*.
- Suzuki, S., Mitsukura, Y., Igarashi, H., Kobayashi, H., & Harashima, F. (2012). Activity recognition for children using self-organizing map. In *2012 ieee ro-man: The 21st ieee international symposium on robot and human interactive communication* (p. 653-658). doi: 10.1109/ROMAN.2012.6343825
- Vaiyapuri, T., Lydia, E. L., Sikkandar, M. Y., Díaz, V. G., Pustokhina, I. V., & Pustokhin, D. A. (2021). Internet of things and deep learning enabled elderly fall detection model for smart homecare. *IEEE Access*, 9, 113879–113888.
- Van Laerhoven, K., Borazio, M., & Burdinski, J. H. (2015). Wear is your mobile? investigating phone carrying and use habits with a wearable device. *Frontiers in ICT*, 2, 10.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2023). *Attention is all you need*. Retrieved from <https://arxiv.org/abs/1706.03762>
- Yadav, S. K., Tiwari, K., Pandey, H. M., & Akbar, S. A. (2021). A review of multimodal human activity recognition with special emphasis on classification, applications, challenges and future directions. *Knowledge-Based Systems*, 223, 106970.
- Yeo, W. J., van der Heever, W., Mao, R., Cambria, E., Satapathy, R., & Mengaldo, G. (2023). A comprehensive review on financial explainable ai. *arXiv preprint arXiv:2309.11960*.